

CG3 Physically based Graphics

numeric solution of differential
equations

Numeric Foundations

Differential Equations

- A differential equation describes a relation between a function in one or several variables and the [partial] derivatives of the function (often a system of equations)

- In case of a single variable, e.g. t one calls this an **ordinary differential equation**:

$$\{t, f(t), \dot{f}(t), \ddot{f}(t), \dddot{f}(t), \dots\}$$

$$\ddot{x}(t) = -\omega^2 x(t)$$

- If derivatives of several variables arise one calls this **partial differential equation** or pde:

$$\{x, y, t, f(x, y, t), \partial_t f, \partial_x f, \partial_y f, \dots\}$$

$$\frac{\partial^2 u(x, t)}{\partial t^2} = c^2 \frac{\partial^2 u(x, t)}{\partial x^2}$$

Numeric Foundations Differential Equations

- We look for unknown functions[s]
 $f(t[,x,y,...])$
- E.g.: $\ddot{f}(t) = -f(t) \Rightarrow f(t) = A \cos t + B \sin t$
 $f(0) = 0, \dot{f}(0) = 1 \Rightarrow A = 0, B = 1$
- For a unique solution with concrete values for A and B additional so called **boundary conditions** are needed
- More examples:
 - ordinary: oscilator, particle-spring system
 - pde: Wave Eq., Maxwell Eg., Schrödinger Eq.,
Einsteinian Field Eq., Navier-Stokes-Eq.

Numeric Foundations

Ordinary Differential Equations

- **Order n** of differential equation corresponds to order of highest derivative

- **implicit representation:**

$$\omega^2 x + \ddot{x} = 0 \quad DE\{t, f(t), \dot{f}(t), \ddot{f}(t), \dots\} = 0$$

- **explicit representation:**

$$\ddot{x} = -\omega^2 x \quad \frac{\partial^n f}{\partial t^n} = DE\{t, f(t), \dot{f}(t), \ddot{f}(t), \dots\}$$

- **Order reduction:** a DE of order n can be transformed into a system of DE of order 1.

$$\begin{aligned} f_0 &= x & f_0(t) &= f(t) & f_2(t) &= \dot{f}_1(t) \\ f_1 &= \dot{f}_0 = \dot{x} = v & f_1(t) &= \dot{f}_0(t) & \dot{f}_{n-1}(t) &= DG\{t, f_0(t), f_1(t), f_2(t), \dots\} \\ \dot{f}_1 &= \dot{v} = -\omega^2 x \end{aligned}$$

Numeric Foundations

Phase Space

$$\vec{y}(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}$$

- In the reduction to a system of 1st order DEs the function value are combined with derivatives up to $(n - 1)$ to a n -d **phase space**:

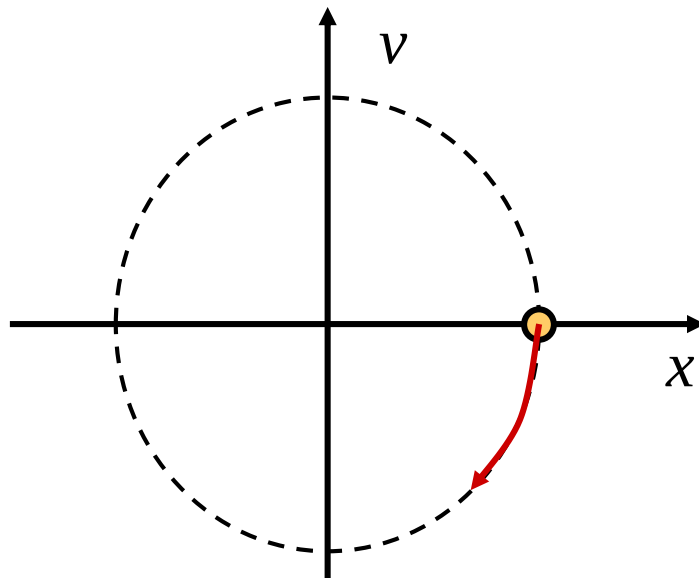
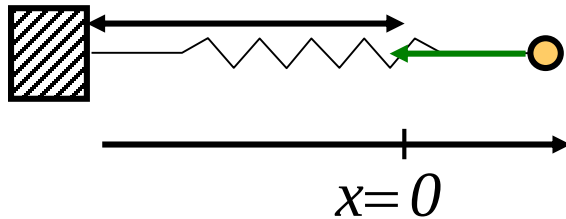
$$\vec{y}(t) = \left(\underline{x}(t) \quad \underline{\dot{x}}(t) \quad \underline{\ddot{x}}(t) \quad \cdots \quad \frac{\partial^{n-1}}{\partial t^{n-1}} \underline{x}(t) \right)$$

- A location $\vec{y}_0$ in phase space **uniquely defines time evolution**
- In physical system of 2nd order, the state is uniquely defined through position $\underline{x}_0$ and **velocity** $\vec{v}_0$ or **momentum** $\vec{p}_0 = m\vec{v}_0$
- **Careful**: instead of $\vec{y}$ often symbol $\underline{x}$ is used for a phase space location in literature

Numeric Foundations

Phase Space

Harmonic Oscillator



$$\ddot{x}(t) = -\omega^2 x(t), \quad \omega = \sqrt{\frac{k}{m}}$$

$$x(t) = A \sin \omega t + B \cos \omega t$$

$$v(t) = \omega (A \cos \omega t - B \sin \omega t)$$

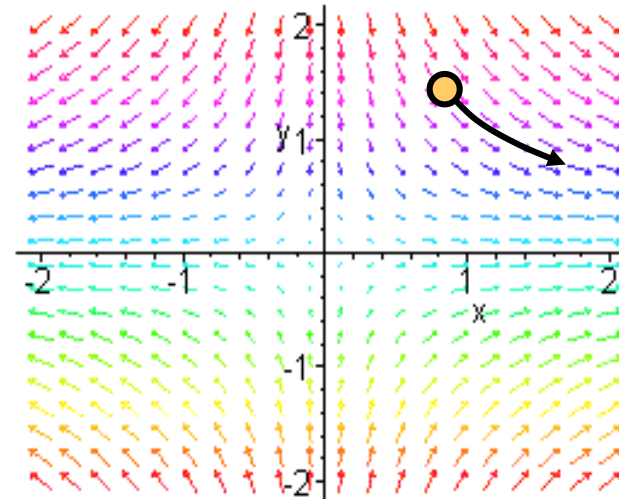
$$\vec{y}(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix} \quad \text{time evolution function}$$

$$\dot{\vec{y}}(t) = \begin{pmatrix} v(t) \\ -\omega^2 x(t) \end{pmatrix} = \vec{f}(t, \vec{y}(t)) \quad \text{equations of motion}$$

Numeric Foundations

Ordinary Differential Equations

- The **time evolution** $\vec{f}$ is vector field on the simulation domain $\Omega \subseteq \mathbf{R}^n$
- It defines in combination with phase space location $\vec{y}_0 = \vec{y}(t = 0)$ an **initial value problem**
- If $\vec{f}$ does not depend explicitly on t , the DE is called **autonomous**, otherwise **non-autonomous**
- In case of autonomous DE, the time evolution correspond to **stream-lines** of the vector field



Numeric Foundations

Ordinary Differential Equations

■ Theorem of Picard / Lindelöf:

If for given $\vec{f}: [0, a] \times \Omega \rightarrow \mathbf{R}^n$ the Lipschitz-condition: $\exists L: \forall t \in [0, a] \wedge \forall \vec{y}_1, \vec{y}_2 \in \Omega$:

$$\|\vec{f}(t, \vec{y}_1) - \vec{f}(t, \vec{y}_2)\| \leq L \cdot \|\vec{y}_1 - \vec{y}_2\|$$

holds, then for each initial value $\vec{y}_0 \in \Omega$ **exists a differentiable function $\vec{y}(t)$** which solves the initial value problem $\{\vec{y}_0, \vec{f}\}$.

