Numerical solution of ODE

General formulation

Consider relation given a system of **first order** differential equation

Mechanical systems are often expressed as

- *single equation of second order in* p
- *system of first order in* $u = (p, v)$

In general, we can write

$$u'(t) = \mathcal{F}(u(t), t)$$

If $\mathcal{F}$ is an affine function in u

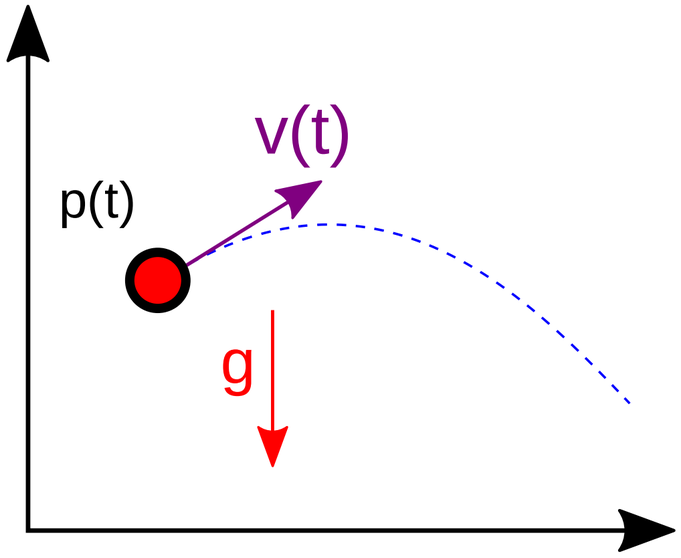
$$u'(t) = \mathrm{A}(t)\, u(t) + b(t)$$

When A is constant through time

$$u'(t) = \mathrm{A}\, u(t) + b(t)$$

Example: Free fall under gravity

- Force $F(t) = m\,g$
- Second order differential equation: $p''(t) = g$
- First order system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} v(t) \\ g \end{pmatrix}}_{\mathcal{F}(u,t)}$
- Linear system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}}_{\mathrm{A}} \underbrace{\begin{pmatrix} p \\ v \end{pmatrix}(t)}_{u(t)} + \underbrace{\begin{pmatrix} 0 \\ g \end{pmatrix}}_{b(t)}$

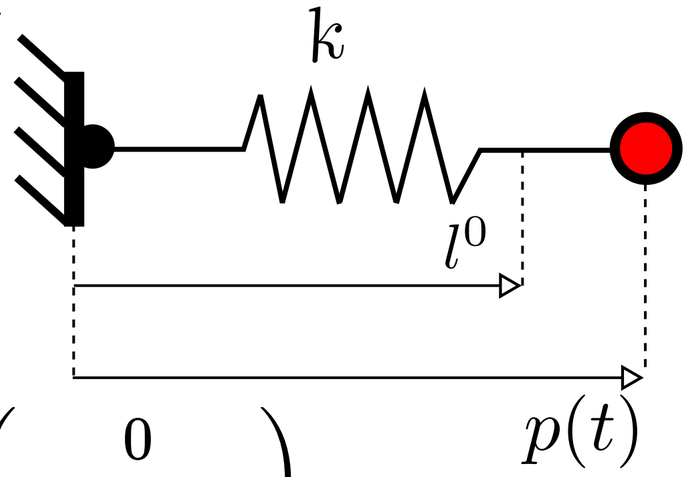


- Exact solution known: $p(t) = \frac{1}{2} g\,t^2 + v_0\,t + p_0$

Note: variables are vectors - matrix can be expressed by block, or per components.

Example: 1D spring (/Harmonic oscillator)

- Force $F(t) = -k\,(p(t) - l^0)$, k spring stiffness, l^0 rest length
- Second order differential equation: $m\,p''(t) + k\,p(t) = k\,l^0$



- First order system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} v(t) \\ -k/m\,(p(t) - l^0) \end{pmatrix}}_{\mathcal{F}(u,t)}$

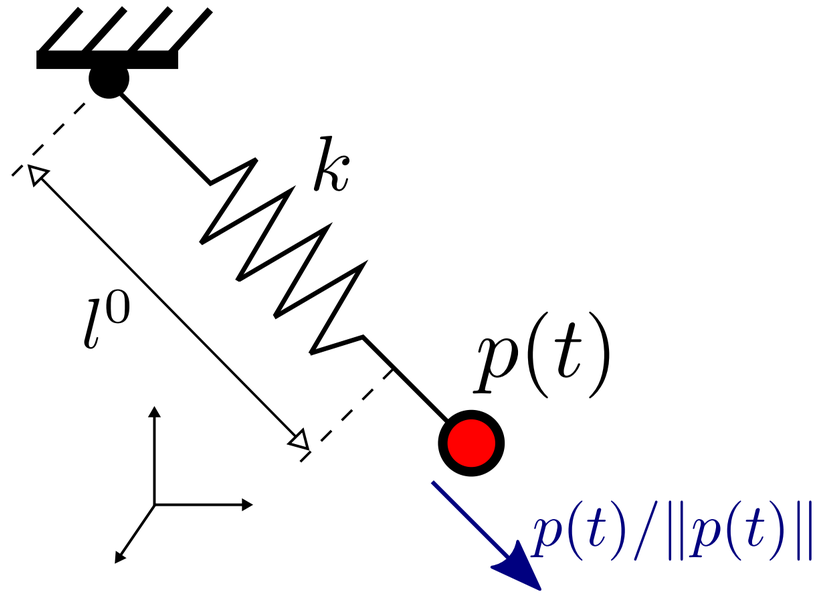
- Linear system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} 0 & 1 \\ -k/m & 0 \end{pmatrix}}_{\mathrm{A}} \underbrace{\begin{pmatrix} p \\ v \end{pmatrix}(t)}_{u(t)} + \underbrace{\begin{pmatrix} 0 \\ k/m\,l^0 \end{pmatrix}}_{b}$

