# CSE 291 (SP24) Physics Simulation Introduction


**Albert Chern**

UC San Diego
JACOBS SCHOOL OF ENGINEERING

# Course information

**Physics Simulation**

- **Instructor:** Albert Chern
- **TA:** Chad Mckell
- **Course website:** https://cseweb.ucsd.edu/~alchern/teaching/cse291_sp24/
- We use **Piazza** and **Gradescope**

# Course information

**Physics Simulation**

- **Goal:** Mathematical principles behind simulation tasks
  Hands-on experience with physics-based animations
- **Applications:** Computer animation, scientific computing
  classical mechanics, theory abstraction
- **Grade:** HW0–4 (written and mini-project)
- **Collaboration:** Final submissions should be individual work, but we encourage you to study the math and solve the problems together!

# Course information

**Physics Simulation**

- **Prerequisites:**
  - Linear algebra, multivariable calculus, elementary physics
  - Using one programming platform with visualization that is capable of using/importing sparse matrix library
    - e.g. graphics software: Houdini, Blender, Unity
    - e.g. C++, Python, MATLAB, Javascript+WebGL
- **What tools can you use:**
  - Build your own solver from lower level (you can use built-in geometry processing functions) Don't use a full-blown built-in simulation solver.

# Syllabus

| Week | Tuesday | Thursday |
|---|---|---|
| 1 | **4/2: Lecture prerecorded**<br>Introduction<br>• | **4/4: Lecture prerecorded**<br>Ordinary Differential Equations<br>• |
| 2 | **4/9:** Dimensional Analysis | **4/11:** Differentials and gradients<br>• **HW0 due (miniproject)** |
| 3 | **4/16:** Calculus of variations | **4/18:** Least action principle<br>• **HW1 due (written)** |
| 4 | **4/23:** Constrained systems | **4/25:** Rigid body motion |
| 5 | **4/30:** Geodesic equation | **5/2:** Incremental potential<br>• **HW2 due (written part 2.1, 2.2)** |
| 6 | **5/7:** Tensors | **5/9:** Tensors<br>• **HW2 due (miniproject part 2.3)** |
| 7 | **5/14:** Elasticity | **5/16:** Elasticity |
| 8 | **5/21: Lecture prerecorded** Fluids | **5/23: Lecture prerecorded** Fluids |
| 9 | **5/28:** Fluids (numerics)<br>• **HW3 due (miniproject)** | **5/30:** Fluids |
| 10 | **6/4:** Hamiltonian mechanics | **6/6: No class (instructor unavailable)** |
| Final | **6/11: No class**<br>• **HW4 due (miniproject)** | **6/13: No class** |
| | **Summer break** | |

# Simulation, Physics, Math

- Simulation, Physics, Math
- Getting started: F = ma
- Solve ODEs numerically

# Physics simulation

- In computational physics, engineering, computer graphics,…
- Generate computer-generated data that mimic that we would observe in the physical world.
- Why?
  - Make predictions, conduct virtual experiments
  - Believable visual effects
- How?

# Physics simulation

- How?
  - **Mathematical modeling**

    *Turn physical phenomena into mathematical equations.*

    *(What are the variables? What are the laws of physics)*
  - **Analysis**

    *Get a general idea of how the solution should behave.*

    *(Is the problem well-posed?)*
  - **Computation**

    *Solve (approximate) solutions analytically or numerically.*

# Physics simulation

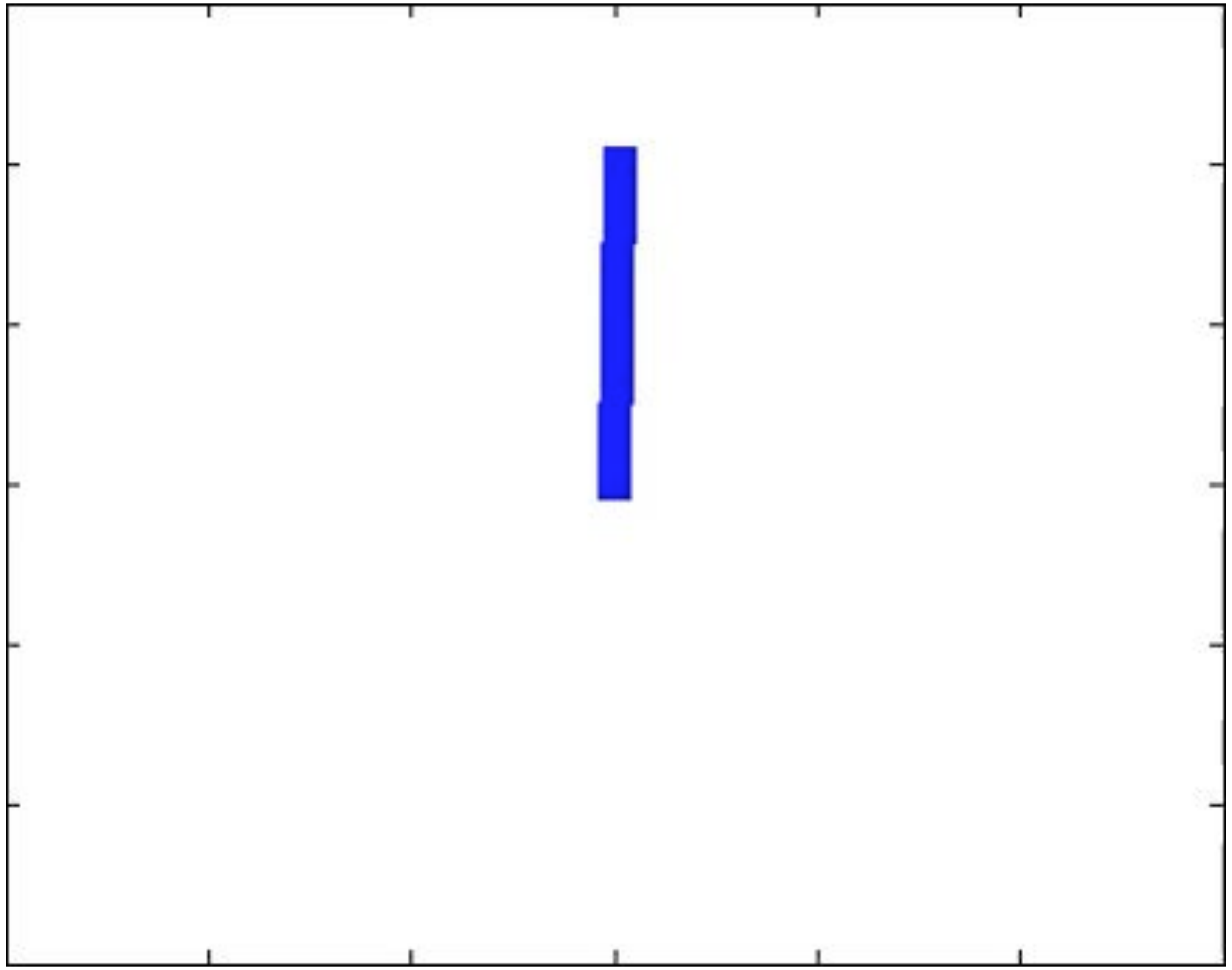
We will focus on **general principles**

- Dimensional analysis
- Least action principle
- Incremental potential formulation
- Constitutive modeling in continuum mechanics

Systems we will cover

- Small mechanical systems
- Rigid body
- Constrained system (linkage, robotics, collision and contact)
- Elastic body
- Fluids

# Physics simulation



# Physics simulation


Youtube “Soft Body Tetris [01]” by ImbaPixel

https://youtu.be/rm44SV8xUDo

Nabizadeh, Wang, Ramamoorthi, C.
Covector Fluids
2022

# Getting started: F=ma

- Simulation, Physics, Math
- Getting started: F = ma
- Solve ODEs numerically

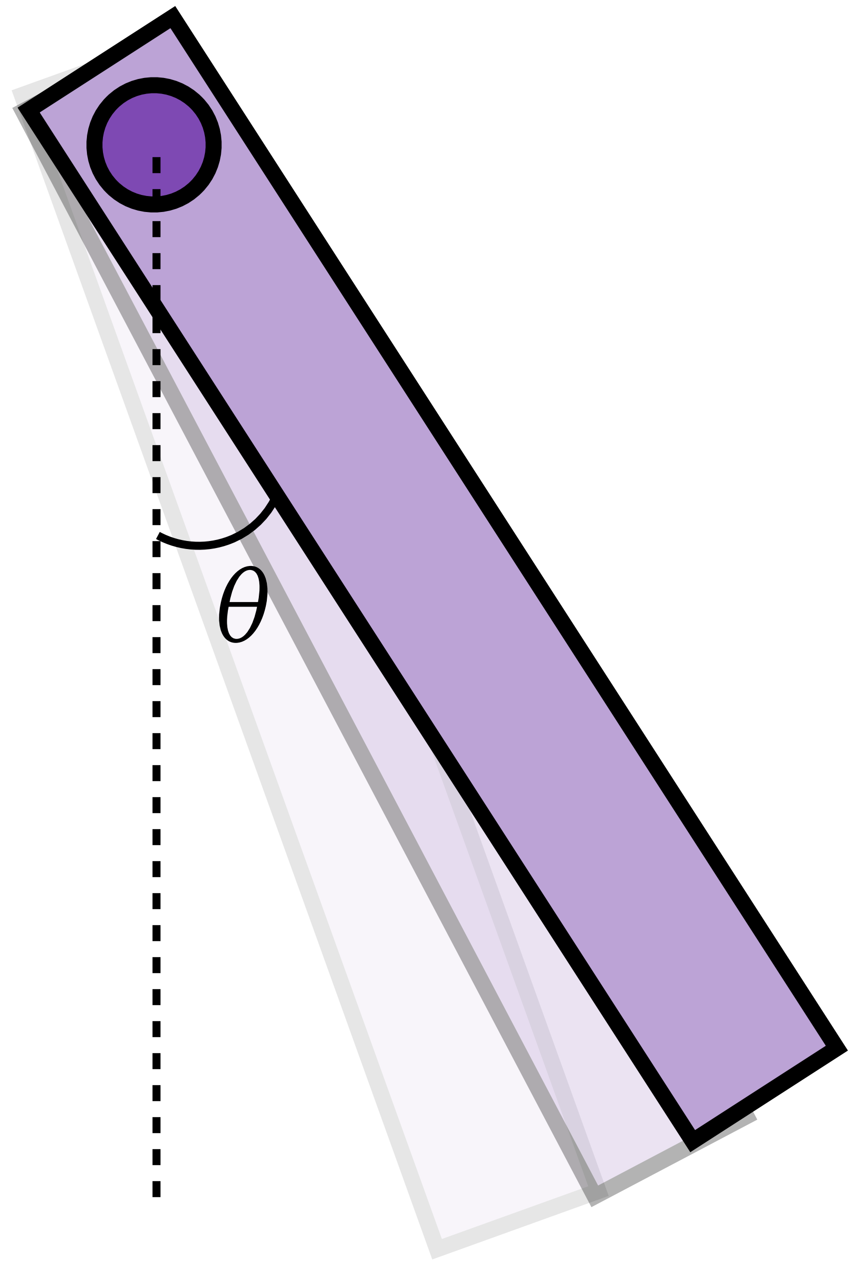
# Physics based motion

**Exercise 0.1 — 5pt.** Using your favorite program to produce an animation of a simple physical system. It could be a pendulum motion like demonstrated in the lecture, or other system.

(a) Upload a video of your result.

(b) Upload a written document that briefly explains the system. (Include the equation of motion, an explanation of what each variable in the equation means, and what the time stepping algorithm looks like.)

(c) Upload the source file(s) (for example .zip). ■



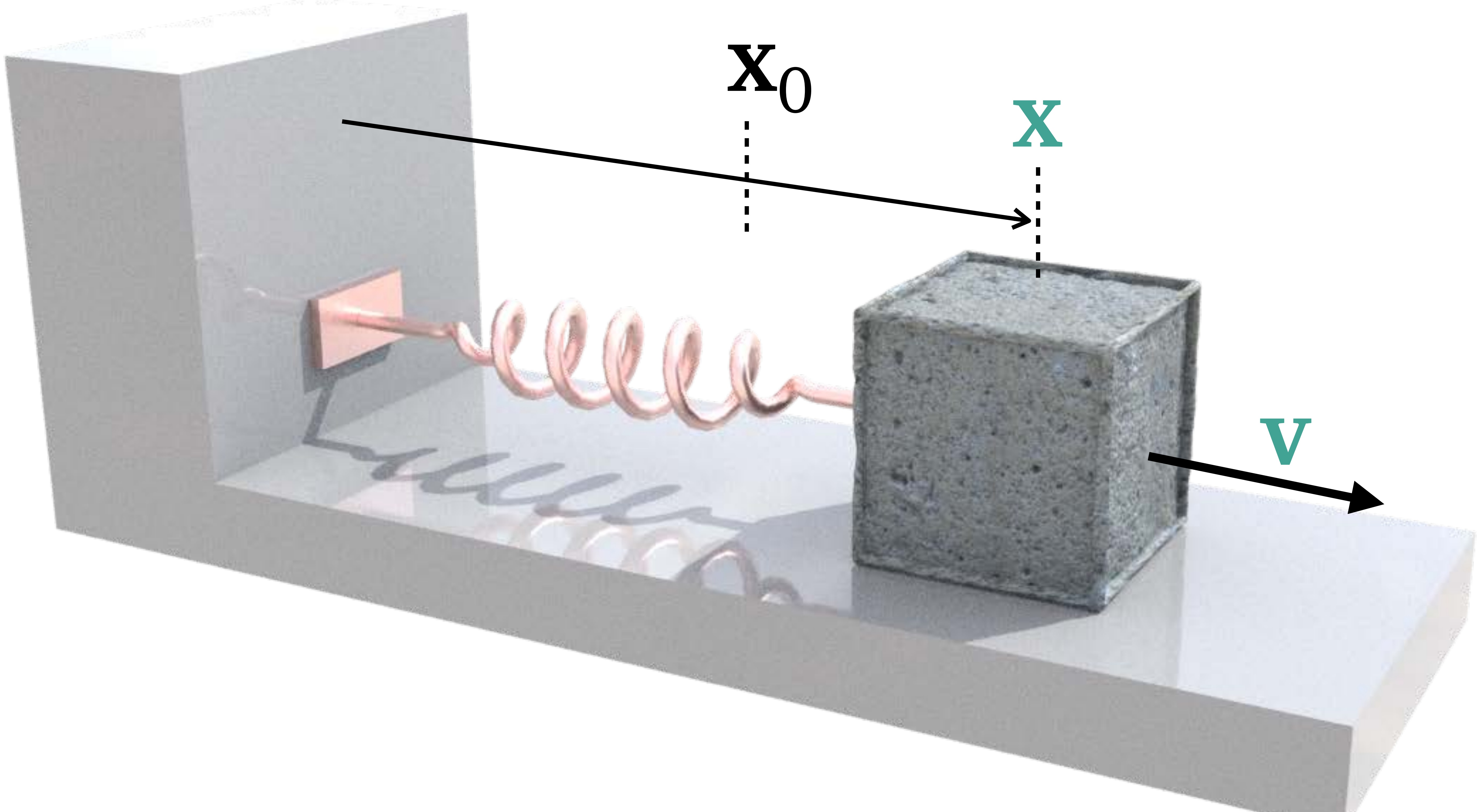
# Physics based motion

Rough idea:

- Position of each object is governed by Newton’s law of motion
- Rate of change of position is called velocity
- Rate of change of velocity is called acceleration
- Model “force” as a function of position and velocity
- Newton’s law of motion: Mass x acceleration = force

# Example

- Animate an object attached to a spring
- Identify the moving position: **x**
- Associated velocity **v**

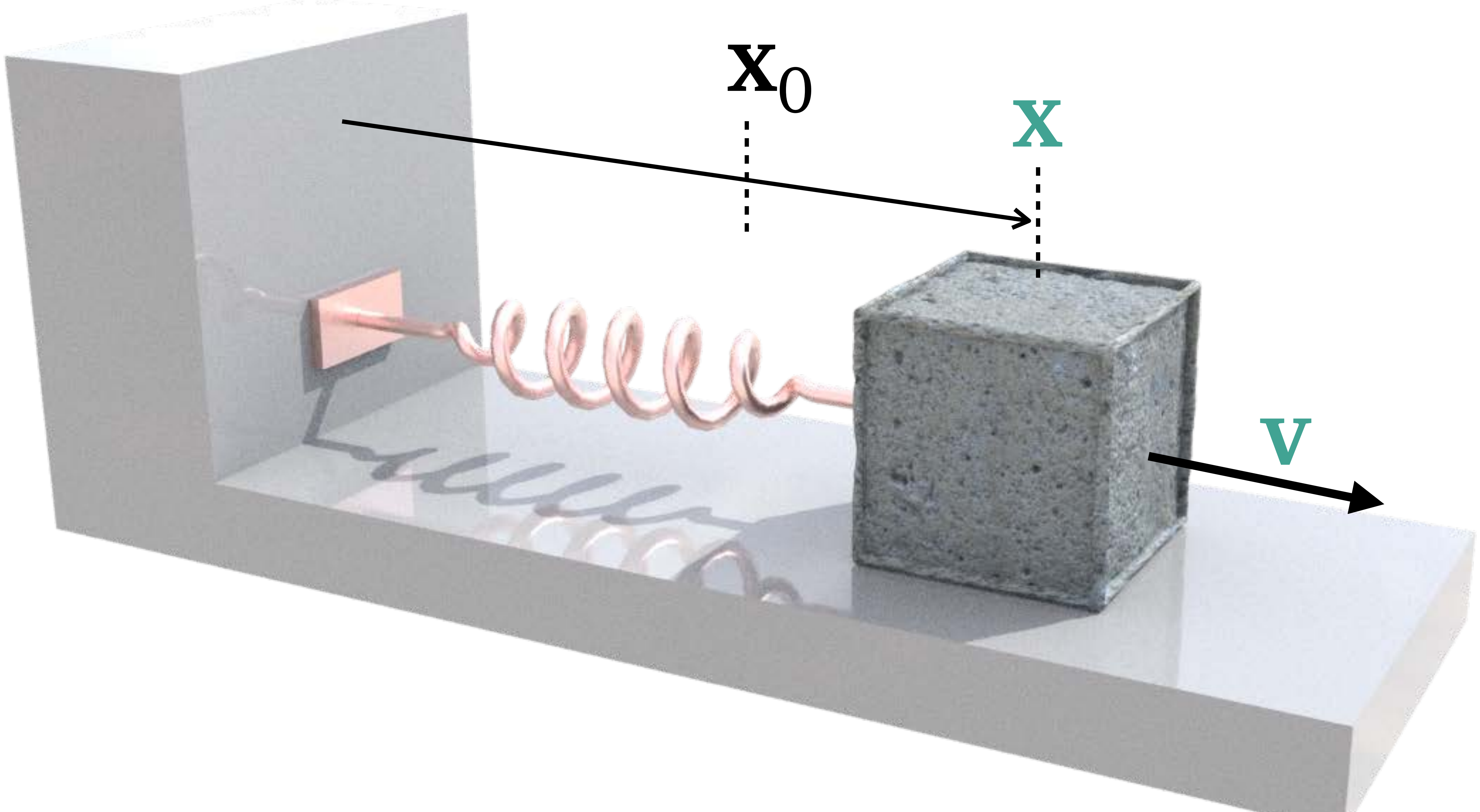


# Example

- Force

$$\mathbf{f}(\mathbf{x},\mathbf{v}) = -k(\mathbf{x}-\boxed{\mathbf{x}_0}) - \mu\mathbf{v}$$

*rest position*

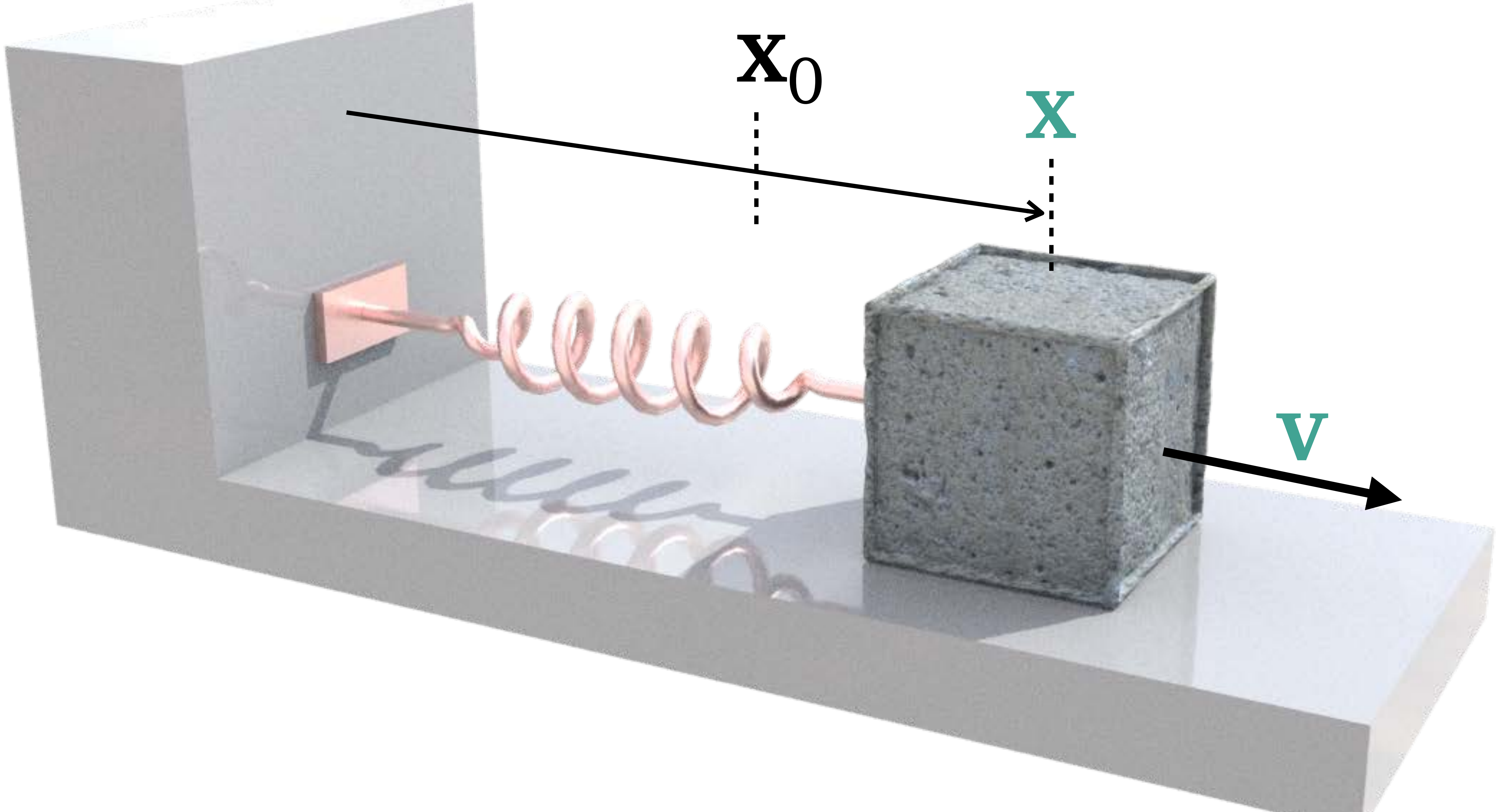


# Example

- Force

$$\mathbf{f}(\mathbf{x}, \mathbf{v}) = -k(\mathbf{x} - \mathbf{x}_0) - \mu\mathbf{v}$$

*stiffness of spring*

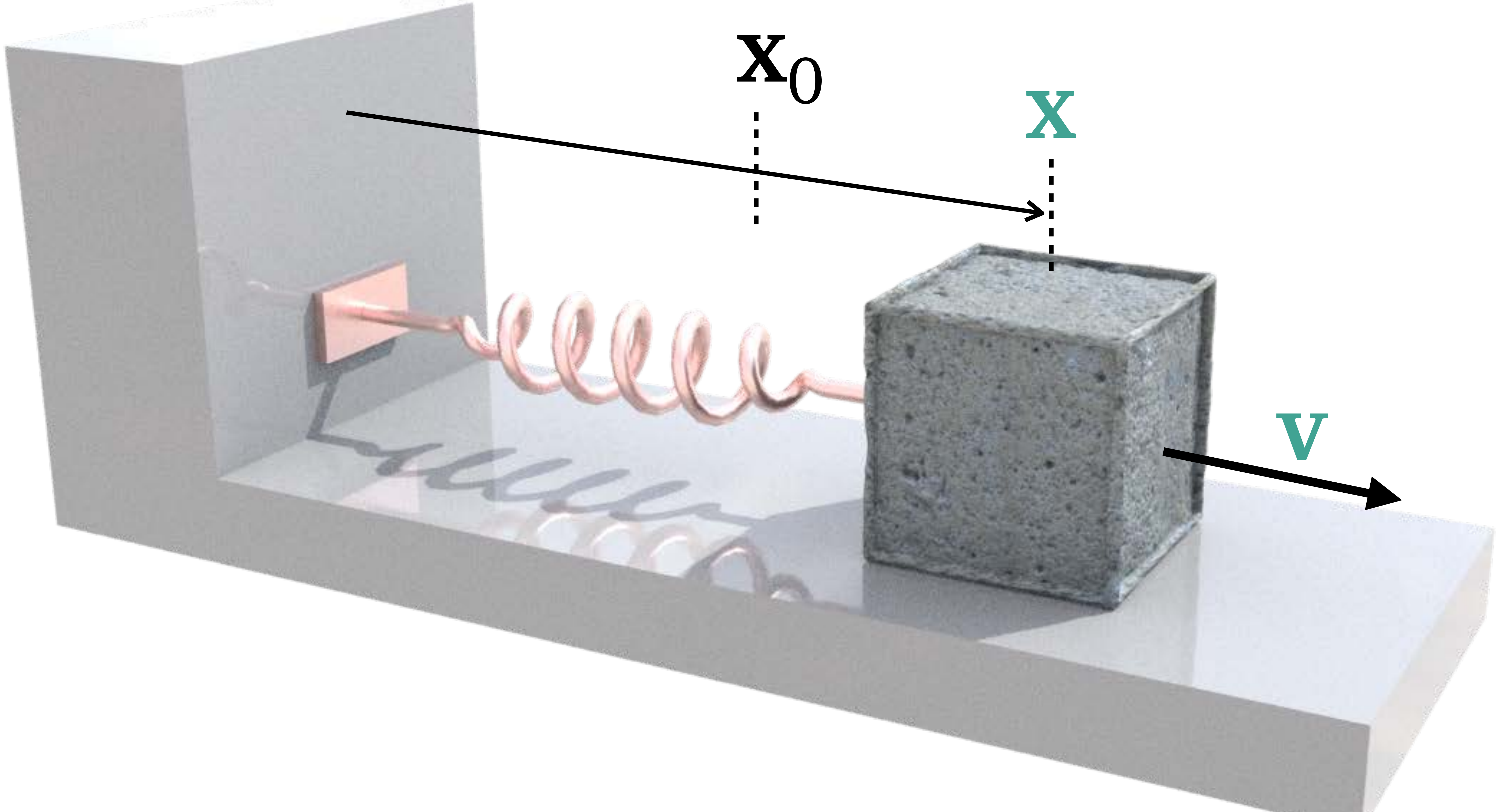


# Example

- Force

$$\mathbf{f}(\mathbf{x}, \mathbf{v}) = -k(\mathbf{x} - \mathbf{x}_0) - \mu \mathbf{v}$$

*friction*



# Example

- Force

$$\mathbf{f}(\mathbf{x},\mathbf{v}) = -k(\mathbf{x}-\mathbf{x}_0) - \mu\mathbf{v}$$

