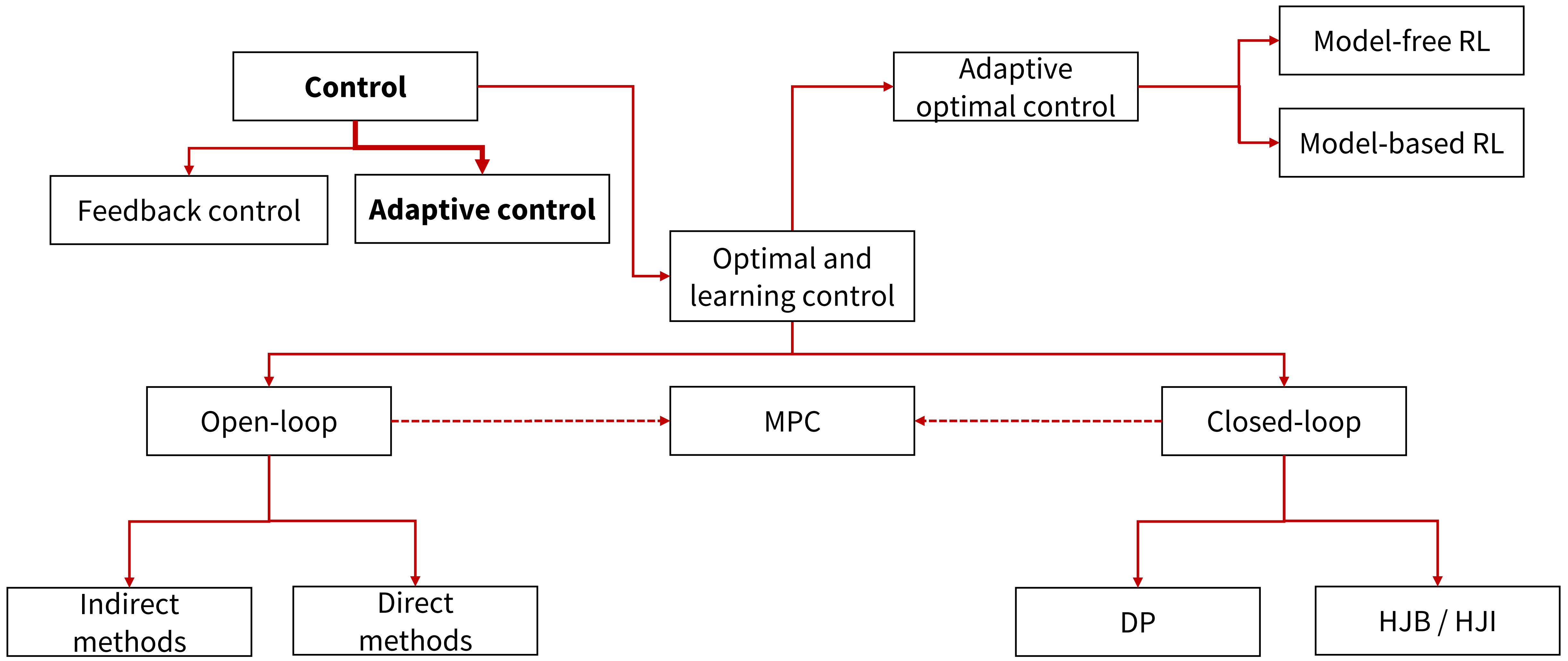
AA203
Optimal and Learning-based Control

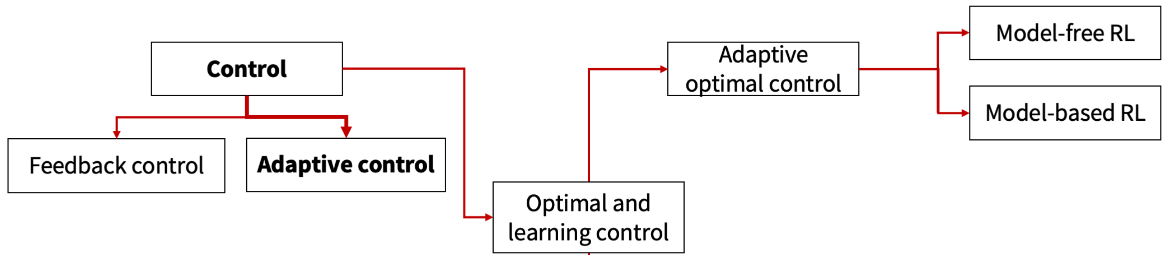
Intro to learning; System identification and adaptive control

Course overview



Outline

Intro to learning settings



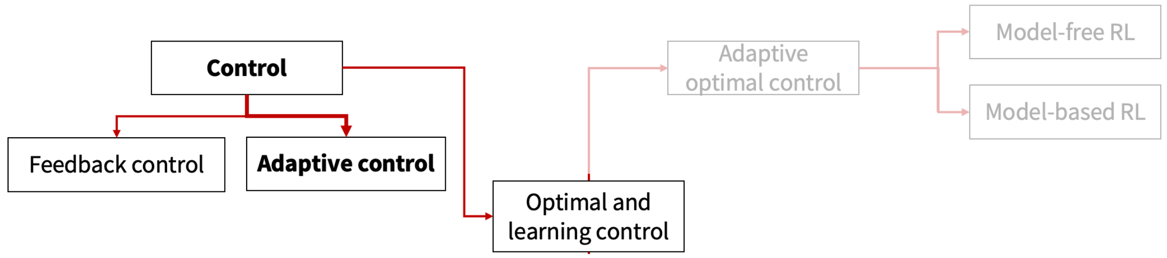
Outline

Intro to learning settings

System ID

Adaptive control

- MRAC: Model Reference Adaptive Control
- MIAC: Model Identification Adaptive Control