- Exact solution known: $p(t) = A\,\sin(\omega\,t + \varphi) + l^0$

$$\omega = \sqrt{k/m},\, A^2 = (p^0 - l^0)^2 + \left(\frac{v_0}{\omega}\right)^2,\, \tan(\varphi) = \frac{p^0 - l^0}{v^0/\omega}$$

Example: 3D mass spring

- Force $F(t) = m\,g - k\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|}$, k spring stiffness, l^0 rest length
- Second order differential equation: $m\,p''(t) = mg - k\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|}$
- First order system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} v(t) \\ g - k/m\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|} \end{pmatrix}}_{\mathcal{F}(u,t)}$
- Not linear
- No simple explicit solution



Existence of solution

Solving the physical system = Solving the **Initial Value Problem** (IVP)

- Find the solution of $u'(t) = \mathcal{F}(u, t), t \geq 0$
- With initial condition $u(t = 0) = u_0$

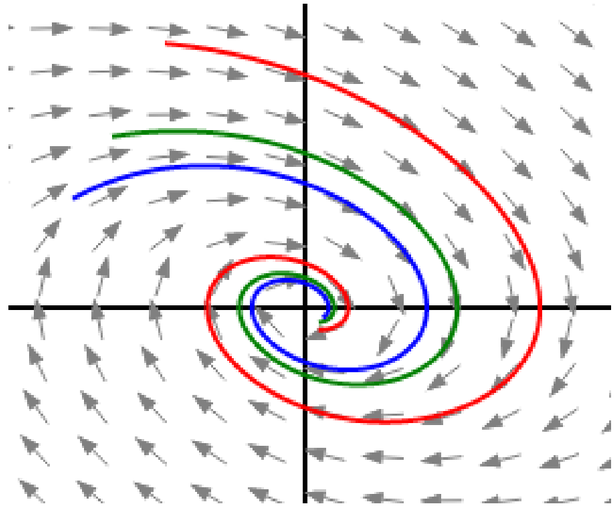
Cauchy-Lipschitz (/Picard-Lindelof) theorem states that

If $\mathcal{F}$ is Lipschitz with respect to u and continuous with respect to t

Then there exists a unique solution $u(t)$.

The solution is called the **integral curve** of the IVP.

- $\mathcal{F}$ can be seen as a vector field
  (Vector field in 6D for $p(t) \in \mathbb{R}^3, v(t) \in \mathbb{R}^3$)
- Solution is a path along this vector field passing by u_0



Solving the ODE exactly

Equation is linear

Homogeneous, constant coefficients

$u'(t) = \mathrm{A}u(t), u(0) = u_0$

$\Rightarrow u(t) = u_0 \, \exp(\mathrm{A}\, t)$

Non-Homogeneous, constant coefficients

$u'(t) = \mathrm{A}u(t) + b(t), u(0) = u_0$

$$\Rightarrow u(t) = u_0 \, \exp(\mathrm{A}\, t) + \exp\left(\mathrm{A}\, t\right) \int_{t'=0}^{t} b(t') \exp\left(-\mathrm{A}\, t'\right) \, \mathrm{d}t'$$

Homogeneous, variable coefficients

$u'(t) = \mathrm{A}(t)\, u(t), u(0) = u_0$

$\Rightarrow$ No closed-form solution in the general case

- *Rem.* Unfortunately, in general, $u(t)$ is **not** $u_0 \, \exp\left(\int_0^t \mathrm{A}(t') \, \mathrm{d}t'\right)$

Equation is non linear

No general method

$\Rightarrow$ Numerical approaches are required most of the time

Numerical solution

1st order Explicit Euler

Naive numerical scheme: Approximation of the derivative

$$\frac{u^{k+1} - u^k}{h} = \mathcal{F}(u^k, t^k)$$

$$\Rightarrow u^{k+1} = u^k + h\,\mathcal{F}(u^k, t^k)$$

In the linear case

$$u^{k+1} = (\mathrm{I} + h\,\mathrm{A})\; u^k + h\,b^k$$

Pro : very easy to implement

Is u^k a good approximation of the true solution $\tilde{u}(t^k)$?

For a single particle with p, v variables:

$$\begin{cases} v^{k+1} = v^k + h\,F(u^k, t^k)/m \\ p^{k+1} = p^k + h\,v^k \end{cases}$$

Explicit Euler - study case

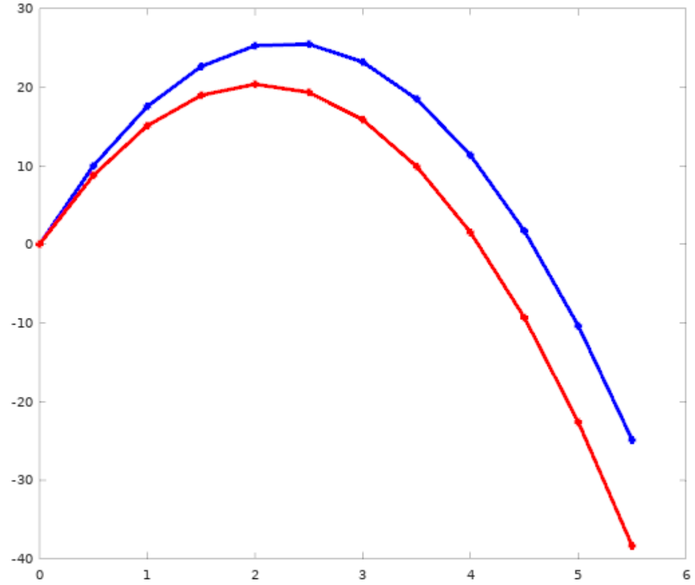
Free fall under gravity

- True solution $\tilde{u}(k\,h) = p_0 + (k\,h)v_0 + \frac{(k\,h)^2}{2}\,g$

- Numerical scheme: $\begin{pmatrix} p \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p \\ v \end{pmatrix}^{k} + \begin{pmatrix} 0 \\ h\,g \end{pmatrix}$