- Equations of motion

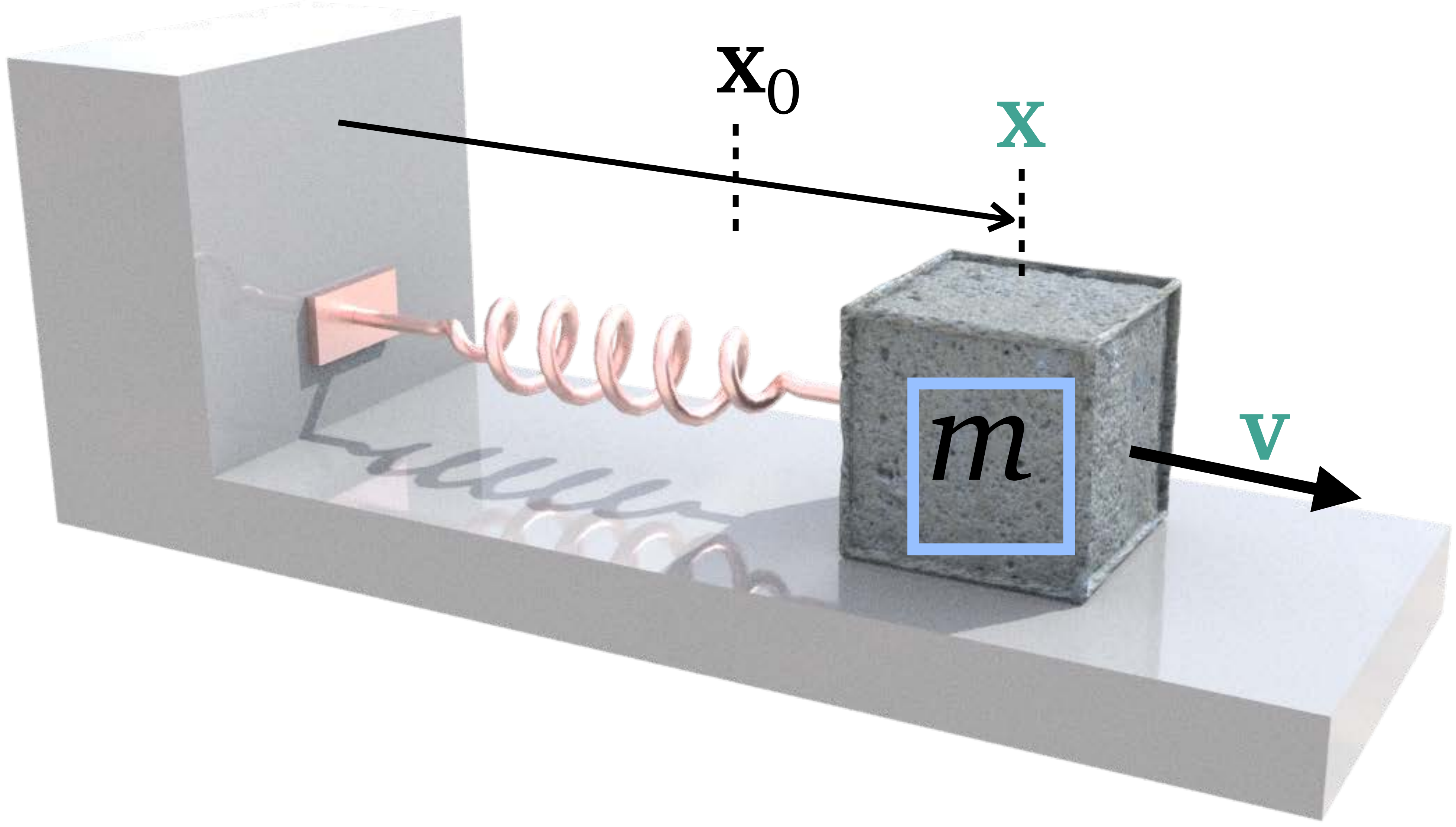
$$\frac{d\mathbf{x}}{dt} = \mathbf{v}$$

relationship between position and velocity

$$\frac{d\mathbf{v}}{dt} = \frac{1}{m}\mathbf{f}(\mathbf{x},\mathbf{v})$$

relationship between acceleration and force

*mass*



# Example

- Force

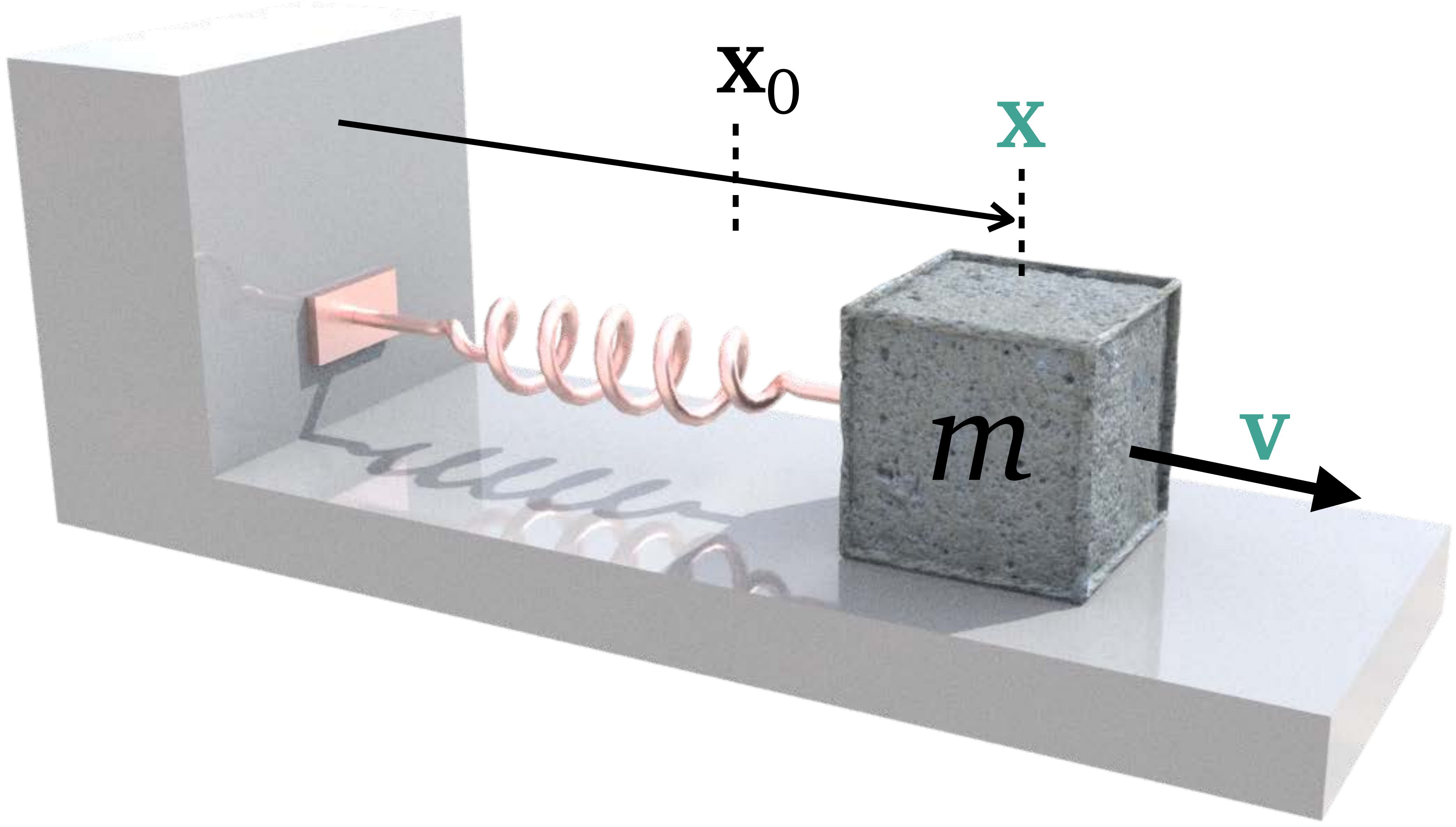
$$\mathbf{f}(\mathbf{x},\mathbf{v}) = -k(\mathbf{x}-\mathbf{x}_0) - \mu\mathbf{v}$$

- Equations of motion

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}$$

$$\frac{d\mathbf{v}}{dt} = \frac{1}{m}\mathbf{f}(\mathbf{x},\mathbf{v})$$

$$= -\frac{k}{m}(\mathbf{x}-\mathbf{x}_0) - \frac{\mu}{m}\mathbf{v}$$



# Example

- We use the overhead dots to indicate time derivatives

$$\frac{d\mathbf{x}(t)}{dt} = \dot{\mathbf{x}}(t) \qquad \frac{d^2\mathbf{x}(t)}{dt^2} = \ddot{\mathbf{x}}(t)$$

# Example

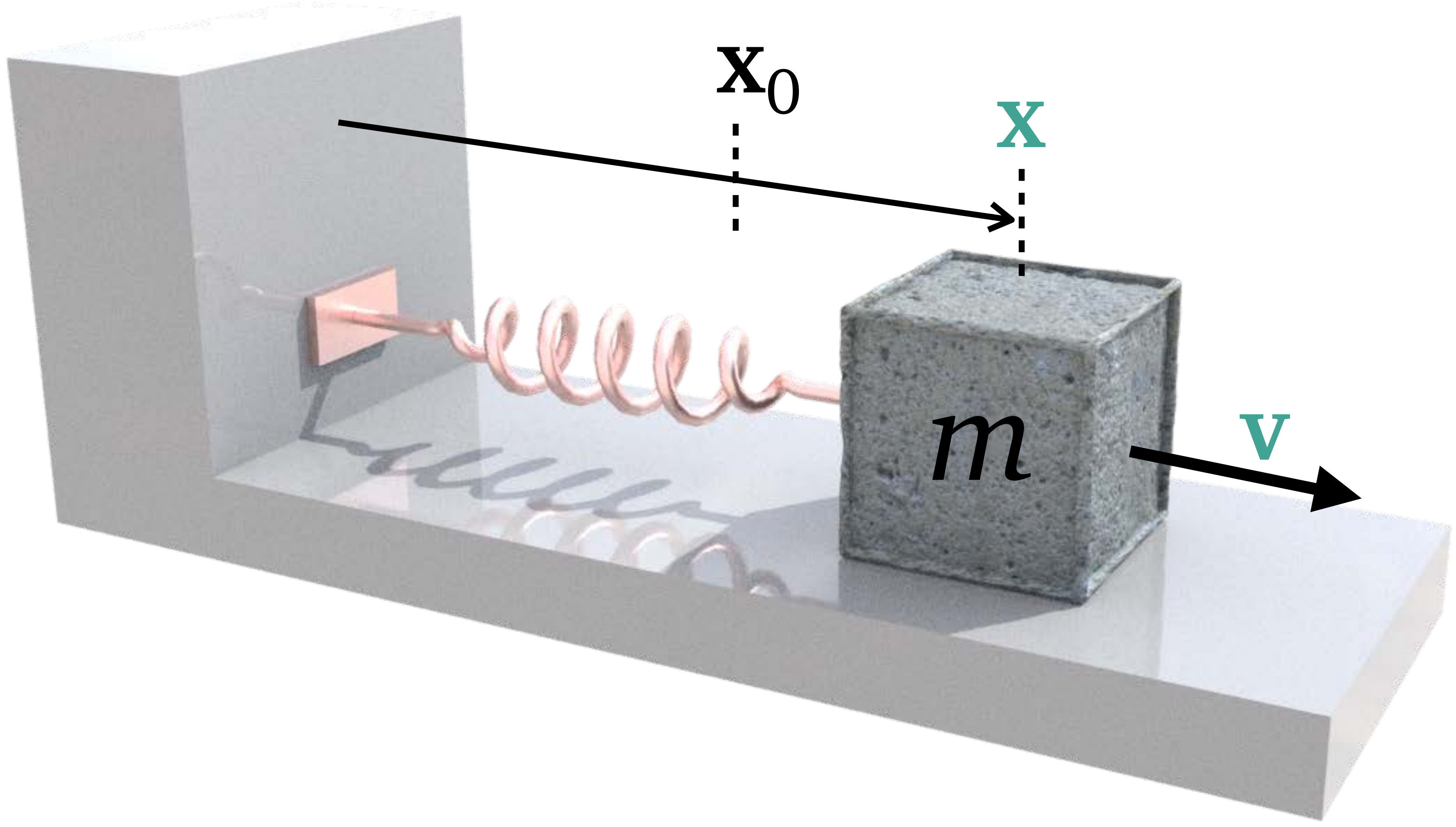
- Equations of motion

$$\dot{\mathbf{x}} = \mathbf{v}$$

$$\dot{\mathbf{v}} = -\frac{k}{m}(\mathbf{x} - \mathbf{x}_0) - \frac{\mu}{m}\mathbf{v}$$

- Substitute **v**:

$$\ddot{\mathbf{x}} = -\frac{k}{m}(\mathbf{x} - \mathbf{x}_0) - \frac{\mu}{m}\dot{\mathbf{x}}$$



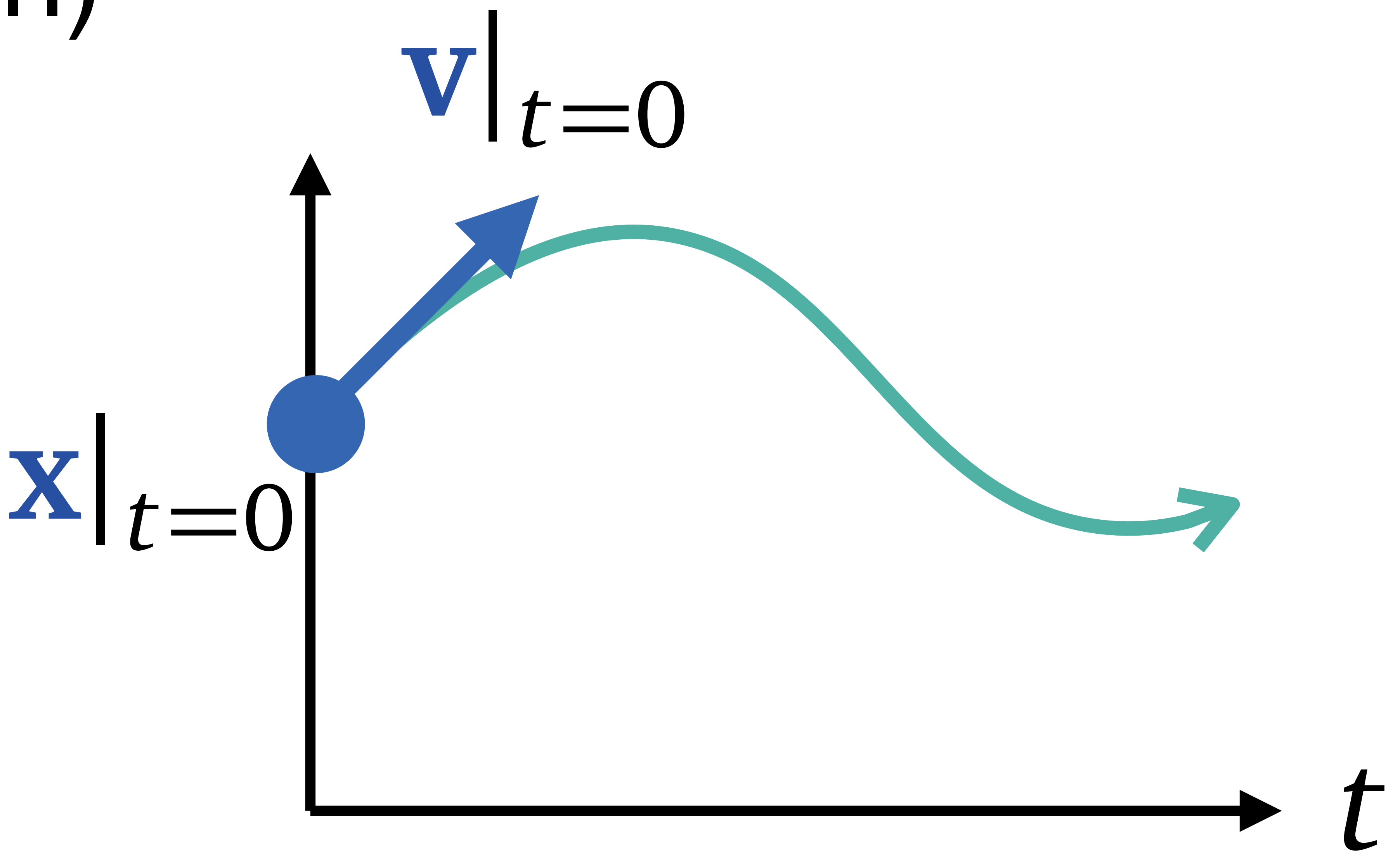

- Equation involving $\mathbf{x}$ and its derivatives
- This is called an *ordinary differential equation (ODE)*

