Principles of Robot Autonomy I

State space dynamics – definitions and modeling

Stanford University

Autonomous Systems Lab

Agenda

- State space dynamics
  - Definitions
  - Modeling (kinematic and dynamic models)
  - Special case: LTI systems and linearization

- Readings
  - B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo. Robotics: Modelling, Planning, and Control. Springer, 2008 (chapter 11)
  - Chapter 1 in PoRA lecture notes

State space models

- We can control a robot through the *inputs* to the system (e.g., motor torques, rotor thrusts, etc.)

- The *state* of a robot is a collection of variables (e.g., position, velocity) that change over time in response to the inputs

- A state space model

$$\dot{x} = f(x, u)$$

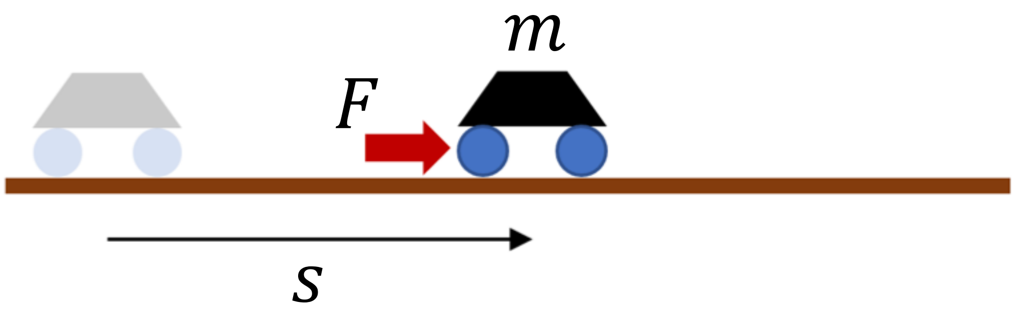
  is a mathematical description of how the state x evolves over time (i.e., $\dot{x}$ or dx/dt) in response to the inputs u

Example: double-integrator

- Suppose we can control the force pushing on a cart

- Newton's second law tells us that

$$F = m\ddot{s}$$



- Let $x = (s, v)$ with $v = \dot{s}$, and $u = {}^{F}/_{m}$. Then we can write

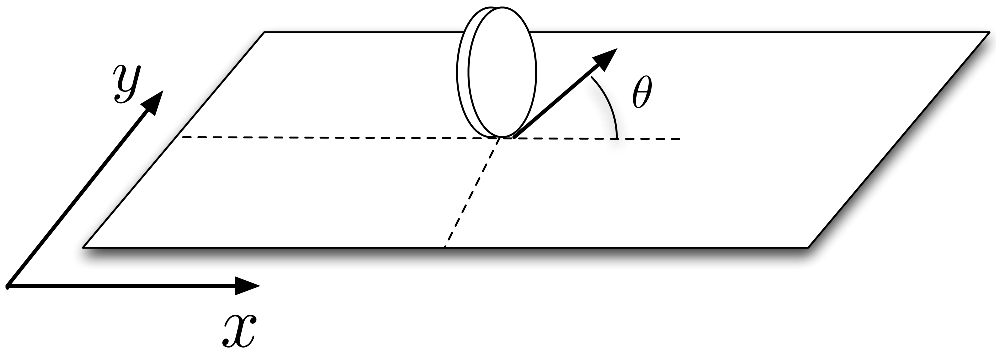
$$\dot{x} = \begin{pmatrix} v \\ u \end{pmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u}_{f(x,u)}$$

Kinematic models

- Kinematic models are mathematical models that describe the motion of a system without consideration of forces
- Kinematic models typically result from *geometric constraints* on the motion of a system, before considering any forces
- For example, the "unicycle" with *generalized coordinates* $q = (x, y, \theta)$ should not slip sideways, i.e.,

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \cdot \begin{pmatrix} \sin\theta \\ -\cos\theta \end{pmatrix} = 0$$

$$\underbrace{\begin{bmatrix} \sin\theta & -\cos\theta & 0 \end{bmatrix}}_{G(q)} \dot{q} = 0$$



