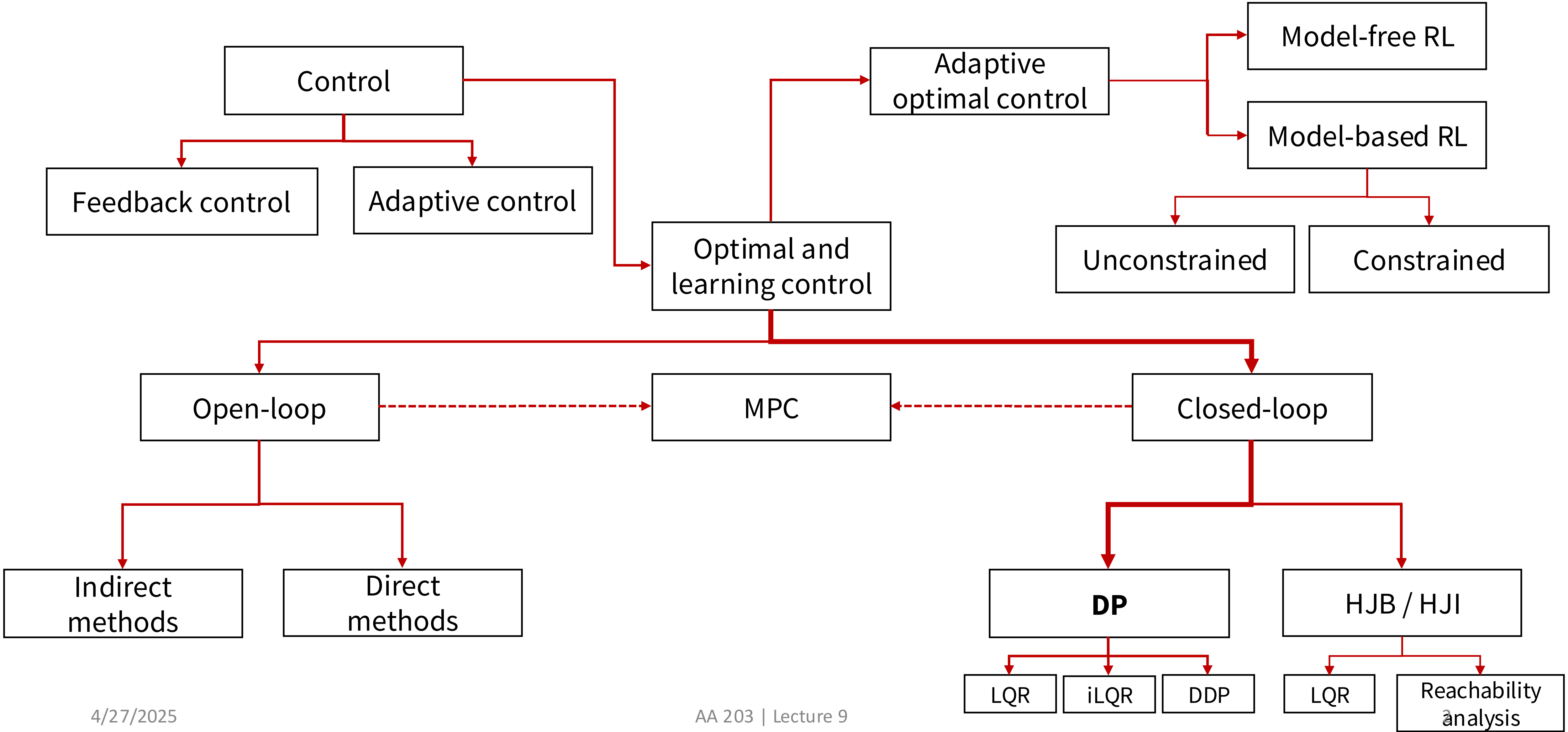
AA203
Optimal and Learning-based Control

Stochastic DP, value iteration, policy iteration

Stanford University

Autonomous Systems Lab

Roadmap


Control
Feedback control
Adaptive control
Optimal and learning control
Adaptive optimal control
Model-free RL
Model-based RL
Unconstrained
Constrained
Open-loop
MPC
Closed-loop
Indirect methods
Direct methods
DP
HJB / HJI
LQR
iLQR
DDP
LQR
Reachability analysis

Stochastic optimal control problem: Markov Decision Problem (MDP)