Outline

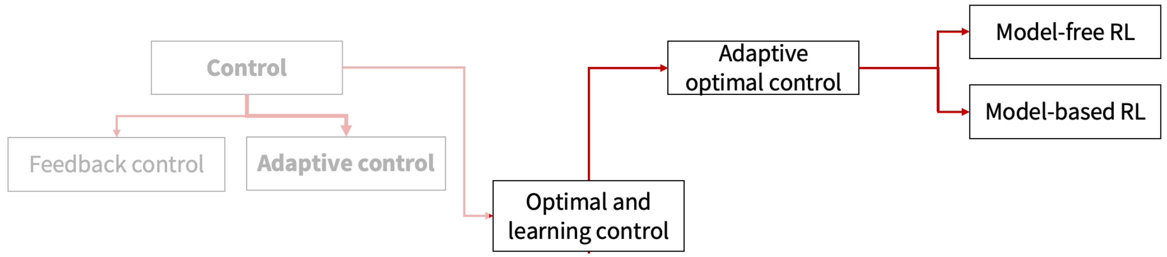
Intro to learning settings

System ID

Adaptive control

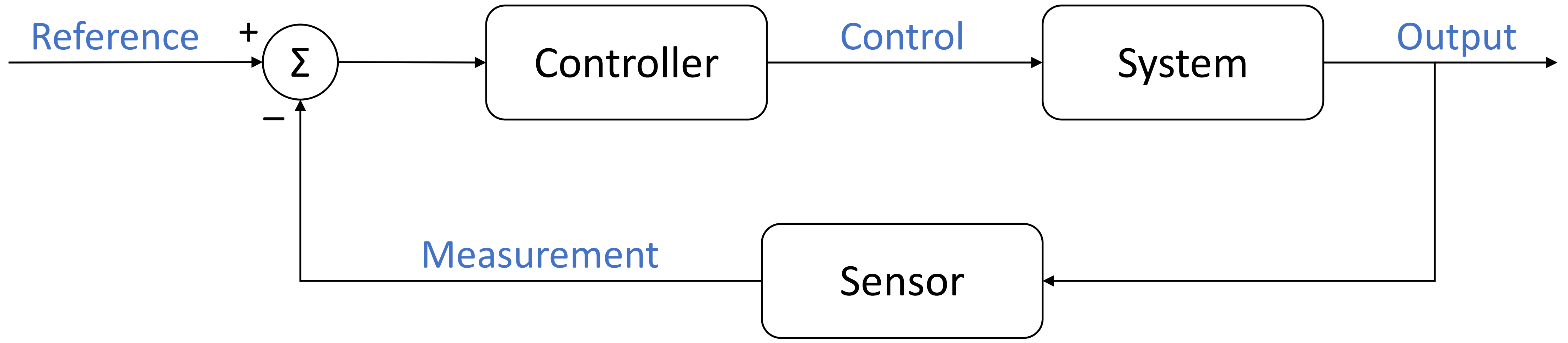
- MRAC: Model Reference Adaptive Control
- MIAC: Model Identification Adaptive Control