# Types of problems

- Equations of motion $$\ddot{\mathbf{x}} = -\frac{k}{m}(\mathbf{x} - \mathbf{x}_0) - \frac{\mu}{m}\dot{\mathbf{x}}$$
  - This is called an *ordinary differential equation (ODE)*

- Initial value problem (forward simulation)
  - Given initial conditions
    i.e. the values of $\mathbf{x}|_{t=0}, \mathbf{v}|_{t=0}$

    Extend them into function of time
    $$\mathbf{x}(t), \mathbf{v}(t) \qquad t \geq 0$$

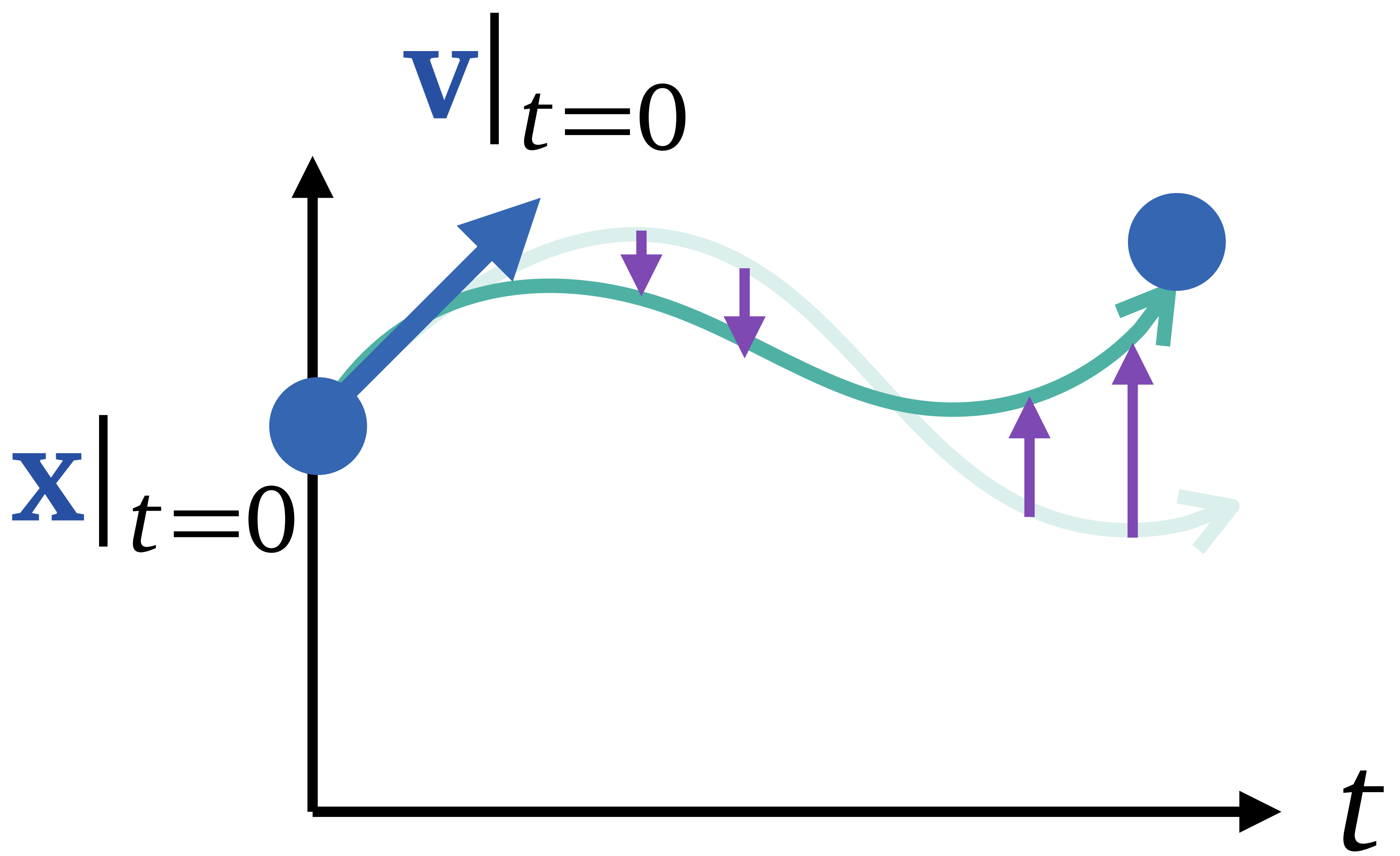


# Types of problems

- Equations of motion $$\ddot{\mathbf{x}} = -\frac{k}{m}(\mathbf{x} - \mathbf{x}_0) - \frac{\mu}{m}\dot{\mathbf{x}}$$
  - This is called an *ordinary differential equation (ODE)*

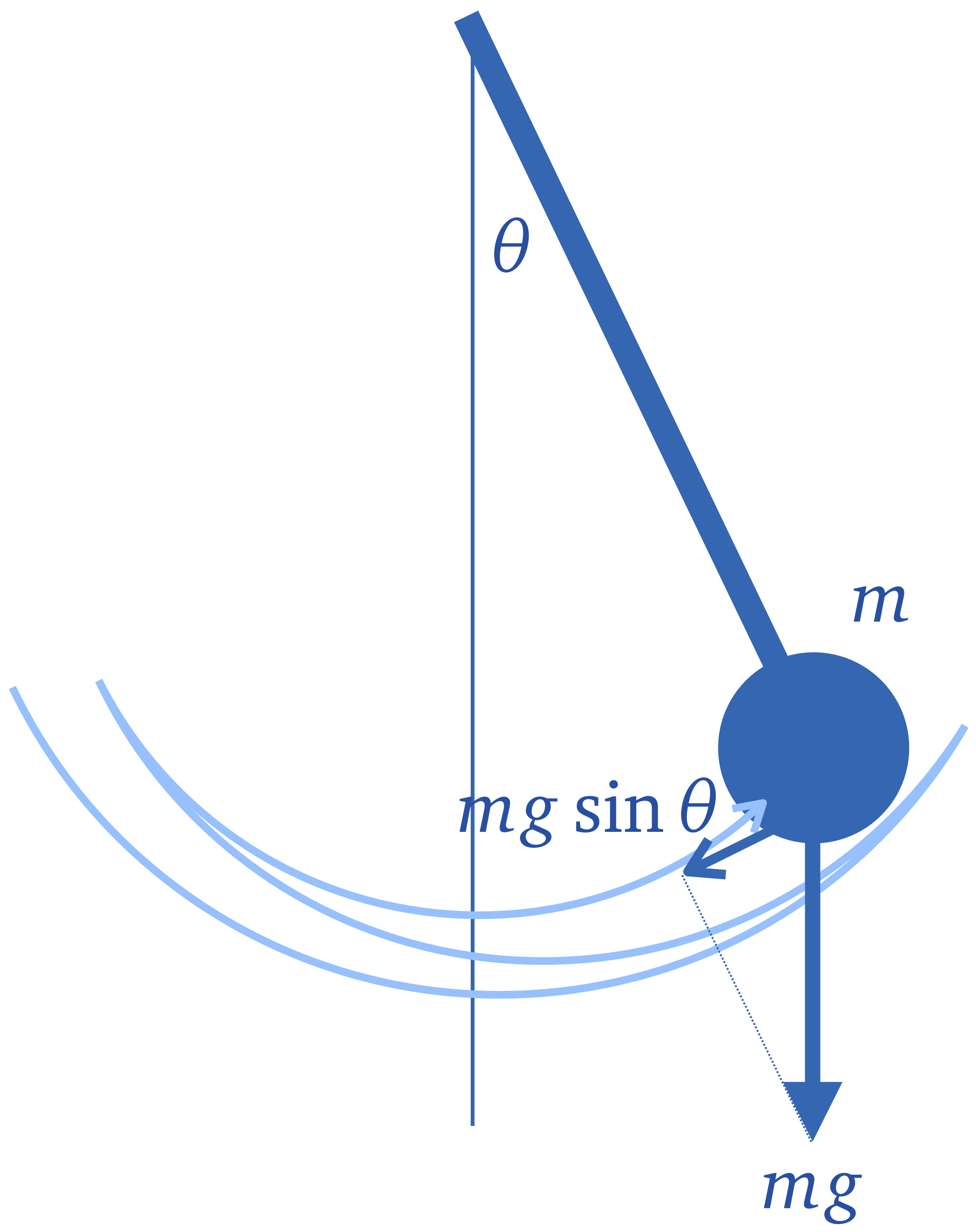
- Control problem
  - Given desired location to arrive at some future time,

    find minimal correction force to achieve the goal.
  - Robotics, control systems, physics-based keyframe animations



# Another example

- Pendulum




$$m\ddot{\theta} = -mg\sin\theta$$

# Solving ODE Numerically

- Simulation, Physics, Math
- Getting started: F = ma
- Solve ODEs numerically

# ODE

- Derive the differential equation (ODE) from physical laws
- Solve the differential equation (ODE)
  - Hard to solve it by hand most of the time
  - Numerical method is needed

# ODE

- Given any differential equation, for example,

$$\dddot{\mathbf{x}} + \ddot{\mathbf{x}}\dot{\mathbf{x}} + \sin(\ddot{\mathbf{x}}) = 1$$

- Convert it into a 1st order system of ODEs (involving at most first derivative)
  - Give each derivative a separate name (except for the highest order derivative) $\mathbf{v} = \dot{\mathbf{x}}$ $\mathbf{a} = \ddot{\mathbf{x}}$
  - Then $\begin{cases} \dot{\mathbf{x}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{a} \\ \dot{\mathbf{a}} = 1 - \mathbf{a}\mathbf{v} - \sin(\mathbf{a}) \end{cases}$

# ODE

- Given any differential equation, for example,

$$\dddot{\mathbf{x}} + \ddot{\mathbf{x}}\dot{\mathbf{x}} + \sin(\ddot{\mathbf{x}}) = 1$$

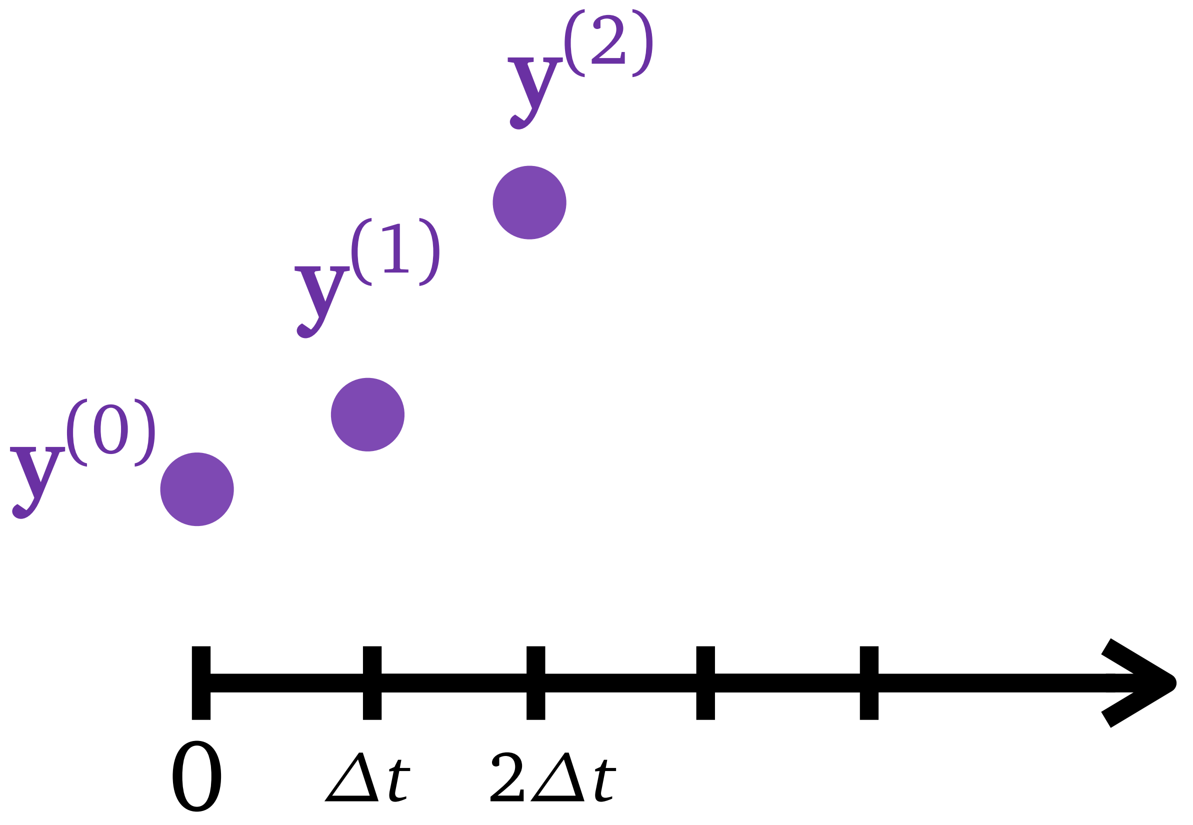
  - Then $\begin{cases} \dot{\mathbf{x}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{a} \\ \dot{\mathbf{a}} = 1 - \mathbf{a}\mathbf{v} - \sin(\mathbf{a}) \end{cases}$