- For existence and uniqueness of solution there are results for special classes of pdes but no general result is known.
- Body collisions violate the Lipschitz-condition such that Picard/Lindelöf cannot be applied.

Numeric Foundations

Analytic Solutions

- Look up solution formulary
 - Use algebra program like Maple
 - Use research related software like Crack
- Analytic Solution often not needed as
- There exist efficient numeric solvers with control of error
 - Collisions and other effects make system of DEs too complicated for analytic solutions

Numeric Integration

- Let us consider the initial value problem

$$\begin{aligned}\vec{y}(0) &= \vec{y}_0 \\ \dot{\vec{y}}(t) &= \vec{f}(t, \vec{y}(t))\end{aligned}$$

- Integration of both sides from 0 to t yields

$$\vec{y}(t) = \vec{y}_0 + \int_0^t \vec{f}(\tilde{t}, \vec{y}(\tilde{t})) d\tilde{t}$$

- Integral on right side is determined numerically

Numeric Integration

- This is not a typical integration as unknown function $\vec{y}(t)$ also appears on right side:

$$\vec{y}(t) = \int_0^t \vec{f}(\tau, \vec{y}(\tau)) d\tau + \vec{y}_0$$

- Approximate $\vec{y}(t)$ with polygonal line at discrete times $t_i := i \cdot h$ at $\vec{y}_0, \vec{y}_1, \vec{y}_2, \vec{y}_3, \dots$, where each phase location $\vec{y}_i$ approximates $\vec{y}(t_i)$.

- discretize one step: $\vec{y}_1 = \vec{f}(0, \vec{y}(0)) \cdot h + \vec{y}_0$

yields explicit Euler:

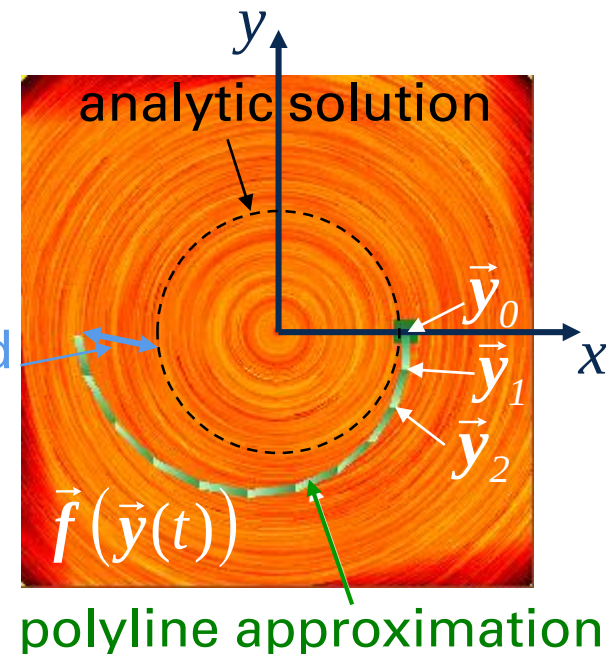
$$\vec{y}_{i+1} = h \cdot \vec{f}(t_i, \vec{y}_i) + \vec{y}_i$$

Numeric Integration

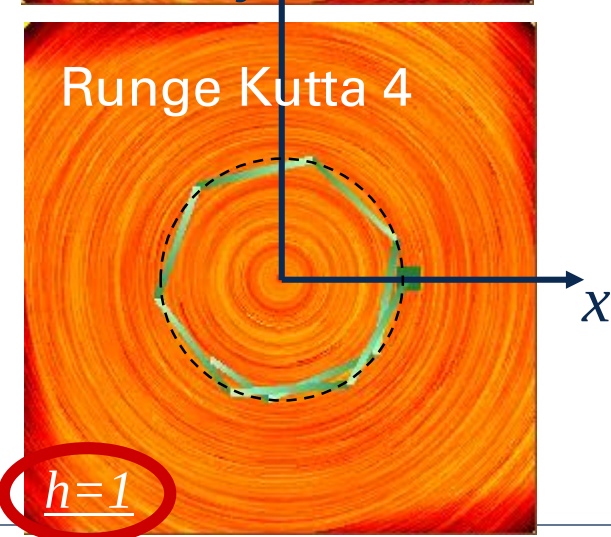
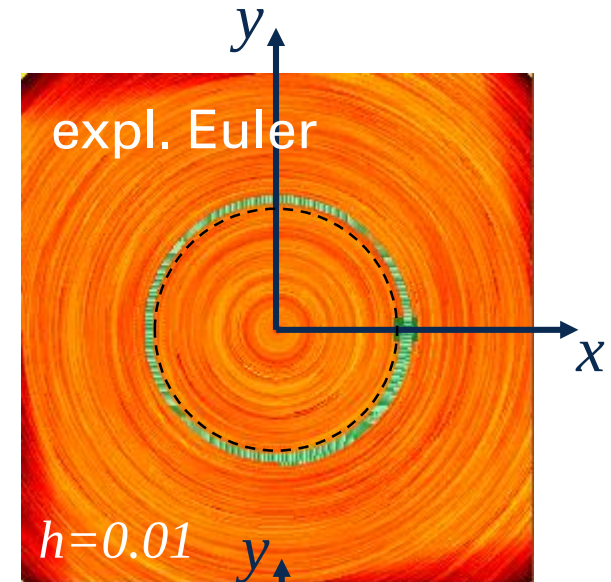
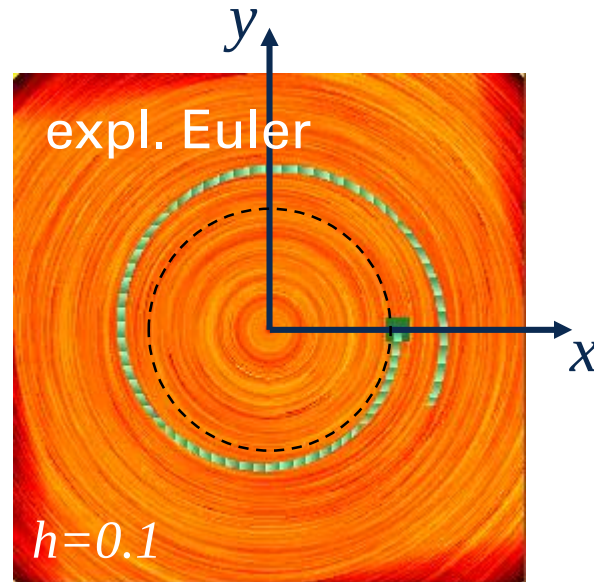
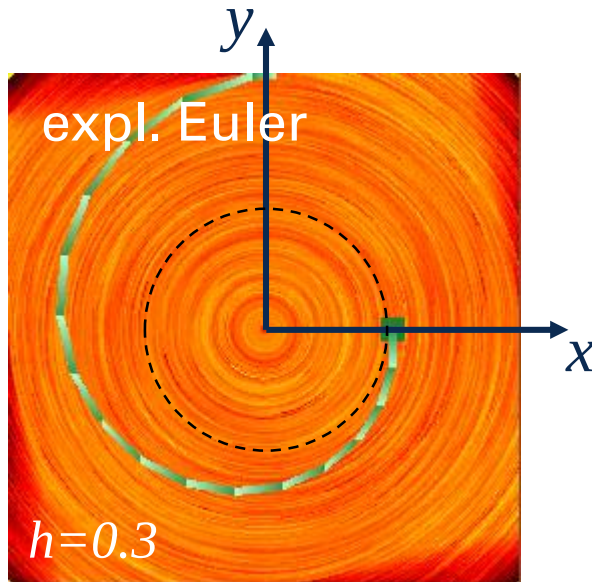
- $\vec{f}(t, \vec{y}(t))$ is a vector field
- For autonomous DE field is static and evolutions are stream-lines of the field, that can be computed approximately
- Each integration step introduces an approximation error with respect to analytic solution **accumulated error**
- Decreasing the size of the time step decreases the approximation error