- System: $\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k), k = 0, \ldots, N-1$
- Control constraints: $\boldsymbol{u}_k \in U(\boldsymbol{x}_k)$
- Probability distribution: $\boldsymbol{w}_k \sim P_k(\cdot \mid \boldsymbol{x}_k, \boldsymbol{u}_k)$
- Policies: $\pi = \{\pi_0 \ldots, \pi_{N-1}\}$, where $\boldsymbol{u}_k = \pi_k(\boldsymbol{x}_k)$
- Expected Cost:

$$J_\pi(\boldsymbol{x}_0) = E_{\boldsymbol{w}_k, k=0,\ldots,N-1}\left[g_N(\boldsymbol{x}_N) + \sum_{k=0}^{N-1} g_k(\boldsymbol{x}_k, \pi_k(\boldsymbol{x}_k), \boldsymbol{w}_k)\right]$$

- Stochastic optimal control problem

$$J^*(x_0) = \min_\pi J_\pi(\boldsymbol{x}_0)$$

Key points

- Discrete-time model
- Markovian model
- Objective: find optimal closed-loop policy
- Additive cost (central assumption)
- Risk-neutral formulation

Key points

- Discrete-time model
- Markovian model
- Objective: find optimal closed-loop policy
- Additive cost (central assumption)
- Risk-neutral formulation

Other communities use different notation: Powell, W. B. *AI, OR and control theory: A Rosetta Stone for stochastic optimization*. Princeton University, 2012.

Principle of optimality

- Let $\pi^* = \{\pi_0^*, \pi_1^*, \dots, \pi_{N-1}^*\}$ be an optimal policy
- Consider tail subproblem

$$E\left[g_N(\boldsymbol{x}_N) + \sum_{k=i}^{N-1} g_k(\boldsymbol{x}_k, \pi_k(\boldsymbol{x}_k), \boldsymbol{w}_k)\right]$$

and the tail policy $\{\pi_i^*, \dots, \pi_{N-1}^*\}$

Principle of optimality: The tail policy is optimal for the tail subproblem

The DP algorithm (stochastic case)

Intuition

- DP first solves ALL tail subproblems at the final stage
- At generic step, it solves ALL tail subproblems of a given time length, using solution of tail subproblems of shorter length

The DP algorithm (stochastic case)

The DP algorithm

- Start with

$$J_N(\boldsymbol{x}_N) = g_N(\boldsymbol{x}_N)$$

and go backwards using

$$J_k(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k \in U(\boldsymbol{x}_k)} E_{w_k} \left[g_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k) + J_{k+1}\left(f(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k)\right) \right]$$

for $k = 0, 1, \ldots, N-1$

- Then $J^*(\boldsymbol{x}_0) = J_0(\boldsymbol{x}_0)$ and optimal policy is constructed by setting

$$\pi_k^*(\boldsymbol{x}_k) = \underset{\boldsymbol{u}_k \in U(\boldsymbol{x}_k)}{\operatorname{argmin}} E_{w_k} \left[g_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k) + J_{k+1}\left(\boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k)\right) \right]$$

Example: Inventory Control Problem

- Stock available $x_k \in \mathbb{N}$, inventory $u_k \in \mathbb{N}$, and demand $w_k \in \mathbb{N}$
- Dynamics: $x_{k+1} = \max(0, x_k + u_k - w_k)$
- Constraints: $x_k + u_k \leq 2$
- Probabilistic structure: $p(w_k = 0) = 0.1, p(w_k = 1) = 0.7$, and $p(w_k = 2) = 0.2$
- Cost

$$E\left[\underbrace{0}_{g_3(x_3)} + \sum_{k=0}^{2} \underbrace{(u_k + (x_k + u_k - w_k)^2)}_{g_k(x_k, u_k, w_k)} \right]$$

Example: Inventory Control Problem

- Algorithm takes form

$$J_k(x_k) = \min_{0 \leq u_k \leq 2 - x_k} E_{w_k}[u_k + (x_k + u_k - w_k)^2 + J_{k+1}(\max(0, x_k + u_k - w_k))]$$

for $k = 0, 1, 2$

- For example

$$J_2(0) = \min_{u_2 = 0,1,2} E_{w_2}[u_2 + (u_2 - w_2)^2] = \min_{u_2 = 0,1,2} u_2 + 0.1(u_2)^2 + 0.7(u_2 - 1)^2 + 0.2(u_2 - 2)^2$$

which yields $J_2(0) = 1.3$, and $\pi_2^*(0) = 1$

Example: Inventory Control Problem

Final solution:

- $J_0(0) = 3.7,$
- $J_0(1) = 2.7,$ and
- $J_0(2) = 2.818$

(see this spreadsheet)

Stochastic LQR

Find control policy that minimizes

$$E\left[\frac{1}{2}\boldsymbol{x}_N^T Q \boldsymbol{x}_N + \frac{1}{2}\sum_{k=0}^{N-1}\left(\boldsymbol{x}_k^T Q_k \boldsymbol{x}_k + \boldsymbol{u}_k^T R_k \boldsymbol{u}_k\right)\right]$$

subject to

- dynamics $\boldsymbol{x}_{k+1} = A_k \boldsymbol{x}_k + B_k \boldsymbol{u}_k + \boldsymbol{w}_k$

with $\boldsymbol{x}_0 \sim \mathcal{N}(\overline{\boldsymbol{x}_0}, \Sigma_{\boldsymbol{x}_0}), \{\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, \Sigma_{\boldsymbol{w}_k})\}$ independent and Gaussian vectors

Stochastic LQR

As before, let's suppose $J_{k+1}^*(\mathbf{x}_{k+1}) = \frac{1}{2}\mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}$ + c_{k+1}. Then (with a slight abuse, as we *neglect* the constant term since it does not affect the optimization):

$$\min_{\mathbf{u}_k} \mathbb{E}_{\mathbf{w}_k} \left[g_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) + J_{k+1}^*(f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)) \right]$$

$$= \min_{\mathbf{u}_k} \frac{1}{2} \mathbb{E}_{\mathbf{w}_k} \left[\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + (A_k \mathbf{x}_k + B_k \mathbf{u}_k + \mathbf{w}_k)^T P_{k+1} (A_k \mathbf{x}_k + B_k \mathbf{u}_k + \mathbf{w}_k) \right]$$

$$= \min_{\mathbf{u}_k} \frac{1}{2} \mathbb{E}_{\mathbf{w}_k} \Big[\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + (A_k \mathbf{x}_k + B_k \mathbf{u}_k)^T P_{k+1} (A_k \mathbf{x}_k + B_k \mathbf{u}_k)$$

$$2(A_k \mathbf{x}_k + B_k \mathbf{u}_k)^T P_{k+1} \mathbf{w}_k + \mathbf{w}_k^T P_{k+1} \mathbf{w}_k \Big]$$

$$= \min_{\mathbf{u}_k} \frac{1}{2} \left(\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + (A_k \mathbf{x}_k + B_k \mathbf{u}_k)^T P_{k+1} (A_k \mathbf{x}_k + B_k \mathbf{u}_k) + \mathrm{tr}(P_{k+1} \Sigma_{\mathbf{w}_k}) \right)$$

Stochastic LQR

As before, let's suppose $J^*_{k+1}(\mathbf{x}_{k+1}) = \frac{1}{2}\mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}$ + c_{k+1}. Then (with a slight abuse, as we *neglect* the constant term since it does not affect the optimization):

$$
\begin{aligned}
&\min_{\mathbf{u}_k} \mathbb{E}_{\mathbf{w}_k}\left[g_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) + J^*_{k+1}(f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k))\right] \\
&= \min_{\mathbf{u}_k} \frac{1}{2}\mathbb{E}_{\mathbf{w}_k}\left[\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + (A_k\mathbf{x}_k + B_k\mathbf{u}_k + \mathbf{w}_k)^T P_{k+1}(A_k\mathbf{x}_k + B_k\mathbf{u}_k + \mathbf{w}_k)\right] \\
&= \min_{\mathbf{u}_k} \frac{1}{2}\mathbb{E}_{\mathbf{w}_k}\left[\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + (A_k\mathbf{x}_k + B_k\mathbf{u}_k)^T P_{k+1}(A_k\mathbf{x}_k + B_k\mathbf{u}_k)\right. \\
&\qquad\qquad\qquad \left. 2(A_k\mathbf{x}_k + B_k\mathbf{u}_k)^T P_{k+1}\mathbf{w}_k + \mathbf{w}_k^T P_{k+1}\mathbf{w}_k\right] \\
&= \min_{\mathbf{u}_k} \frac{1}{2}\left(\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + (A_k\mathbf{x}_k + B_k\mathbf{u}_k)^T P_{k+1}(A_k\mathbf{x}_k + B_k\mathbf{u}_k) + \mathrm{tr}(P_{k+1}\Sigma_{\mathbf{w}_k})\right)
\end{aligned}
$$