Short recap on RL



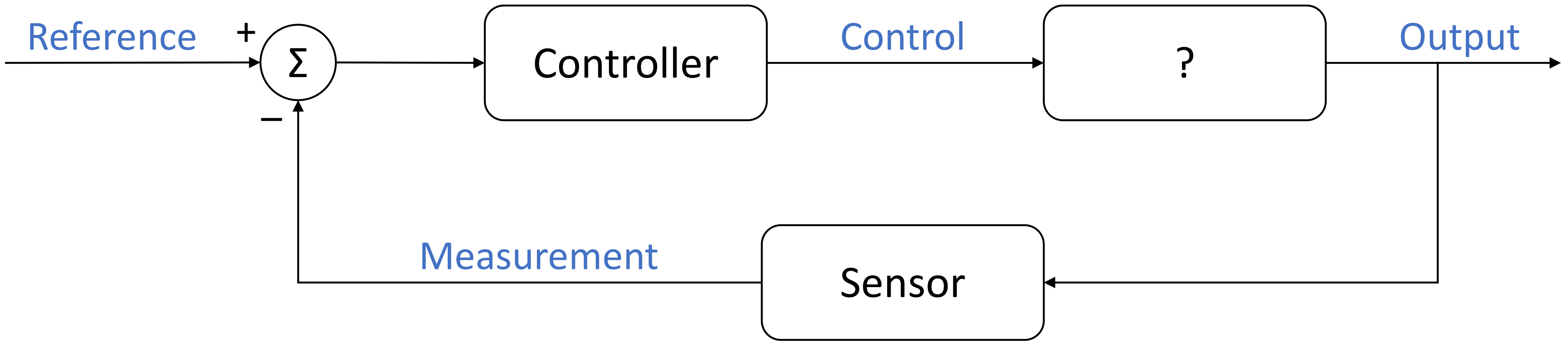
Feedback control

- Reference tracking, with uncertainty

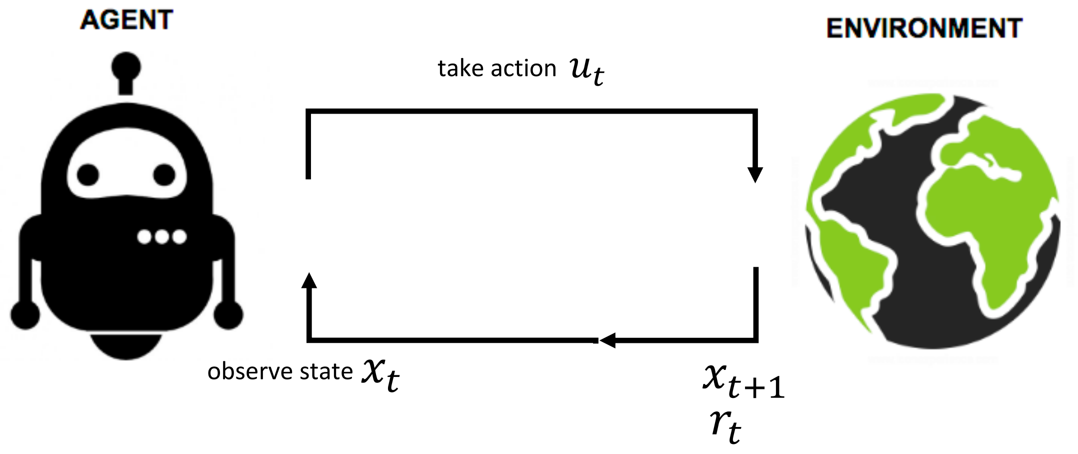


Feedback control

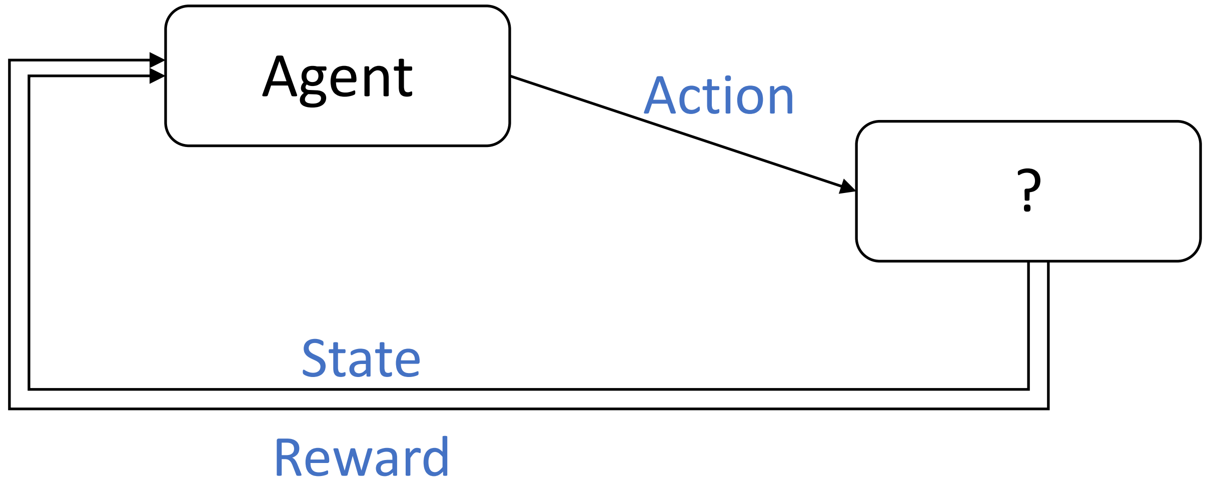
- Reference tracking, with uncertainty



Reinforcement learning



$$\tau = (x_0, u_0, \dots, x_N, u_N)$$



Approaches

How do we handle uncertainty?

- In many cases, when uncertainties have only a small effect, a feedback controller will adequately compensate for model error
  - Small wind disturbances in a quadrotor
- We can use robust control approaches (e.g., minimax control strategies)
  - Harrier VTOL near hover
- We can use observed state transitions to attempt to identify patterns and improve our control strategy
  - F-16 under wind resistance

What can we learn?

Want to use measurements to improve control performance.

Can either:

- Use measurements to directly improve controller
  - Direct adaptive control
  - Model-free reinforcement learning

- Use measurements to learn model, use model to improve controller
  - System identification
  - Indirect adaptive control
  - Model-based reinforcement learning

What can we learn?

Want to use measurements to improve control performance.

Can either:

- Use measurements to directly improve controller
  - Direct adaptive control Today
  - Model-free reinforcement learning Upcoming weeks

- Use measurements to learn model, use model to improve controller
  - System identification Today
  - Indirect adaptive control Today
  - Model-based reinforcement learning Upcoming weeks

How does learning happen?

Three possible learning settings:

- **“Zero” episodes**: the system identification approach, in which learning is done based on data gathered before operation
- **One episode**: want to learn and re-optimize our controller online -> this is the standard setting for adaptive control
- **Multiple episodes**: interact with the environment in episodes, in which the system is reset at the start of each episode; learning and policy optimization can happen between episodes → this is the standard setting for reinforcement learning

Outline

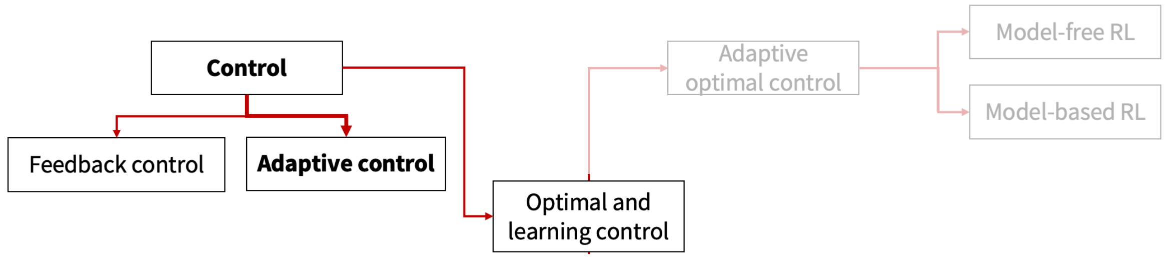
Intro to learning settings

System ID

Adaptive control

- MRAC: Model Reference Adaptive Control
- MIAC: Model Identification Adaptive Control