Ex.: harmon. Oscillator

$$\vec{f}(\vec{y}(t)) = \begin{pmatrix} v(t) \\ -\omega^2 x(t) \end{pmatrix}$$



Numeric Integration



Careful: for large time steps as supported by integration methods of high approximation order polyline can deviate significantly from solution between integration points

Numeric Integration

Approximation Order

Approximation Error

- notation for analytic solution of integration:

$$\underline{\vec{y}}_i(s) = \vec{y}_i + \int_{t_i}^s \vec{f}(\tau, \vec{y}(\tau)) d\tau$$

- error per step h :

$$\varepsilon_i = \left\| \underbrace{\vec{y}_{i+1}}_{\text{numeric}} - \underbrace{\vec{y}_i(t_{i+1})}_{\text{analytic}} \right\|$$

- Ex.: explicit Euler

$$\vec{y}_{i+1} = \vec{y}_i + h \cdot \vec{f}(t_i, \vec{y}_i)$$

- Taylor series
around t_i :

$$\underline{\vec{y}}_i(t_i + h) = \vec{y}_i + h \cdot \vec{f}(t_i, \vec{y}_i) + O(h^2)$$

- ➡ step error is proportional to h^2

Numeric Integration

Approximation Order

- Accumulated error from integration over interval Δt :
 - Number of necessary steps: $n = \Delta t/h$
 - For a per step error proportional to h^2 , an upper bound for the accumulated error is given by:

$$\varepsilon_{\Delta t} = \|\vec{y}_n - \vec{y}(\Delta t)\| \leq \frac{(1 + Lh)^n - 1}{2L} \|\ddot{\vec{y}}\|_{[0, \Delta t]} \cdot h \leq \underbrace{\frac{e^{L\Delta t} - 1}{2L} \|\ddot{\vec{y}}\|_{[0, \Delta t]}}_{\text{constant}} \cdot h$$

Lipschitz constant of $\vec{f}$

- i.e. the accumulated error grows with $O(h)$
- For a step error of h^{k+1} the accumulated error grows with $O(h^k)$. One calls k the approximation order of the method (careful: don't mix up approximation order with the order of the DE!)
- Ex.: explicit Euler is method of approximation order 1

Numeric Integration

Verlet Method

- Method is specifically designed for physical systems of order 2 with position $\underline{x}$ and velocity $\vec{v}$.

- Derivation:

- Exploit Taylor series adding $+h$ and $-h$:

$$\underline{x}(t_i + h) = \underline{x}(t_i) + h \cdot \dot{\underline{x}}(t_i) + \frac{1}{2} h^2 \cdot \ddot{\underline{x}}(t_i) + \frac{1}{6} h^3 \cdot \ddot{\underline{x}}(t_i) + O(h^4)$$

$$\underline{x}(t_i - h) = \underline{x}(t_i) - h \cdot \dot{\underline{x}}(t_i) + \frac{1}{2} h^2 \cdot \ddot{\underline{x}}(t_i) - \frac{1}{6} h^3 \cdot \ddot{\underline{x}}(t_i) + O(h^4)$$

$$\underline{x}(t_i + h) + \underline{x}(t_i - h) = 2\underline{x}(t_i) + h^2 \cdot \ddot{\underline{x}}(t_i) + O(h^4), \text{ mit } \ddot{\underline{x}}(t_i) = \vec{a}(t_i)$$

- i.e. the 3rd order term vanished and $\underline{x}(t)$ is approximated with approximation order 3:

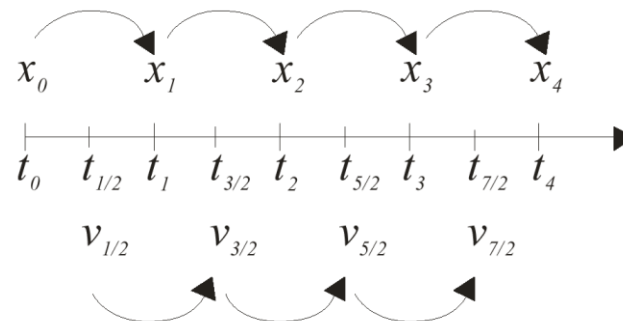
$$\underline{x}_{i+1} = 2\underline{x}_i - \underline{x}_{i-1} + h^2 \frac{\vec{F}_i}{m} + O(h^4)$$

$\vec{v}_{i+1}$ is not directly available

Numeric Integration

Verlet Method with Velocity

- estimate velocity from $\vec{v} = \frac{x_{i+1} - x_i}{h}$
- Different interpretations are possible
 - forward difference ... $\vec{v} = \vec{v}_i$
 - backward difference ... $\vec{v} = \vec{v}_{i+1}$
 - central difference ... $\vec{v} = \vec{v}_{i+\frac{1}{2}}$
- central difference can be interpreted on staggered temporal grid and have best approximation order:



Numeric Integration

Verlet Method

Plugging $\vec{v}_{i-\frac{1}{2}} = \frac{\underline{x}_i - \underline{x}_{i-1}}{h}$ in yields:

$$\underline{x}_{i+1} = 2\underline{x}_i - \underline{x}_{i-1} + h^2 \frac{\vec{F}_i}{m}$$

$$\underline{x}_{i+1} = \underline{x}_i + h \frac{\underline{x}_i - \underline{x}_{i-1}}{h} + h^2 \frac{\vec{F}_i}{m}$$

$$\underline{x}_{i+1} = \underline{x}_i + h \vec{v}_{i-\frac{1}{2}} + h^2 \frac{\vec{F}_i}{m}$$

$$\underline{x}_{i+1} = \underline{x}_i + h \left(\vec{v}_{i-\frac{1}{2}} + h \frac{\vec{F}_i}{m} \right)$$

Verlet with
velocity has
app. ord. 2

$$\vec{v}_{i+\frac{1}{2}} = \vec{v}_{i-\frac{1}{2}} + h \frac{\vec{F}_i}{m}, \quad \underline{x}_{i+1} = \underline{x}_i + h \vec{v}_{i+\frac{1}{2}}$$

Numeric Integration

Euler-Cromer Methode

also called semi-implicit

$$\begin{aligned}\vec{v}_{i+1} &= \vec{v}_i + h \frac{\vec{F}_i}{m} \\ \underline{x}_{i+1} &= \underline{x}_i + h \cdot \vec{v}_{i+1}\end{aligned}$$

- Method is of approx. order 1 but is symplectic, what means that it is energy preserving and therefore more stable than other methods of approximation order 1.

