Pencil Code school on early Universe physics and gravitational waves

October 2025, CERN (Switzerland)

Lecture: Numerical schemes for differential equations

Part I: timestepping

Antonino Salvino Midiri
(University of Geneva, Switzerland)

Part II: finite differences

Philippe Bourdin
(University of Graz, Austria)

Outline of Part I

- Introduction to numerical schemes for ODEs
- Symplectic methods
- Non-symplectic methods

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state...

$$\boxed{x(t_0),\ \dot{x}(t_0)}$$

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state…

$$\boxed{x(t_0),\ \dot{x}(t_0)}$$

… we want to compute

$$\boxed{x(t_0 + \Delta t),\ \ \dot{x}(t_0 + \Delta t)}$$

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state... ... we want to compute

$$\boxed{x(t_0),\ \dot{x}(t_0)} \longrightarrow \boxed{x(t_0 + \Delta t),\ \ \dot{x}(t_0 + \Delta t)}$$

We need an ALGORITHM

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state... ... we want to compute

$$\boxed{x(t_0),\ \dot{x}(t_0)} \longrightarrow \boxed{x(t_0 + \Delta t),\ \ \dot{x}(t_0 + \Delta t)}$$

We need an ALGORITHM $\longrightarrow$ First idea: Taylor expansion

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state... ... we want to compute

$$\boxed{x(t_0),\ \dot{x}(t_0)} \longrightarrow \boxed{x(t_0+\Delta t),\ \dot{x}(t_0+\Delta t)}$$

We need an ALGORITHM $\longrightarrow$ First idea: Taylor expansion

$$x(t_0+\Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + \mathcal{O}[(\Delta t)^2]$$

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\, \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^2]$$

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state… … we want to compute

$$\boxed{x(t_0),\ \dot{x}(t_0)} \longrightarrow \boxed{x(t_0 + \Delta t),\ \ \dot{x}(t_0 + \Delta t)}$$

We need an ALGORITHM ⟶ First idea: Taylor expansion

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + \mathcal{O}[(\Delta t)^2]$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)] + \mathcal{O}[(\Delta t)^2]$$

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state... ... we want to compute

$$\boxed{x(t_0),\ \dot{x}(t_0)} \longrightarrow \boxed{x(t_0+\Delta t),\ \ \dot{x}(t_0+\Delta t)}$$

With Taylor we are limited to order Δt [we do not know $\ddot{x}(t_0)$]

$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0)$$

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$

Numerical schemes for Ordinary Differential Equations (ODEs)

How to solve numerically $\ddot{x}(t) = \mathcal{F}[x(t)]$

Given an initial state... ... we want to compute

$$\boxed{x(t_0),\ \dot{x}(t_0)} \longrightarrow \boxed{x(t_0+\Delta t),\ \ \dot{x}(t_0+\Delta t)}$$

With Taylor we are limited to order Δt [we do not know $\ddot{x}(t_0)$]

$$x(t_0+\Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

How good is this algorithm?

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

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

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$
$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

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

$$x(t_0),\ \dot{x}(t_0) \longrightarrow \begin{cases} x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) \\ \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) - \Delta t\, \omega^2\, x(t_0) \end{cases}$$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$\omega^2 = 1$

$m = 1$

$$\ddot{x}(t) = -x(t)$$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$\omega^2 = 1$

$m = 1$

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1$$

$$\dot{x}(t_0) = 0$$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$
$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$\omega^2 = 1$

$m = 1$

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1 \qquad\qquad x(t) = \cos t$$
$$\dot{x}(t_0) = 0 \qquad\qquad \dot{x}(t) = -\sin t$$

$\xrightarrow{\text{analytical solution}}$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0)$$
$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$\omega^2 = 1$

$m = 1$

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1 \qquad \dot{x}(t_0) = 0$$

$\xrightarrow{\text{analytical solution}}$

$$x(t) = \cos t \qquad \dot{x}(t) = -\sin t$$

$\xrightarrow{\text{energy is conserved}}$

$$E(t) = \frac{1}{2} m(\dot{x}^2 + \omega^2 x^2) = \frac{1}{2}$$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1$$

$$\dot{x}(t_0) = 0$$

We can solve the problem numerically with the Euler method and compare with the analytical solution

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) \;=\; \mathcal{F}[x(t)]$$

$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0)$$

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) \;= -x(t)$$

$$x(t_0) = 1$$

$$\dot{x}(t_0) = 0$$

We can solve the problem numerically with the Euler method and compare with the analytical solution

Choose $t_0 = 0$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$
$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

We can solve the problem numerically with the Euler method and compare with the analytical solution

$$x(t_0) = 1$$

Choose $t_0 = 0$

$$\dot{x}(t_0) = 0$$

Choose Δt Period $T = 2\pi \longrightarrow \Delta t = T/100$

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$
$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1$$

$$\dot{x}(t_0) = 0$$

We can solve the problem numerically with the Euler method and compare with the analytical solution

Choose $t_0 = 0$

Choose Δt Period $T = 2\pi$ $\longrightarrow$ $\Delta t = T/100$

Choose $t_{fin} = 10\,T$ $\longrightarrow$ $N = \frac{t_{fin} - t_0}{\Delta t} = 10^3$ timesteps

Numerical schemes for Ordinary Differential Equations (ODEs)

$$\ddot{x}(t) \;=\; \mathcal{F}[x(t)]$$

$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0)$$

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) \;= -x(t)$$

$$x(t_0) = 1$$

$$\dot{x}(t_0) = 0$$

We can solve the problem numerically with the Euler method and compare with the analytical solution

Choose $t_0 = 0$

Choose Δt Period $T = 2\pi \longrightarrow \Delta t = T/100$

Choose $t_{fin} = 10\,T \longrightarrow N = \frac{t_{fin}-t_0}{\Delta t} = 10^3$ timesteps

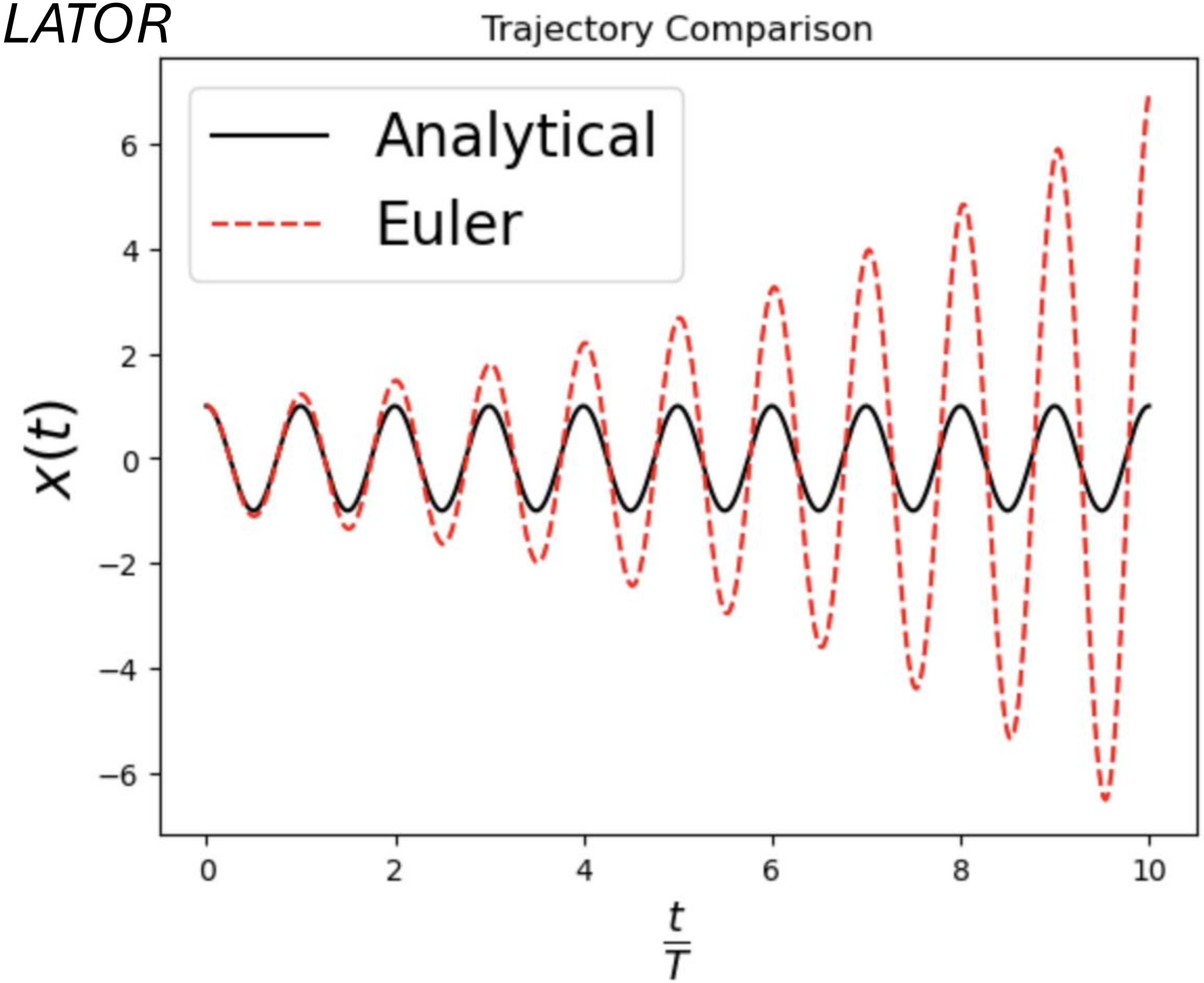
Compare at all timesteps

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \dot{x}(t_0) = 0$$



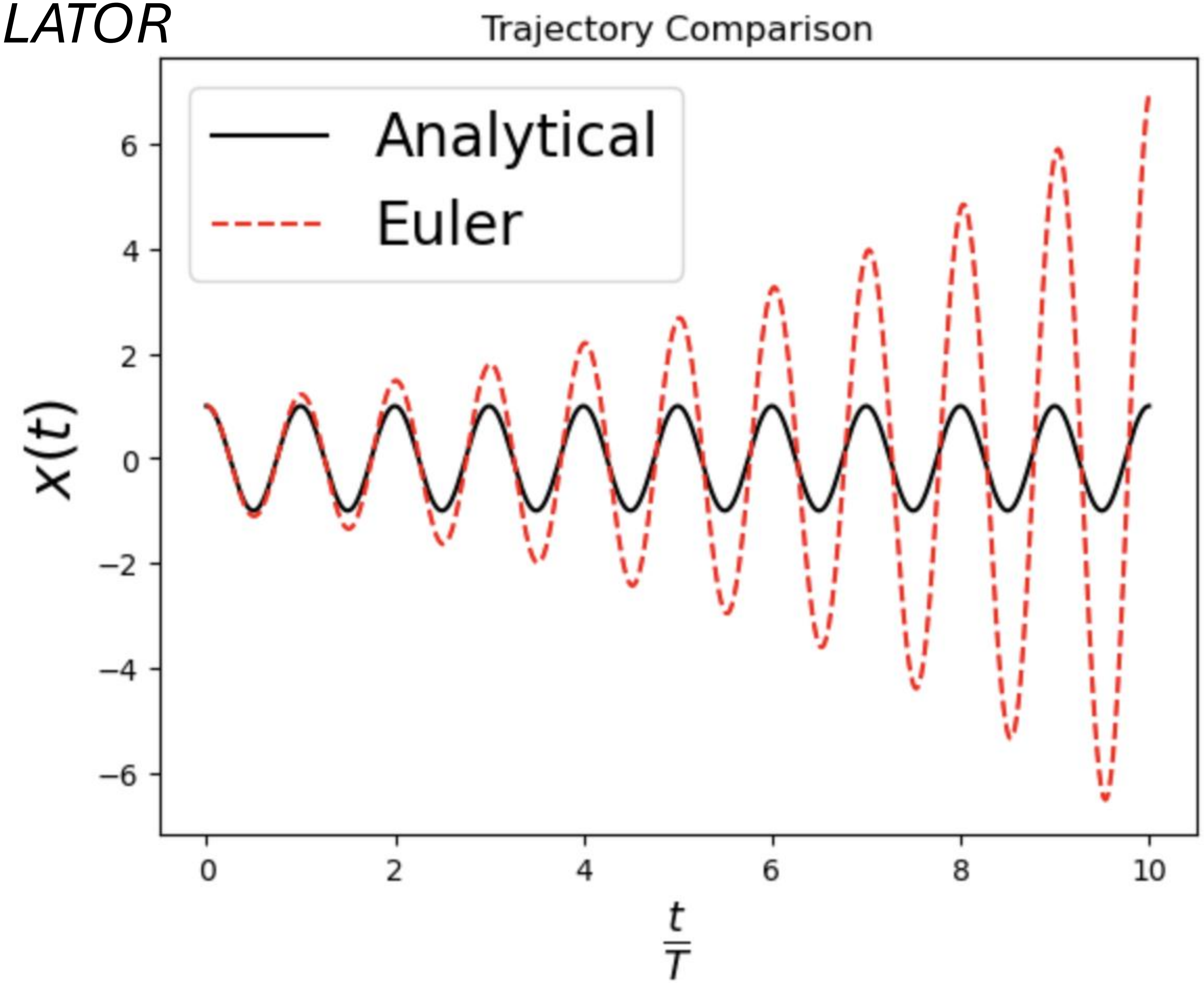
Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \dot{x}(t_0) = 0$$

Euler method fails at keeping the trajectory bounded as in the analytical solution



Numerical schemes for Ordinary Differential Equations (ODEs)