This relation induces a kinematic model, as we will see shortly

Holonomic and nonholonomic constraints

- More broadly, constraints on degrees of freedom come in various forms:

$$\underbrace{h(q) = 0}_{\text{holonomic}} \qquad \underbrace{g(q, \dot{q}) = 0}_{\text{nonholonomic}} \qquad \underbrace{G(q)\dot{q} = 0}_{\text{semi-holonomic / Pfaffian}}$$

  Pfaffian constraints are a special, yet common case of nonholonomic constraints

- If $G(q)$ has k rows (constraints) and d columns (DOFs), then

Kinematic model of the constrained system

$$\dot{q} = \sum_{j=1}^{d-k} u_j b_j(q) = [b_1(q) \quad b_2(q) \quad \cdots \quad b_{d-k}(q)]u = B(q)u$$

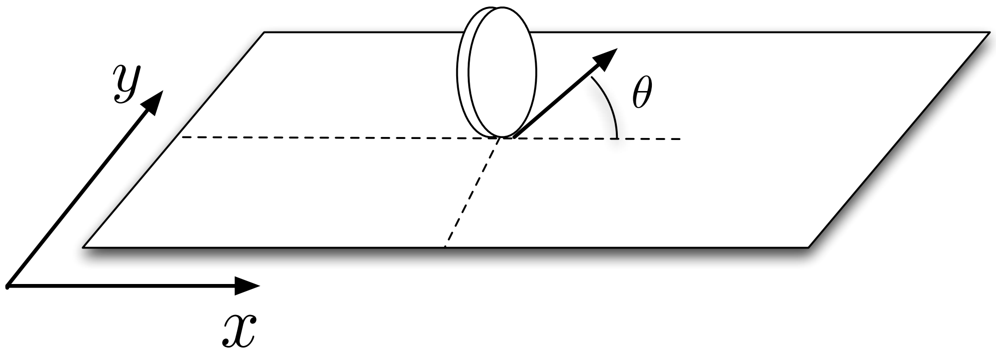
where $\{b_j(q)\}_{j=1}^{d-k}$ is a basis for admissible velocities, i.e., the null space of $G(q)$.

Back to unicycle example

- The "unicycle" with DOFs $q = (x, y, \theta)$ should not slip sideways, i.e.,

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \cdot \begin{pmatrix} \sin\theta \\ -\cos\theta \end{pmatrix} = 0$$

$$\underbrace{[\sin\theta \quad -\cos\theta \quad 0]}_{G(q)} \dot{q} = 0$$

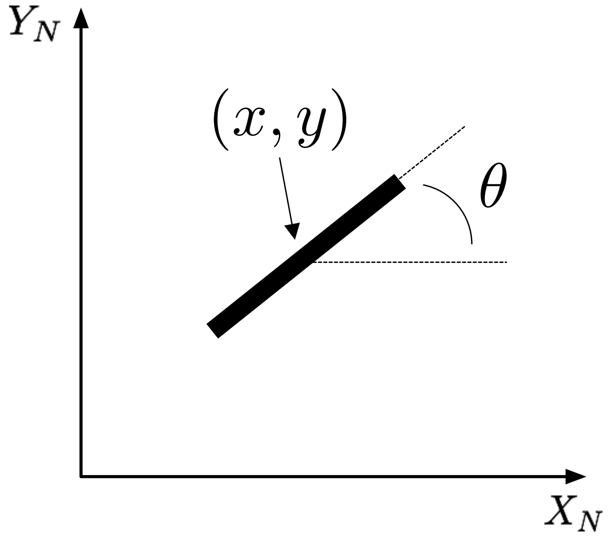


$$\dot{q} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u_2 = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} u$$

- Physically, $u_1 = v$ is the forward velocity of the wheel, and $u_2 = \omega$ is its rotational steering velocity