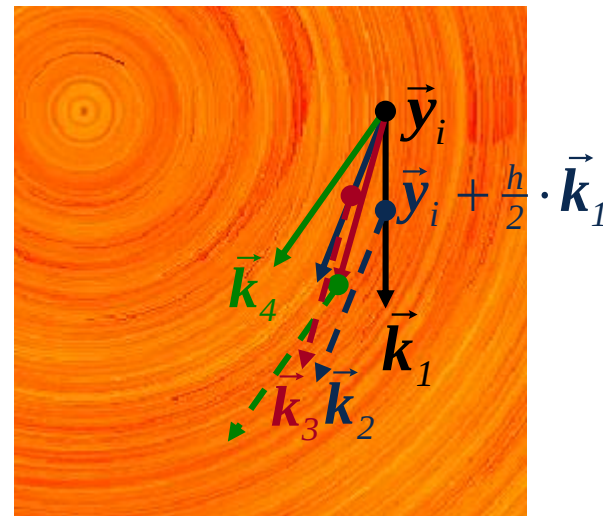
Numeric Integration

Runge-Kutta Methods

- Family of methods with methods for different approximation order
- App. ord. n builds on n function evaluations
- Very popular is RK4 (Runge Kutta 4. App.Ord.)

$$\vec{y}_{i+1} = \vec{y}_i + h \cdot \frac{1}{6} (\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4) + O(h^5)$$

$$\begin{aligned}\vec{k}_1 &= \vec{f}(t_i, \vec{y}_i) \\ \vec{k}_2 &= \vec{f}\left(t_i + \frac{h}{2}, \vec{y}_i + \frac{h}{2} \cdot \vec{k}_1\right) \\ \vec{k}_3 &= \vec{f}\left(t_i + \frac{h}{2}, \vec{y}_i + \frac{h}{2} \cdot \vec{k}_2\right) \\ \vec{k}_4 &= \vec{f}\left(t_i + h, \vec{y}_i + h \cdot \vec{k}_3\right)\end{aligned}$$



Numeric Integration

A-Stability

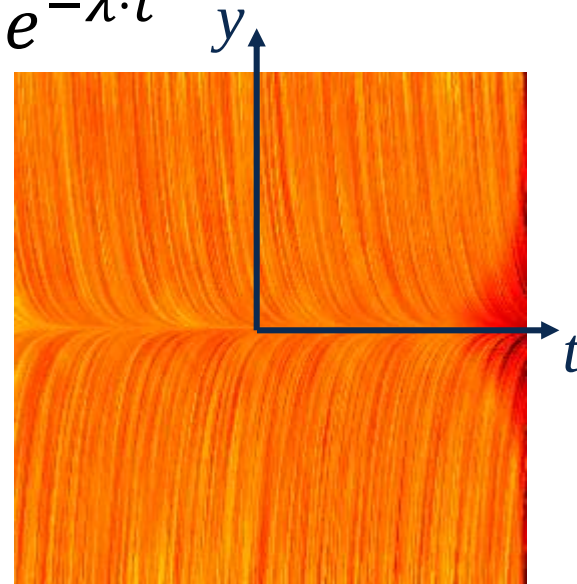
- A-Stability is analyzed on simple test case – the following 1st order DE: $\dot{y}(t) = -\lambda \cdot y(t)$