- Numerical solution: $p^k = p_0 + k\,h\,v_0 + \frac{k(k-1)}{2}\,h^2\,g$

$\Rightarrow$ **Not exact** : Error $e^k = |u^k - \tilde{u}(k\,h)| = \frac{g}{2} k\,h^2$



red: true solution

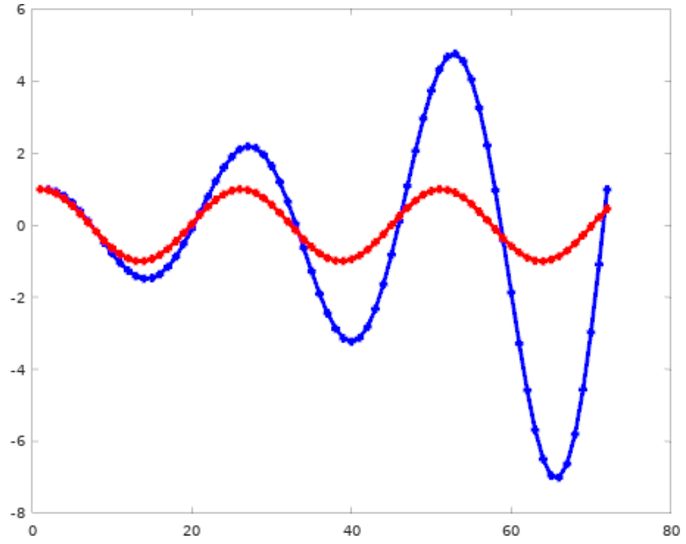
blue: numerical solution

Explicit Euler - study case

1D spring

- True solution: permanent oscillation

- Numerical scheme: $\begin{pmatrix} p \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & h \\ -K/m\,h & 1 \end{pmatrix} \begin{pmatrix} p \\ v \end{pmatrix}^{k} + \begin{pmatrix} 0 \\ K/m\,h\,l^0 \end{pmatrix}$



red: true solution

red: numerical solution

- Numerical solution diverge to ∞
- Worse than bad accuracy for Graphics

Explicit Euler - study case

1D spring: Analysis of the system energy

- Energy $E = \frac{1}{2}mv^2 + \frac{K}{2}(p - l^0)^2$

$$E^{k+1} = \frac{1}{2}m\left(-\frac{K}{m}\Delta t\left(p^k - l^0\right) + v^k\right)^2 + \frac{1}{2}K\left(p^k + \Delta t\, v^k - l^0\right)^2$$

$$E^{k+1} = \underbrace{\frac{1}{2}m\left(v^k\right)^2 + \frac{1}{2}K(p^k - l^0)^2}_{E^k} + \frac{1}{2}\left[\underbrace{\frac{K^2}{m}(\Delta t)^2(p^k - l^0)^2}_{>0} - \underbrace{2\,K\Delta t\left(p^k - l^0\right)\,v^k + 2K\Delta t\left(p^k - l^0\right)v^k}_{=0} + \underbrace{K(\Delta t)^2\left(v^k\right)^2}_{>0}\right]$$

$$E^{k+1} = E^k + \epsilon\,(\Delta t)^2, \epsilon > 0$$

$\Rightarrow$ gain of energy

$\Rightarrow$ divergence

Accuracy of a numerical method - general definition

- Define the **local truncation error** τ:

  Error accumulated during one step, assuming perfect knowledge of the true solution

  $\tau_k = \|\tilde{u}(t^k) - u^k\|$, assuming $u^{k-1} = \tilde{u}(t^{k-1})$

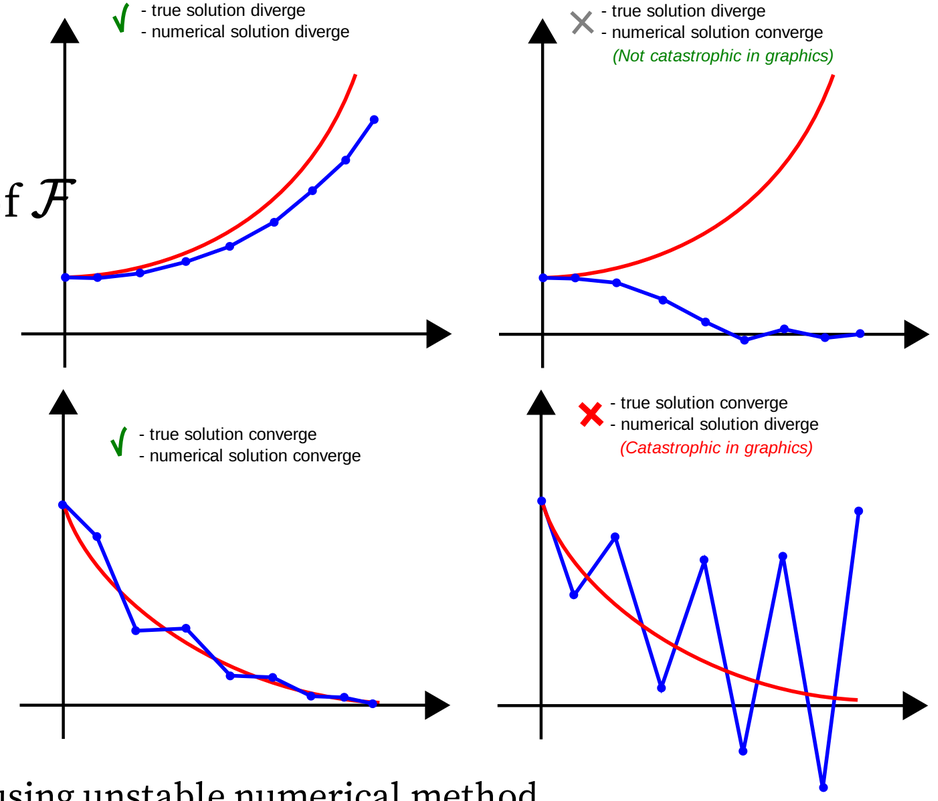
- A numerical scheme is said to be accurate of order k, if its local truncation error is in $\mathcal{O}(h^{k+1})$.

Explicit Euler is of order 1*, at every step we add an error in* $\mathcal{O}(h^2)$.