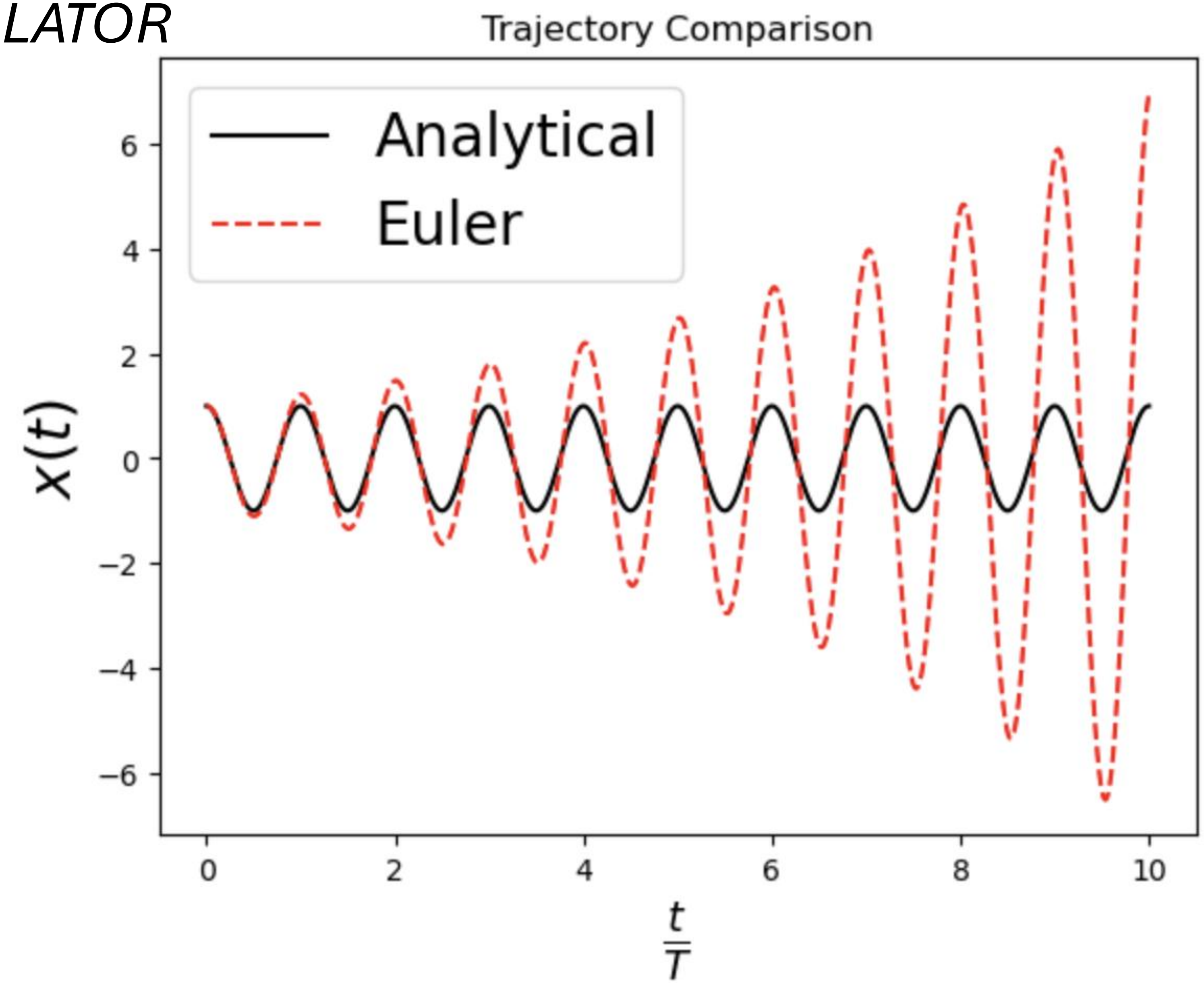
EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \dot{x}(t_0) = 0$$

Euler method fails at keeping the trajectory bounded as in the analytical solution

Let us check energy conservation



Numerical schemes for Ordinary Differential Equations (ODEs)

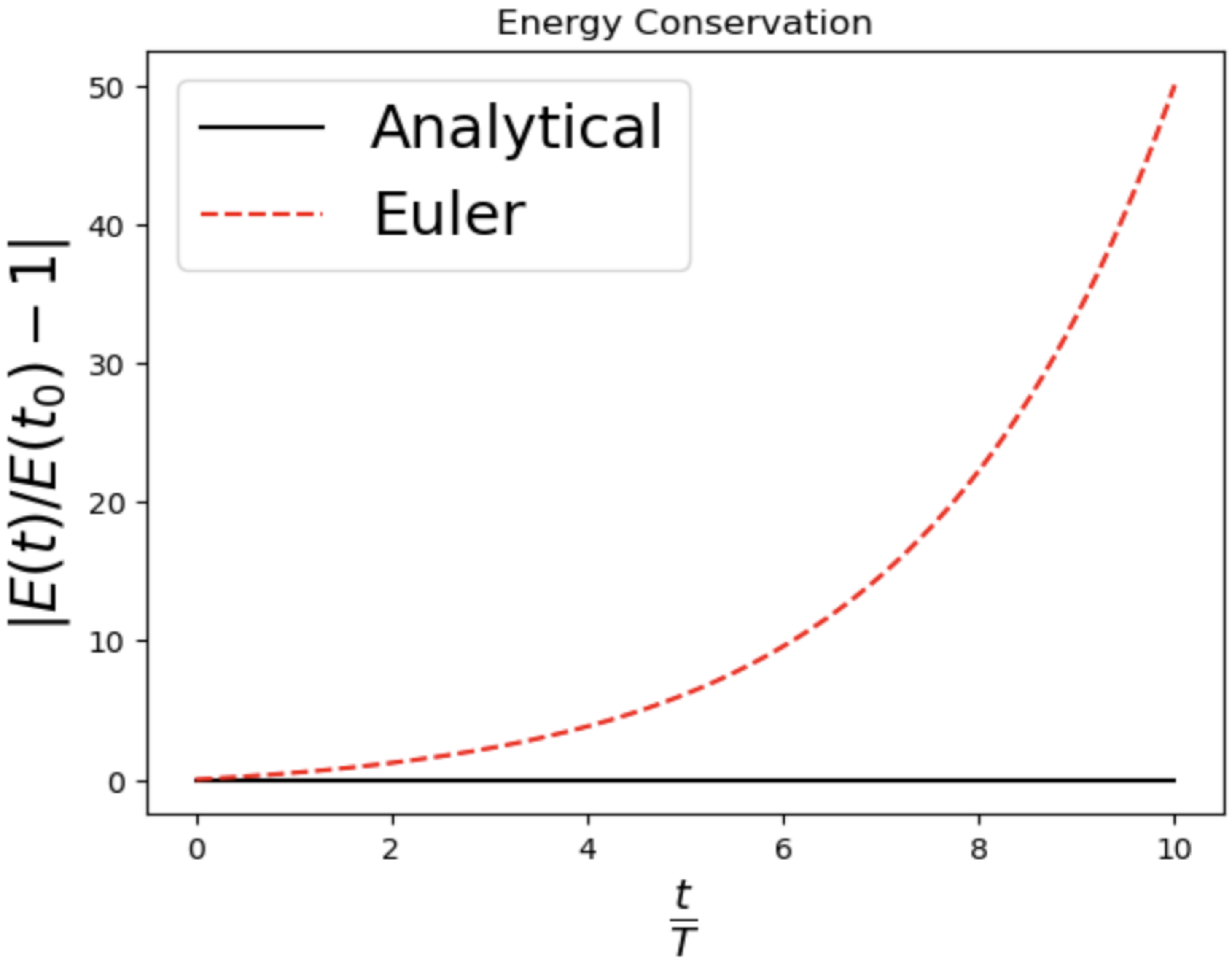
EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \dot{x}(t_0) = 0$$

Euler method fails at keeping the trajectory bounded as in the analytical solution

Let us check energy conservation



Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

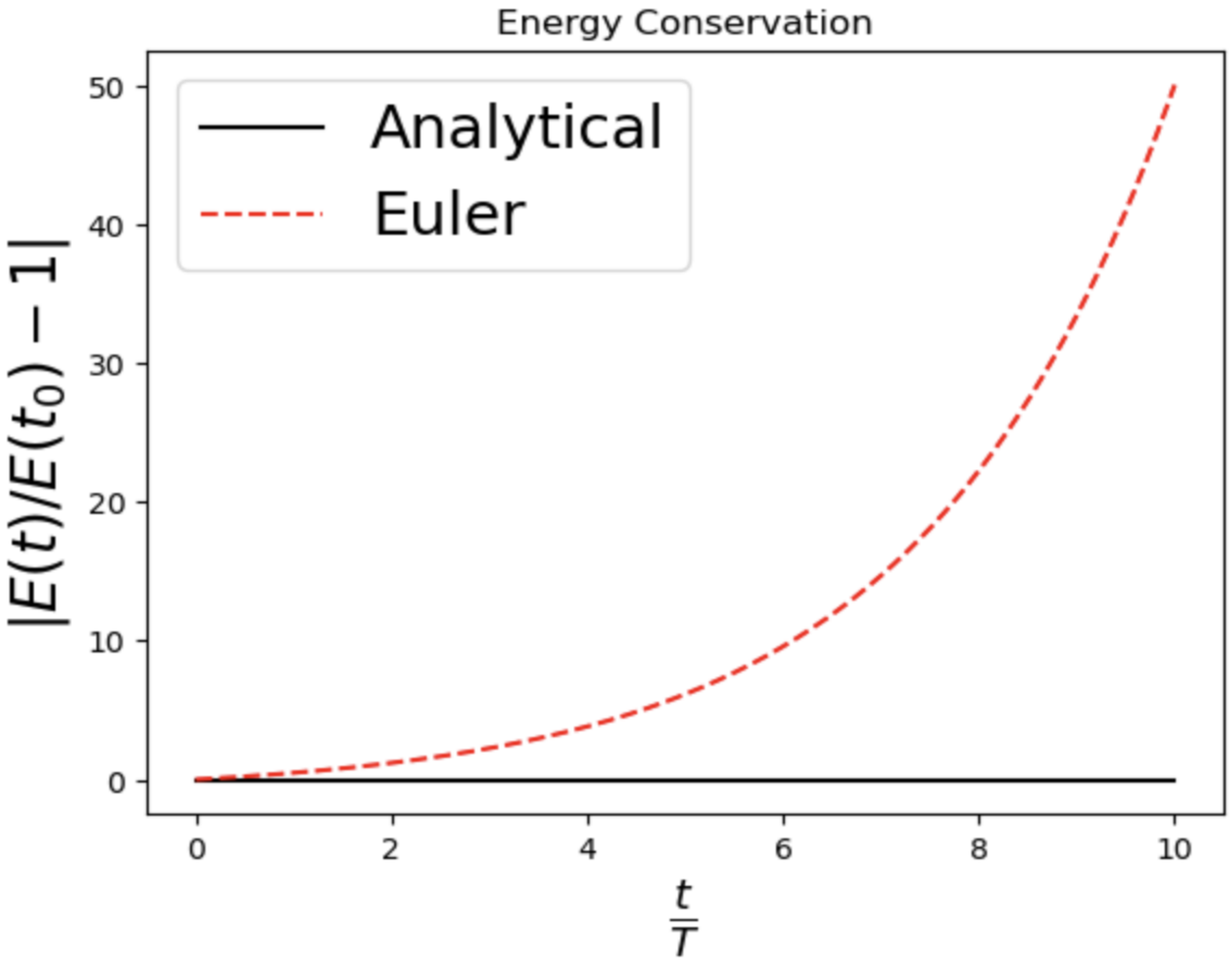
$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \dot{x}(t_0) = 0$$

Euler method fails at keeping the trajectory bounded as in the analytical solution

Let us check energy conservation

Euler methods fails at keeping the energy bounded



Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

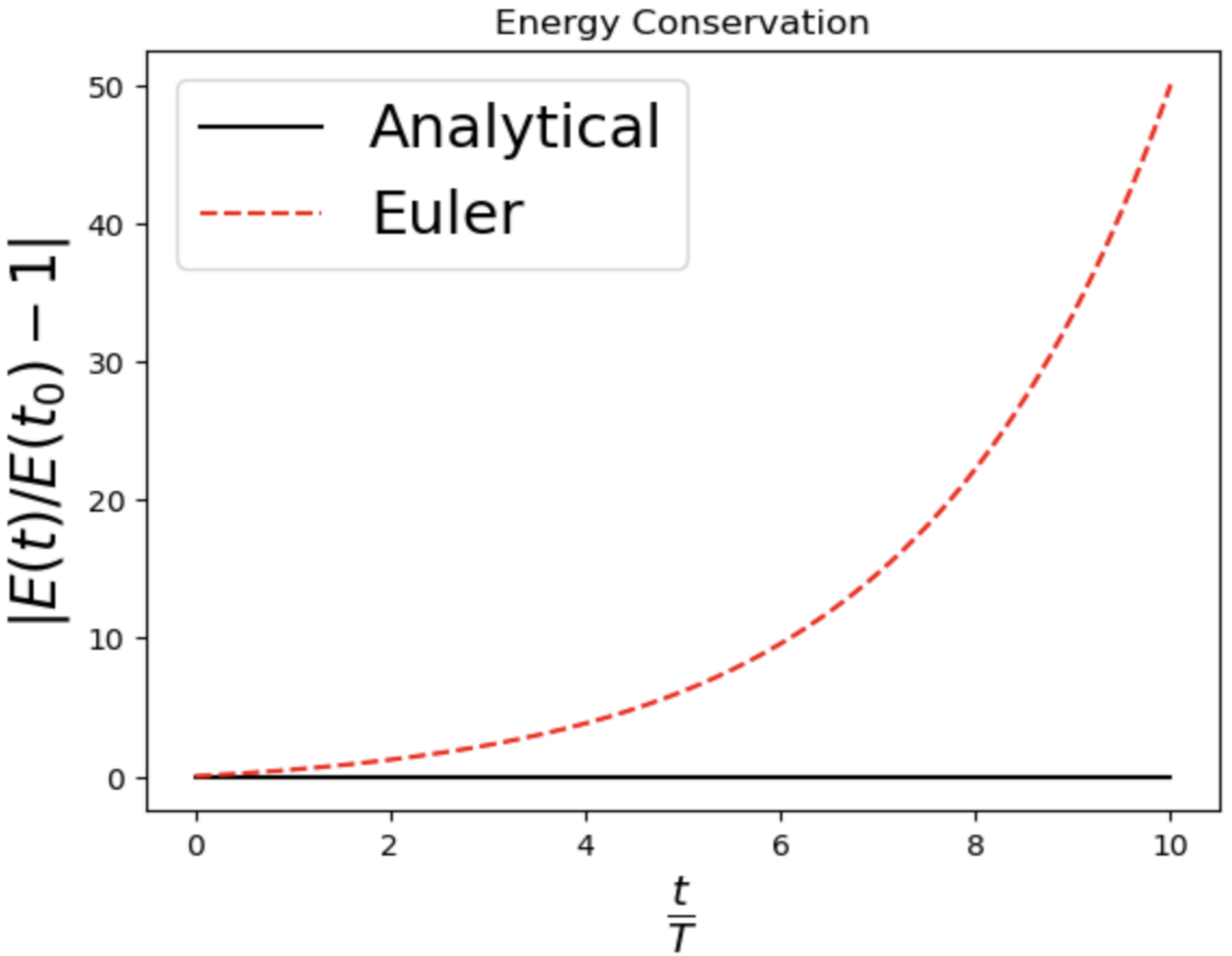
$$x(t_0) = 1, \dot{x}(t_0) = 0$$

Euler method fails at keeping the trajectory bounded as in the analytical solution