➔ optimal policy is the same as in the deterministic case; cost-to-go is increased by some constant related to magnitude of noise

Infinite Horizon MDPs

State: $x \in \mathcal{X}$ (often $s \in \mathcal{S}$)

Action: $u \in \mathcal{U}$ (often $a \in \mathcal{A}$)

Transition Function: $T(x_t \,|x_{t-1}\,, u_{t-1}) = p(x_t|x_{t-1}, u_{t-1})$

Reward Function: $r_t = \; R(x_t, u_t)$

Discount Factor: γ

MDP (stationary model): $\mathcal{M} = (\mathcal{X}, \mathcal{U}, T, R, \gamma)$

Infinite Horizon MDPs

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

<u>Stationary</u> policy: $u_t = \pi(x_t)$

Goal: Choose policy that **maximizes cumulative (discounted) reward**

$$V^* = \max_{\pi} E\left[\sum_{t\geq 0} \gamma^t R\big(x_t, \pi(x_t)\big)\right];$$

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

Infinite Horizon MDPs

- The optimal value function $V^*(x)$ satisfies Bellman's equation

$$V^*(x) = \max_u \left(R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u)\, V^*(x') \right)$$

- For any stationary policy π, the values $V_\pi(x) \coloneqq E\left[\sum_{t \geq 0} \gamma^t R\big(x_t, \pi(x_t)\big) \middle| \, x_0 = x\right]$ are the unique solution to the equation

$$V_\pi(x) = R(x, \pi(x)) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x, \pi(x))\, V_\pi(x')$$

State-action value functions (Q functions)

- The expected cumulative discounted reward starting from x, applying u, and following the optimal policy thereafter

$$V^*(x) = \max_u \left(\underbrace{R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u)\, V^*(x')}_{Q^*(x,u)} \right)$$

- The optimal Q function, $Q^*(x,u)$, satisfies Bellman's equation

$$Q^*(x,u) = R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u) \max_{u'} Q^*(x',u')$$

- For any stationary policy π, the corresponding Q function satisfies

$$Q_\pi(x,u) = R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u)\, Q_\pi(x',\pi(x'))$$

Solving infinite-horizon MDPs

If you know the model (i.e., the transition function T and reward function R), use ideas from dynamic programming

- Value Iteration / Policy Iteration

Reinforcement Learning: learning from interaction

- Model-based
- Model-free

Solving infinite-horizon MDPs

If you know the model (i.e., the transition function T and reward function R), use ideas from dynamic programming

- Value Iteration / Policy Iteration

Reinforcement Learning: learning from interaction

- Model-based
- Model-free

Value Iteration

- Initialize $V_0(x) = 0$ for all states x
- Loop until finite horizon / convergence:

$$V_{k+1}(x) = \max_u \left(R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u)\, V_k(x') \right)$$

- Value iteration for Q functions

$$Q_{k+1}(x,u) = R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u) \max_{u'} Q_k(x',u')$$

Policy Iteration

Starting with a policy $\pi_k(x)$, alternate two steps:

1. <u>Policy Evaluation</u>
   Compute $V_{\pi_k}(x)$ as the solution of
   $$V_{\pi_k}(x) = R(x, \pi_k(x)) + \gamma \sum_{x' \in \mathcal{X}} T(x' | x, \pi(x))\, V_{\pi_k}(x')$$
2. <u>Policy Improvement</u>
   Define $\pi_{k+1}(x) = \arg\max_{u} \left(R(x, u) + \gamma \sum_{x' \in \mathcal{X}} T(x' | x, u)\, V_{\pi_k}(x') \right)$

Proposition: $V_{\pi_{k+1}}(x) \geq V_{\pi_k}(x)\ \forall\, x \in \mathcal{X}$

Inequality is strict if π_k is suboptimal

Use this procedure to iteratively improve policy until convergence

Recap

- Value Iteration
  - Estimate optimal value function
  - Compute optimal policy from optimal value function
- Policy Iteration
  - Start with random policy
  - Iteratively improve it until convergence to optimal policy
- Requires **model of MDP** to work!

Next time

- HJB, HJI
- Reachability analysis