Short recap on RL



System identification for learning-based control

- For many problems, we don’t need to learn online
- A standard control engineering pipeline is to do experiments in advance to build a *data-driven* model of the dynamics
- Then, we can use this model for planning and control without further learning online
- Relies on having an engineer in the loop for learning, designing experiments, resetting the system, etc.
- *Linear regression* is one of the main system id tools

Least squares

System model

$$y = \boldsymbol{\theta}^T \boldsymbol{z} + \epsilon$$

Given data $y_1, \dots, y_N, \boldsymbol{z}_1, \dots, \boldsymbol{z}_N$, want to minimize mean squared error:

$$\sum_{i=1}^{N} (y_i - \boldsymbol{\theta}^T \boldsymbol{z}_i)^2$$

Rewrite as

$$\|\boldsymbol{y} - Z\, \boldsymbol{\theta}\|_2^2$$

Solution (full rank Z): $\widehat{\boldsymbol{\theta}} = (Z^T Z)^{-1} Z^T \boldsymbol{y}$

Example: first order model

- Consider a 1D first-order system with discrete-time dynamics
$$x_{t+1} = ax_t + bu_t + \epsilon_t$$
- Linear regression representation
  - $\boldsymbol{z_t} = [x_t, u_t], t = 0, \dots, N$
  - $\boldsymbol{\theta} = [a, b]$
- Comments
  - Practically, least squares can be written in recursive form for efficient updates as new observations stream in
  - Least squares regression may also be considered in continuous time (minimizing an error *integral*)

Linear regression for system id

- As seen before, the solution is

$$\hat{\boldsymbol{\theta}} = (Z^T Z)^{-1} Z^T \mathbf{y}$$

- Gauss-Markov theorem: $\hat{\boldsymbol{\theta}}$ is the **best linear unbiased estimator** (for any noise distribution that obeys assumptions)
    - Errors ϵ_t are zero mean, uncorrelated, and all have the same finite variance
- If noise distribution is Gaussian, $\hat{\boldsymbol{\theta}}$ is the maximum likelihood estimator

Persistent excitation

Classical question: what are sufficient conditions for our estimator $\widehat{\boldsymbol{\theta}}$ to converge?

$$\begin{aligned}\widehat{\boldsymbol{\theta}} &= (Z^T Z)^{-1} Z^T \mathbf{y} \\ &= (Z^T Z)^{-1} Z^T (Z\boldsymbol{\theta} + \boldsymbol{\epsilon}) \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}\end{aligned}$$

$$\mathbb{E}[\widehat{\boldsymbol{\theta}}] = \qquad\qquad \operatorname{cov}[\widehat{\boldsymbol{\theta}}] =$$

Persistent excitation

Classical question: what are sufficient conditions for our estimator $\widehat{\boldsymbol{\theta}}$ to converge?

$$\begin{aligned}\widehat{\boldsymbol{\theta}} &= (Z^T Z)^{-1} Z^T \mathbf{y} \\ &= (Z^T Z)^{-1} Z^T (Z\boldsymbol{\theta} + \boldsymbol{\epsilon}) \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}\end{aligned}$$

$$\begin{aligned}\mathbb{E}[\widehat{\boldsymbol{\theta}}] &= \mathbb{E}[\boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}] \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \mathbb{E}[\boldsymbol{\epsilon}] \\ &= \boldsymbol{\theta}\end{aligned}$$

$$\mathrm{cov}[\widehat{\boldsymbol{\theta}}] =$$

Persistent excitation

Classical question: what are sufficient conditions for our estimator $\widehat{\boldsymbol{\theta}}$ to converge?

$$\begin{aligned}\widehat{\boldsymbol{\theta}} &= (Z^T Z)^{-1} Z^T \mathbf{y} \\ &= (Z^T Z)^{-1} Z^T (Z\boldsymbol{\theta} + \boldsymbol{\epsilon}) \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}\end{aligned}$$

$$\begin{aligned}\mathbb{E}[\widehat{\boldsymbol{\theta}}] &= \mathbb{E}[\boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}] \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \mathbb{E}[\boldsymbol{\epsilon}] \\ &= \boldsymbol{\theta}\end{aligned}$$

$$\begin{aligned}\operatorname{cov}[\widehat{\boldsymbol{\theta}}] &= \operatorname{cov}[\boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}] \\ &= (Z^T Z)^{-1} Z^T \operatorname{cov}[\boldsymbol{\epsilon}] Z (Z^T Z)^{-1} \\ &= \sigma^2 (Z^T Z)^{-1} \\ &= \sigma^2 \left(\sum_{i=1}^{N} \mathbf{z}_i \mathbf{z}_i^T \right)^{-1} = \sigma^2 \left(\sum_{i=1}^{N} \begin{bmatrix} x_t^2 & x_t u_t \\ x_t u_t & u_t^2 \end{bmatrix} \right)^{-1}\end{aligned}$$

$\operatorname{cov}[AX] = A \operatorname{cov}[X] A^T$

$\operatorname{cov}[\epsilon] = \sigma^2 I$

Persistent excitation

Classical question: what are sufficient conditions for our estimator $\widehat{\boldsymbol{\theta}}$ to converge?

