CS 295:
Optimal Control and Reinforcement Learning
Winter 2020

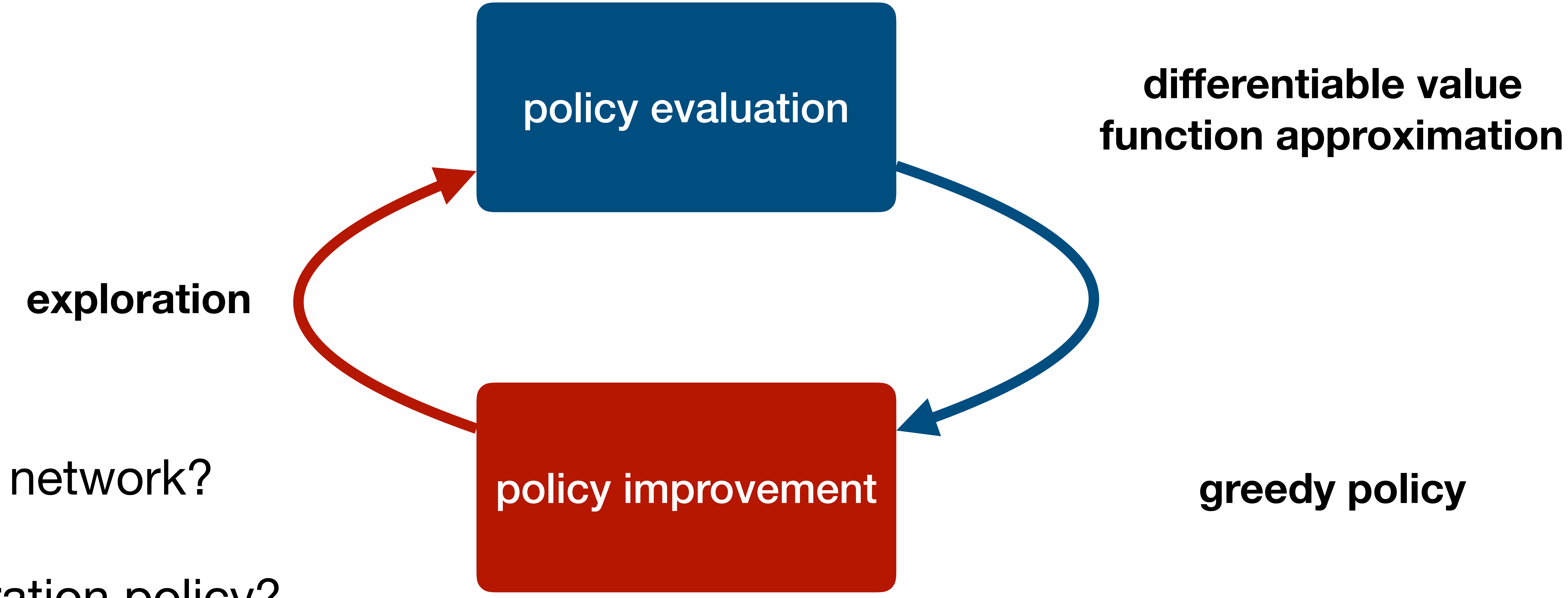
Lecture 6: Policy-Gradient Methods

Roy Fox
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine

Today's lecture

- DQN in practice
- Policy Gradient (PG) methods
  - REINFORCE
- Variance reduction
  - Baselines
  - Reward-to-go

Deep Q-Learning (DQN)



- Target network?
- Exploration policy?
- Replay buffer?

DQN pseudocode

Algorithm 1 DQN

initialize θ for Q_θ, set $\bar{\theta} \leftarrow \theta$
for each step **do**
    if new episode, reset to s_0
    observe current state s_t
    take ϵ-greedy action a_t based on $Q_\theta(s_t, \cdot)$

$$\pi(a_t|s_t) = \begin{cases} 1 - \frac{|\mathcal{A}|-1}{|\mathcal{A}|}\epsilon & a_t = \operatorname{argmax}_a Q_\theta(s_t, a) \\ \frac{1}{|\mathcal{A}|}\epsilon & \text{otherwise} \end{cases}$$

    get reward r_t and observe next state s_{t+1}
    add (s_t, a_t, r_t, s_{t+1}) to replay buffer $\mathcal{D}$
    for each (s, a, r, s') in minibatch sampled from $\mathcal{D}$ **do**

$$y \leftarrow \begin{cases} r & \text{if episode terminated at } s' \\ r + \gamma \max_{a'} Q_{\bar{\theta}}(s', a') & \text{otherwise} \end{cases}$$

        compute gradient $\nabla_\theta (y - Q_\theta(s, a))^2$
    take minibatch gradient step
    every K steps, set $\bar{\theta} \leftarrow \theta$