Let us check energy conservation

Euler methods fails at keeping the energy bounded

Let us look at the phase space

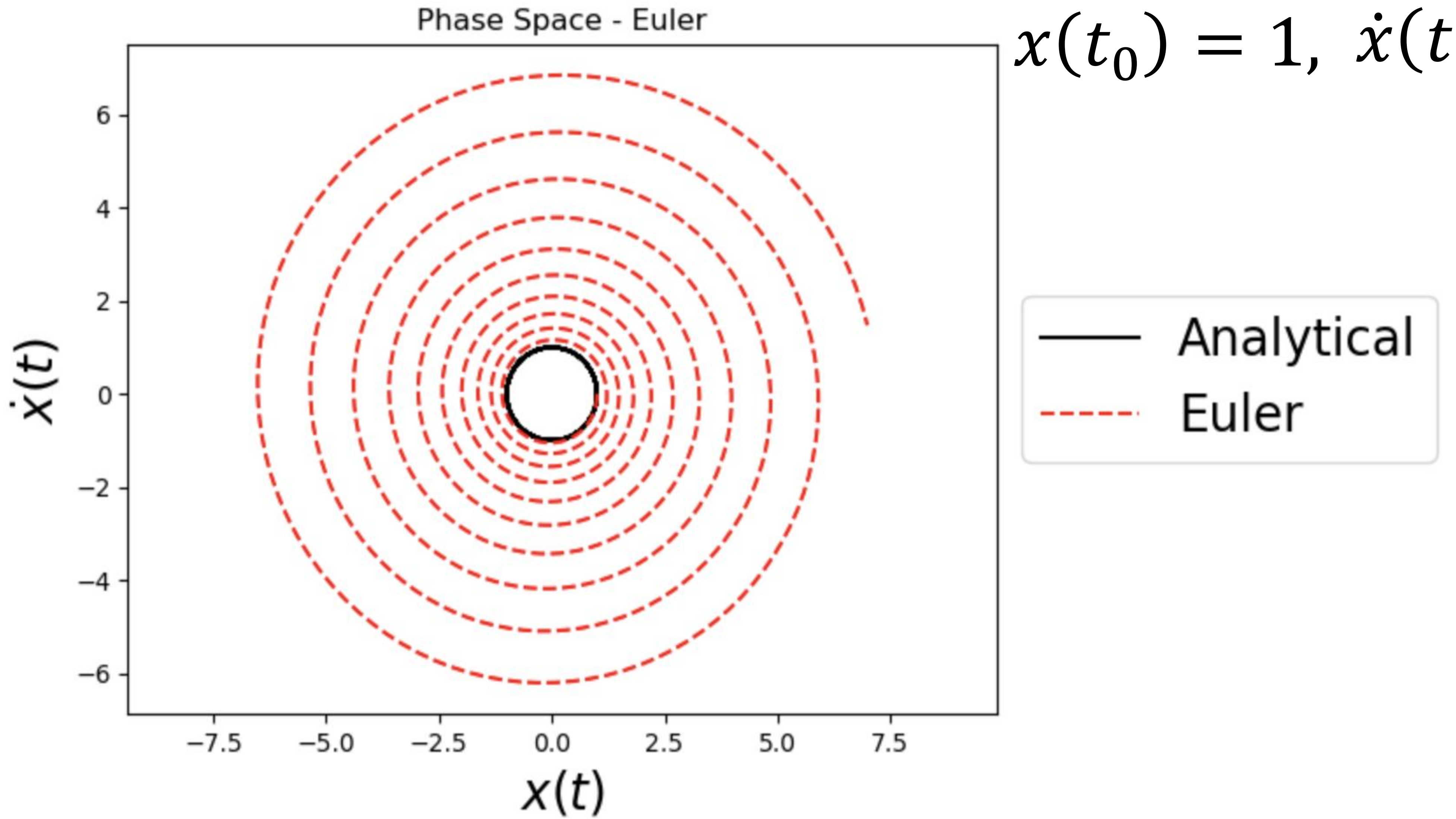


Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$

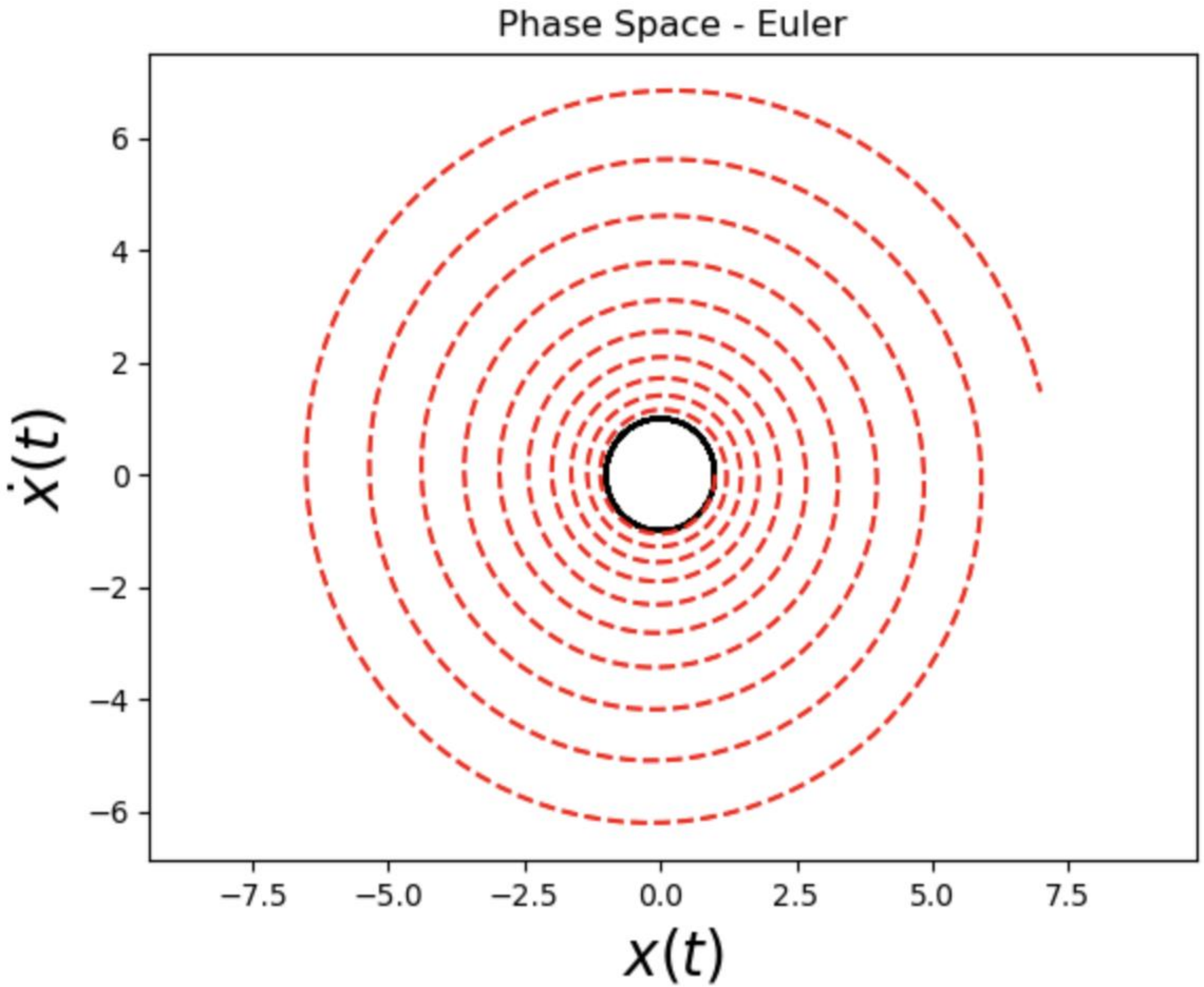


Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$



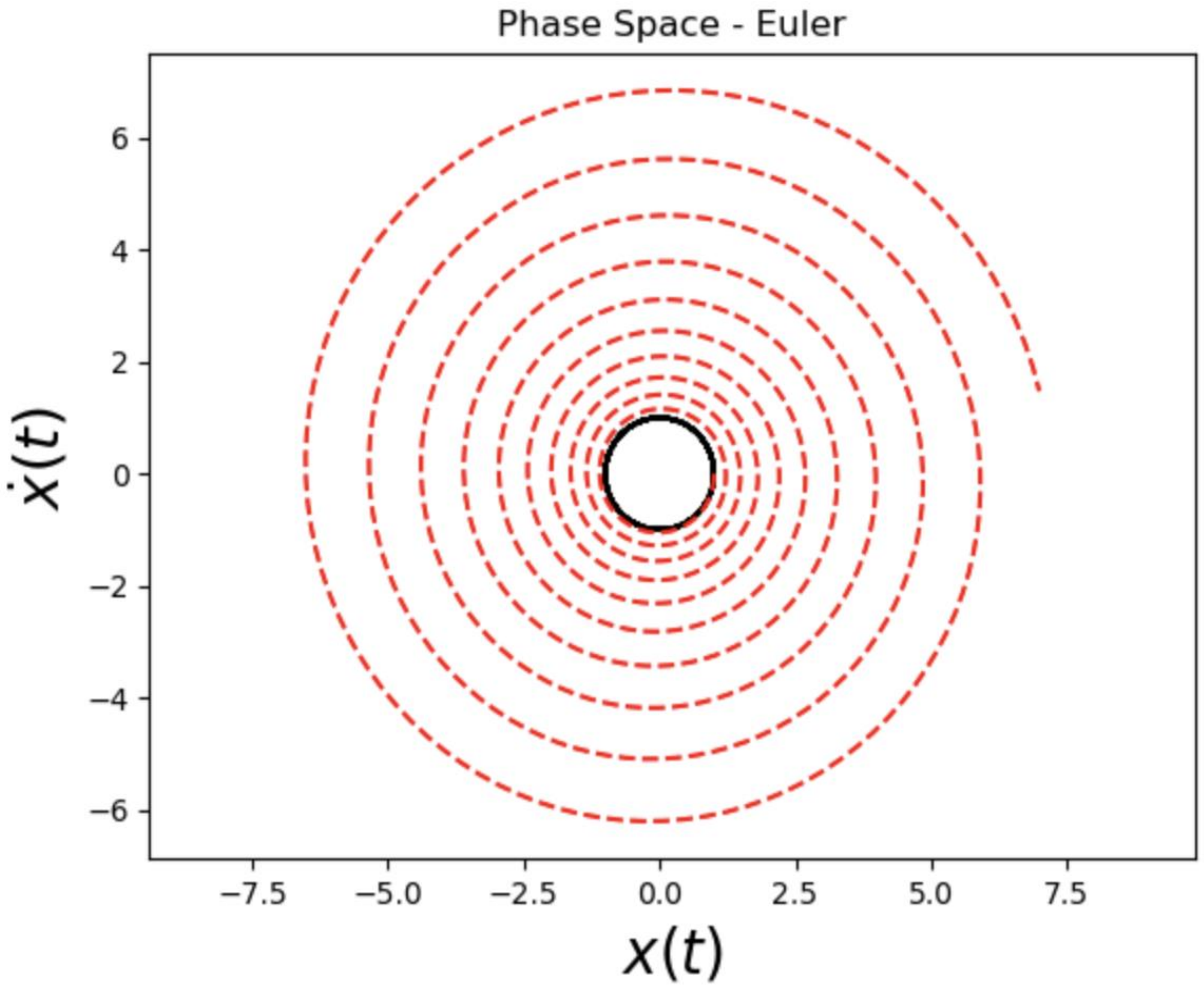
Liouville's theorem for Hamiltonian systems here is *numerically* violated: the phase space area is not preserved throughout the evolution

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$



Liouville's theorem for Hamiltonian systems here is *numerically* violated: the phase space area is not preserved throughout the evolution

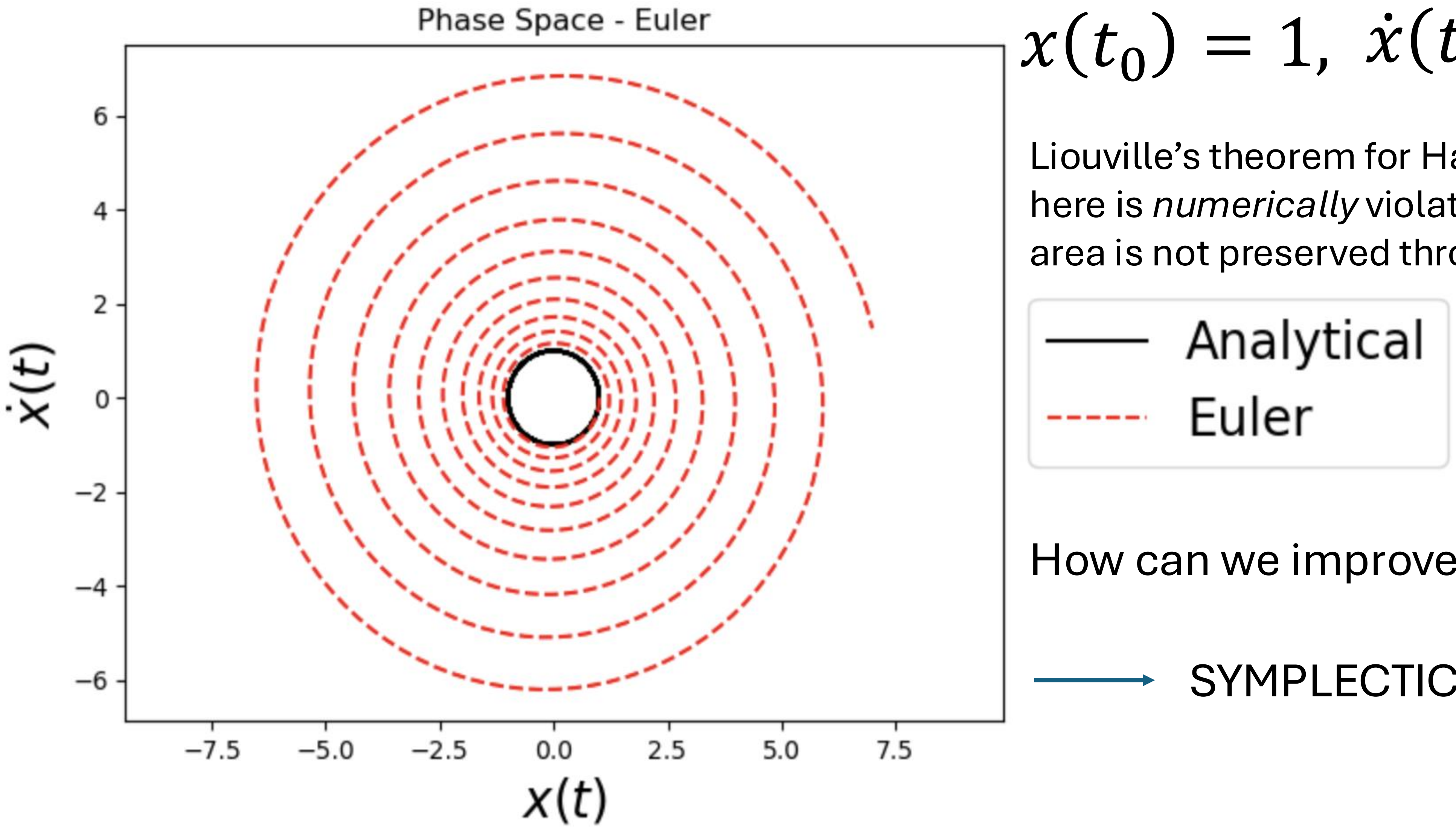
How can we improve our algorithm?