  - Let $\mathbf{y} = \begin{bmatrix} \mathbf{x} \\ \mathbf{v} \\ \mathbf{a} \end{bmatrix}$ ODE becomes $\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y})$

# Numerical ODE

- Generic ODE $\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y})$
- Discretize time into time-frames $\mathbf{y}^{(n)} = \mathbf{y}(n\Delta t)$
- **(Forward) Euler method** $\frac{\mathbf{y}^{(n+1)} - \mathbf{y}^{(n)}}{\Delta t} \approx \mathbf{f}(\mathbf{y}^{(n)})$

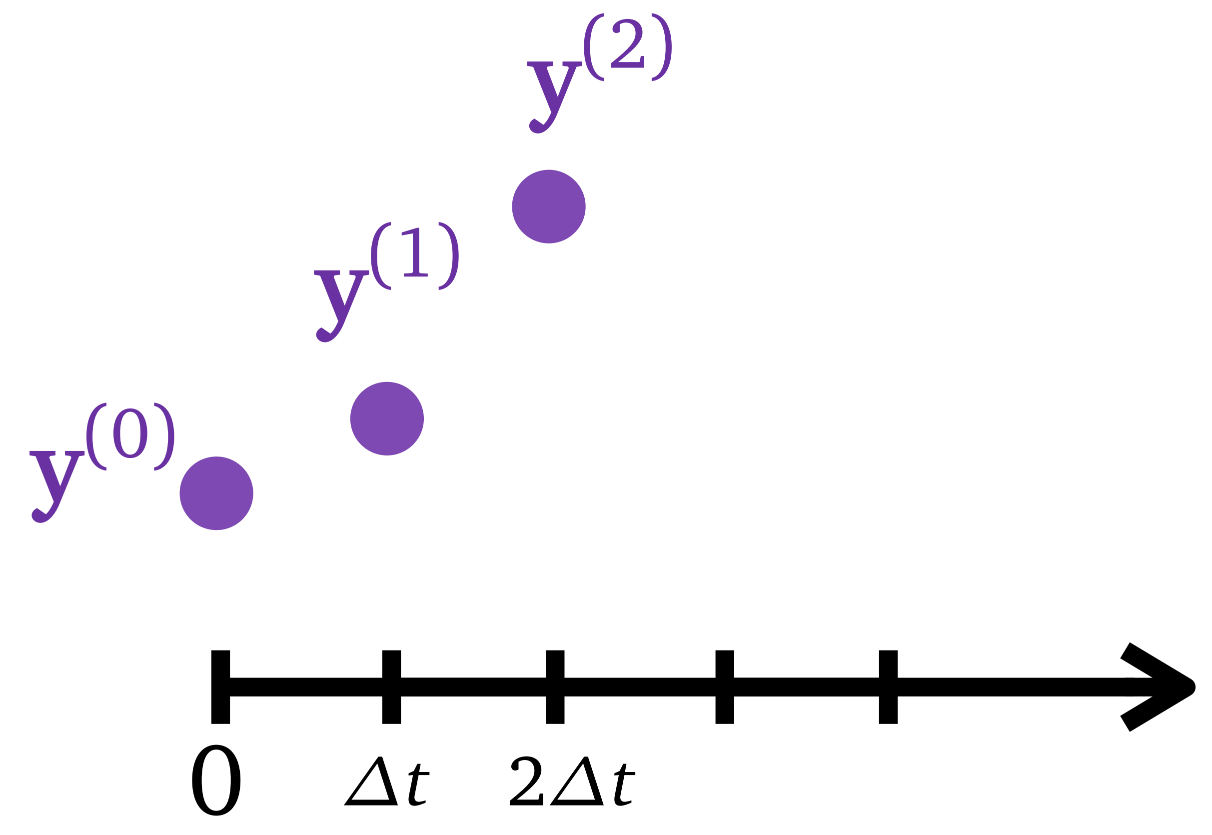
$$\mathbf{y}^{(n+1)} \approx \mathbf{y}^{(n)} + \Delta t \cdot \mathbf{f}(\mathbf{y}^{(n)})$$



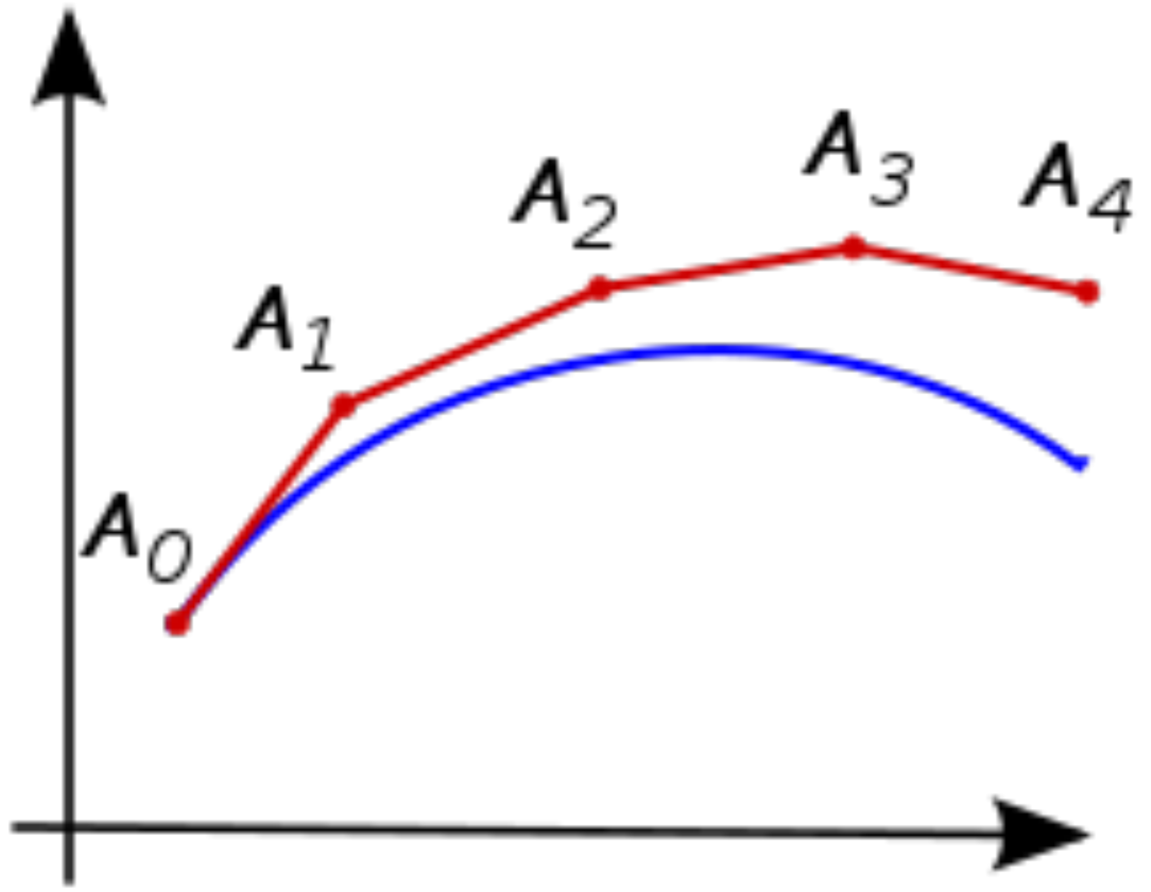
# Numerical ODE

- **(Forward) Euler method** $\frac{\mathbf{y}^{(n+1)} - \mathbf{y}^{(n)}}{\Delta t} \approx \mathbf{f}(\mathbf{y}^{(n)})$

$$\mathbf{y}^{(n+1)} \approx \mathbf{y}^{(n)} + \Delta t \cdot \mathbf{f}(\mathbf{y}^{(n)})$$



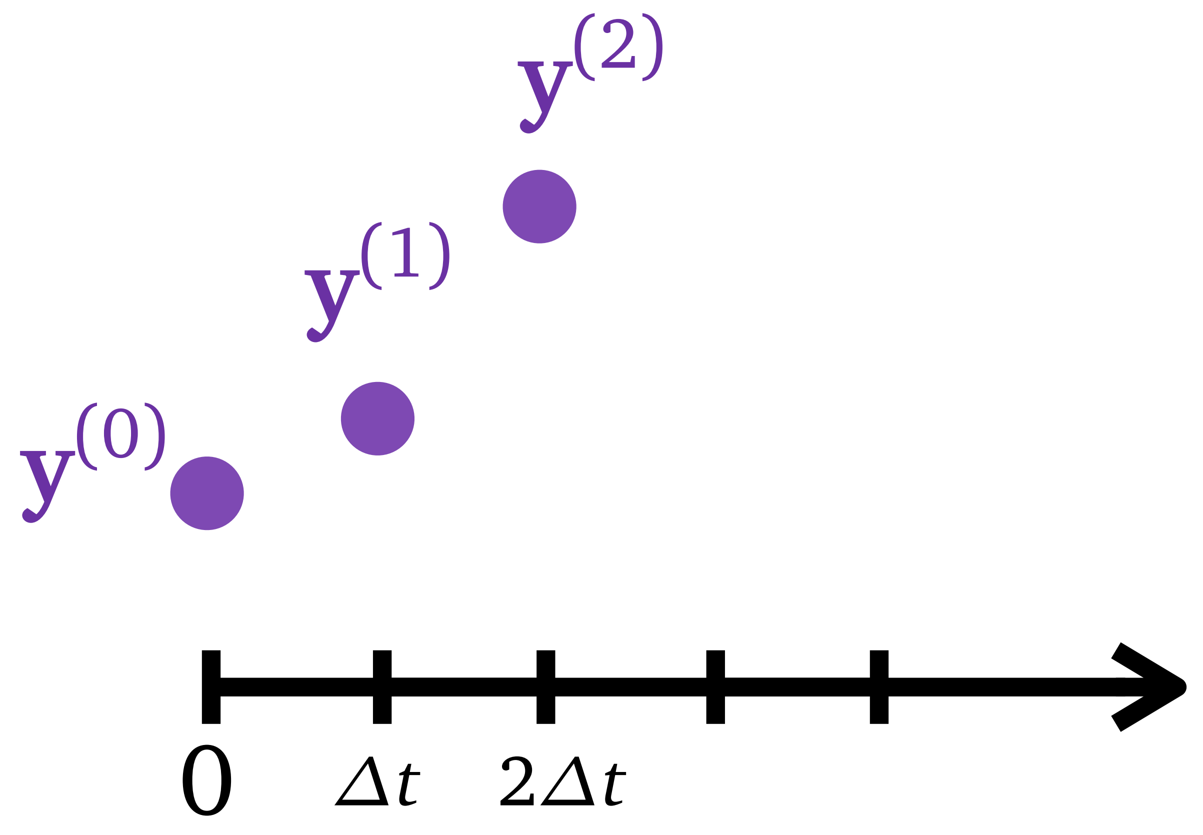

- Advantages
  - There is an **explicit** formula to plug in old state to get new state
  - Fast and simple
- Limitations
  - Not very accurate unless $\Delta t$ is tiny
  - Can be energy increasing (unphysical)



# Numerical ODE

- **(Forward) Euler method** $\frac{\mathbf{y}^{(n+1)} - \mathbf{y}^{(n)}}{\Delta t} \approx \mathbf{f}(\mathbf{y}^{(n)})$

$$\mathbf{y}^{(n+1)} \approx \mathbf{y}^{(n)} + \Delta t \cdot \mathbf{f}(\mathbf{y}^{(n)})$$




- **Backward Euler method**

$$\frac{\mathbf{y}^{(n+1)} - \mathbf{y}^{(n)}}{\Delta t} = \mathbf{f}(\mathbf{y}^{(n+1)})$$

- Limitations
  - Not very accurate unless $\Delta t$ is tiny
  - Have to solve for new state (**implicit**) instead of explicit update

# Numerical ODE

- **Backward Euler method**

$$\frac{\mathbf{y}^{(n+1)} - \mathbf{y}^{(n)}}{\Delta t} = \mathbf{f}(\mathbf{y}^{(n+1)})$$

- Limitations
  - Not very accurate unless $\Delta t$ is tiny
  - Have to solve for new state (**implicit**) instead of explicit update
- Advantages
  - Energy decreasing (dissipating), which looks physical
  - Can take larger time steps $\Delta t$ without instability
  - Can incorporate collision (just add constraint to the implicit solves)

# Numerical ODE

- **Euler method** $\mathbf{y}^{(n+1)} \approx \mathbf{y}^{(n)} + \Delta t \cdot \mathbf{f}(\mathbf{y}^{(n)})$ $\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y})$

- **Runge–Kutta method (RK4)** (Accurate, stable, explicit)

$$\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}(\mathbf{y}^{(n)}) \\
\mathbf{k}_2 &= \mathbf{f}(\mathbf{y}^{(n)} + \tfrac{\Delta t}{2}\mathbf{k}_1) \\
\mathbf{k}_3 &= \mathbf{f}(\mathbf{y}^{(n)} + \tfrac{\Delta t}{2}\mathbf{k}_2) \\
\mathbf{k}_4 &= \mathbf{f}(\mathbf{y}^{(n)} + \Delta t\,\mathbf{k}_3)
\end{aligned}$$

$$\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} + \tfrac{\Delta t}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)$$