Value estimation bias

- Q-value estimation is optimistically biased
- Jensen's inequality: $\mathbb{E}[\max_a Q(a)] \geqslant \max_a \mathbb{E}[Q(a)]$
- While there's uncertainty in $Q_{\bar{\theta}}$, y is more positively biased than $Q_{\bar{\theta}}$
- So how can this converge?
  - As certainty increases, new bias decreases
  - Old bias attenuates with repeated discounting by γ

Double Q-Learning

- One solution: keep two estimates of Q^*, Q_0 and Q_1
- Target for $Q_i(s, a)$:

$$y_i = r + \gamma Q_{1-i}(s', \operatorname*{argmax}_{a'} Q_i(s', a'))$$

- How to use this with DQN?
- One idea: use target network as the other estimate

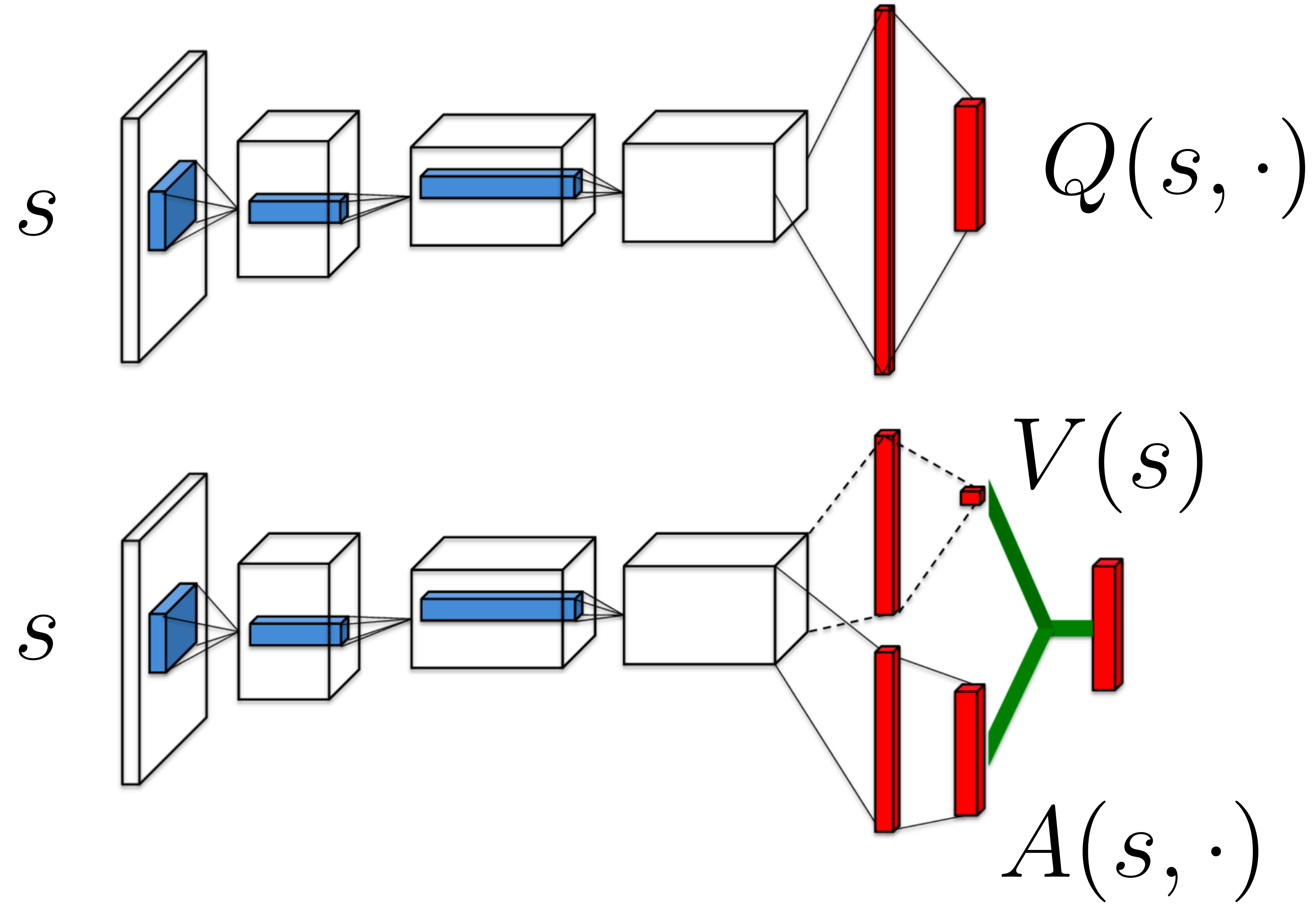
$$y = r + \gamma Q_{\bar{\theta}}(s', \operatorname*{argmax}_{a'} Q_{\theta}(s', a'))$$