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$



Liouville's theorem for Hamiltonian systems here is *numerically* violated: the phase space area is not preserved throughout the evolution

How can we improve our algorithm?

→ SYMPLECTIC METHODS

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

We can in fact compute the Jacobian of the transformation $[x(t_0), \dot{x}(t_0)] \rightarrow [x(t_1), \dot{x}(t_1)]$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

We can in fact compute the Jacobian of the transformation $[x(t_0), \dot{x}(t_0)] \rightarrow [x(t_1), \dot{x}(t_1)]$

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix}$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

We can in fact compute the Jacobian of the transformation $[x(t_0), \dot{x}(t_0)] \rightarrow [x(t_1), \dot{x}(t_1)]$

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 & \Delta t \\ \Delta t \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & 1 \end{pmatrix}$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

We can in fact compute the Jacobian of the transformation $[x(t_0), \dot{x}(t_0)] \rightarrow [x(t_1), \dot{x}(t_1)]$

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 & \Delta t \\ \Delta t \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & 1 \end{pmatrix}$$

$$\det \mathbb{J} = 1 - (\Delta t)^2 \frac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} \neq 1$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

Can we modify in order to have $\det \mathbb{J} = 1$?

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 & \Delta t \\ \Delta t \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & 1 \end{pmatrix}$$

$$\det \mathbb{J} = 1 - (\Delta t)^2 \frac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} \neq 1$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_1)$$

Can we modify in order to have $\det \mathbb{J} = 1$?

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 & \Delta t \\ \Delta t\, \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & 1 \end{pmatrix}$$

$$\det \mathbb{J} = 1 - (\Delta t)^2 \frac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} \neq 1$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_1)$$

Can we modify in order to have $\det \mathbb{J} = 1$?

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 + (\Delta t)^2 \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & \Delta t \\ \Delta t \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & 1 \end{pmatrix}$$

$$\det \mathbb{J} = 1 - (\Delta t)^2 \frac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} \neq 1$$

Numerical schemes for Ordinary Differential Equations (ODEs)

Euler method does not preserve phase space area

$$\dot{x}(t_1) \equiv \dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_1) \equiv x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_1)$$

Can we modify in order to have $\det \mathbb{J} = 1$?

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_1)}{\partial x(t_0)} & \dfrac{\partial x(t_1)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_1)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_1)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 + (\Delta t)^2 \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & \Delta t \\ \Delta t \dfrac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} & 1 \end{pmatrix}$$

$$\det \mathbb{J} = 1 + (\Delta t)^2 \frac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} - (\Delta t)^2 \frac{\partial \mathcal{F}[x(t_0)]}{\partial x(t_0)} = 1$$

Numerical schemes for Ordinary Differential Equations (ODEs)

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

$$= x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

Numerical schemes for Ordinary Differential Equations (ODEs)

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

$$= x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

Symplectic = preserves phase space area $(\det \mathbb{J} = 1)$

Numerical schemes for Ordinary Differential Equations (ODEs)

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

$$= x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

Symplectic = preserves phase space area $(\det \mathbb{J} = 1)$

→ Trajectories cannot go unbounded

→ Energy, even if not exactly conserved, is bounded

Numerical schemes for Ordinary Differential Equations (ODEs)

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

$$= x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

Symplectic = preserves phase space area ($\det \mathbb{J} = 1$)

→ Trajectories cannot go unbounded

→ Energy, even if not exactly conserved, is bounded

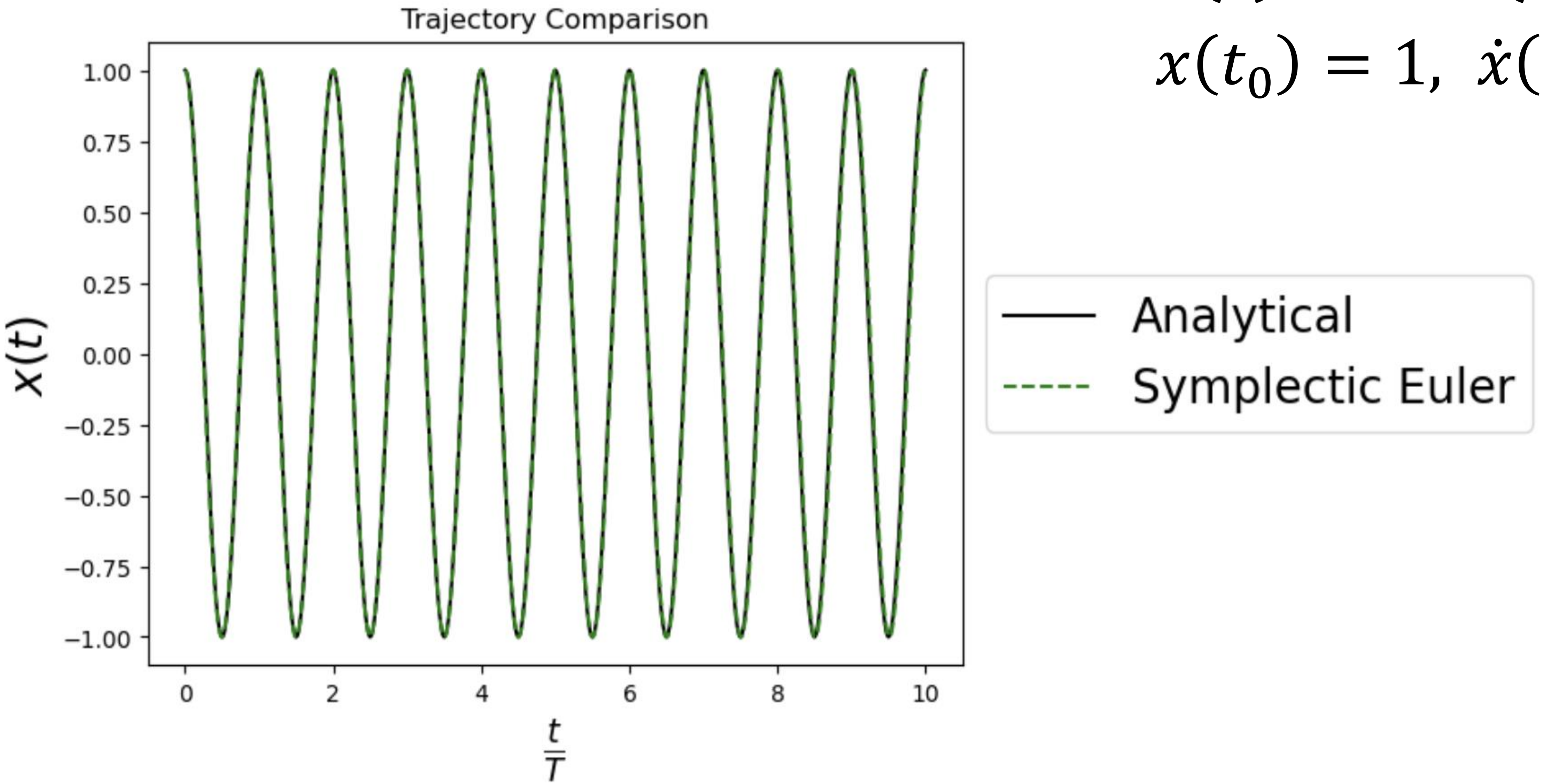
Let us apply it to the *1D HARMONIC OSCILLATOR*!