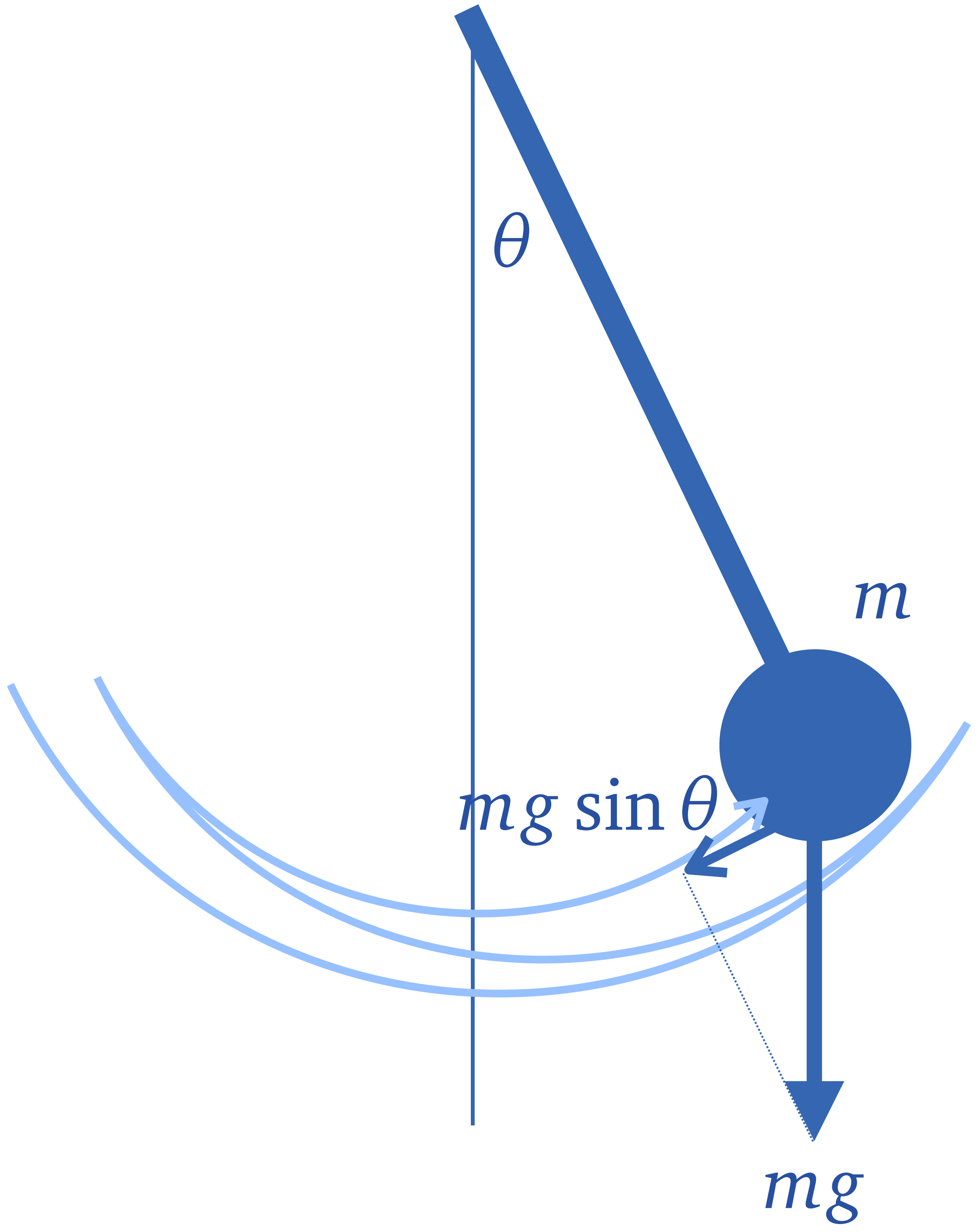
(collision handling is not as elegant as backward Euler)

# Numerical ODE

- In most cases the RK4 method works very well
- Sometimes the underlying physical system has additional structures (energy conservation, momentum conservation)
- Special algorithm (non-RK4) aims at preserving energy or momentum
  - Variational integrator
  - Symplectic integrator
  - Lie group integrator

# Pendulum equation

Example: pendulum equation.

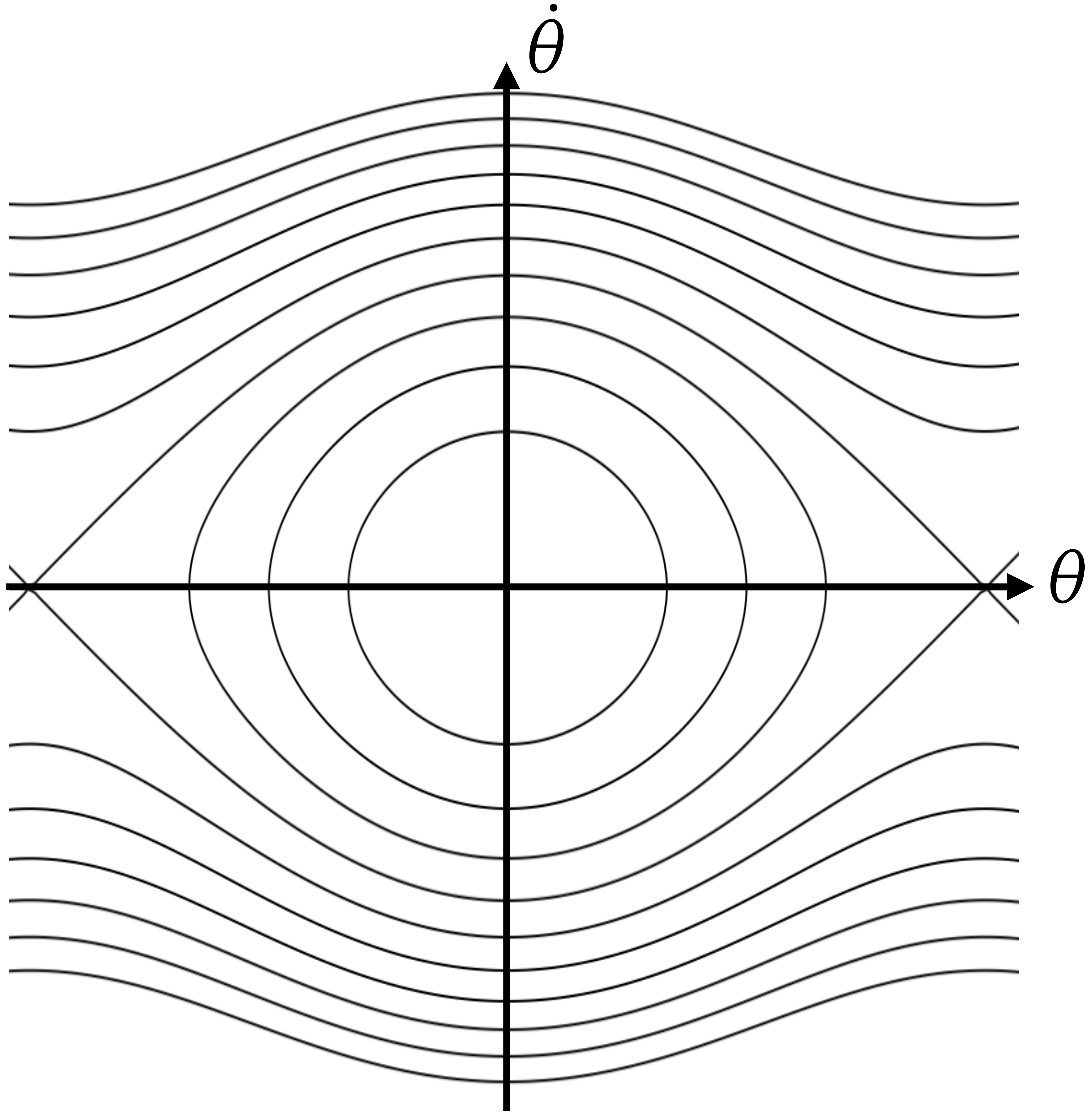



$$m\ddot{\theta} = -mg\sin\theta$$

✓ Energy conservation

$$\frac{1}{2}m\dot{\theta}^2 - mg\cos\theta = \text{const}$$

✓ Integrable system

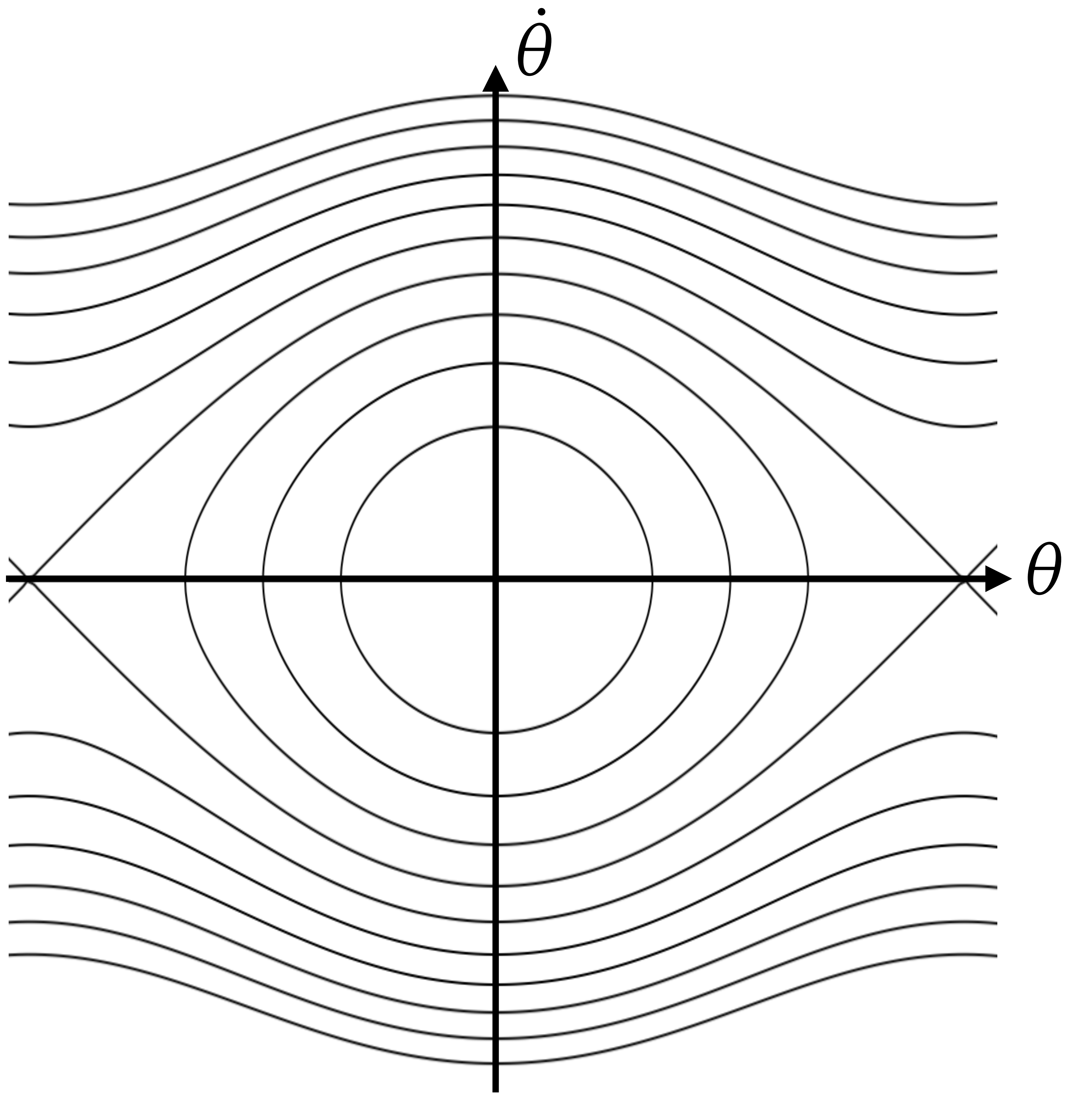


# Pendulum equation

## Example: pendulum equation.

$$m\ddot{\theta} = -mg\sin\theta$$

- High order differential equation solver (4th order Runge–Kutta method)




Given $(\theta_i, v_i = \dot{\theta}_i)$

- $\theta^*_{i+1/2} = \theta_i + \frac{\Delta t}{2} v_i \qquad v^*_{i+1/2} = v_i - \frac{\Delta t}{2}\sin\theta_i$
- $\theta^{**}_{i+1/2} = \theta_i + \frac{\Delta t}{2} v^*_{i+1/2} \qquad v^{**}_{i+1/2} = v_i - \frac{\Delta t}{2}\sin\theta^*_{i+1/2}$
- $\theta^{***}_{i+1} = \theta_i + \Delta t\, v^{**}_{i+1/2} \qquad v^{***}_{i+1} = v_i - \Delta t\sin\theta^{**}_{i+1/2}$

Output $\theta_{i+1} = \theta_i + \frac{\Delta t}{6}\left(v_i + 2v^*_{i+1/2} + 2v^{**}_{i+1/2} + v^{***}_{i+1}\right)$

$$v_{i+1} = v_i - \frac{\Delta t}{6}\left(\sin\theta_i + 2\sin\theta^*_{i+1/2} + 2\sin\theta^{**}_{i+1/2} + \sin\theta^{***}_{i+1}\right)$$

# Pendulum equation

## Example: pendulum equation.

$$m\ddot{\theta} = -mg\sin\theta$$

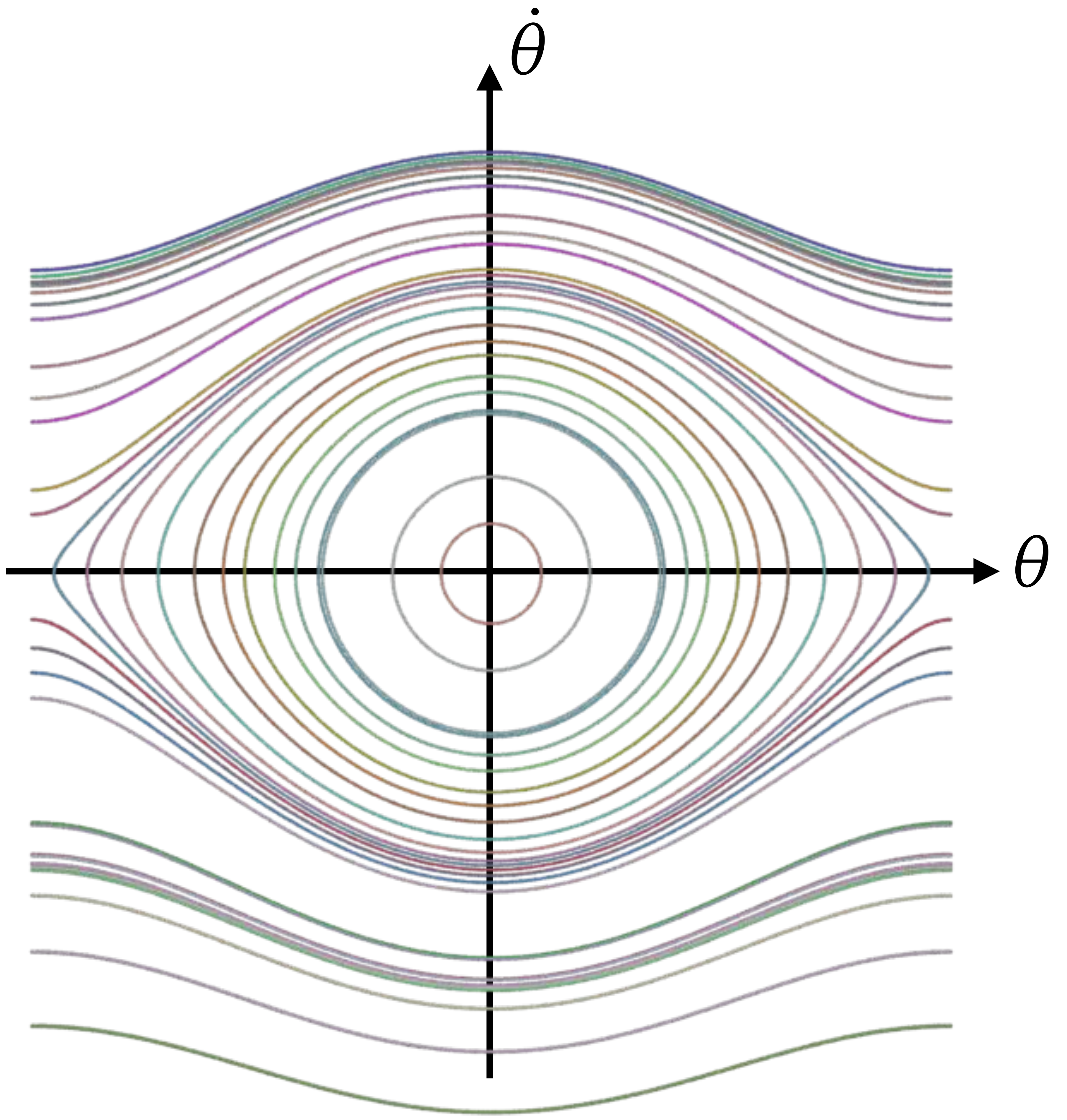
- High order differential equation solver (4th order Runge–Kutta method)