$$\begin{aligned}\widehat{\boldsymbol{\theta}} &= (Z^T Z)^{-1} Z^T \mathbf{y} \\ &= (Z^T Z)^{-1} Z^T (Z\boldsymbol{\theta} + \boldsymbol{\epsilon}) \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}\end{aligned}$$

$$\begin{aligned}\mathbb{E}[\widehat{\boldsymbol{\theta}}] &= \mathbb{E}[\boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}] \\ &= \boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \mathbb{E}[\boldsymbol{\epsilon}] \\ &= \boldsymbol{\theta}\end{aligned}$$

$$\mathrm{cov}[AX] = A\,\mathrm{cov}[X]A^T$$

$$\mathrm{cov}[\epsilon] = \sigma^2 I$$

$$\begin{aligned}\mathrm{cov}[\widehat{\boldsymbol{\theta}}] &= \mathrm{cov}[\boldsymbol{\theta} + (Z^T Z)^{-1} Z^T \boldsymbol{\epsilon}] \\ &= (Z^T Z)^{-1} Z^T \,\mathrm{cov}[\boldsymbol{\epsilon}] Z (Z^T Z)^{-1} \\ &= \sigma^2 (Z^T Z)^{-1} \\ &= \sigma^2 \left(\sum_{i=1}^{N} \mathbf{z}_i \mathbf{z}_i^T\right)^{-1} = \sigma^2 \left(\sum_{i=1}^{N} \begin{bmatrix} x_t^2 & x_t u_t \\ x_t u_t & u_t^2 \end{bmatrix}\right)^{-1}\end{aligned}$$

→ criterion such that if our sys id/adaptive control system operates forever, then we'll have identified the system perfectly

Performance questions

Practically, the system identification approach leads to several questions:

- What if we have a data budget?
  - How much data is required to learn the model?
  - How can we quantify a “good” estimate?
  - We care about controller performance, not model accuracy, so do we require an accurate model?
- How should we design the inputs used for data collection?
  - What if we’re learning while doing?
  - What if an engineer can’t intervene to prevent system failure during data collection?
- What if our system does not fall in the class of systems we are considering?

Outline

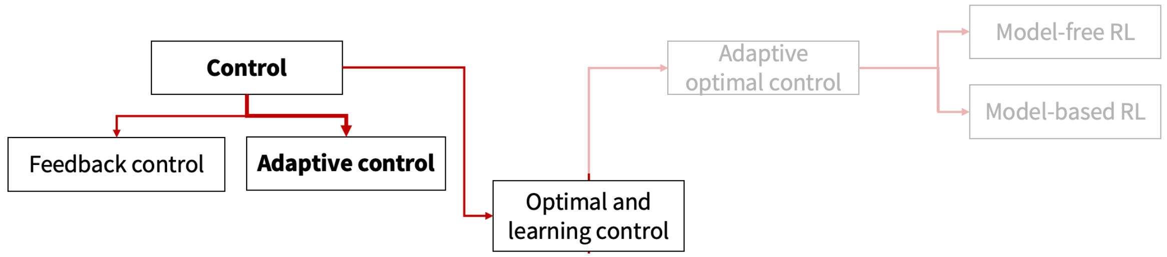
Intro to learning settings

System ID

Adaptive control

- MRAC: Model Reference Adaptive Control
- MIAC: Model Identification Adaptive Control

Short recap on RL



Adaptive control

- Broadly, adaptive control aims to perform *online adaptation* of the policy to improve performance
- This can be done via directly updating the policy or updating the model and re-optimizing or re-computing the controller
- Most classical adaptive control work does not consider the *optimal adaptive control* problem; they focus on proving stability of the coupled controller + adaptive component + system dynamics

Detour: Lyapunov theorem for global stability

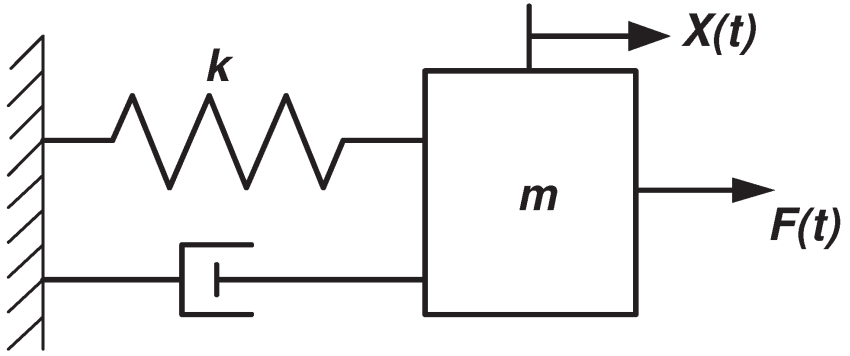
Consider an autonomous system $\dot{\boldsymbol{x}} = \mathrm{f}(\boldsymbol{x})$ (e.g., $\dot{\boldsymbol{x}} = \mathrm{f}(\boldsymbol{x}, \boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x}))$).
Equilibrium point: $\boldsymbol{x}^*$ s.t. $\mathrm{f}(\boldsymbol{x}^*) = 0$

1. $V(\boldsymbol{x}) > 0$ for all $\boldsymbol{x} \neq \boldsymbol{x}^*$, $V(\boldsymbol{x}^*) = 0$
2. $\dot{V}(\boldsymbol{x}) < 0$ for all $\boldsymbol{x} \neq \boldsymbol{x}^*$
3. $V(\boldsymbol{x}) \rightarrow \infty$ as $\|\boldsymbol{x}\| \rightarrow \infty$

If such a function exists, then equilibrium $\boldsymbol{x}^*$ is globally asymptotically stable.