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$

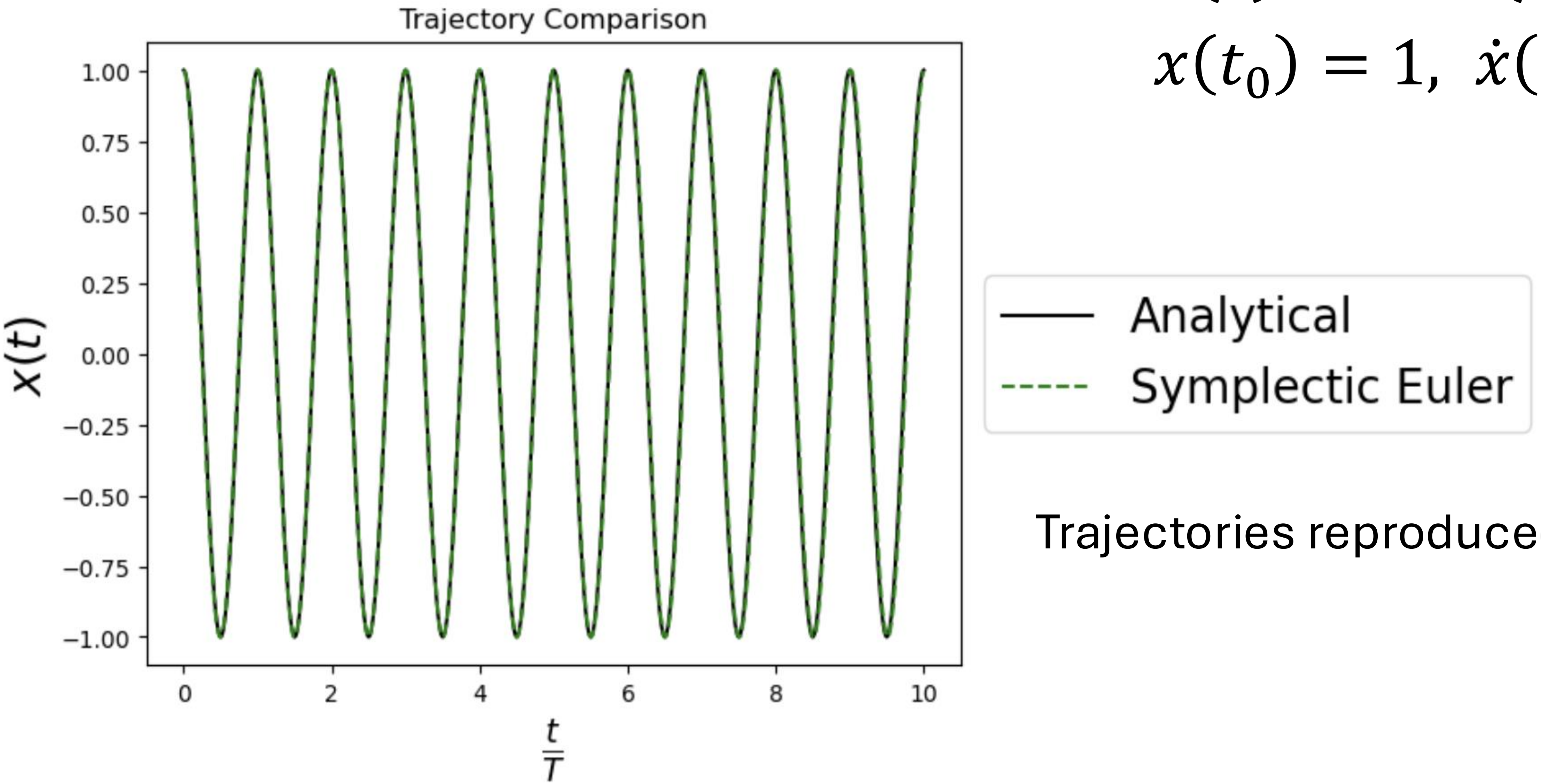


Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$



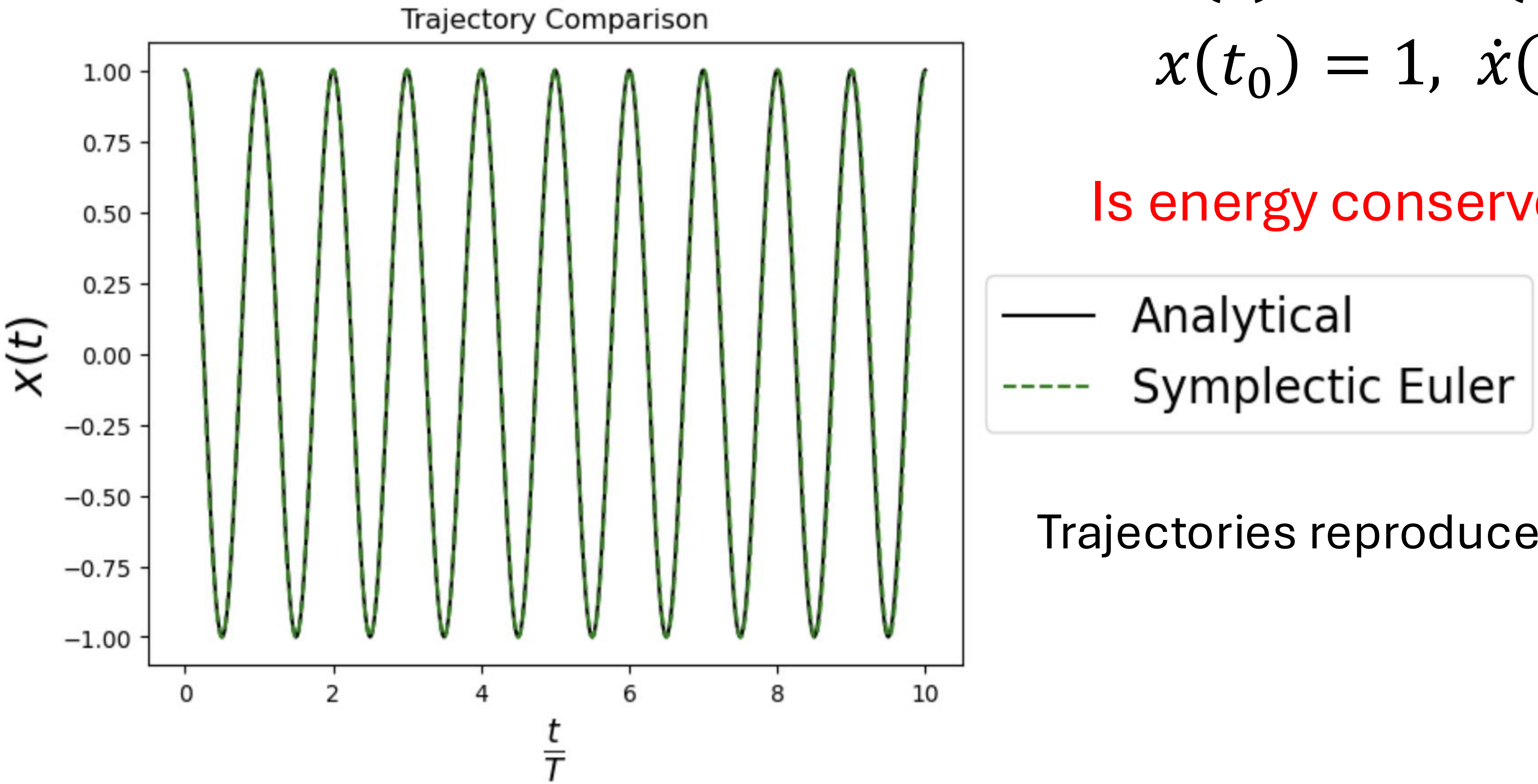
Trajectories reproduced ✓

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE:*1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \; \dot{x}(t_0) = 0$$



Is energy conserved?

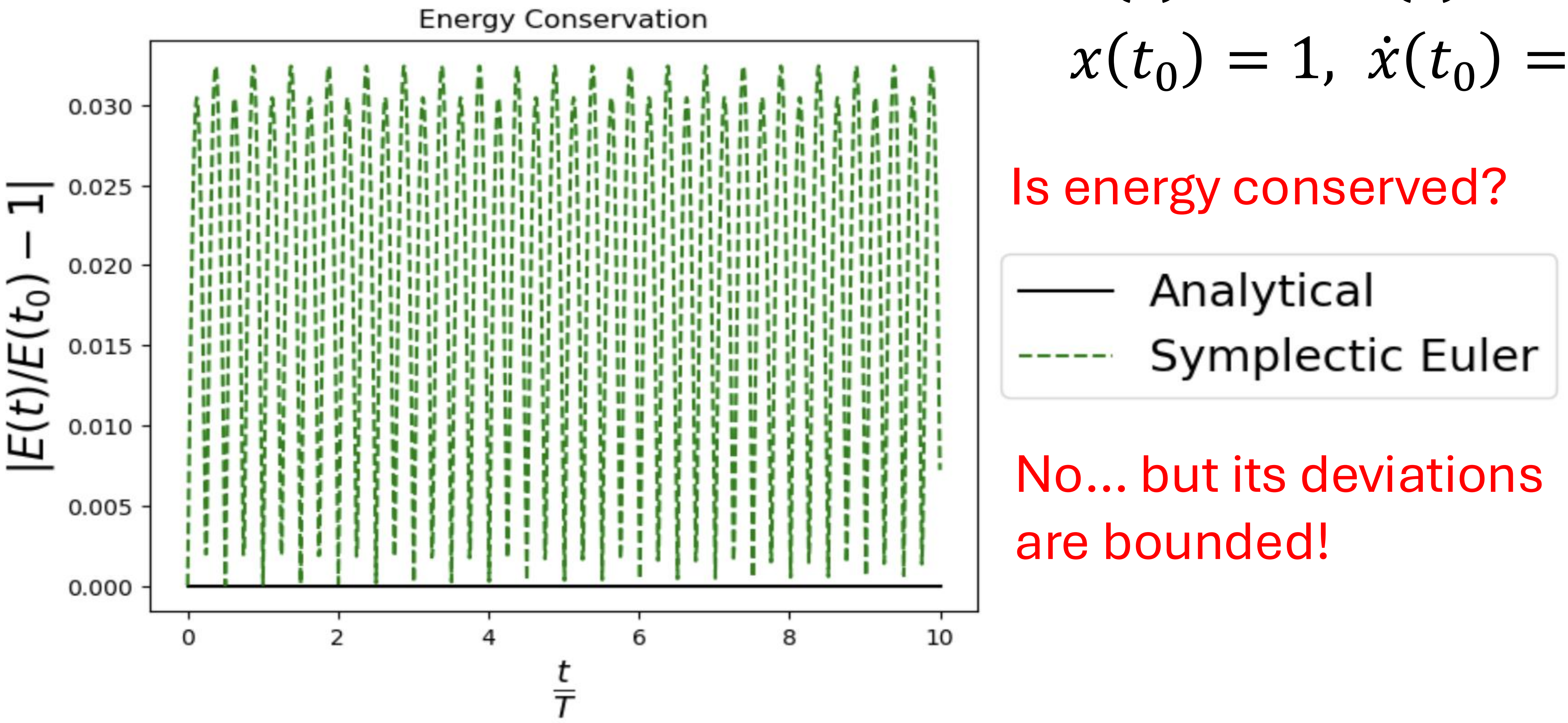
Trajectories reproduced ✓

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1, \ \dot{x}(t_0) = 0$$



Is energy conserved?

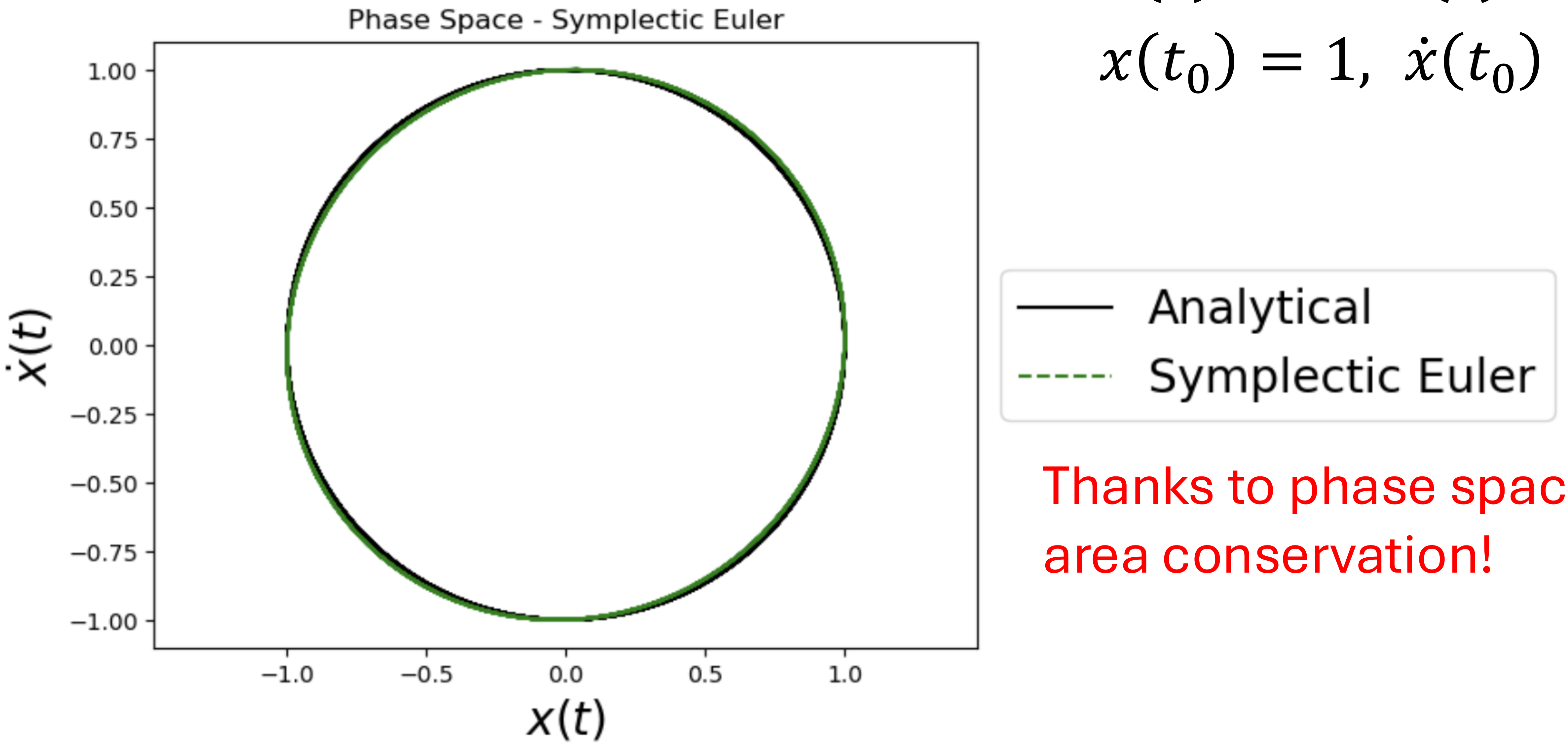
No... but its deviations are bounded!

Numerical schemes for Ordinary Differential Equations (ODEs)

EXAMPLE: *1D HARMONIC OSCILLATOR*

$$\ddot{x}(t) = -x(t)$$

$$x(t_0) = 1,\ \dot{x}(t_0) = 0$$



Thanks to phase space area conservation!

Symplectic methods

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

Symplectic Euler preserves phase space area $(\det \mathbb{J} = 1)$

Symplectic methods

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

Symplectic Euler preserves phase space area $(\det \mathbb{J} = 1)$

Consequence of the algorithmic structure but also of dealing with a CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Symplectic methods

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

Symplectic Euler preserves phase space area $(\det \mathbb{J} = 1)$

Consequence of the algorithmic structure but also of dealing with a CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

If we had instead a NON-CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$$

Symplectic methods

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

Symplectic Euler preserves phase space area $(\det \mathbb{J} = 1)$

Consequence of the algorithmic structure but also of dealing with a CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

If we had instead a NON-CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$$

The same algorithm would give

$$\mathbb{J} = \begin{pmatrix} \dfrac{\partial x(t_0+\Delta t)}{\partial x(t_0)} & \dfrac{\partial x(t_0+\Delta t)}{\partial \dot{x}(t_0)} \\ \dfrac{\partial \dot{x}(t_0+\Delta t)}{\partial x(t_0)} & \dfrac{\partial \dot{x}(t_0+\Delta t)}{\partial \dot{x}(t_0)} \end{pmatrix} = \begin{pmatrix} 1 + (\Delta t)^2 \dfrac{\partial \mathcal{F}}{\partial x(t_0)} & \Delta t + (\Delta t)^2 \dfrac{\partial \mathcal{F}}{\partial \dot{x}(t_0)} \\ \Delta t \dfrac{\partial \mathcal{F}}{\partial x(t_0)} & 1 + \Delta t \dfrac{\partial \mathcal{F}}{\partial \dot{x}(t_0)} \end{pmatrix}$$

Symplectic methods

SYMPLECTIC EULER

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0 + \Delta t)$$

Symplectic Euler preserves phase space area $(\det \mathbb{J} = 1)$

Consequence of the algorithmic structure but also of dealing with a CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

If we had instead a NON-CONSERVATIVE system

$$\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$$

The same algorithm would give

$$\mathbb{J} = \begin{pmatrix} 1 + (\Delta t)^2 \dfrac{\partial \mathcal{F}}{\partial x(t_0)} & \Delta t + (\Delta t)^2 \dfrac{\partial \mathcal{F}}{\partial \dot{x}(t_0)} \\ \Delta t \dfrac{\partial \mathcal{F}}{\partial x(t_0)} & 1 + \Delta t \dfrac{\partial \mathcal{F}}{\partial \dot{x}(t_0)} \end{pmatrix}$$

$$\longrightarrow \det \mathbb{J} = 1 + \Delta t \frac{\partial \mathcal{F}}{\partial \dot{x}(t_0)} \neq 1$$

The algorithm is no longer symplectic

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

They are thus a natural choice for separable Hamiltonian systems

$$H(p, x) = T(p) + V(x) = \frac{p^2}{2m} + V(x)$$

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

They are thus a natural choice for separable Hamiltonian systems

$$\mathrm{H}(\mathrm{p},\mathrm{x}) = \mathrm{T}(\mathrm{p}) + \mathrm{V}(\mathrm{x}) = \frac{\mathrm{p}^2}{2\mathrm{m}} + \mathrm{V}(\mathrm{x})$$

$$\begin{cases} \dot{x} = \dfrac{\partial H}{\partial p} = \dfrac{p}{m} \\ \ddot{x} = \dfrac{\dot{p}}{m} = -\dfrac{1}{m}\dfrac{\partial H}{\partial x} = -\dfrac{1}{m}\dfrac{\partial V}{\partial x} = \mathcal{F}[x(t)] \end{cases}$$

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER...

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

... is a *first-order* method (*accurate to order Δt*)

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER…

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

… is a *first-order* method (*accurate to order Δt*)

The error with respect to the continuum is proportional to $(\Delta t)^2$

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER…

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

… is a *first-order* method (*accurate to order Δt*)

The error with respect to the continuum is proportional to $(\Delta t)^2$

This can be shown by Taylor expanding the L.H.S. around t_0

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER...

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$
$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0) + (\Delta t)^2\mathcal{F}[x(t_0)]$$

... is a *first-order* method (*accurate to order Δt*)

The error with respect to the continuum is proportional to $(\Delta t)^2$

This can be shown by Taylor expanding the L.H.S. around t_0

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\ddot{x}(t_0) + \frac{1}{2}(\Delta t)^2\dddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$
$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0) + \frac{1}{2}(\Delta t)^2\ddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER...