Given $(\theta_i, v_i = \dot{\theta}_i)$

- $\theta^*_{i+1/2} = \theta_i + \frac{\Delta t}{2} v_i$ $\quad v^*_{i+1/2} = v_i - \frac{\Delta t}{2}\sin\theta_i$
- $\theta^{**}_{i+1/2} = \theta_i + \frac{\Delta t}{2} v^*_{i+1/2}$ $\quad v^{**}_{i+1/2} = v_i - \frac{\Delta t}{2}\sin\theta^*_{i+1/2}$
- $\theta^{***}_{i+1} = \theta_i + \Delta t\, v^{**}_{i+1/2}$ $\quad v^{***}_{i+1} = v_i - \Delta t\sin\theta^{**}_{i+1/2}$

Output $\theta_{i+1} = \theta_i + \frac{\Delta t}{6}\left(v_i + 2v^*_{i+1/2} + 2v^{**}_{i+1/2} + v^{***}_{i+1}\right)$

$$v_{i+1} = v_i - \frac{\Delta t}{6}\left(\sin\theta_i + 2\sin\theta^*_{i+1/2} + 2\sin\theta^{**}_{i+1/2} + \sin\theta^{***}_{i+1}\right)$$




$\Delta t = 0.1$

# Pendulum equation

## Example: pendulum equation.

$$m\ddot{\theta} = -mg\sin\theta$$

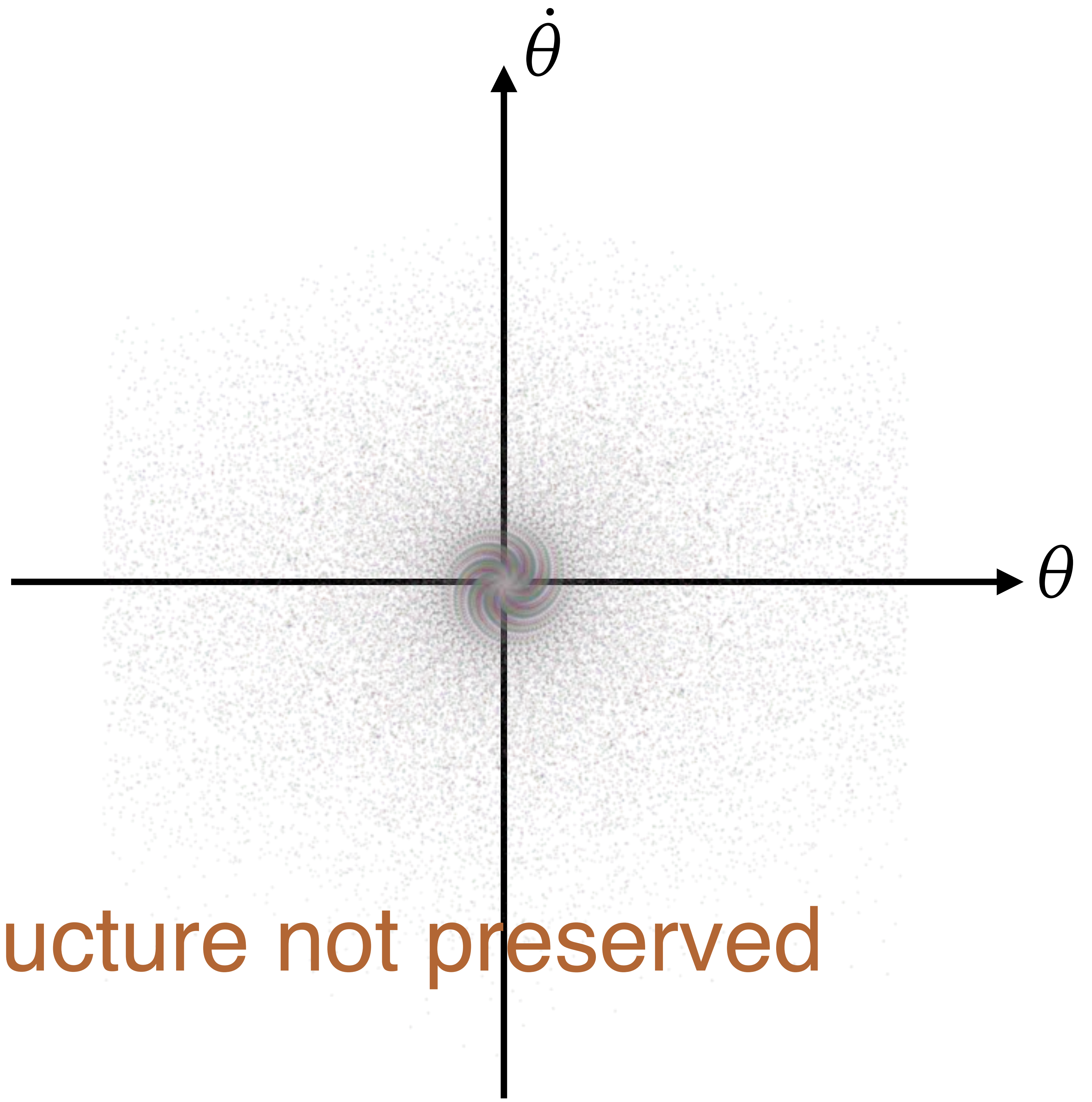
- High order differential equation solver (4th order Runge–Kutta method)

Given $(\theta_i, v_i = \dot{\theta}_i)$

- $\theta^{*}_{i+1/2} = \theta_i + \frac{\Delta t}{2} v_i \qquad v^{*}_{i+1/2} = v_i - \frac{\Delta t}{2}\sin\theta_i$
- $\theta^{**}_{i+1/2} = \theta_i + \frac{\Delta t}{2} v^{*}_{i+1/2} \qquad v^{**}_{i+1/2} = v_i - \frac{\Delta t}{2}\sin\theta^{*}_{i+1/2}$
- $\theta^{***}_{i+1} = \theta_i + \Delta t\, v^{**}_{i+1/2} \qquad v^{***}_{i+1} = v_i - \Delta t \sin\theta^{**}_{i+1/2}$

Output $\theta_{i+1} = \theta_i + \frac{\Delta t}{6}\left(v_i + 2v^{*}_{i+1/2} + 2v^{**}_{i+1/2} + v^{***}_{i+1}\right)$

$$v_{i+1} = v_i - \frac{\Delta t}{6}\left(\sin\theta_i + 2\sin\theta^{*}_{i+1/2} + 2\sin\theta^{**}_{i+1/2} + \sin\theta^{***}_{i+1}\right)$$




Structure not preserved

$\Delta t = 0.9$

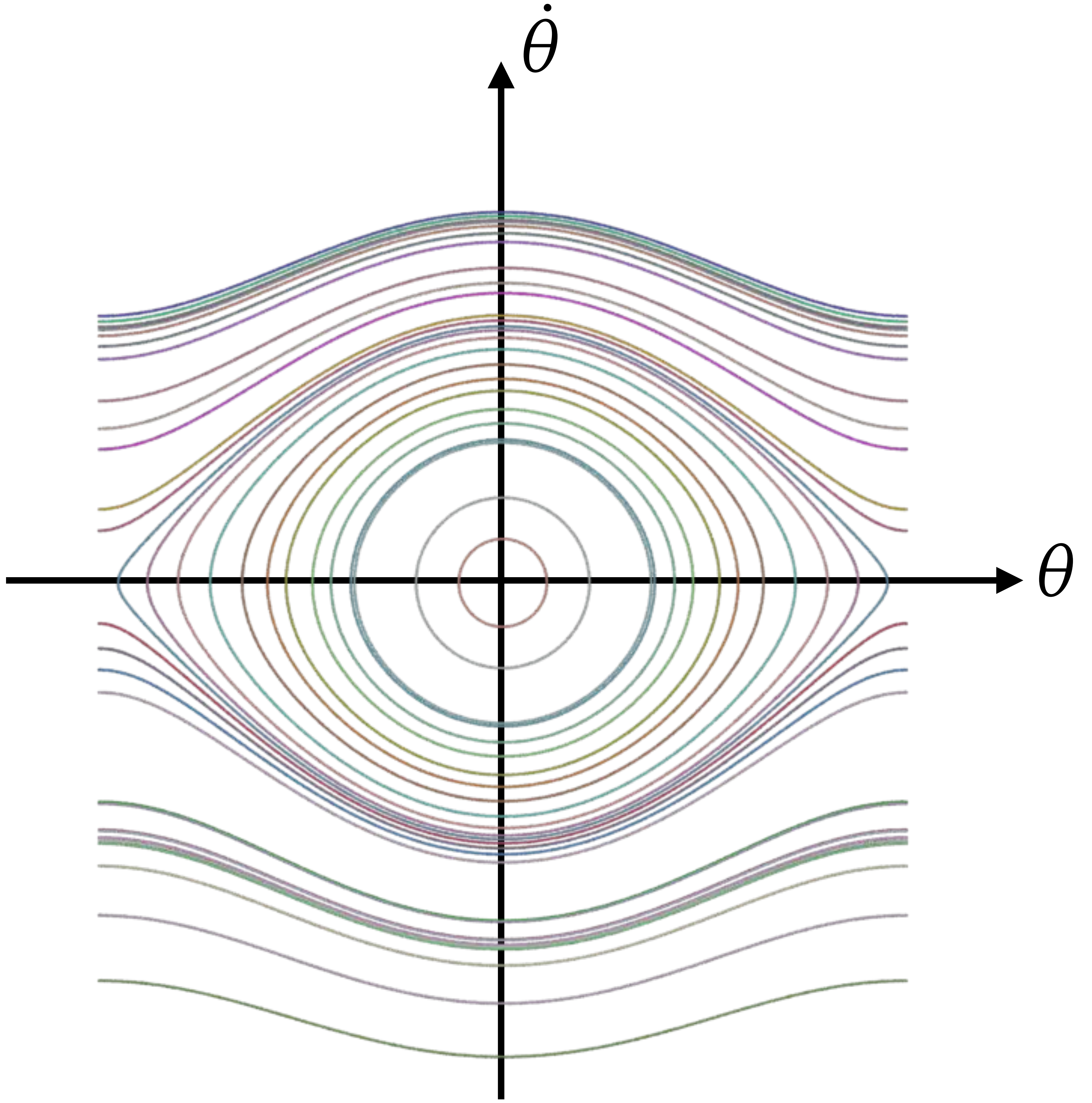
# Pendulum equation

Example: pendulum equation.

$$m\ddot{\theta} = -mg\sin\theta$$

- A 2nd order discretization

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin\theta_i$$




$\Delta t = 0.1$

# Pendulum equation

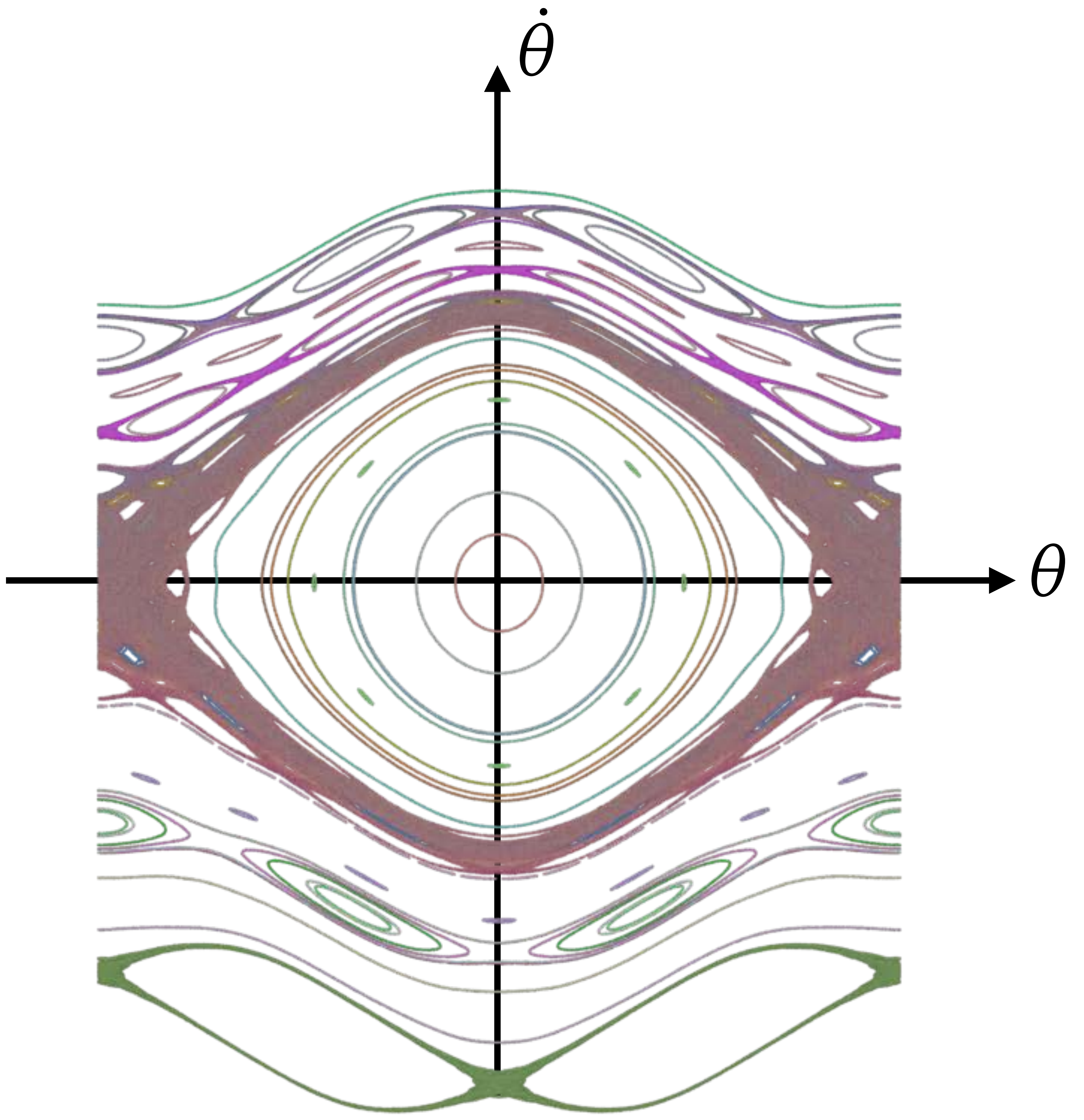
Example: pendulum equation.