Detour: stability analysis via Lyapunov

Mass/Spring/Damper



$$m\ddot{x} + d\dot{x} + kx = 0$$

Consider the total energy of the system:

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

$$\implies \frac{d}{dt}E(x, \dot{x}) = -d\dot{x}^2$$

Adaptive control approaches

Encompasses a large variety of techniques, including :

- Model reference adaptive control (MRAC)
- Model identification adaptive control (MIAC)

Model reference adaptive control (MRAC)

- A model reference adaptive controller is composed of four parts:
  1. A plant containing unknown parameters
  2. A *reference model* for compactly specifying the desired output
  3. A feedback control law containing adjustable parameters
  4. An adaptation mechanism for updating the adjustable parameters

- Compared to reward functions, a *reference model* is an alternative way to specify how a system should behave
  - The reference model provides the ideal plant response which the adaptation mechanism should seek in adjusting the parameters

Example of MRAC control

- Consider double integrator
$$m\ddot{x} = u$$
- Assume a human operator provides the positioning command $r(t)$ to the control system
- A reasonable way of specifying the ideal response of the controlled mass to the external command $r(t)$ is to use the reference model
$$\ddot{x}_m + k_d\dot{x}_m + k_p x_m = k\, r(t)$$
where the reference model output x_m is the *ideal* output

Example of MRAC control

- If the mass is known exactly, one can achieve perfect tracking via

$$u = m(\ddot{x}_m - 2\lambda\dot{\tilde{x}} - \lambda^2\tilde{x})$$

where $\lambda > 0$ is some chosen gain and $\tilde{x} \coloneqq x - x_m$ is the tracking error

- This control leads to exponentially convergent tracking dynamics

$$\ddot{\tilde{x}} + 2\lambda\,\dot{\tilde{x}} + \lambda^2\tilde{x} = 0$$

Example of MRAC control

- If the mass is *not* known exactly, we can use the control law
$$u = \hat{m}(\ddot{x}_m - 2\lambda\dot{\tilde{x}} - \lambda^2\tilde{x})$$
which contains the adjustable parameter $\hat{m}$
- This control leads to the closed-loop dynamics
$$m\dot{s} + \lambda m s = \tilde{m}v$$
where:
    - s is a combined tracking error measure (the “sliding mode” variable), defined by $s = \dot{\tilde{x}} + \lambda\tilde{x}$
    - the signal quantity v is given by $v = \ddot{x}_m - 2\lambda\dot{\tilde{x}} - \lambda^2\tilde{x}$
    - and the parameter estimation error is $\tilde{m}$= $\hat{m} - m$

Example of MRAC control

- One way of adjusting the parameter $\hat{m}$ is to use the (nonlinear) update law

$$\dot{\hat{m}} = -\gamma v s$$

where $\gamma > 0$ is called the *adaptation gain*

- Stability and convergence can be analyzed via Lyapunov theory
- Consider Lyapunov function candidate

$$V = \frac{1}{2}\left[m s^2 + \frac{1}{\gamma}\tilde{m}^2\right]$$

- Its derivative is $\dot{V} = -\lambda m s^2$
- Thus $s \to 0$, and hence $\tilde{x} \to 0$ and $\dot{\tilde{x}} \to 0$

MRAC

- An excellent reference for systematic MRAC design is:
  Jean-Jacques Slotine, Weiping Li, *Applied Nonlinear Control*, Chapter 8

- If the reference signal $r(t)$ is very simple, such as zero or a constant, it is possible for many vectors of parameters, besides the ideal parameter vector, to lead to tracking error convergence

- However, if the reference signal $r(t)$ is so complex that only the “true” parameter vector can lead to tracking convergence, then one shall have parameter convergence -> *persistent excitation condition*

Model identification adaptive control

- MIAC (also referred to as self-tuning) simply combines model estimation with a controller that uses the estimated model
- Important distinction between *certainty-equivalent* and *cautious* approaches
    - **Certainty-equivalent:** maintains point estimate of model and uses that model for policy selection/optimization. Can be suboptimal, risky.
    - **Cautious:** Maintains measure of estimator uncertainty, incorporates the uncertainty into the controller. This is often overly robust because it does not account for future info gain!

MRAC vs. MIAC

- MRAC and MIAC arise from two different perspectives:
    1. parameters in MRAC are updated so as to minimize tracking error between the plant output and the reference model output
    2. parameters in MIAC are updated so as to minimize the data-fitting error
- MIAC controllers are in general more flexible, as one can couple various controllers with various estimators
- However, correctness of MIAC controllers is more difficult to guarantee, as if the signals are not rich, the estimated parameters may not be close to the "true" values, and stability and convergence may not be ensured
- In contrast, for MRAC, stability and convergence are usually guaranteed *regardless* of the richness of the signals