Unicycle and differential drive models

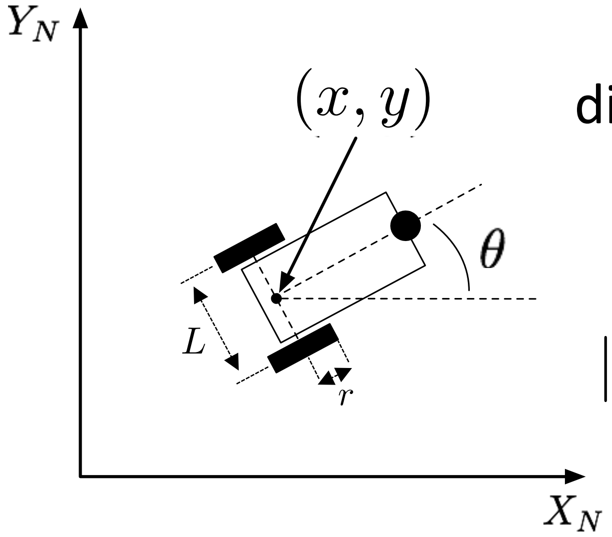
$$\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}\cos\theta\\ \sin\theta\\ 0\end{pmatrix} v + \begin{pmatrix}0\\ 0\\ 1\end{pmatrix} \omega$$



unicycle

$$|v| \leq v_{\max}$$
$$|\omega| \leq \omega_{\max}$$

$$\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}\frac{r}{2}(\omega_l + \omega_r)\cos\theta\\ \frac{r}{2}(\omega_l + \omega_r)\sin\theta\\ \frac{r}{L}(\omega_r - \omega_l)\end{pmatrix}$$



differential drive

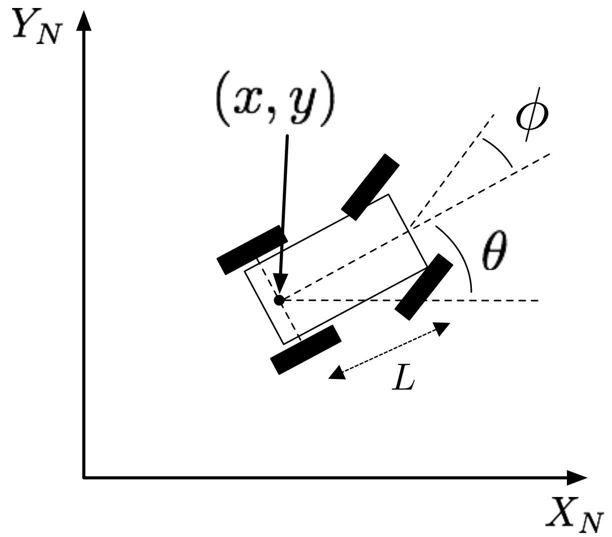
$$|\omega_l| \leq \omega_{l,\max}$$
$$|\omega_r| \leq \omega_{r,\max}$$

We can alternate between these kinematic models via the one-to-one input mappings:

$$v = \frac{r}{2}(\omega_r + \omega_l) \qquad \omega = \frac{r}{L}(\omega_r - \omega_l)$$

Simple car model

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} v\cos\theta \\ v\sin\theta \\ \frac{v}{L}\tan\phi \end{pmatrix}$$



states: (x, y, θ)

inputs: (v, ϕ)

$$|v| \leq v_{\max}, \ |\phi| \leq \phi_{\max} < \frac{\pi}{2}$$ → Simple car model

$$v \in \{-v_{\max}, v_{\max}\}, \ |\phi| \leq \phi_{\max} < \frac{\pi}{2}$$ → Reeds-Shepp car

$$v = v_{\max}, \ |\phi| \leq \phi_{\max} < \frac{\pi}{2}$$ → Dubins car

References:

- J.-P. Laumond. *Robot motion planning and control*. 1998.
- S. LaValle. *Planning algorithms.* 2006.

From kinematic to dynamic models

- A kinematic state space model should be interpreted only as a subsystem of a more general dynamical model

- Improvements to the previous kinematic models can be made by placing integrators in front of action variables

- For example, for the unicycle model, one can set the speed as the integration of an action a representing acceleration, that is

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = \omega, \quad \dot{v} = a$$

states: (x, y, θ, v) inputs: (ω, a)