Stability of a numerical method - general definition

- Classical stability of a method studied on $u'(t) = \lambda\, u(t), \lambda \in \mathbb{C}$.
  - The true solution $\tilde{u}(t) = \exp(\lambda\, t)$ converge if $\mathbb{R}_e(\lambda) \leq 0$.
  - For linear system, λ refers to eigenvalues of A.
  - For non linear system, λ refers to eigenvalues of the Jacobian of $\mathcal{F}$

- A numerical method is *unconditionnaly stable* if $\mathbb{R}_e(\lambda) \leq 0$
  $\Rightarrow$ stable discrete solution.
- Otherwise, it is conditionnally stable/unstable.
- Region of stability:
  Set of conditions on λ such that the discrete solution doesn't diverge.



Rem.

- A numerical solution can diverge even when the true ODE solution converge when using unstable numerical method.
- Converseley, a numerical solution can converge even when the true ODE solution diverge when using stable numerical method.
- Stability $!=$ Accuracy.

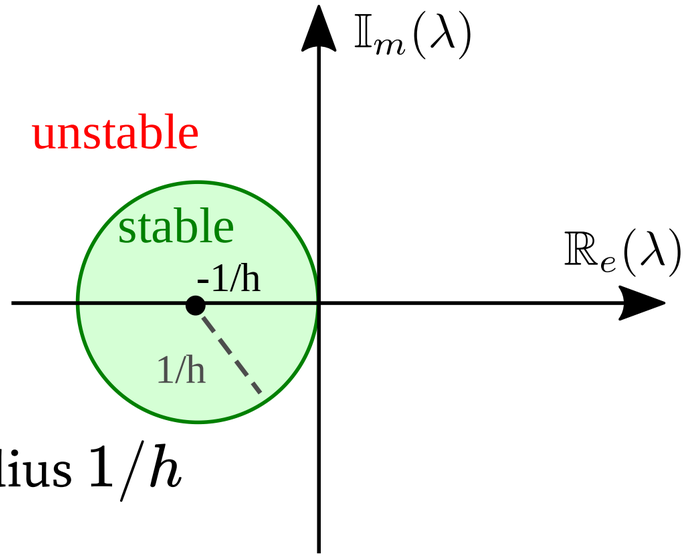
Stability analysis of explicit Euler

$u'(t) = \lambda\, u(t)$

$\Rightarrow u^{k+1} = u^k + \lambda\, u^k$ using explicit Euler

$\Rightarrow u^{k+1} = (1 + \lambda\, h)\, u^k$

$\Rightarrow$ Stable if $|1 + \lambda\, h| \leq 1$ (conditionnal stability)



$|1/h + \lambda| \leq 1/h$: Interior of a disc centered on $(-1/h, 0)$ with radius $1/h$

Rem. For 1D elastic spring

$\lambda = \pm i\sqrt{K/m}$

$\Rightarrow |1 + i\sqrt{K/m}h| = \sqrt{1 + K/m\, h^2} > 1$

$\Rightarrow$ Explicit euler is always unstable on the elastic spring problem.

Other approach: Implicit method

Other approach: Implicit Euler

Explicit Euler

$u^{k+1} = u^k + h\,\mathcal{F}(u^k, t^k)$

u^k is known to compute u^{k+1}

- In the linear case

$u^{k+1} = (\mathrm{I} + h\,\mathrm{A})\,u^k + h\,b(t^k)$

Implicit Euler

$u^{k+1} = u^k + h\,\mathcal{F}(u^{k+1}, t^{k+1})$

u^{k+1} appears in RHS

$\Rightarrow$ Need to solve $u^{k+1} - h\,\mathcal{F}(u^{k+1}, t^{k+1}) = u^k$

- In the linear case : solve a linear system

$u^{k+1} = u^k + h\,\left(\mathrm{A}u^{k+1} + b(t^{k+1})\right)$

$\Rightarrow u^{k+1} = (\mathrm{I} - h\,\mathrm{A})^{-1}\,\left(u^k + h\,b(t^{k+1})\right)$

Implicit Euler - study case

Free fall under gravity

- True solution $\tilde{u}(k\,\Delta t) = p_0 + (k\,\Delta t)v_0 + \frac{(k\,\Delta t)^2}{2}\,g$

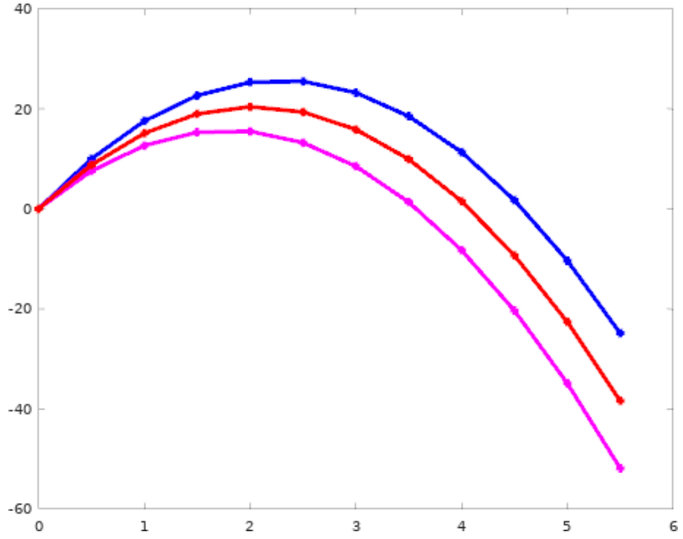
- Numerical scheme: $\begin{pmatrix} p \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & -h \\ 0 & 1 \end{pmatrix}^{-1} \left(\begin{pmatrix} p \\ v \end{pmatrix}^{k} + \begin{pmatrix} 0 \\ g \end{pmatrix} \right)$

$$\Rightarrow \begin{pmatrix} p \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p \\ v \end{pmatrix}^{k} + \begin{pmatrix} h^2\,g \\ h\,g \end{pmatrix}$$

- Numerical solution: $p^k = p_0 + (k\,\Delta t)v_0 + \frac{k(k+1)}{2}\,(\Delta t)^2\,g$

$\Rightarrow$ Not exact : Error $e^k = |u^k - \tilde{u}(k\,\Delta t)| = \frac{k}{2}(\Delta t)^2 g$