Outline

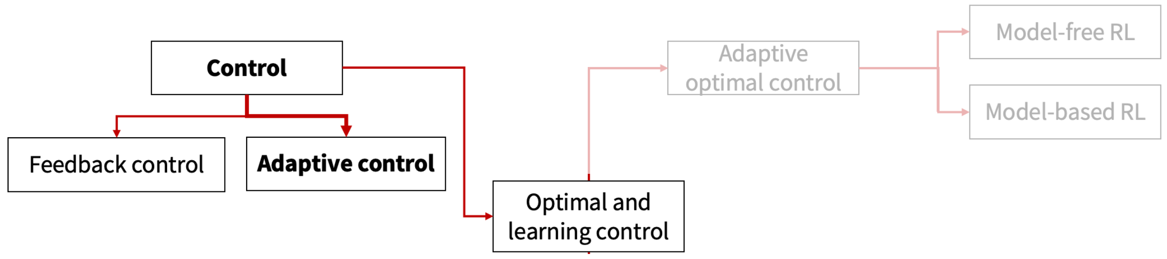
Intro to learning settings

System ID

Adaptive control

- MRAC: Model Reference Adaptive Control
- MIAC: Model Identification Adaptive Control

Short recap on RL



Markov Decision Process

State:	$x \in \mathcal{X}$
Action:	$u \in \mathcal{U}$
Transition function / Dynamics:	$T(x_t \mid x_{t-1}, u_{t-1}) = p(x_t \mid x_{t-1}, u_{t-1})$
Reward function:	$r_t = R(x_t, u_t) : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$
Discount factor:	$\gamma \in (0,1)$

Typically represented as a tuple

$$\mathcal{M} = (\mathcal{X}, \mathcal{U}, T, R, \gamma)$$

Goal: choose a policy that maximizes cumulative (discounted) reward

$$\pi^* = \underset{\pi}{\mathrm{arg}max}\ \mathbb{E}_p\left[\sum_{t \geq 0} \gamma^t R\big(x_t, \pi(x_t)\big)\right]$$

Review

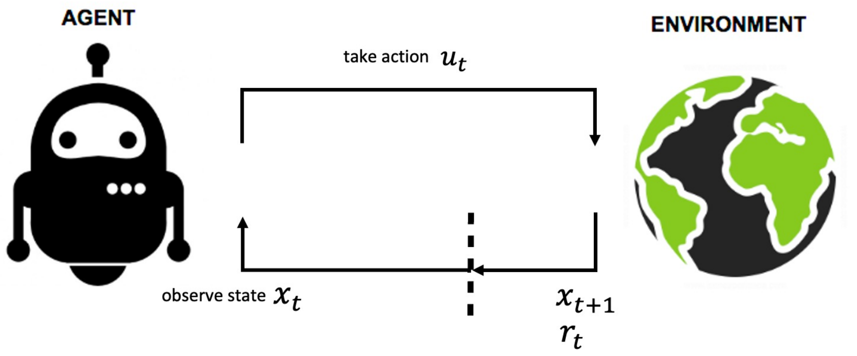
In previous lectures, we made the distinction between *prediction* (given a policy π, estimate V_π, Q_π) and *control* (learn the optimal policy π^*)

Motivated by Dynamic Programming, we discussed *exact methods* for solving MDPs:
- Policy Iteration
- Value Iteration

Problem	Bellman Equation	Algorithm
Prediction	Bellman Expectation Equation	Iterative Policy Evaluation
Control	Bellman Expectation Equation + Greedy Policy Improvement	Policy Iteration
Control	Bellman Optimality Equation	Value Iteration

Limitation: Update equations (i.e., Bellman equations) require access to dynamics model $T(x_{t+1} \mid x_t, u_t)$

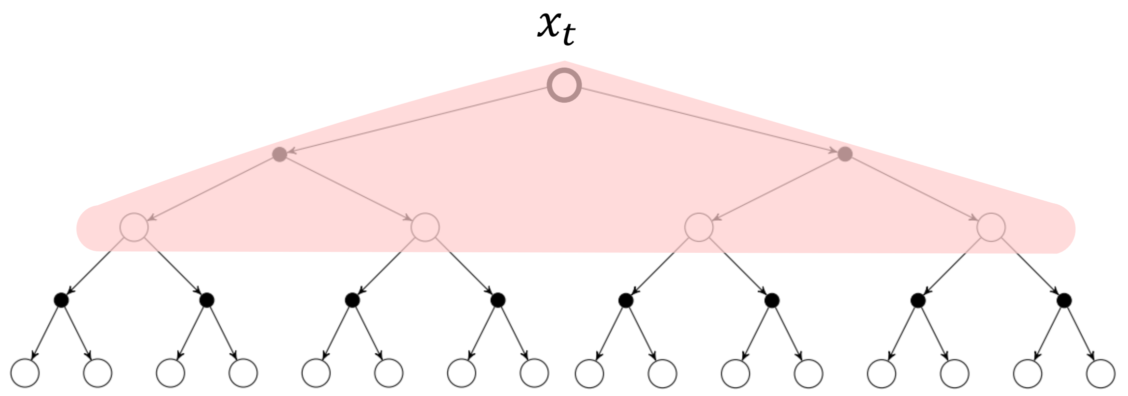
We saw how to use **sampling and bootstrapping** to approximate the expectations in the update equations:
- Monte Carlo (MC) Learning
- Temporal-Difference (TD) Learning