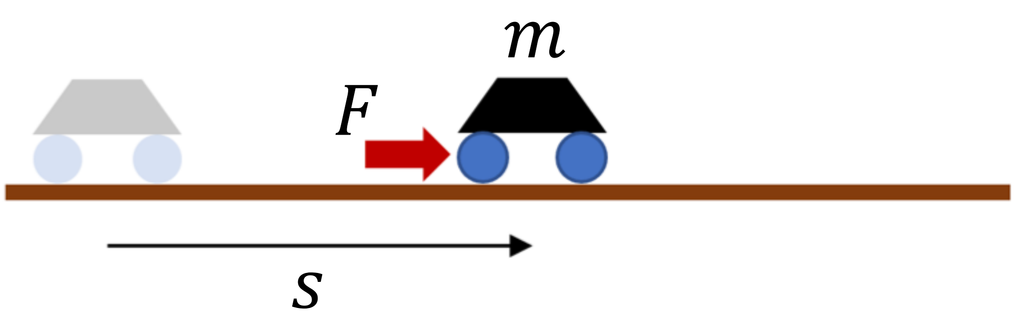
Linear time-invariant models

- In general, $\dot{x} = f(x, u)$ is nonlinear, which can make it difficult to analyze
- *Linear time-invariant (LTI)* models take the form
$$\dot{x} = Ax + Bu$$
with constant matrices A and B
- For $\dot{x} = \alpha x$ with $x(0) = x_0$, the solution is $x(t) = x_0 e^{\alpha t}$. If $\alpha < 0$, the system is *stable*, i.e., $x(t)$ converges to zero over time
- For $\dot{x} = Ax$ with $x(0) = x_0$, the solution is $x(t) = x_0 e^{At}$, where e^{At} is the *matrix exponential*
- Analogously to the scalar case, if $\mathrm{Real}(\lambda) < 0$ for each eigenvalue λ of A, then the system is stable

Example: PD control for a double-integrator

- Let $x = (s, v)$ with $v = \dot{s}$, and $u = {}^{F}/_{m}$. Then

$$\dot{x} = \begin{pmatrix} v \\ u \end{pmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u$$



- Choose $u = -\kappa_p s - \kappa_d v$. Then

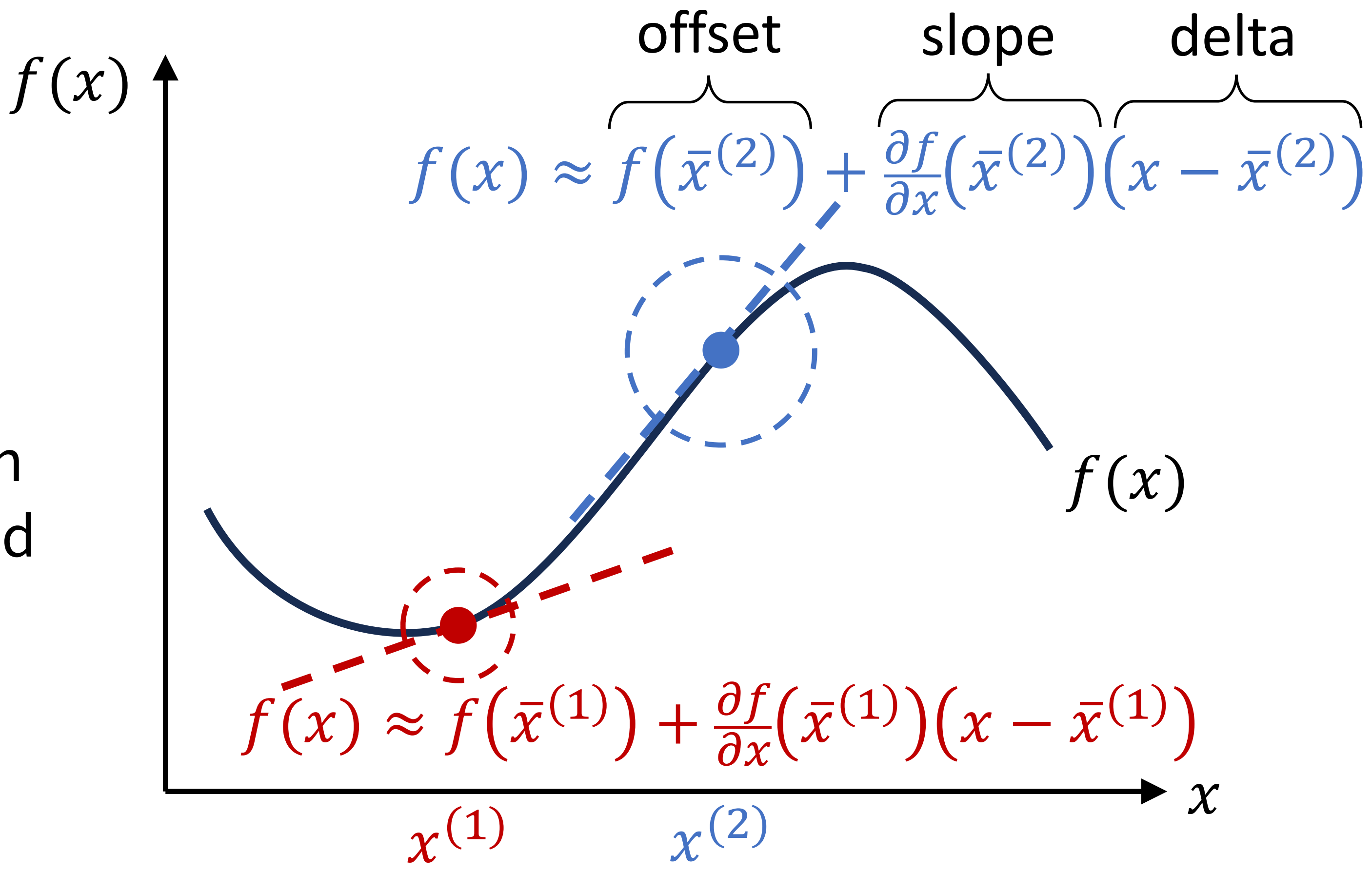
$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -\kappa_p & -\kappa_d \end{bmatrix} x$$