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

... is a *first-order* method (*accurate to order* Δt)

The error with respect to the continuum is proportional to $(\Delta t)^2$

This can be shown by Taylor expanding the L.H.S. around t_0

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \ddot{x}(t_0) + \frac{1}{2}(\Delta t)^2 \dddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + \frac{1}{2}(\Delta t)^2 \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$

Difference between Taylor and algorithm (error) proportional to $(\Delta t)^2$!

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER…

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}[x(t_0)]$$
$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0) + (\Delta t)^2\mathcal{F}[x(t_0)]$$

… is a *first-order* method (*accurate to order* Δt)

The error with respect to the continuum is proportional to $(\Delta t)^2$

This can be shown by Taylor expanding the L.H.S. around t_0

$$\dot{x}(t_0+\Delta t) = \dot{x}(t_0) + \Delta t\,\ddot{x}(t_0) + \frac{1}{2}(\Delta t)^2\dddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$
$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}(t_0) + \frac{1}{2}(\Delta t)^2\ddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$

To reach a certain desired accuracy, one may need (depending on the problem) a very small Δt, which makes the computation longer!

Symplectic methods

General property of symplectic methods

They are *simple and easy to write* for CONSERVATIVE SYSTEMS

SYMPLECTIC EULER...

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0)]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + (\Delta t)^2 \mathcal{F}[x(t_0)]$$

... is a *first-order* method (*accurate to order* Δt)

The error with respect to the continuum is proportional to $(\Delta t)^2$

This can be shown by Taylor expanding the L.H.S. around t_0

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\, \ddot{x}(t_0) + \frac{1}{2}(\Delta t)^2 \dddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$
$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + \frac{1}{2}(\Delta t)^2 \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$

How to write a higher order symplectic method?

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion gives

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}(t_0) + \frac{1}{2}(\Delta t)^2 \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion gives

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \left[\dot{x}(t_0) + \frac{1}{2}\Delta t\, \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^2]\right]$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion gives

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \underbrace{\left[\dot{x}(t_0) + \frac{1}{2}\Delta t\, \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^2]\right]}_{\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^2]}$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion gives

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods ⟶ (staggered) LEAPFROG

Second order Taylor expansion also for $\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \ddot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \frac{1}{2}(\Delta t)^2 \dddot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion also for $\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t \left[\ddot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \frac{1}{2}\Delta t\, \dddot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^2]\right]$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion also for $\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t \left[\underbrace{\ddot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \frac{1}{2}\Delta t\, \dddot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^2]}_{\ddot{x}(t_0) + \mathcal{O}[(\Delta t)^2]}\right]$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods ⟶ (staggered) LEAPFROG

Second order Taylor expansion also for $\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \ddot{x}(t_0) + \mathcal{O}[(\Delta t)^3]$$

Symplectic methods

$\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

Second order Taylor expansion also for $\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \mathcal{O}[(\Delta t)^3]$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)] + \mathcal{O}[(\Delta t)^3]$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods ⟶ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

$$\dot{x}\left(t_0 - \frac{\Delta t}{2}\right) \xrightarrow{x(t_0)} \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

$$x(t_0) \longrightarrow x(t_0 + \Delta t)$$

$$\dot{x}\left(t_0 - \frac{\Delta t}{2}\right) \qquad \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

$x(t_0)$ $\qquad$ $x(t_0 + \Delta t)$

$\dot{x}\left(t_0 - \frac{\Delta t}{2}\right)$ $\qquad$ $\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) \longrightarrow \dot{x}\left(t_0 + \frac{3\Delta t}{2}\right)$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

$$x(t_0) \qquad\qquad x(t_0 + \Delta t) \longrightarrow x(t_0 + 2\,\Delta t)$$

$$\dot{x}\left(t_0 - \frac{\Delta t}{2}\right) \qquad \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) \qquad \dot{x}\left(t_0 + \frac{3\Delta t}{2}\right) \qquad \ldots$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods $\longrightarrow$ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t\, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

However, measurement of several physical quantities may involve both x and $\dot{\mathrm{x}}$ at the same timestep

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods ⟶ (staggered) LEAPFROG

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}\left(t_0 - \frac{\Delta t}{2}\right) + \Delta t \, \mathcal{F}[x(t_0)]$$

If we compute x at integer timesteps and $\dot{x}$ at half-integer timesteps we find a simple second order symplectic method

However, measurement of several physical quantities may involve both x and $\dot{\mathrm{x}}$ at the same timestep

⟶ one more step needed (either advancing x or $\dot{x}$)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (synchronized) LEAPFROG

$x(t_0),\ \dot{x}(t_0)$

Symplectic methods

$\ddot{x}(t) = \mathcal{F}[x(t)]$

Second order methods ⟶ (synchronized) LEAPFROG

$x(t_0), \dot{x}(t_0)$

Velocity Verlet (VV)

1)

2)

3)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (synchronized) LEAPFROG

$x(t_0), \dot{x}(t_0)$

Velocity Verlet (VV)

1) Advance first the velocity by a half timestep

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}(t_0) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0)]$$

2)

3)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods → (synchronized) LEAPFROG

$x(t_0),\ \dot{x}(t_0)$

Velocity Verlet (VV)

1) Advance first the velocity by a half timestep

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}(t_0) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0)]$$

2) Advance the position using the velocity at half timestep

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\,\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

3)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods ⟶ (synchronized) LEAPFROG

$x(t_0),\ \dot{x}(t_0)$

Velocity Verlet (VV)

1) Advance first the velocity by a half timestep

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}(t_0) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0)]$$

2) Advance the position using the velocity at half timestep

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\,\dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

3) Advance the velocity by another half timestep

$$\dot{x}(t_0 + \Delta t) = \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0 + \Delta t)]$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods ⟶ (synchronized) LEAPFROG

$x(t_0),\, \dot{x}(t_0)$

Velocity Verlet (VV)

1) Advance first the velocity by a half timestep

$$\dot{x}\left(t_0 + \frac{\Delta t}{2}\right) = \dot{x}(t_0) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0)]$$

2) Advance the position using the velocity at half timestep

$$x(t_0 + \Delta t) = x(t_0) + \Delta t\, \dot{x}\left(t_0 + \frac{\Delta t}{2}\right)$$

3) Advance the velocity by another half timestep

$$\dot{x}(t_0 + \Delta t) = \dot{x}\left(t_0 + \frac{\Delta t}{2}\right) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0 + \Delta t)]$$

Second-order symplectic method → *accurate to order* $(\Delta t)^2$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (synchronized) LEAPFROG $\quad x(t_0),\, \dot{x}(t_0)$

Position Verlet (PV)

1)

2)

3)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (synchronized) LEAPFROG

$x(t_0),\ \dot{x}(t_0)$

Position Verlet (PV)

1) Advance first the position by a half timestep

$$x\left(t_0 + \frac{\Delta t}{2}\right) = x(t_0) + \frac{\Delta t}{2}\,\dot{x}(t_0)$$

2)

3)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (synchronized) LEAPFROG

$x(t_0),\ \dot{x}(t_0)$

Position Verlet (PV)

1) Advance first the position by a half timestep

$$x\left(t_0 + \frac{\Delta t}{2}\right) = x(t_0) + \frac{\Delta t}{2}\,\dot{x}(t_0)$$

2) Advance the velocity using the position at half timestep

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}\left[x\left(t_0 + \frac{\Delta t}{2}\right)\right]$$

3)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods ⟶ (synchronized) LEAPFROG

$x(t_0),\ \dot{x}(t_0)$

Position Verlet (PV)

1) Advance first the position by a half timestep

$$x\left(t_0 + \frac{\Delta t}{2}\right) = x(t_0) + \frac{\Delta t}{2}\,\dot{x}(t_0)$$

2) Advance the velocity using the position at half timestep

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}\left[x\left(t_0 + \frac{\Delta t}{2}\right)\right]$$

3) Advance the position by another half timestep

$$x(t_0 + \Delta t) = x\left(t_0 + \frac{\Delta t}{2}\right) + \frac{\Delta t}{2}\,\dot{x}(t_0 + \Delta t)$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Second order methods $\longrightarrow$ (synchronized) LEAPFROG $\quad x(t_0),\ \dot{x}(t_0)$

Position Verlet (PV)

1) Advance first the position by a half timestep

$$x\left(t_0 + \frac{\Delta t}{2}\right) = x(t_0) + \frac{\Delta t}{2}\,\dot{x}(t_0)$$

2) Advance the velocity using the position at half timestep

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t\,\mathcal{F}\left[x\left(t_0 + \frac{\Delta t}{2}\right)\right]$$

3) Advance the position by another half timestep

$$x(t_0 + \Delta t) = x\left(t_0 + \frac{\Delta t}{2}\right) + \frac{\Delta t}{2}\,\dot{x}(t_0 + \Delta t)$$

Second-order symplectic method → *accurate to order* $(\Delta t)^2$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Higher order LEAPFROG $\longrightarrow$ YOSHIDA methods

[H. Yoshida, 1990]

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Higher order LEAPFROG → **YOSHIDA methods**

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t$ $(i = 1,2,\dots,M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Higher order LEAPFROG → YOSHIDA methods

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t$ $(i = 1,2, \dots, M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Then apply Leapfrog (either VV or PV) M times in sequence (step i with timestep Δt_i)

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Higher order LEAPFROG $\longrightarrow$ YOSHIDA methods

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t \ (i = 1,2, \dots, M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Then apply Leapfrog (either VV or PV) M times in sequence (step i with timestep Δt_i)

$$\begin{bmatrix} x(t_0) \\ \dot{x}(t_0) \end{bmatrix} \xrightarrow{\text{Leapfrog 1}} \begin{bmatrix} x(t_0 + \Delta t_1) \\ \dot{x}(t_0 + \Delta t_1) \end{bmatrix}$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Higher order LEAPFROG $\longrightarrow$ YOSHIDA methods

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t$ $(i = 1,2, \dots, M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Then apply Leapfrog (either VV or PV) M times in sequence (step i with timestep Δt_i)

$$\begin{bmatrix} x(t_0) \\ \dot{x}(t_0) \end{bmatrix} \xrightarrow{\text{Leapfrog 1}} \begin{bmatrix} x(t_0 + \Delta t_1) \\ \dot{x}(t_0 + \Delta t_1) \end{bmatrix} \xrightarrow{\text{Leapfrog 2}} \begin{bmatrix} x(t_0 + \Delta t_1 + \Delta t_2) \\ \dot{x}(t_0 + \Delta t_1 + \Delta t_2) \end{bmatrix}$$

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Higher order LEAPFROG ⟶ YOSHIDA methods

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t$ $(i = 1,2, \dots, M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Then apply Leapfrog (either VV or PV) M times in sequence (step i with timestep Δt_i)

$$\begin{bmatrix} x(t_0) \\ \dot{x}(t_0) \end{bmatrix} \xrightarrow{\text{Leapfrog 1}} \begin{bmatrix} x(t_0 + \Delta t_1) \\ \dot{x}(t_0 + \Delta t_1) \end{bmatrix} \xrightarrow{\text{Leapfrog 2}} \begin{bmatrix} x(t_0 + \Delta t_1 + \Delta t_2) \\ \dot{x}(t_0 + \Delta t_1 + \Delta t_2) \end{bmatrix} \quad \dots \quad \xrightarrow{\text{Leapfrog M}} \begin{bmatrix} x(t_0 + \Delta t) \\ \dot{x}(t_0 + \Delta t) \end{bmatrix}$$

Symplectic methods $\ddot{x}(t) = \mathcal{F}[x(t)]$

Higher order LEAPFROG $\longrightarrow$ YOSHIDA methods

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t$ $(i = 1,2, \dots, M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Then apply Leapfrog (either VV or PV) M times in sequence (step i with timestep Δt_i)

$$\begin{bmatrix} x(t_0) \\ \dot{x}(t_0) \end{bmatrix} \xrightarrow{\text{Leapfrog 1}} \begin{bmatrix} x(t_0 + \Delta t_1) \\ \dot{x}(t_0 + \Delta t_1) \end{bmatrix} \xrightarrow{\text{Leapfrog 2}} \begin{bmatrix} x(t_0 + \Delta t_1 + \Delta t_2) \\ \dot{x}(t_0 + \Delta t_1 + \Delta t_2) \end{bmatrix} \quad \dots \quad \xrightarrow{\text{Leapfrog M}} \begin{bmatrix} x(t_0 + \Delta t) \\ \dot{x}(t_0 + \Delta t) \end{bmatrix}$$

There are specific values of M and ω_i which give accuracy to order $(\Delta t)^4$, $(\Delta t)^6$, $(\Delta t)^8$ ...

Symplectic methods

$$\ddot{x}(t) = \mathcal{F}[x(t)]$$

Higher order LEAPFROG $\longrightarrow$ **YOSHIDA methods**

[H. Yoshida, 1990]

Instead of directly applying a synchronized leapfrog from t_0 to $t_0 + \Delta t$, we can split the timestep Δt in M sub-timesteps $\Delta t_i = \omega_i \Delta t \ (i = 1,2, \dots, M)$ such that $\sum_{i=1}^{M} \omega_i = 1$

Then apply Leapfrog (either VV or PV) M times in sequence (step i with timestep Δt_i)

$$\begin{bmatrix} x(t_0) \\ \dot{x}(t_0) \end{bmatrix} \xrightarrow{\text{Leapfrog 1}} \begin{bmatrix} x(t_0+\Delta t_1) \\ \dot{x}(t_0+\Delta t_1) \end{bmatrix} \xrightarrow{\text{Leapfrog 2}} \begin{bmatrix} x(t_0+\Delta t_1+\Delta t_2) \\ \dot{x}(t_0+\Delta t_1+\Delta t_2) \end{bmatrix} \quad \dots \quad \xrightarrow{\text{Leapfrog M}} \begin{bmatrix} x(t_0+\Delta t) \\ \dot{x}(t_0+\Delta t) \end{bmatrix}$$

Symplectic higher order methods require many steps but no auxiliary variables to store!