Same error magnitude than explicit Euler



- *red: True solution*
- *magenta: Implicit Euler*
- *blue: Explicit Euler*

Implicit Euler - study case

1D spring

- True solution: permanent oscillations

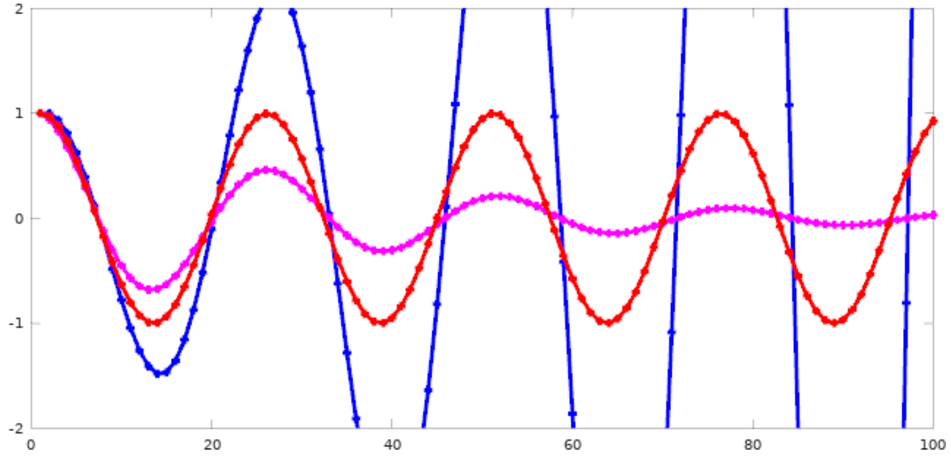
- Numerical scheme: $\begin{pmatrix} p \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & -h \\ K/m\,h & 1 \end{pmatrix}^{-1} \left(\begin{pmatrix} p \\ v \end{pmatrix}^{k} + \begin{pmatrix} 0 \\ K/m\,h\,l^0 \end{pmatrix} \right)$

$$\Rightarrow \begin{pmatrix} p \\ v \end{pmatrix}^{k+1} = \frac{1}{1+h^2\,K/m} \left(\begin{pmatrix} 1 & h \\ -K/m\,h & 1 \end{pmatrix} \begin{pmatrix} p \\ v \end{pmatrix}^{k} + \begin{pmatrix} K/m\,h^2\,l^0 \\ K/m\,h\,l^0 \end{pmatrix} \right)$$

Eigenvalues of $(\mathrm{I} - \mathrm{A}h)^{-1}$ are $\frac{1 \pm i\sqrt{\frac{K}{m}}h}{1+\frac{K}{m}h^2}$

$$\Rightarrow \left| \frac{1 \pm i\sqrt{\frac{K}{m}}h}{1+\frac{K}{m}h^2} \right| = \frac{1}{\sqrt{1+\frac{K}{m}h^2}} < 1 \Rightarrow \text{always converge}$$

Even if the true solution oscillates

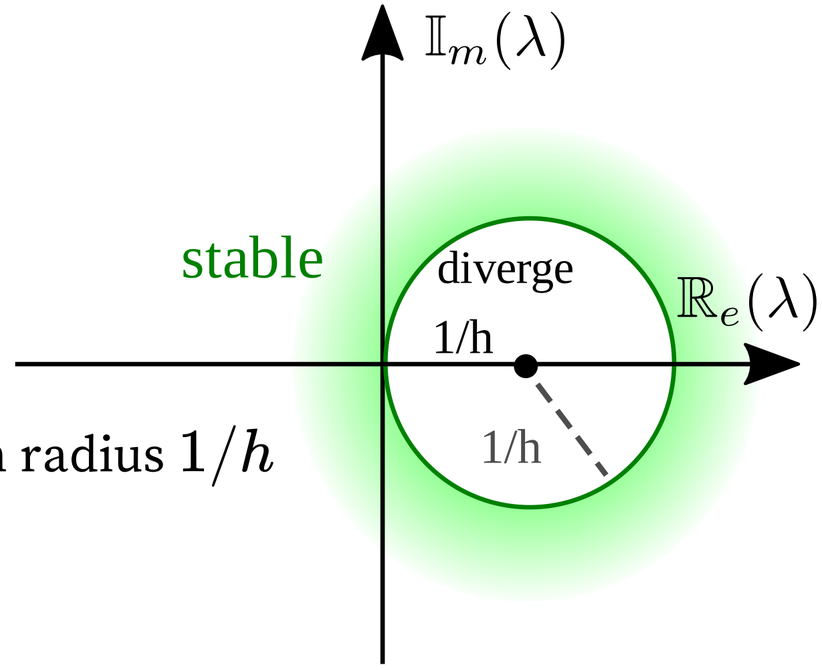


Stability analysis of Implicit Euler

What are the general conditions for which $u'(t) = \lambda\, u(t)$ converge using Implicit Euler ?

$u^{k+1} = u^k + h\,\lambda\, u^{k+1}$
$\Rightarrow (1 - h\,\lambda)\, u^{k+1} = u^k$
$\Rightarrow u^{k+1} = \frac{1}{1-h\,\lambda}\, u^k$

Stability condition: $\left|\frac{1}{1-h\lambda}\right| \leq 1 \Rightarrow |1 - h\lambda| \geq 1$



$|1/h - \lambda| \geq 1/h$: exterior of a disc centered on $(1/h, 0)$ with radius $1/h$
$\Rightarrow$ Fully enclose $\mathbb{R}_e(\lambda) \leq 0$
$\Rightarrow$ Implicit euler is unconditionally stable
Rem. May converges even when the true solution does not.