- Analytic solution: $y(t) = e^{-\lambda \cdot t}$

- Figure shows plot of

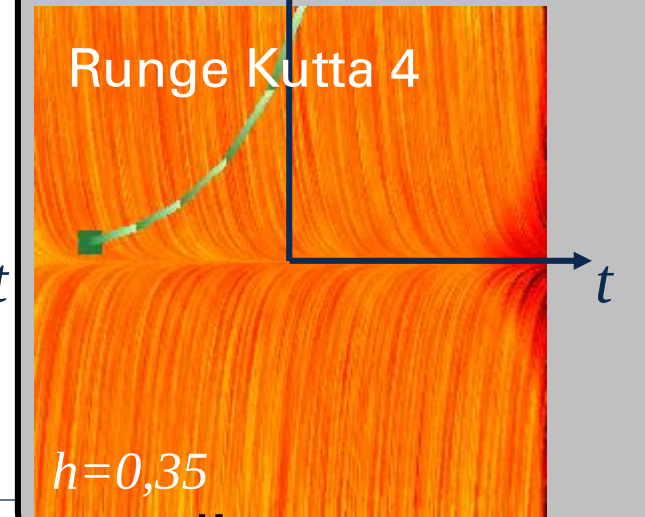
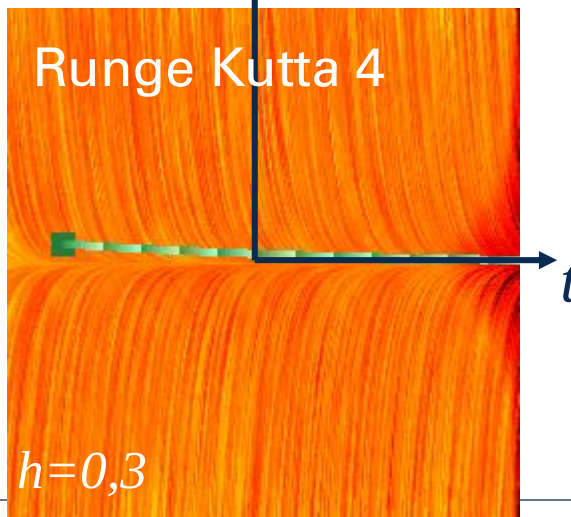
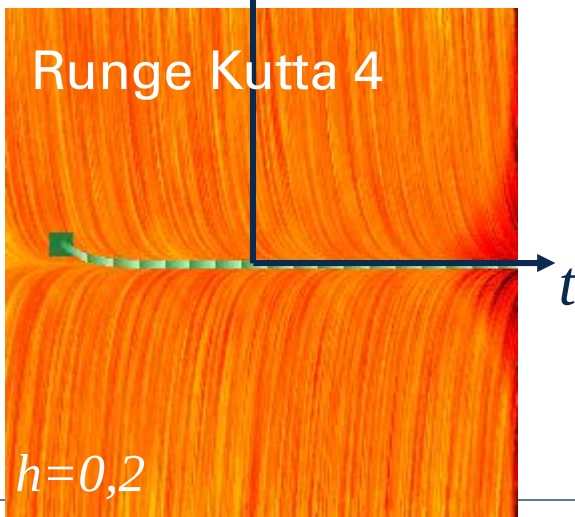
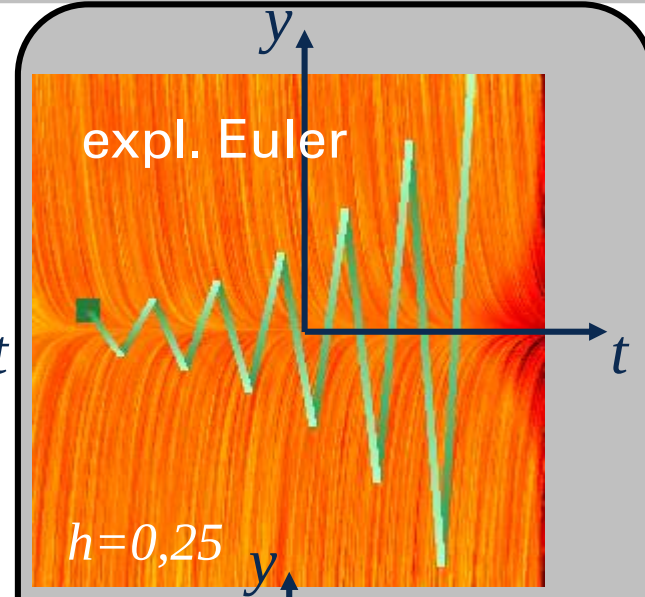
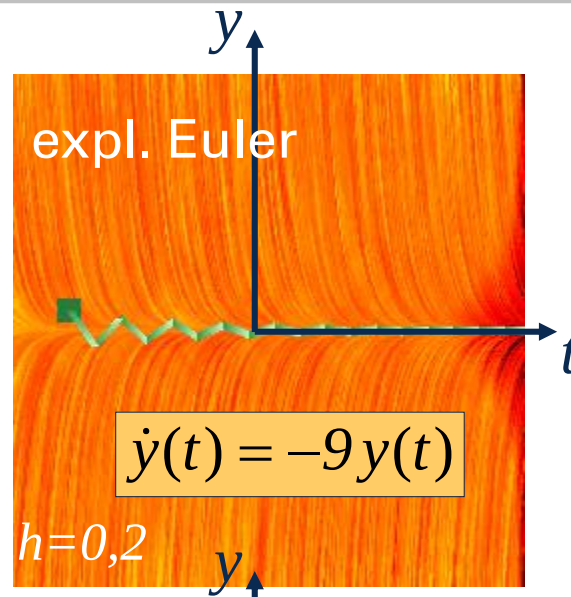
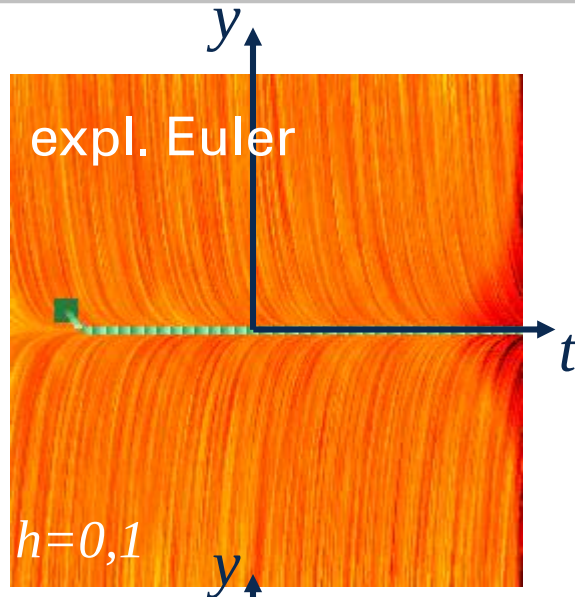
$$\frac{d}{dt} \begin{pmatrix} t \\ y(t) \end{pmatrix} = \begin{pmatrix} 1 \\ -\lambda \cdot y(t) \end{pmatrix}$$

with $\lambda=9$



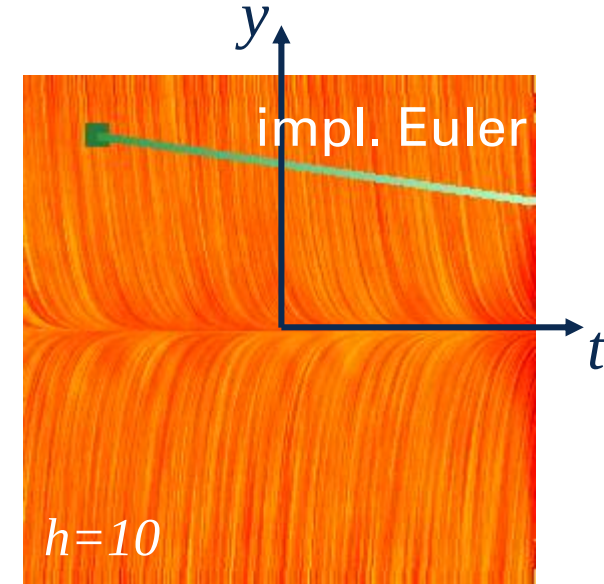
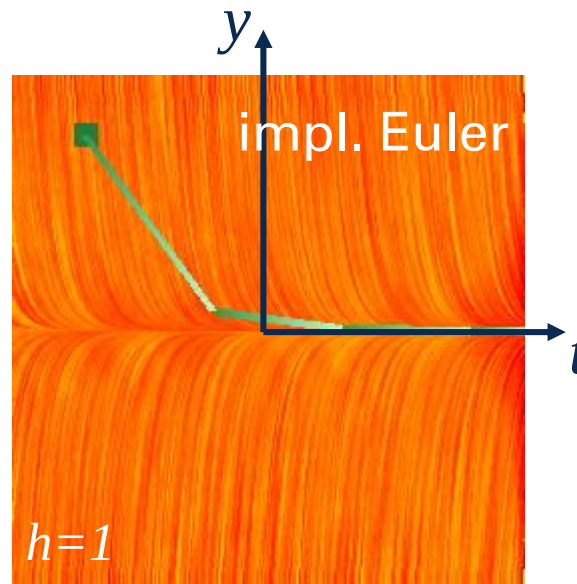
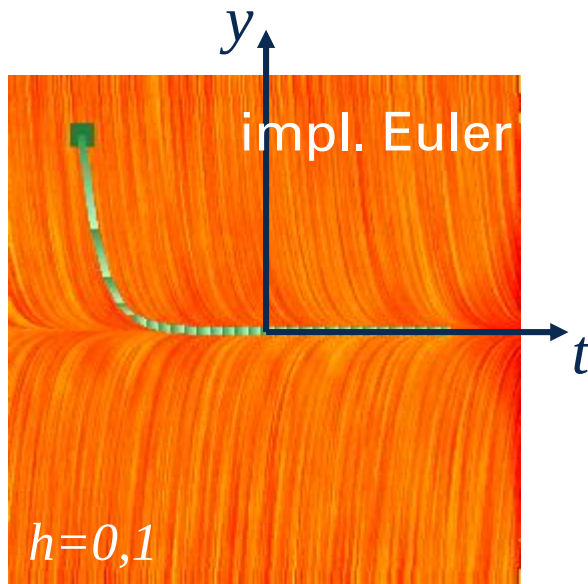
- **Definition:** a numeric integration method is called **A-stable**, iff generated approximations converge for any step width and any λ towards the t -axis.