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \underline{\dot{x}(t)}]$ given $x(t_0), \dot{x}(t_0)$

For *NON-CONSERVATIVE SYSTEMS* Liouville's theorem does not hold

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \underline{\dot{x}(t)}]$ given $x(t_0), \dot{x}(t_0)$

For *NON-CONSERVATIVE SYSTEMS* Liouville's theorem does not hold

No reason to look for a symplectic method in these cases

Non-symplectic methods

We want to solve $\ddot{x}(t) \; = \; \mathcal{F}[x(t), \underline{\dot{x}(t)}]$ given $x(t_0), \dot{x}(t_0)$

For *NON-CONSERVATIVE SYSTEMS* Liouville's theorem does not hold

No reason to look for a symplectic method in these cases

Runge Kutta methods are a natural choice for these systems
(easily generalized to high orders)

e. g. HYDRODYNAMICS

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

First order Runge-Kutta

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

First order Runge-Kutta (*Euler method*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \; \dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \; \mathcal{F}[x(t_0), \dot{x}(t_0)]$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

First order Runge-Kutta (*Euler method*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \; \dot{x}(t_0)$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \; \mathcal{F}[x(t_0), \dot{x}(t_0)]$$

→ Accurate to order $(\Delta t)^1$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

Second order Runge-Kutta (*Modified Euler*)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

Intermediate step

$$x(t_1) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0), \dot{x}(t_0)]$$

Second order Runge-Kutta (*Modified Euler*)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

Intermediate step

$$x(t_1) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0), \dot{x}(t_0)]$$

Second order Runge-Kutta (*Modified Euler*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \frac{\dot{x}(t_0) + \dot{x}(t_1)}{2}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)] + \mathcal{F}[x(t_1), \dot{x}(t_1)]}{2}$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0), \dot{x}(t_0)$

Intermediate step

$$x(t_1) = x(t_0) + \Delta t\, \dot{x}(t_0)$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t\, \mathcal{F}[x(t_0), \dot{x}(t_0)]$$

Second order Runge-Kutta (*Modified Euler*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \frac{\dot{x}(t_0) + \dot{x}(t_1)}{2}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)] + \mathcal{F}[x(t_1), \dot{x}(t_1)]}{2}$$

→ Accurate to order $(\Delta t)^2$

Non-symplectic methods

We want to solve $\ddot{x}(t) \; = \; \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

Third order Runge-Kutta (*Williamson*)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

Third order Runge-Kutta (*Williamson*)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

$$x(t_2) = x(t_0) + \Delta t \, \frac{-3\,\dot{x}(t_0) + 15\,\dot{x}(t_1)}{16}$$

$$\dot{x}(t_2) = \dot{x}(t_0) + \Delta t \, \frac{-3\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 15\,\mathcal{F}[x(t_1), \dot{x}(t_1)]}{16}$$

Third order Runge-Kutta (*Williamson*)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

$$x(t_2) = x(t_0) + \Delta t \, \frac{-3\,\dot{x}(t_0) + 15\,\dot{x}(t_1)}{16}$$

$$\dot{x}(t_2) = \dot{x}(t_0) + \Delta t \, \frac{-3\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 15\,\mathcal{F}[x(t_1), \dot{x}(t_1)]}{16}$$

Third order Runge-Kutta (*Williamson*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \, \frac{5\,\dot{x}(t_0) + 9\,\dot{x}(t_1) + 16\,\dot{x}(t_2)}{30}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \, \frac{5\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 9\,\mathcal{F}[x(t_1), \dot{x}(t_1)] + 16\,\mathcal{F}[x(t_2), \dot{x}(t_2)]}{30}$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

$$x(t_2) = x(t_0) + \Delta t \frac{-3\,\dot{x}(t_0) + 15\,\dot{x}(t_1)}{16}$$

$$\dot{x}(t_2) = \dot{x}(t_0) + \Delta t \frac{-3\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 15\,\mathcal{F}[x(t_1), \dot{x}(t_1)]}{16}$$

Third order Runge-Kutta (*Williamson*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \frac{5\,\dot{x}(t_0) + 9\,\dot{x}(t_1) + 16\,\dot{x}(t_2)}{30}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \frac{5\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 9\,\mathcal{F}[x(t_1), \dot{x}(t_1)] + 16\,\mathcal{F}[x(t_2), \dot{x}(t_2)]}{30}$$

→ Accurate to order $(\Delta t)^3$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

$$x(t_2) = x(t_0) + \Delta t \frac{-3\,\dot{x}(t_0) + 15\,\dot{x}(t_1)}{16}$$

$$\dot{x}(t_2) = \dot{x}(t_0) + \Delta t \frac{-3\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 15\,\mathcal{F}[x(t_1), \dot{x}(t_1)]}{16}$$

Non-symplectic higher order methods require auxiliary variables...

Third order Runge-Kutta (*Williamson*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \frac{5\,\dot{x}(t_0) + 9\,\dot{x}(t_1) + 16\,\dot{x}(t_2)}{30}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \frac{5\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 9\,\mathcal{F}[x(t_1), \dot{x}(t_1)] + 16\,\mathcal{F}[x(t_2), \dot{x}(t_2)]}{30}$$

→ Accurate to order $(\Delta t)^3$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

... however some schemes require only few extra variables

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

$$x(t_2) = x(t_0) + \Delta t \frac{-3\,\dot{x}(t_0) + 15\,\dot{x}(t_1)}{16}$$

$$\dot{x}(t_2) = \dot{x}(t_0) + \Delta t \frac{-3\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 15\,\mathcal{F}[x(t_1), \dot{x}(t_1)]}{16}$$

Third order Runge-Kutta (*Williamson*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \frac{5\,\dot{x}(t_0) + 9\,\dot{x}(t_1) + 16\,\dot{x}(t_2)}{30}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \frac{5\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 9\,\mathcal{F}[x(t_1), \dot{x}(t_1)] + 16\,\mathcal{F}[x(t_2), \dot{x}(t_2)]}{30}$$

→ Accurate to order $(\Delta t)^3$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

given $x(t_0), \dot{x}(t_0)$

Intermediate steps

$$x(t_1) = x(t_0) + \Delta t \frac{\dot{x}(t_0)}{3}$$

$$\dot{x}(t_1) = \dot{x}(t_0) + \Delta t \frac{\mathcal{F}[x(t_0), \dot{x}(t_0)]}{3}$$

$$x(t_2) = x(t_0) + \Delta t \frac{-3\,\dot{x}(t_0) + 15\,\dot{x}(t_1)}{16}$$

$$\dot{x}(t_2) = \dot{x}(t_0) + \Delta t \frac{-3\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 15\,\mathcal{F}[x(t_1), \dot{x}(t_1)]}{16}$$

Low-storage

Third order Runge-Kutta (*Williamson*)

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \frac{5\,\dot{x}(t_0) + 9\,\dot{x}(t_1) + 16\,\dot{x}(t_2)}{30}$$

$$\dot{x}(t_0 + \Delta t) = \dot{x}(t_0) + \Delta t \frac{5\,\mathcal{F}[x(t_0), \dot{x}(t_0)] + 9\,\mathcal{F}[x(t_1), \dot{x}(t_1)] + 16\,\mathcal{F}[x(t_2), \dot{x}(t_2)]}{30}$$

→ Accurate to order $(\Delta t)^3$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) $\longleftarrow$ Low-storage

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0)$, $\dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0)$, $\dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 0

$$\left.\begin{aligned} x^{(0)} &= x(t_0) \\ \dot{x}^{(0)} &= \dot{x}(t_0) \end{aligned}\right] \text{ degrees of freedom}$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 0

$$\left.\begin{aligned} x^{(0)} &= x(t_0) \\ \dot{x}^{(0)} &= \dot{x}(t_0) \end{aligned}\right] \text{ degrees of freedom}$$

$$\left.\begin{aligned} \delta x^{(0)} &= \Delta t\, \dot{x}^{(0)} \\ \delta \dot{x}^{(0)} &= \Delta t\, \mathcal{F}[x^{(0)}, \dot{x}^{(0)}] \end{aligned}\right] \text{ extra variables to store}$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 0

$$x^{(0)} = x(t_0)$$

$$\dot{x}^{(0)} = \dot{x}(t_0)$$

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(0)} = \Delta t\ \dot{x}^{(0)}$$

$$\delta \dot{x}^{(0)} = \Delta t\ \mathcal{F}[x^{(0)}, \dot{x}^{(0)}]$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) $\longleftarrow$ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 0

$$x^{(0)} = x(t_0)$$

$$\dot{x}^{(0)} = \dot{x}(t_0)$$

$$\delta x^{(0)} = \Delta t\ \dot{x}^{(0)}$$

$$\delta \dot{x}^{(0)} = \Delta t\ \mathcal{F}[x^{(0)}, \dot{x}^{(0)}]$$

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(1)} = -\frac{5}{9}\delta x^{(0)} + \Delta t\ \dot{x}^{(1)}$$

$$\delta \dot{x}^{(1)} = -\frac{5}{9}\delta \dot{x}^{(0)} + \Delta t\ \mathcal{F}\left[x^{(1)}, \dot{x}^{(1)}\right]$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 0

$$x^{(0)} = x(t_0)$$

$$\dot{x}^{(0)} = \dot{x}(t_0)$$

$$\delta x^{(0)} = \Delta t\, \dot{x}^{(0)}$$

$$\delta \dot{x}^{(0)} = \Delta t\, \mathcal{F}[x^{(0)}, \dot{x}^{(0)}]$$

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(1)} = -\frac{5}{9}\delta x^{(0)} + \Delta t\, \dot{x}^{(1)}$$

$$\delta \dot{x}^{(1)} = -\frac{5}{9}\delta \dot{x}^{(0)} + \Delta t\, \mathcal{F}\left[x^{(1)}, \dot{x}^{(1)}\right]$$

STEP 2

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) $\longleftarrow$ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 0

$$\cancel{x^{(0)} = x(t_0)}$$

$$\cancel{\dot{x}^{(0)} = \dot{x}(t_0)}$$

$$\cancel{\delta x^{(0)} = \Delta t\, \dot{x}^{(0)}}$$

$$\cancel{\delta \dot{x}^{(0)} = \Delta t\, \mathcal{F}[x^{(0)}, \dot{x}^{(0)}]}$$

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(1)} = -\frac{5}{9}\delta x^{(0)} + \Delta t\, \dot{x}^{(1)}$$

$$\delta \dot{x}^{(1)} = -\frac{5}{9}\delta \dot{x}^{(0)} + \Delta t\, \mathcal{F}\left[x^{(1)}, \dot{x}^{(1)}\right]$$

STEP 2

For this step we only need
$x^{(1)}, \dot{x}^{(1)}, \delta x^{(1)}, \delta \dot{x}^{(1)}$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(1)} = -\frac{5}{9}\delta x^{(0)} + \Delta t\ \dot{x}^{(1)}$$

$$\delta \dot{x}^{(1)} = -\frac{5}{9}\delta \dot{x}^{(0)} + \Delta t\ \mathcal{F}\left[x^{(1)}, \dot{x}^{(1)}\right]$$

STEP 2

$$x^{(2)} = x^{(1)} + \frac{15}{16}\delta x^{(1)}$$

$$\dot{x}^{(2)} = \dot{x}^{(1)} + \frac{15}{16}\delta \dot{x}^{(1)}$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) $\longleftarrow$ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(1)} = -\frac{5}{9}\delta x^{(0)} + \Delta t\, \dot{x}^{(1)}$$

$$\delta \dot{x}^{(1)} = -\frac{5}{9}\delta \dot{x}^{(0)} + \Delta t\, \mathcal{F}\left[x^{(1)}, \dot{x}^{(1)}\right]$$

STEP 2

$$x^{(2)} = x^{(1)} + \frac{15}{16}\delta x^{(1)}$$

$$\dot{x}^{(2)} = \dot{x}^{(1)} + \frac{15}{16}\delta \dot{x}^{(1)}$$

$$\delta x^{(2)} = -\frac{153}{128}\delta x^{(1)} + \Delta t\, \dot{x}^{(2)}$$

$$\delta \dot{x}^{(2)} = -\frac{153}{128}\delta \dot{x}^{(1)} + \Delta t\, \mathcal{F}\left[x^{(2)}, \dot{x}^{(2)}\right]$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 1

$$x^{(1)} = x^{(0)} + \frac{1}{3}\delta x^{(0)}$$

$$\dot{x}^{(1)} = \dot{x}^{(0)} + \frac{1}{3}\delta \dot{x}^{(0)}$$

$$\delta x^{(1)} = -\frac{5}{9}\delta x^{(0)} + \Delta t\, \dot{x}^{(1)}$$

$$\delta \dot{x}^{(1)} = -\frac{5}{9}\delta \dot{x}^{(0)} + \Delta t\, \mathcal{F}\left[x^{(1)}, \dot{x}^{(1)}\right]$$

STEP 2

$$x^{(2)} = x^{(1)} + \frac{15}{16}\delta x^{(1)}$$

$$\dot{x}^{(2)} = \dot{x}^{(1)} + \frac{15}{16}\delta \dot{x}^{(1)}$$

$$\delta x^{(2)} = -\frac{153}{128}\delta x^{(1)} + \Delta t\, \dot{x}^{(2)}$$

$$\delta \dot{x}^{(2)} = -\frac{153}{128}\delta \dot{x}^{(1)} + \Delta t\, \mathcal{F}\left[x^{(2)}, \dot{x}^{(2)}\right]$$

For STEP 3 we only need

$x^{(2)}, \dot{x}^{(2)}, \delta x^{(2)}, \delta \dot{x}^{(2)}$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) $\longleftarrow$ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 2

$$x^{(2)} = x^{(1)} + \frac{15}{16}\delta x^{(1)}$$

$$\dot{x}^{(2)} = \dot{x}^{(1)} + \frac{15}{16}\delta \dot{x}^{(1)}$$

$$\delta x^{(2)} = -\frac{153}{128}\delta x^{(1)} + \Delta t\, \dot{x}^{(2)}$$

$$\delta \dot{x}^{(2)} = -\frac{153}{128}\delta \dot{x}^{(1)} + \Delta t\, \mathcal{F}\left[x^{(2)}, \dot{x}^{(2)}\right]$$

STEP 3

$$x^{(3)} = x^{(2)} + \frac{8}{15}\delta x^{(2)}$$

$$\dot{x}^{(3)} = \dot{x}^{(2)} + \frac{8}{15}\delta \dot{x}^{(2)}$$

Non-symplectic methods

We want to solve $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$ given $x(t_0),\ \dot{x}(t_0)$

Third order Runge-Kutta (*Williamson*) ⟵ Low-storage

We only need to store one additional variable per degree of freedom (x and $\dot{x}$)