with eigenvalues $\lambda = -\frac{\kappa_d}{2} \pm \frac{1}{2}\sqrt{\kappa_d{}^2 - 4\kappa_p}$. If $\kappa_p > 0$ and $\kappa_d > 0$, then $\mathrm{Real}(\lambda) < 0$ for each eigenvalue , so the cart converges to a stand-still at $s = 0$

- This is nice, can we use linear control tools if the system is non-linear?

Linearization

- *Linearization* approximates a nonlinear function f near $\bar{x}$ by a line, i.e., linear function
- The "slope" of the line is the derivative of f at $\bar{x}$. The change in $f(\bar{x})$ near $\bar{x}$ is the slope multiplied by the distance from $\bar{x}$
- The quality of the approximation can vary with the linearization point $\bar{x}$ and distance from $\bar{x}$



Linearization of non-linear state-space models

- For the nonlinear system $\dot{x} = f(x, u)$, the linearization around $(\bar{x}, \bar{u})$ is

$$\dot{x} \approx f(\bar{x}, \bar{u}) + \underbrace{\frac{\partial f}{\partial x}(\bar{x}, \bar{u})}_{A}(x - \bar{x}) + \underbrace{\frac{\partial f}{\partial u}(\bar{x}, \bar{u})}_{B}(u - \bar{u})$$

  Since x and u can be vectors, we generalize derivatives to Jacobian matrices

- If $(\bar{x}, \bar{u})$ is an *equilibrium*, i.e., $f(\bar{x}, \bar{u}) = 0$, we can consider an LTI approximation of the system near $(\bar{x}, \bar{u})$, with state $\Delta x = x - \bar{x}$ and input $\Delta u = u - \bar{u}$:

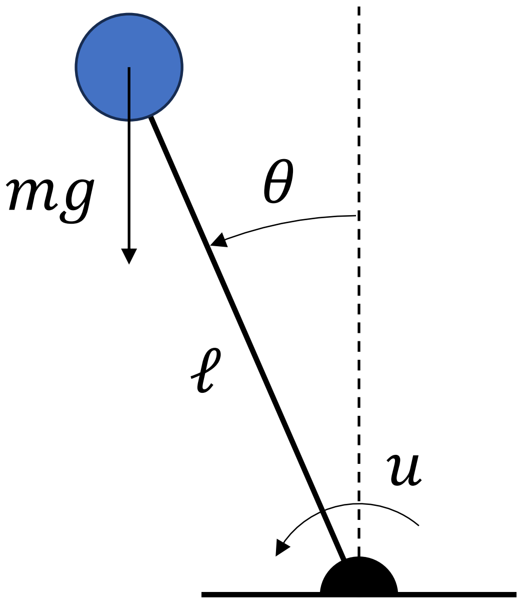
$$\dot{\Delta x} = A\Delta x + B\Delta u$$

- When (x, u) is near $(\bar{x}, \bar{u})$, we can use tools from linear systems analysis and control on nonlinear systems -- more on this later with LQR control!