Numeric Integration Stability



Numeric Integration

Stability – Implicit Euler



$$\dot{y}(t) = -\lambda \cdot y(t), \quad \lambda > 0$$

$$y_{i+1} = y_i + h \cdot \dot{y}(t_{i+1})$$

$$y_{i+1} = y_i - h \cdot \lambda \cdot y_{i+1}$$

$$(1 + h \cdot \lambda) \cdot y_{i+1} = y_i$$

$$y_{i+1} = y_i / (1 + h \cdot \lambda)$$

$$y_{i+1} < y_i \Leftrightarrow 1 + h \cdot \lambda > 1 \Leftrightarrow h \cdot \lambda > 0$$

Numeric Integration

Stability – Implicit Euler

- The DE $\dot{y}(t) = -\lambda \cdot y(t)$ is called **stiff** as explicit methods do not always converge.
- The implicit Euler Method is A-stable
- The general form:

$$\vec{y}_{i+1} = \vec{y}_i + h \cdot \vec{f}(t_{i+1}, \vec{y}_{i+1})$$

yields a system of typically **non-linear equations** in the components of $\vec{y}_{i+1}$

- A **Newton-Iteration** can be used to solve the system of equations
- For stiff problems this additional work **pays off** as larger steps can be taken

Numeric Integration

Midpoint methods

- simplest explicit midpoint method as modified Euler:

$$\vec{y}_{i+1} = \vec{y}_i + h\vec{f}\left(t_{i+\frac{1}{2}}, \vec{y}_i + \frac{h}{2}\vec{f}(t_i, \vec{y}_i)\right)$$

- simplest implicit midpoint method:

$$\vec{y}_{i+1} = \vec{y}_i + h\vec{f}\left(t_{i+\frac{1}{2}}, \frac{1}{2}(\vec{y}_i + \vec{y}_{i+1})\right)$$

- Both methods are of approximation order 2
- implicit midpoint is also symplectic

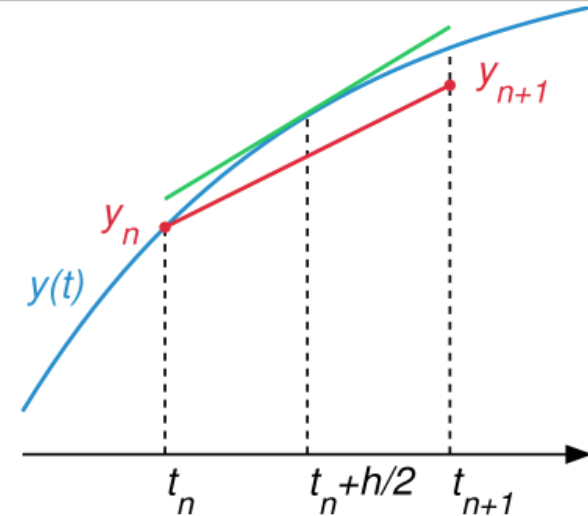


Illustration of the midpoint method assuming that $\vec{y}_n$ equals the exact value $\vec{y}(t_n)$. The midpoint method computes $\vec{y}_{n+1}$ so that the red chord is approximately parallel to the tangent line at the midpoint (the green line).

Numeric Integration

Adaptation of Stepwidth

- let **step** be a k^{th} app.ord. method
- estimate step error ε by comparing to two steps with half stepwidth:

$$\vec{y}_{i+1} = \mathbf{step}(h, t_i, \vec{y}_i) = \underline{\vec{y}}_i(t_{i+1}) + ch^{k+1} + O(h^{k+2})$$

$$\vec{y}_{i+1}^* = \mathbf{step}\left(\frac{h}{2}, t_{i+\frac{1}{2}}, \mathbf{step}\left(\frac{h}{2}, t_i, \vec{y}_i\right)\right) \approx \underline{\vec{y}}_i(t_{i+1}) + 2c\left(\frac{h}{2}\right)^{k+1} + O(h^{k+2})$$

➔ $\varepsilon = \left\| \vec{y}_{i+1}^* - \vec{y}_{i+1} \right\|_{\infty} = g(c)h^{k+1} + O(h^{k+2})$

- if ε is larger than toleranz τ discard step and re-estimate h , otherwise estimate h for next step:

$$h_{\text{new}} = h \cdot \sqrt[k+1]{\rho \cdot \frac{\tau}{\varepsilon}}, \text{ with safety factor } \rho < 1$$

Numeric Integration

Adaptation of Stepwidth

- Derivation of stepwidth update:

$$\varepsilon(h) \approx g(c)h^{k+1}$$

$$\Rightarrow g(c) \approx \frac{\varepsilon(h)}{h^{k+1}}$$

$$\varepsilon(h_{\text{new}}) \approx g(c)h_{\text{new}}^{k+1} < \tau$$

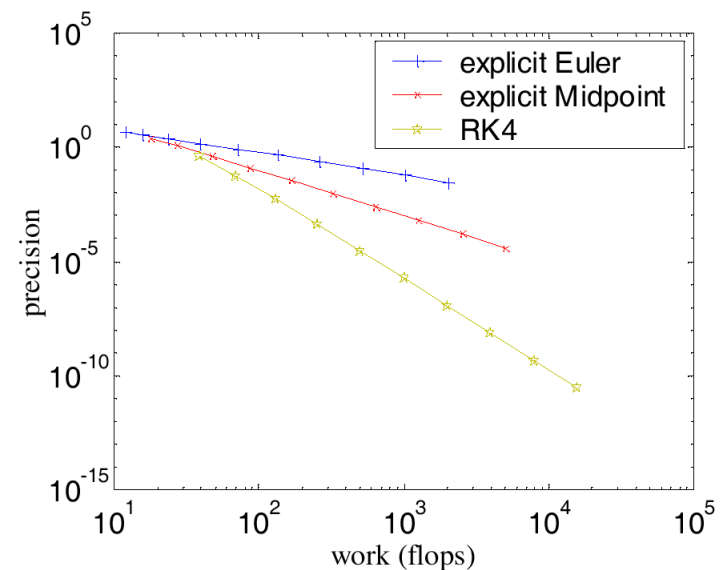
$$\varepsilon(h) \frac{h_{\text{new}}^{k+1}}{h^{k+1}} < \tau$$

$$h_{\text{new}}^{k+1} < h^{k+1} \frac{\tau}{\varepsilon(h)}$$

$$h_{\text{new}} < \sqrt[k+1]{h^{k+1} \frac{\tau}{\varepsilon(h)}} = h \cdot \sqrt[k+1]{\frac{\tau}{\varepsilon(h)}}$$

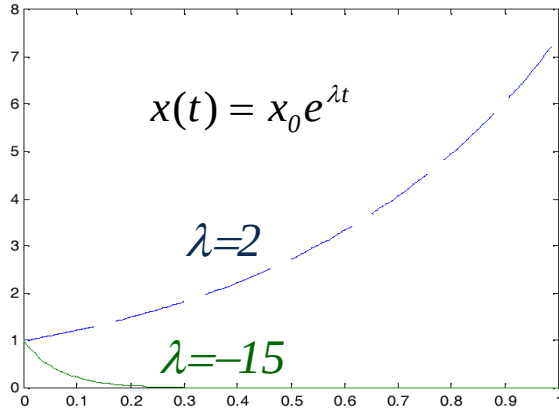
Numeric Integration Efficiency Analysis

- Hauth, M., Etzmuß, O., & Straßer, W. (2003). Analysis of numerical methods for the simulation of deformable models. *The Visual Computer*, 19(7-8), 581-600.
- Compare methods through the number of floating point operations necessary to achieve a given precision (error bound)
- Fraction of precision over number of necessary operations (slope in diagrams) is measure of efficiency of a method