STEP 2

$$x^{(2)} = x^{(1)} + \frac{15}{16}\delta x^{(1)}$$

$$\dot{x}^{(2)} = \dot{x}^{(1)} + \frac{15}{16}\delta \dot{x}^{(1)}$$

$$\delta x^{(2)} = -\frac{153}{128}\delta x^{(1)} + \Delta t\ \dot{x}^{(2)}$$

$$\delta \dot{x}^{(2)} = -\frac{153}{128}\delta \dot{x}^{(1)} + \Delta t\ \mathcal{F}\left[x^{(2)}, \dot{x}^{(2)}\right]$$

STEP 3

$$x^{(3)} = x^{(2)} + \frac{8}{15}\delta x^{(2)} \qquad \equiv x(t_0 + \Delta t)$$

$$\dot{x}^{(3)} = \dot{x}^{(2)} + \frac{8}{15}\delta \dot{x}^{(2)} \qquad \equiv \dot{x}(t_0 + \Delta t)$$

Leapfrog vs Runge Kutta

EXAMPLE: *1D DAMPED OSCILLATOR*

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

Leapfrog vs Runge Kutta

EXAMPLE: *1D DAMPED OSCILLATOR*

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

The system has the following analytical solution ($\gamma^2 < \omega^2$)

$$x(t) = e^{-\gamma t}\left[A\cos\sqrt{\omega^2-\gamma^2}\,t + B\sin\sqrt{\omega^2-\gamma^2}\,t\right]$$

Leapfrog vs Runge Kutta

EXAMPLE: *1D DAMPED OSCILLATOR*

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

The system has the following analytical solution ($\gamma^2 < \omega^2$)

$$x(t) = e^{-\gamma t}\left[A\cos\sqrt{\omega^2-\gamma^2}\,t + B\sin\sqrt{\omega^2-\gamma^2}\,t\right]$$

We can adapt a second order Velocity Verlet (VV2)...

$$\dot{x}\left(t_0+\frac{\Delta t}{2}\right) = \dot{x}(t_0) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0),\dot{x}(t_0)]$$

$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}\left(t_0+\frac{\Delta t}{2}\right)$$

$$\dot{x}(t_0+\Delta t) = \dot{x}\left(t_0+\frac{\Delta t}{2}\right) + \frac{\Delta t}{2}\,\mathcal{F}\left[x(t_0+\Delta t),\dot{x}\left(t_0+\frac{\Delta t}{2}\right)\right]$$

Leapfrog vs Runge Kutta

EXAMPLE: *1D DAMPED OSCILLATOR* $\ddot{x}(t) = -\omega^2 x(t) - 2\gamma\dot{x}(t)$

The system has the following analytical solution ($\gamma^2 < \omega^2$)

$$x(t) = e^{-\gamma t}\left[A\cos\sqrt{\omega^2-\gamma^2}\,t + B\sin\sqrt{\omega^2-\gamma^2}\,t\right]$$

We can adapt a second order Velocity Verlet (VV2)...

$$\dot{x}\left(t_0+\frac{\Delta t}{2}\right) = \dot{x}(t_0) + \frac{\Delta t}{2}\,\mathcal{F}[x(t_0),\dot{x}(t_0)]$$

$$x(t_0+\Delta t) = x(t_0) + \Delta t\,\dot{x}\left(t_0+\frac{\Delta t}{2}\right)$$

$$\dot{x}(t_0+\Delta t) = \dot{x}\left(t_0+\frac{\Delta t}{2}\right) + \frac{\Delta t}{2}\,\mathcal{F}\left[x(t_0+\Delta t),\dot{x}\left(t_0+\frac{\Delta t}{2}\right)\right]$$

... and compare it with the result given by a second order Runge-Kutta (RK2 – Modified Euler)

Leapfrog vs Runge Kutta

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

EXAMPLE: *1D DAMPED OSCILLATOR*

$x(t_0) = 1,\ \dot{x}(t_0) = 0$

$\omega^2 = 1, \gamma = 0.1$

Pseudoperiod $T = \frac{2\pi}{\sqrt{\omega^2 - \gamma^2}}$

$\Delta t = T/100$

Leapfrog vs Runge Kutta

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

EXAMPLE: *1D DAMPED OSCILLATOR*

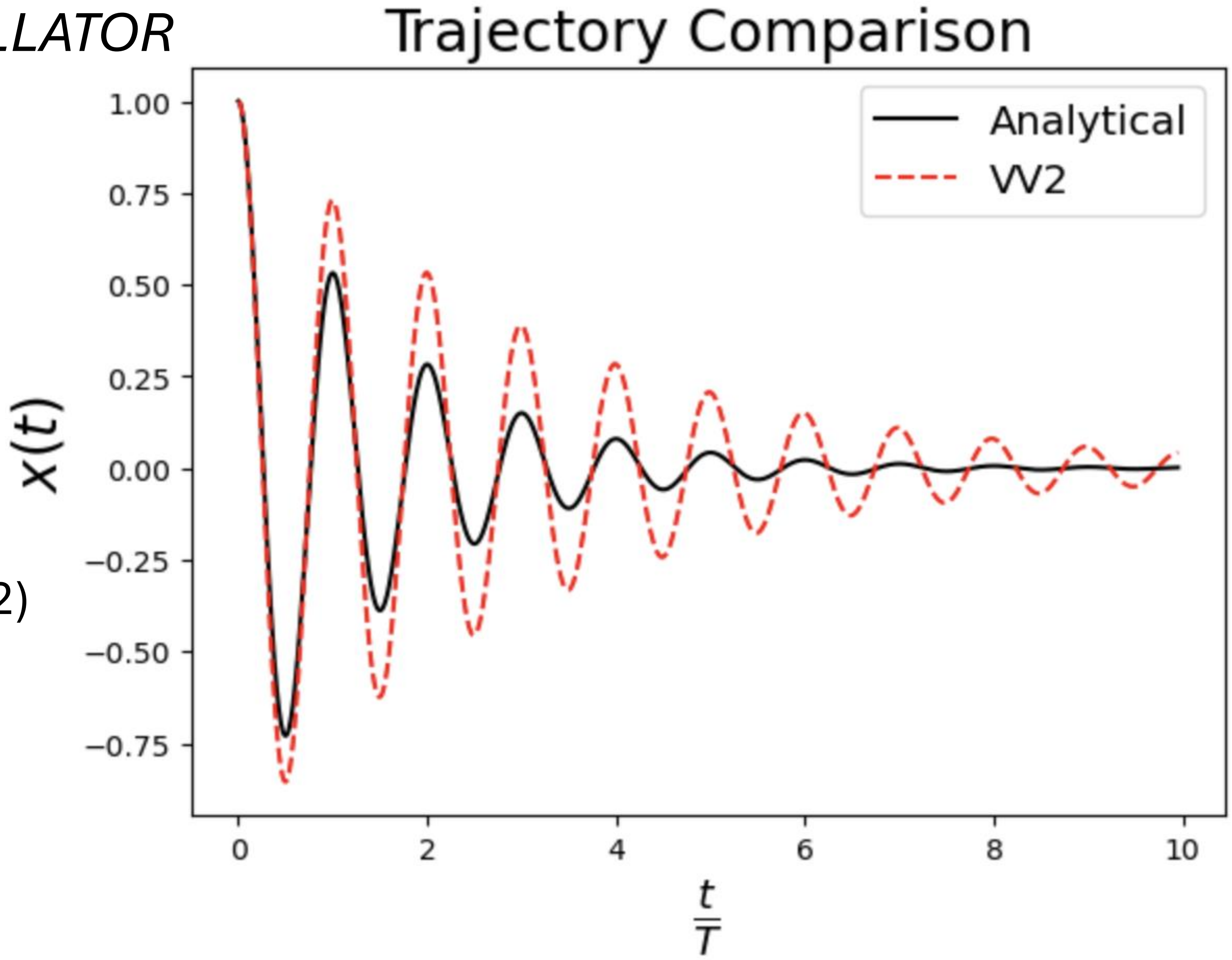
$x(t_0) = 1,\ \dot{x}(t_0) = 0$

$\omega^2 = 1, \gamma = 0.1$

Pseudoperiod $T = \frac{2\pi}{\sqrt{\omega^2 - \gamma^2}}$

$\Delta t = T/100$

Second-order Velocity Verlet (VV2) does not reproduce correctly the analytical solution



Leapfrog vs Runge Kutta

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

EXAMPLE: *1D DAMPED OSCILLATOR*

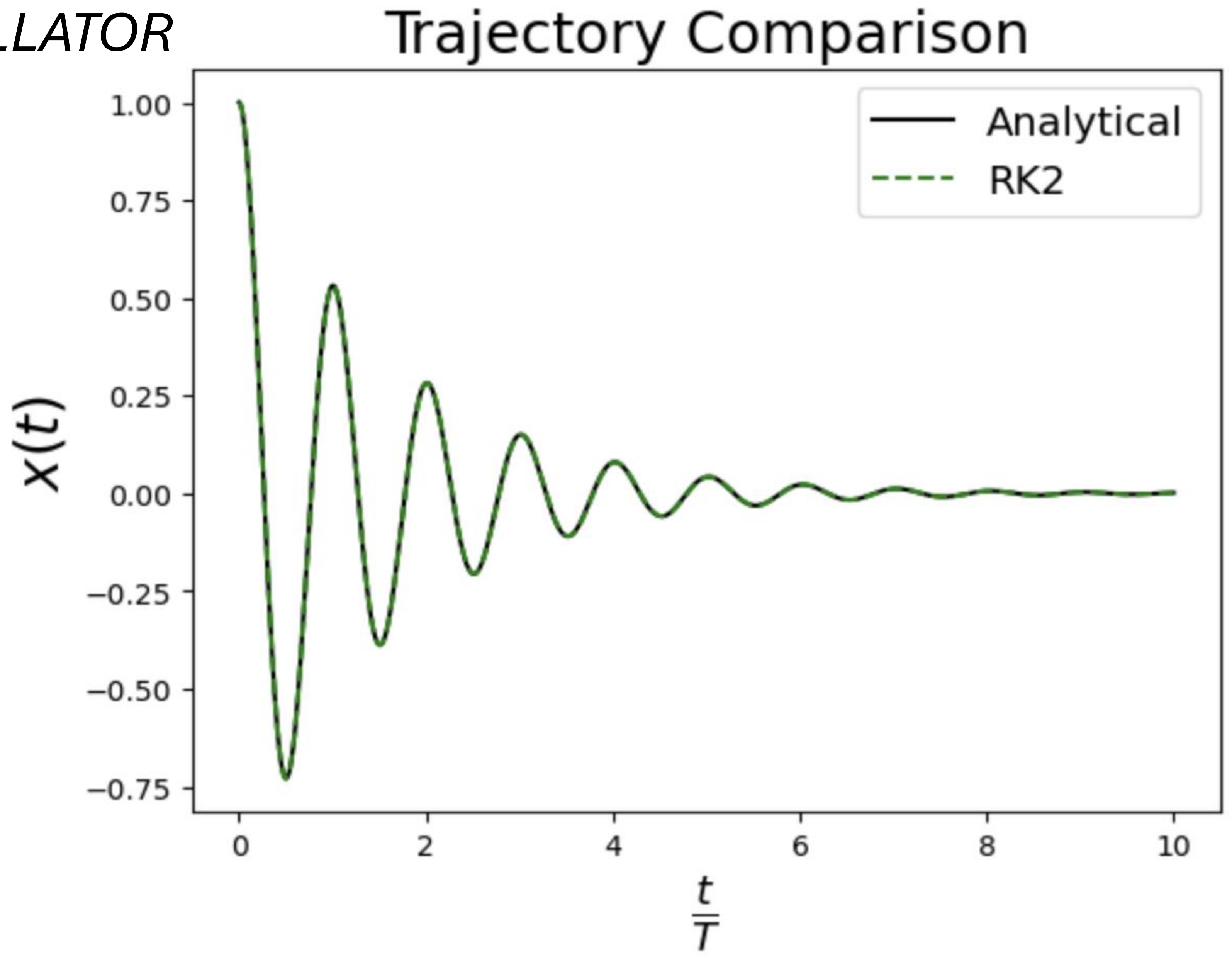
$x(t_0) = 1,\ \dot{x}(t_0) = 0$

$\omega^2 = 1, \gamma = 0.1$

Pseudoperiod $T = \frac{2\pi}{\sqrt{\omega^2 - \gamma^2}}$

$\Delta t = T/100$

Second-order Runge-Kutta (RK2) follows correctly the damping of the analytical solution



Numerical schemes for Partial Differential Equations (PDEs)

How to deal with PDEs? $\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t), \partial_x f(x,t), \ldots\right]$

Numerical schemes for Partial Differential Equations (PDEs)

How to deal with PDEs? $\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t), \partial_x f(x,t), \ldots\right]$

We have shown timestepping schemes for the case

$$\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t)\right]$$

Numerical schemes for Partial Differential Equations (PDEs)

How to deal with PDEs? $\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t), \partial_x f(x,t), \dots\right]$

We have shown timestepping schemes for the case

$$\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t)\right]$$

However, once all derivative operators are discretized
(i.e. written as a linear superposition of field values at difference lattice points),
PDEs become ODEs *in the lattice*

Numerical schemes for Partial Differential Equations (PDEs)

How to deal with PDEs? $\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t), \partial_x f(x,t), \ldots\right]$

We have shown timestepping schemes for the case

$$\ddot{f}(x,t) = \mathcal{F}\left[f(x,t), \dot{f}(x,t)\right]$$

However, once all derivative operators are discretized
(i.e. written as a linear superposition of field values at difference lattice points),
PDEs become ODEs *in the lattice*

How to discretize derivative operators? $\longrightarrow$ see Part II

Conclusions of part I

Conclusions of part I

- Symplectic methods are a natural approach for conservative systems $\ddot{x}(t) = \mathcal{F}[x(t)]$

Conclusions of part I

- Symplectic methods are a natural approach for conservative systems $\ddot{x}(t) = \mathcal{F}[x(t)]$

- Runge-Kutta methods are a natural approach for non-conservative systems $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

Conclusions of part I

- Symplectic methods are a natural approach for conservative systems $\ddot{x}(t) = \mathcal{F}[x(t)]$

- Runge-Kutta methods are a natural approach for non-conservative systems $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

- What is the best algorithm depends on the physical problem!

Conclusions of part I

- Symplectic methods are a natural approach for conservative systems $\ddot{x}(t) = \mathcal{F}[x(t)]$

- Runge-Kutta methods are a natural approach for non-conservative systems $\ddot{x}(t) = \mathcal{F}[x(t), \dot{x}(t)]$

- What is the best algorithm depends on the physical problem!

THANKS FOR YOUR ATTENTION!