$$\tau = (x_0, u_0, \dots, x_N, u_N)$$

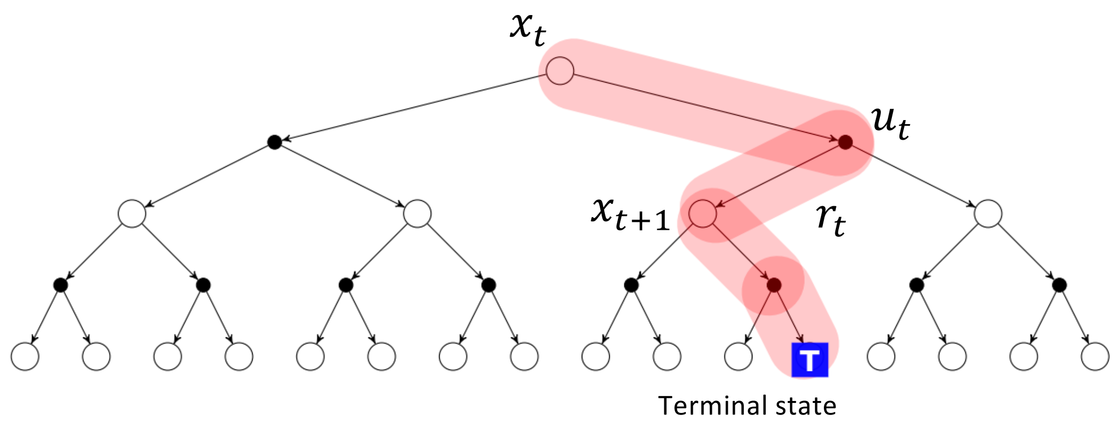
Dynamic Programming backup

$$\hat{V}(x_t) \leftarrow \mathbb{E}[R_t + \gamma \hat{V}(x_{t+1})]$$



Monte Carlo backup

$$\hat{V}(x_t) \leftarrow \hat{V}(x_t) + \alpha\left(G_t - \hat{V}(x_t)\right)$$

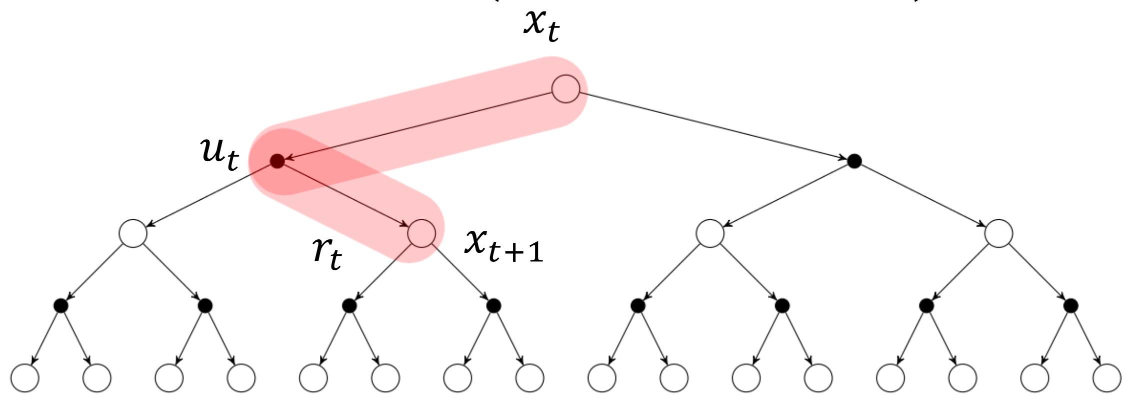


- **Sampling:** define the update through samples to approximate expectations
  - MC samples
  - TD samples
  - DP does not sample

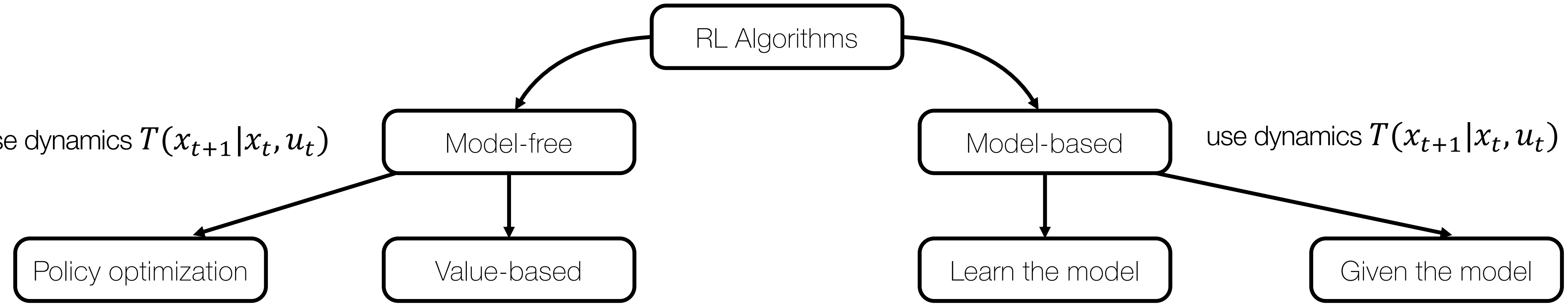
- **Bootstrapping:** define the update through an estimate
  - MC does not bootstrap
  - TD bootstraps
  - DP bootstraps

Temporal-Difference backup

$$\hat{V}(x_t) \leftarrow \hat{V}(x_t) + \alpha\left(R_t + \gamma \hat{V}(x_{t+1}) - \hat{V}(x_t)\right)$$



A taxonomy of RL



do not use dynamics $T(x_{t+1}|x_t, u_t)$

use dynamics $T(x_{t+1}|x_t, u_t)$

directly maximize the RL objective

$$\mathbb{E}_{\tau\sim p_\pi(\tau)}\left[\sum_{t=0}^{H}\gamma^t r(x_t, u_t)\right]$$

policy implicitly defined via $V(x)$ or $Q(x, u)$

$$\pi(x_t) = \arg\max_u Q(x_t, u_t)$$

Estimate $f_\theta \approx T(x_{t+1}|x_t, u_t)$

$T(x_{t+1}|x_t, u_t)$ is known

Next class

- Model-free Reinforcement Learning (Value-based Methods)