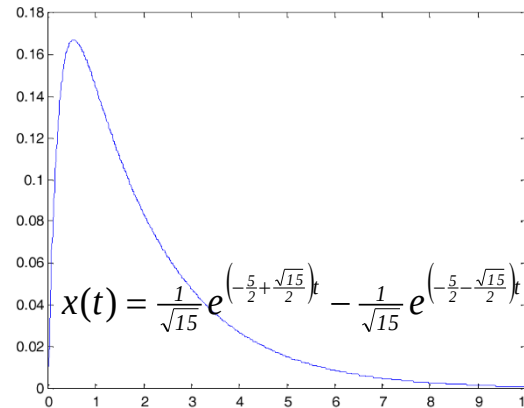


Numeric Integration Efficiency Analysis

- Hauth et al. use the stiff DE of A-stability analysis with two parameter settings and the damped harmonic oscillator with and without gravity
- For all DEs the **analytic solution** is known and can be used to compute precision of methods exactly

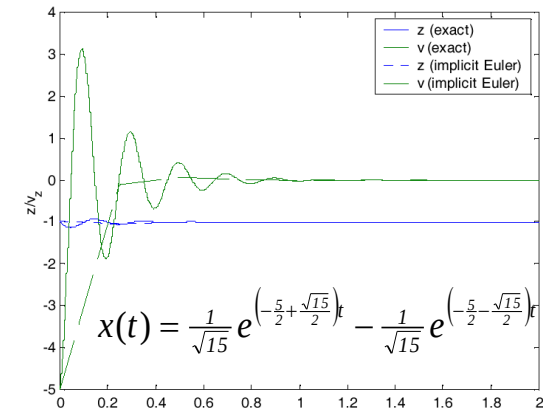


$$x(0) = 1 \quad \dot{x} = \lambda x$$



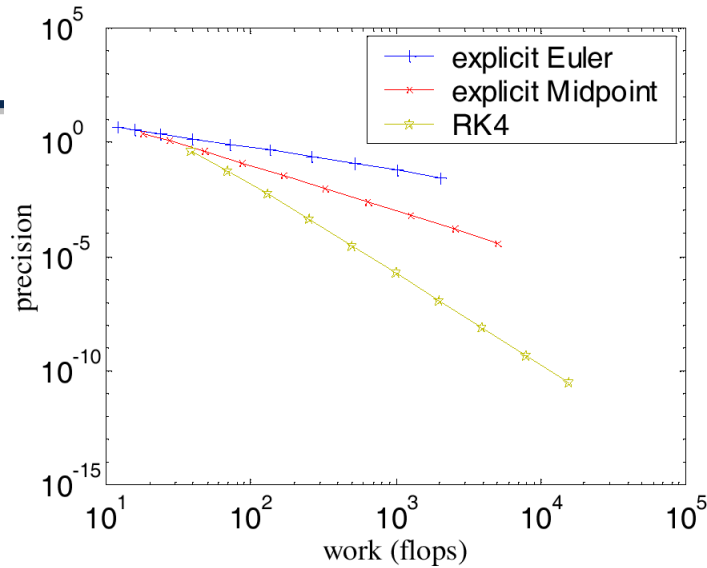
$$x(0) = 0 \quad \ddot{x} = \frac{\lambda}{2} x + \lambda \dot{x}$$

$$\dot{x}(0) = 1 \quad \lambda = -5$$

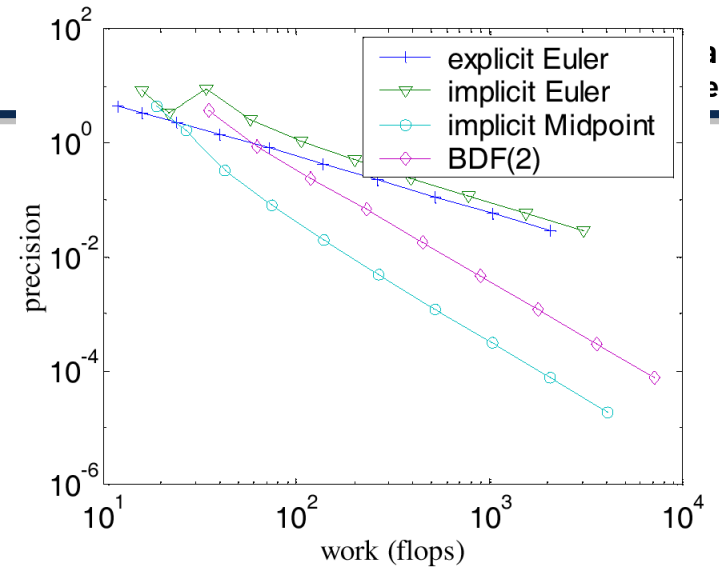


$$x(0) = 0, \dot{x}(0) = -5 \quad \ddot{x} = \frac{k}{m}(l_0 - x) + \frac{d}{m}\dot{x} + g$$