Example: Inverted pendulum

- The dynamics are described by $m\ell^2\ddot{\theta} = mg\ell\sin\theta + u$. In state space form with $x = (\theta, \dot{\theta})$, they are

$$\dot{x} = f(x,u) = \begin{pmatrix} \dot{\theta} \\ \frac{g}{\ell}\sin\theta + \frac{1}{m\ell^2}u \end{pmatrix}$$



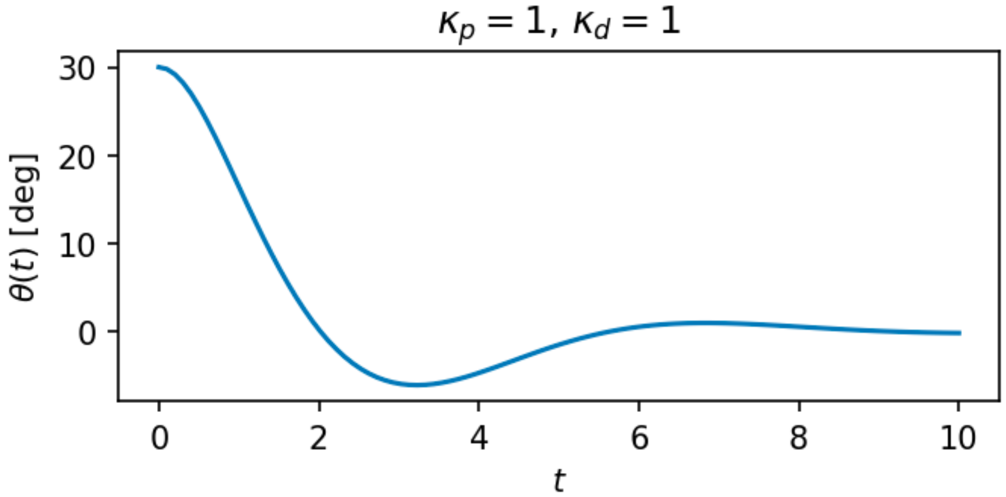
- Since $(x,u) = 0$ is an equilibrium, the linearization here is

$$\dot{x} \approx \begin{pmatrix} \dot{\theta} \\ \frac{g}{\ell}\theta + u \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ {}^{g}/_{\ell} & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ {}^{1}/_{m\ell^2} \end{bmatrix} u$$

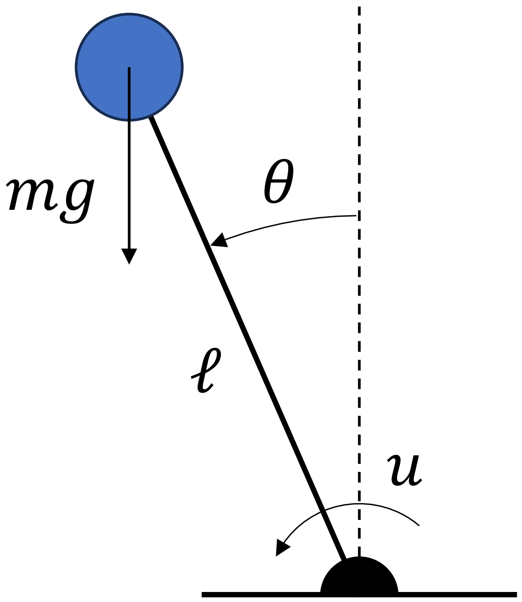
- This is close to a double-integrator! We could try $\frac{1}{m\ell^2}u = -\left(\frac{g}{\ell} + \kappa_p\right)\theta - \kappa_d\dot{\theta}$ to stabilize the pendulum near the upright equilibrium