$$m\ddot{\theta} = -mg\sin\theta$$

- A 2nd order discretization

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin\theta_i$$

✓ Energy conservation in "asteroid belts"




$\Delta t = 0.9$

# Pendulum equation

## Example: pendulum equation.

$$m\ddot{\theta} = -mg\sin\theta$$

Least action principle

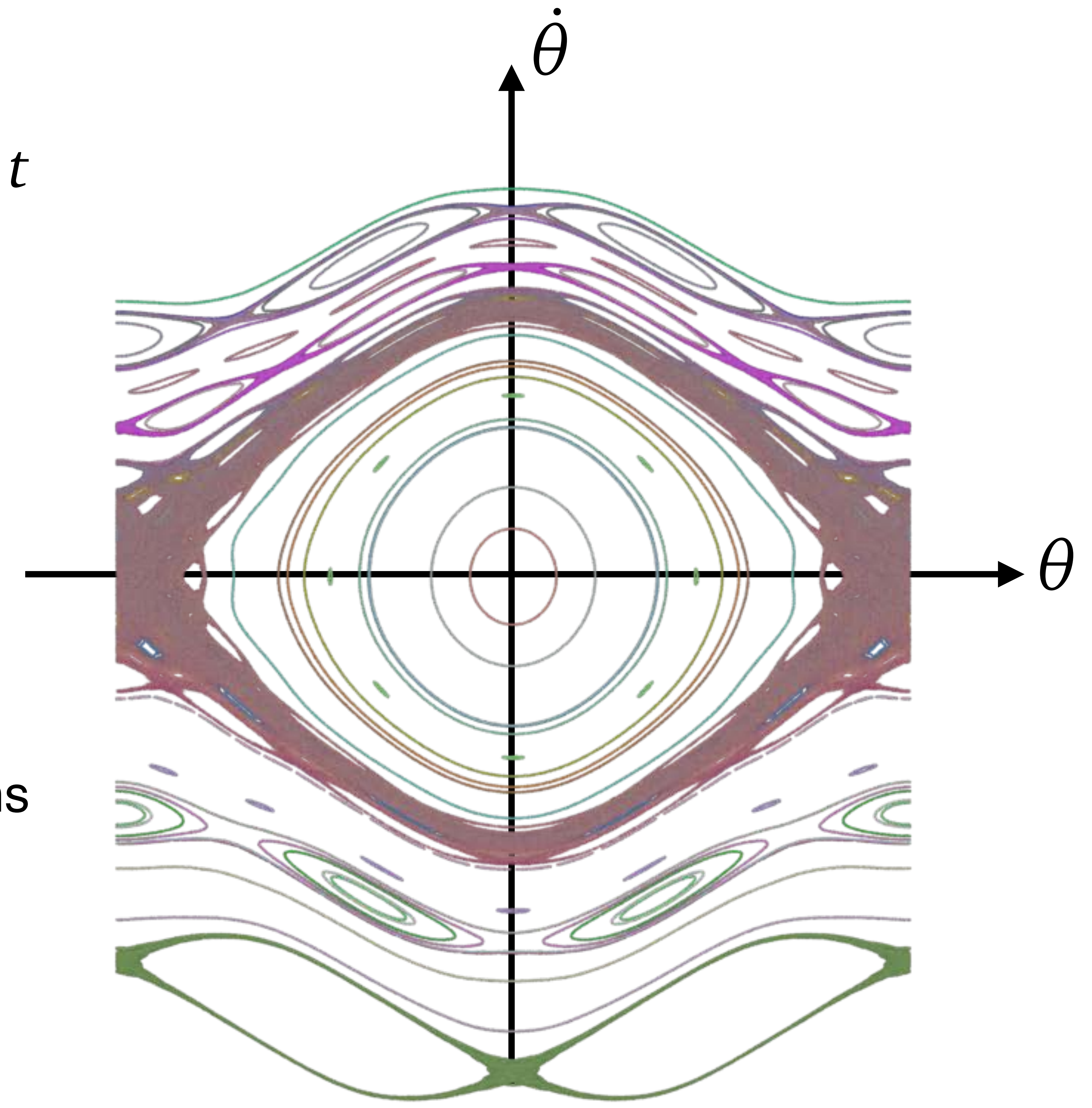
$$\int \left(\frac{m}{2}\dot{\theta}^2 + mg\cos\theta\right) dt$$

- A 2nd order discretization

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin\theta_i$$

First introduce discrete action,
then derive the least action paths

✓Energy conservation
in “asteroid belts”




$\Delta t = 0.9$

# Pendulum equation

Example: pendulum equation.

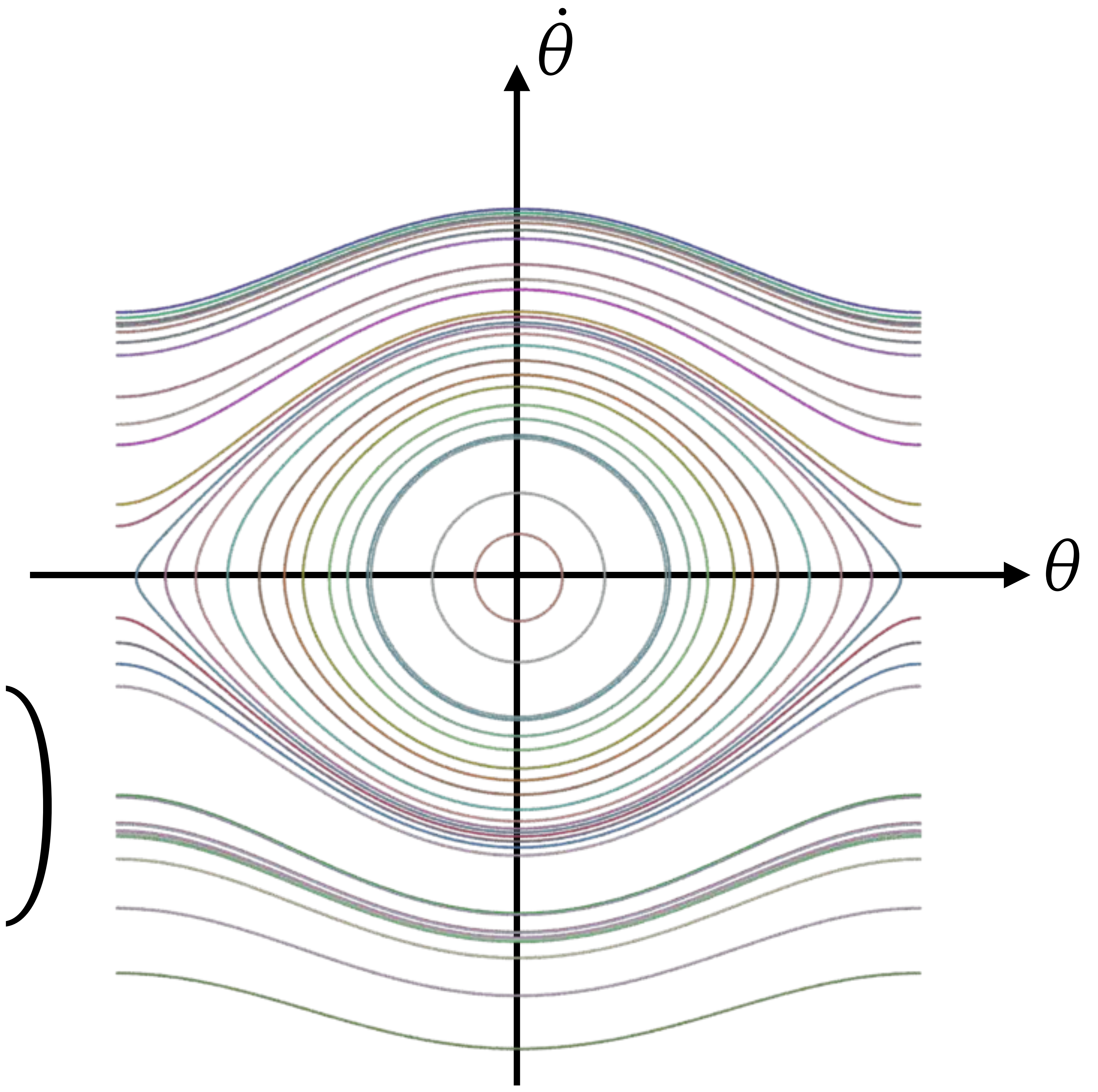
$$m\ddot{\theta} = -mg\sin\theta$$

- Another 2nd order discretization

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin\theta_i$$

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = 4\arg\left(1 + \frac{\Delta t^2}{4} e^{-\mathrm{i}\theta_i}\right)$$

✓ Integrable system




$\Delta t = 0.1$

# Pendulum equation

Example: pendulum equation.

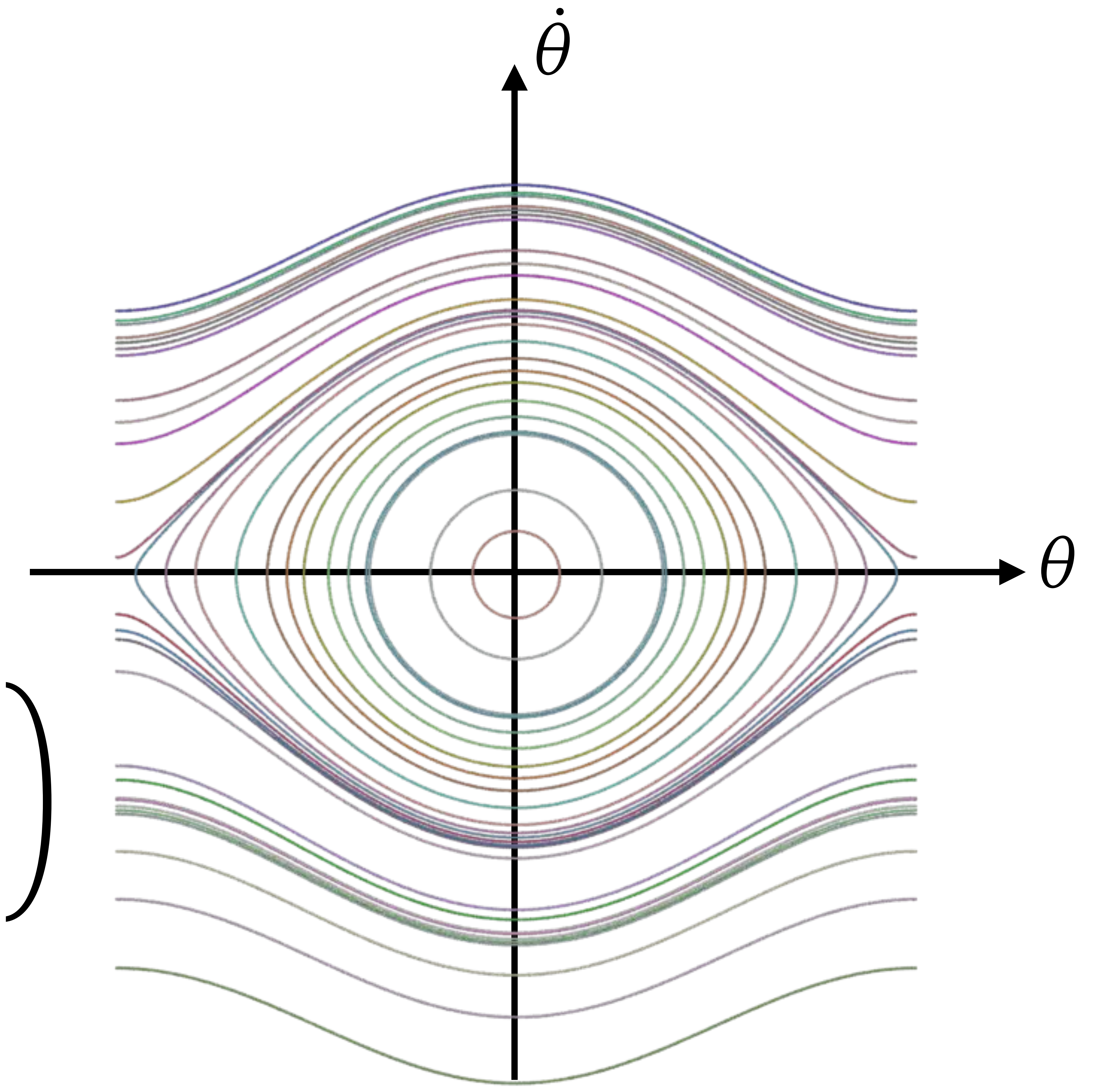
$$m\ddot{\theta} = -mg\sin\theta$$

- Another 2nd order discretization

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin\theta_i$$

$$\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = 4\arg\left(1 + \frac{\Delta t^2}{4}e^{-\mathbf{i}\theta_i}\right)$$

✓ Integrable system




$\Delta t = 0.9$

# Pendulum equation

general

no-structure

quantitative
high-precision

rare

structure
preserving

qualitative
exact

4th order Runge–Kutta | Variational integrator | Discrete integrable system

$\Delta t = 0.1$

$\Delta t = 0.9$