Comparison Explicit Euler / Implicit Euler

Explicit Euler

- $u^{k+1} = u^k + h\,\mathcal{F}(u^k, t^k)$
- $u^{k+1} = (\mathrm{I} + h\,\mathrm{A})\,u^k + h\,b(t^k)$
- Accuracy: 1st Order
- Stability:

  Conditional $\left|1 + h\lambda\right| \geq 1$

  Solution to spring model diverges

Implicit Euler

- $u^{k+1} = u^k + h\,\mathcal{F}(u^{k+1}, t^{k+1})$
- $u^{k+1} = (\mathrm{I} - h\,\mathrm{A})^{-1}\left(u^k + h\,b(t^{k+1})\right)$
- Accuracy: 1st Order
- Stability:

  Unconditionnaly stable

  Solution to spring model converge to l^0

Higher order methods

Higher order method - RK

Higher accuracy can be achived using Taylor expansion.

ex. $u^{k+1} = u^k + h\,\mathcal{F}(u^k, t^k) + \frac{h^2}{2}\frac{\mathrm{d}\mathcal{F}}{\mathrm{d}t}(u^k, t^k)$

- Second order accurate: **2nd order explicit Euler**

- Higher order can achieve arbitrary accuracy.
- In practice: computing derivatives of $\mathcal{F}$ is complex.

$\Rightarrow$ Not often used in practice

Instead: **Runge Kutta**

- Reach higher order accuracy
- Only involve the knowledge of $\mathcal{F}$, without its derivatives
- Involves several successives evaluations of $\mathcal{F}$

Runge-Kutta methods

RK2

$$u^{k+1} = u^k + \frac{1}{2}(k_1 + k_2)$$
$$k_1 = h\,\mathcal{F}(u^k, t^k)$$
$$k_2 = h\,\mathcal{F}(u^k + k_1, t^k + h)$$

Midpoint method

$$k_1 = h\,\mathcal{F}(u^k, t^k)$$
$$k_2 = h\,\mathcal{F}(u^k + k_1/2, t^k + h/2)$$
$$u^{k+1} = u^k + h\,k_2$$

RK2 and Midpoint are 2nd order accurate, still not stable for harmonic oscillator.

RK4 (Classical Runge-Kutta)

$$k_1 = h\,\mathcal{F}(u^k, t^k)$$
$$k_2 = h\,\mathcal{F}\left(u^k + \frac{k_1}{2}, t^k + \frac{h}{2}\right)$$
$$k_3 = h\,\mathcal{F}\left(u^k + \frac{k_2}{2}, t^k + \frac{h}{2}\right)$$
$$k_4 = h\,\mathcal{F}\left(u^k + k_3, t^k + h\right)$$
$$u^{k+1} = u^k + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

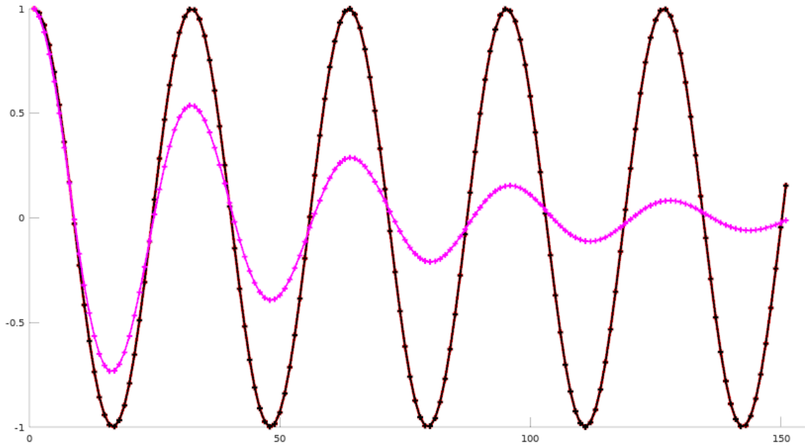
4th order, conditionally stable

RK4 - Case study

- Rk4 conditionally stable for oscillating spring

  Not permanent oscillations: Slight decrease of magnitude through time

- Large improvement of accuracy compared to implicit Euler



$h = 0.2$, error rk4 $\simeq 10^{-4}$

- *red: true solution*
- *black: rk4*
- *magenta: implicit Euler*

Stability region: $\left|1 + h\lambda + \frac{h^2}{2}\lambda + \frac{h^3}{6}\lambda^3 + \frac{1}{24}h^4\lambda^4\right| \leq 1$

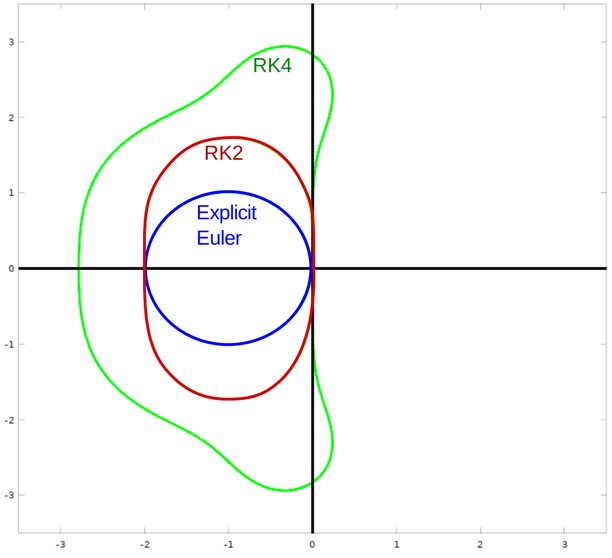
In the case of oscillating spring: $\lambda = i\sqrt{K/m} = i\omega$

Stable if $\left|1 + ih\omega - \frac{h^2}{2}\omega^2 - i\frac{h^3}{6}\omega^3 + \frac{h^4}{24}\omega^4\right| \leq 1$

$\Rightarrow 1 - \frac{h^6\omega^6}{72} + \frac{h^8\omega^8}{576} \leq 1$

$\Rightarrow h^6\,\omega^6\,(h^2\omega^2 - 8)/576 \leq 0$

$\Rightarrow h \leq \frac{2^{3/2}}{\omega} = \frac{2^{3/2}}{\sqrt{K/m}} \simeq \frac{2.8}{\sqrt{K/m}}$



Introduction to symplectic methods

Standard approaches trade-off

- Explicit methods: (+) Simple to compute, (-) limited stability
- Implicit methods: (-) Hard to compute (especially on non linear functions), (+) very stable
- Oscillatory systems are not easy to model
  (-) Numerical solution either diverge or converge.