Example: Inverted pendulum

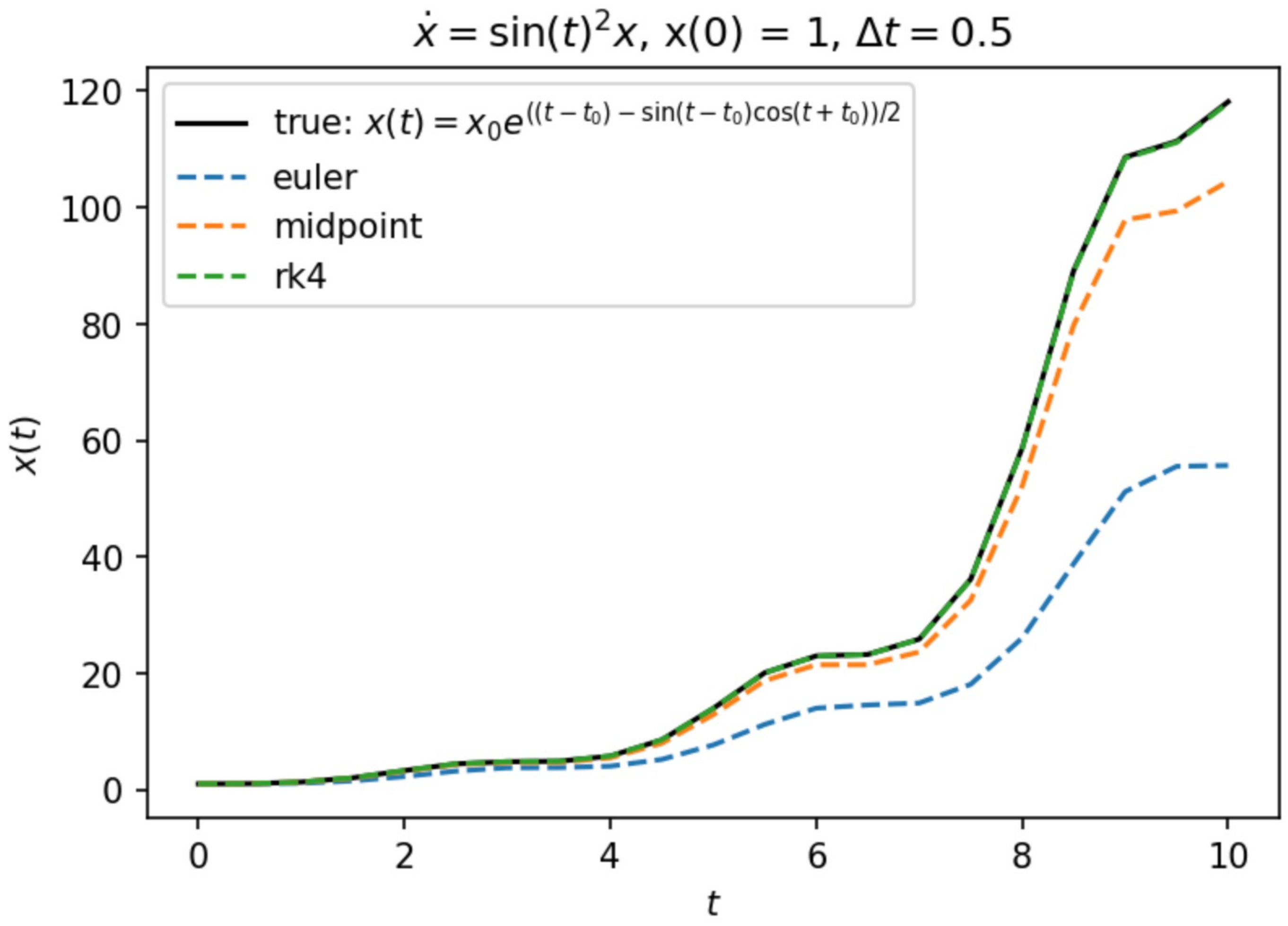
- We try $\frac{1}{m\ell^2}u = -\left(\frac{g}{\ell} + \kappa_p\right)\theta - \kappa_d\dot{\theta}$ to stabilize the pendulum near the upright equilibrium:



- We will later discuss how we actually simulate this system on a computer



Next time



$\dot{x} = \sin(t)^2 x$, x(0) = 1, $\Delta t = 0.5$
true: $x(t) = x_0 e^{((t-t_0) - \sin(t-t_0)\cos(t+t_0))/2}$
euler
midpoint
rk4
120
100
80
60
40
20
0
$x(t)$
0
2
4
6
8
10
t