$$k = 1000, l = -1, d = 10, g = -10, m = 1$$

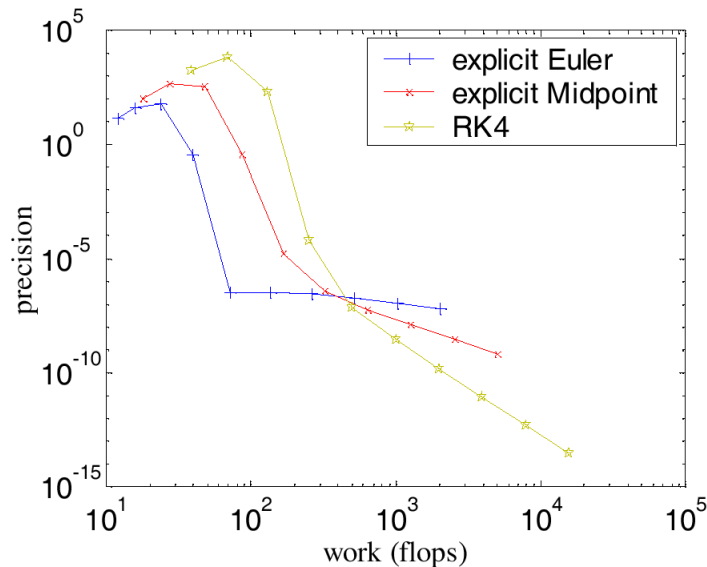


$$\lambda = 2$$

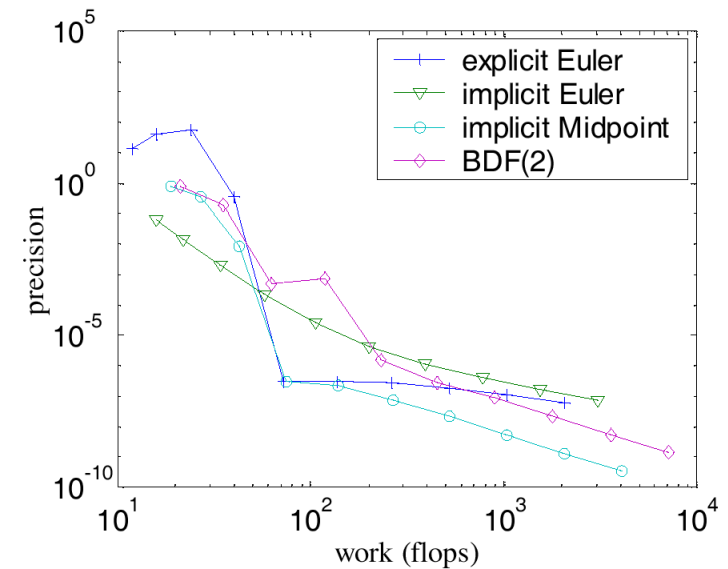


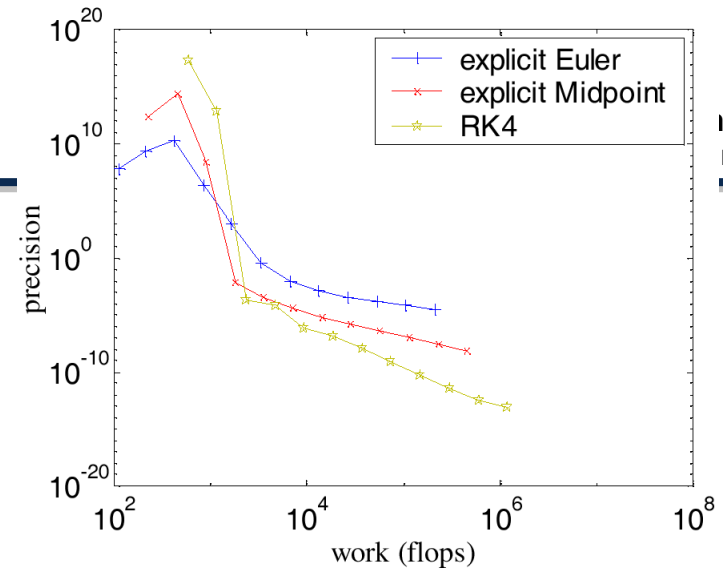
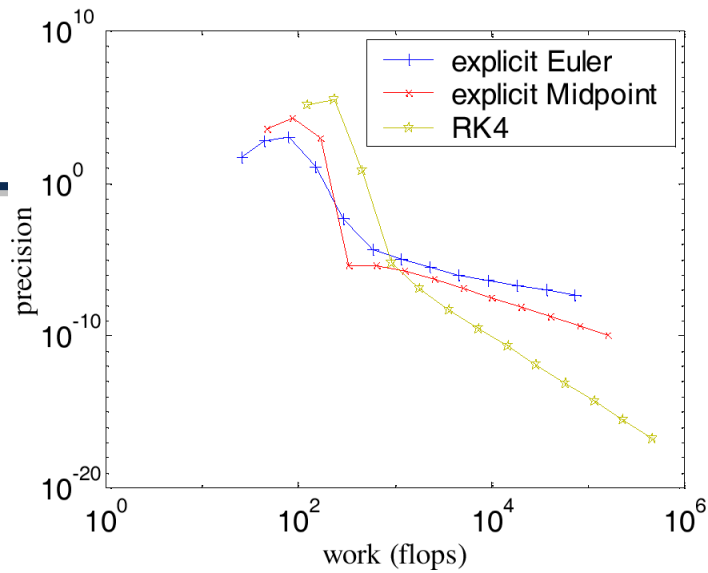
aphik
erung

$$x(0) = 1 \quad \dot{x} = \lambda x \quad \lambda = 2$$

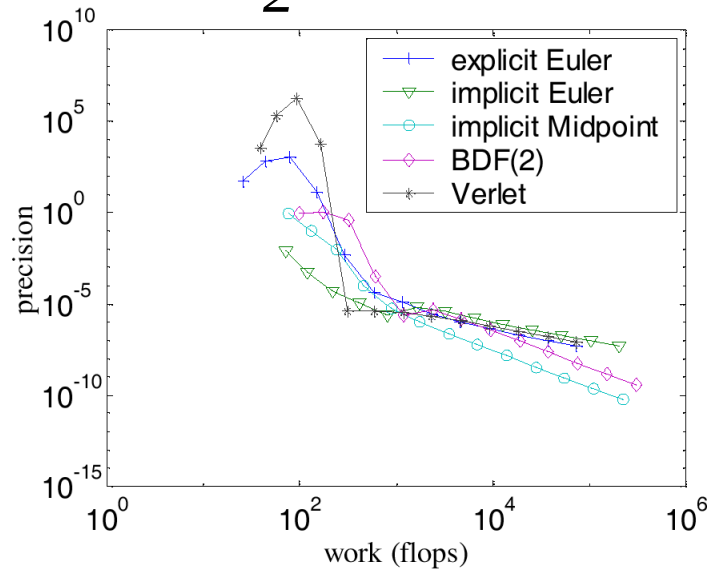


$$\lambda = -15$$

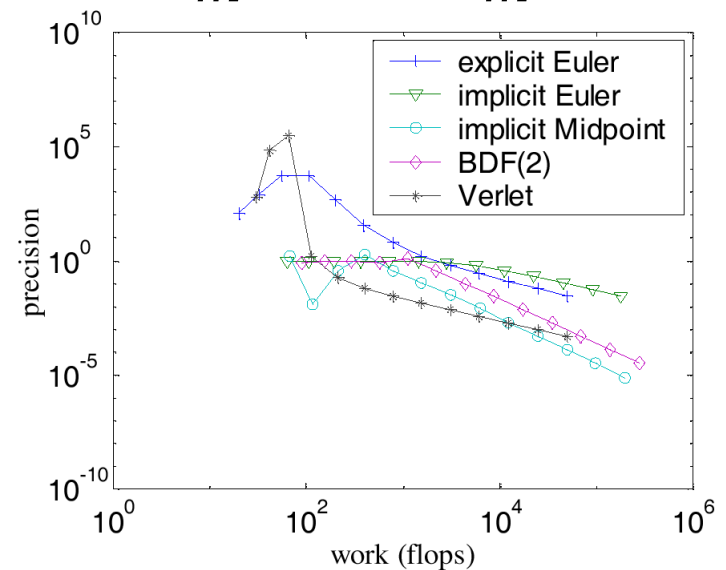




$$\ddot{x} = \frac{\lambda}{2} x + \lambda \dot{x} \quad \lambda = -5$$



$$\ddot{x} = \frac{k}{m} (l_0 - x) + \frac{d}{m} \dot{x} + g$$



Summary

- an initial value problem is composed of a DE and an initial value
- ordinary DEs only depend on single variable
- order k of a DE is highest derivative
- order k DE can be transformed to linear system of order 1 DEs.
- numeric integration methods can be characterized by their approximation order and whether they are stable
- implicit methods are stable but lead to system of non-linear equations to be solved
- step width can be adapted
- most efficient scheme depends on application

Physics Engines in JavaScript

3D

- ammo.js: <https://github.com/kripken/ammo.js>
- cannon.js: <https://schteppe.github.io/cannon.js>
- oimo.js: <https://github.com/lo-th/Oimo.js>

2D

- verlet-js: <https://github.com/subprotocol/verlet-js>
- matter-js: <https://brm.io/matter-js>
- overview page:
<https://www.tapirgames.com/blog/open-source-physics-engines>