Symplectic approach

- *Remark*: Mechanical systems have position and velocity variables
  - Derivative of position is linear w/r velocity
  - Derivative of velocity is more complex (forces - non linear)

$\Rightarrow$ *General idea*: separate treatment of velocity and position

Semi-implicit:

- Implicit scheme for position p^{k+1} (linear part)
- Explicit scheme for velocity v^{k+1} (non linear part)

$\Rightarrow$ In practice: use velocity v^{k+1} to evaluate p^{k+1}.

Pro

(+) As simple as explicit method to implement
(+) Improved stability
(+) Well adapted to oscillatory systems

Semi implicit method

Simplest semi-implicit method: **Semi-implicit Euler** / **Verlet**

General case

$$v^{k+1} = v^k + h\,\mathcal{F}_v(p^k, v^k, t^k)$$
$$p^{k+1} = p^k + h\,\mathcal{F}_p(p^k, v^{k+1}, t^k)$$

Application to classical mechanical cases

$$p'(t) = v(t), m\,v'(t) = F(p,t)$$

$$\Rightarrow \begin{cases} v^{k+1} = v^k + h\,F(p^k, t^k)/m \\ p^{k+1} = p^k + h\,v^{k+1} \end{cases}$$

(+) Trivially easy to convert explicit Euler to semi-implicit Euler

Expressed using positions only

$$p^{k+1} = p^k + h(v^k + h\,F(p^k, t^k))$$
$$p^{k+1} = p^k + h\left(\frac{p^k - p^{k-1}}{h} + h\,F(p^k, t^k)/m\right)$$
$$\Rightarrow p^{k+1} = 2\,p^k - p^{k-1} + h^2\,F(p,t)/m$$

1st order accurate (like explicit/implicit Euler) in position and speed.

Stability on oscillatory system

1D spring system: $F(p^k, v^k, t^k) = -K\,p^k$

$p^{k+1} = 2\,p^k - p^{k-1} - h^2\,K/m\,p^k$

$\Rightarrow p^{k+1} = (2 - h^2\,K/m)\,p^k - p^{k-1}$

Stable and permanent oscillation when $h < \frac{2}{\sqrt{K/m}} = \frac{2}{\omega}$

Q. How can we demonstrate it ?

Note: $h < 2/\omega$ only valid for one 1D spring. Coupled 3D springs may require smaller value of h.

Use of semi-implicit

Ex. Elastic Spring: $F(p) = -K(p - L_0)$

Explicit Euler

```
for(int k=0; k<N; ++k) {
    p = p + h*v;
    v = v + h * F/m;
}
```

(-) Always diverges for elastic problem

Semi-Implicit Euler

```
for(int k=0; k<N; ++k) {
    v = v + h * F/m;
    p = p + h*v;
}
```

(+) Permanent oscillation for sufficiently small h.

$\Rightarrow$ Extremly simple change !
Big gain in stability

Higher order symplectic

2nd order Velocity Verlet

$$\begin{cases} v^{k+1/2} & = v^k + \frac{h}{2} F(p^k, v^k, t^k)/m \\ p^{k+1} & = p^k + h\, v^{k+1/2} \\ v^{k+1} & = v^{k+1/2} + \frac{h}{2} F(p^{k+1}, v^{k+1/2}, t^{k+1})/m \end{cases}$$

- Remains stable with permanent oscillation for $h \leq \frac{2}{\sqrt{K/m}}$

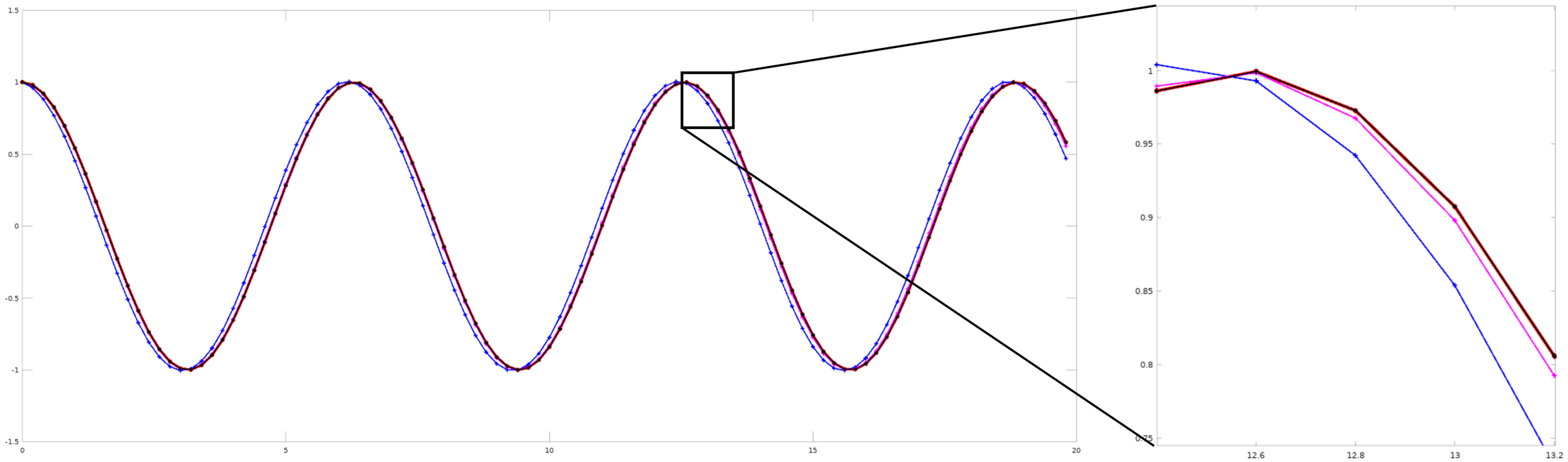
- Still simple to implement

Example for 1D elastic spring

```
for(int k=0; k<N; ++k) {
    v = v + h/2 * (-K/m*p);
    p = p + h*v;
    v = v + h/2 * (-K/m*p);
}
```

Comparison between approaches

Small $h = 0.2/\omega$

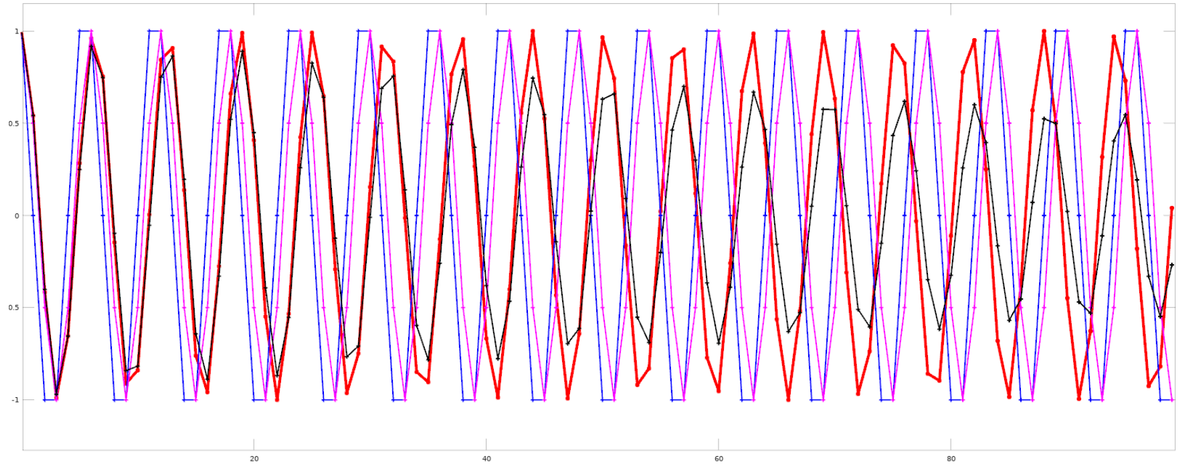


True solution , *Semi-implicit Euler* , *Velocity Verlet* , *Runge-Kutta RK4*

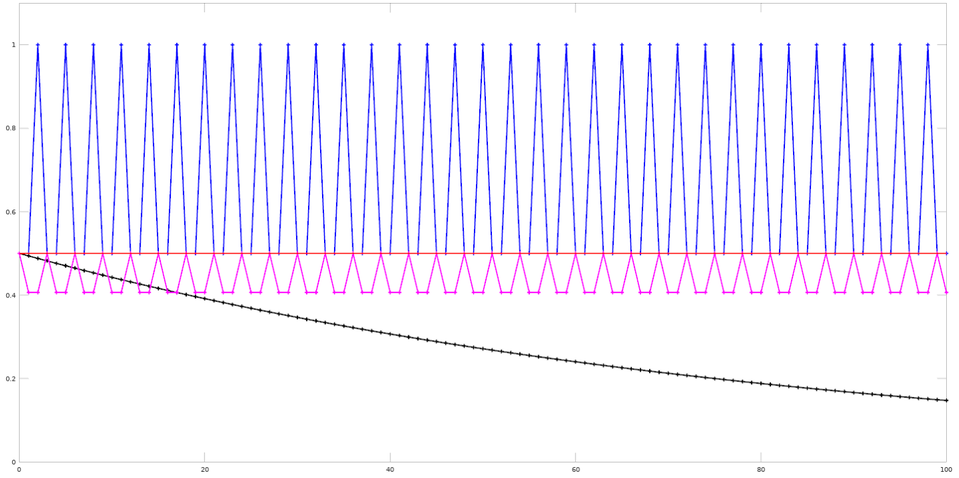
- RK4 - best behavior (undistinguishable from true solution)

Comparison between approaches

Larger $h = 1.0/\omega$



Temporal evolution of p^k



Temporal evolution of energy $E = 1/2m\,(v^k)^2 + 1/2K(p^k)^2$

True solution , *Semi-implicit Euler* , *Velocity Verlet* , *Runge-Kutta RK4*

- RK4 loose energy and p^k converge toward l^0
- Symplectic integrator keep oscillating

Summary and extensions

Numerical integration of ODE

General formulation: $u'(t) = \mathcal{F}(u,t), u(t) = (p(t), v(t))$.

Explicit Euler

$u^{k+1} = u^k + \Delta t\, \mathcal{F}(u^k, t^k)$

(+) Easy to implement

(-) Worst scheme in all cases (divergence, low accuracy)

Explicit Runge-Kutta

$u^{k+1} = u^k + \Delta t \sum_j \alpha_j k_j$

(+) Good **accuracy**

(+) Efficient to apply

(+/-) Stability OK for non-stiff problem, diverge on stiff problem

(-) Artificial damping for constant energy system

Implicit methods

$u^{k+1} = u^k + \Delta t\, \mathcal{F}(u^{k+1}, t^{k+1})$

(+) Good to deal with **stiff problem** - very stable

(-) Add numerical damping (converge even if solution oscillates)

(-) Hard/computationally costly to apply on non linear problem

Symplectic integrator

$v^{k+1} = v^k + \Delta t\, F^k / m$

$p^{k+1} = p^k + \Delta t\, v^{k+1}$

(+) Handle well constant energy system, preserves energy (Hamiltonian systems)

(+) Simple and efficient to implement

(-) Less accurate than RK

(-) Diverge on stiff problem

Position based dynamics (PBD)

PBD - Use symplectic integrator expressed **using position only**

- Velocity is computed implicitly as $(p^{k+1} - p^k)/h$
- Handle constraints using explicit projection of positions.
  - Interesting to handle non-linear/stiff constraints

```
PBD algorithm
For all k
    Integrate position (without constraints) p[k+1]=integrator(p[k],v[k],t)
    For all constraints
        Project p[k+1] on constraints
    Compute new speed v[k+1]=(p[k+1]-p[k])/h
```

(+) Unconditionnaly stable even for stiff constraints

(+) Simple to implement

(-) Low accuracy, no energy preserving

Very popular in Computer Graphics

[M. Muller et al. Position Based Dynamics. VRIPHYS 2006]

Extensions: XPBD, Projective dynamics, ...