- Another idea: Clipped Double Q-Learning

$$y_i = r + \gamma \min_{i=1,2} Q_{\bar{\theta}_i}(s', \operatorname*{argmax}_{a'} Q_{\theta_i}(s', a'))$$

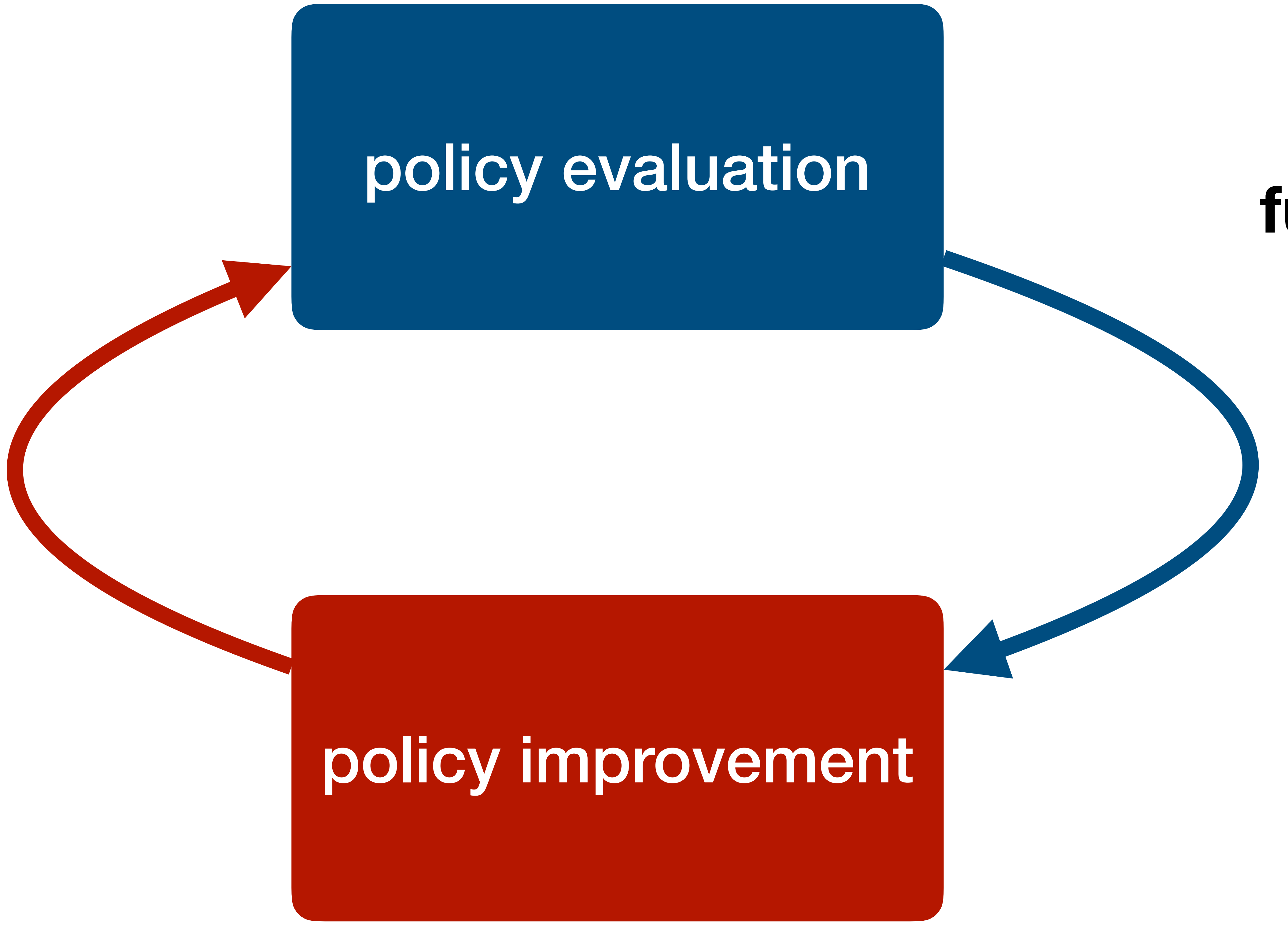
Dueling Networks

- Advantage function: $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$



- Stabilize with $Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|\mathcal{A}|} \sum_{\bar{a}} A(s,\bar{a}) \right)$

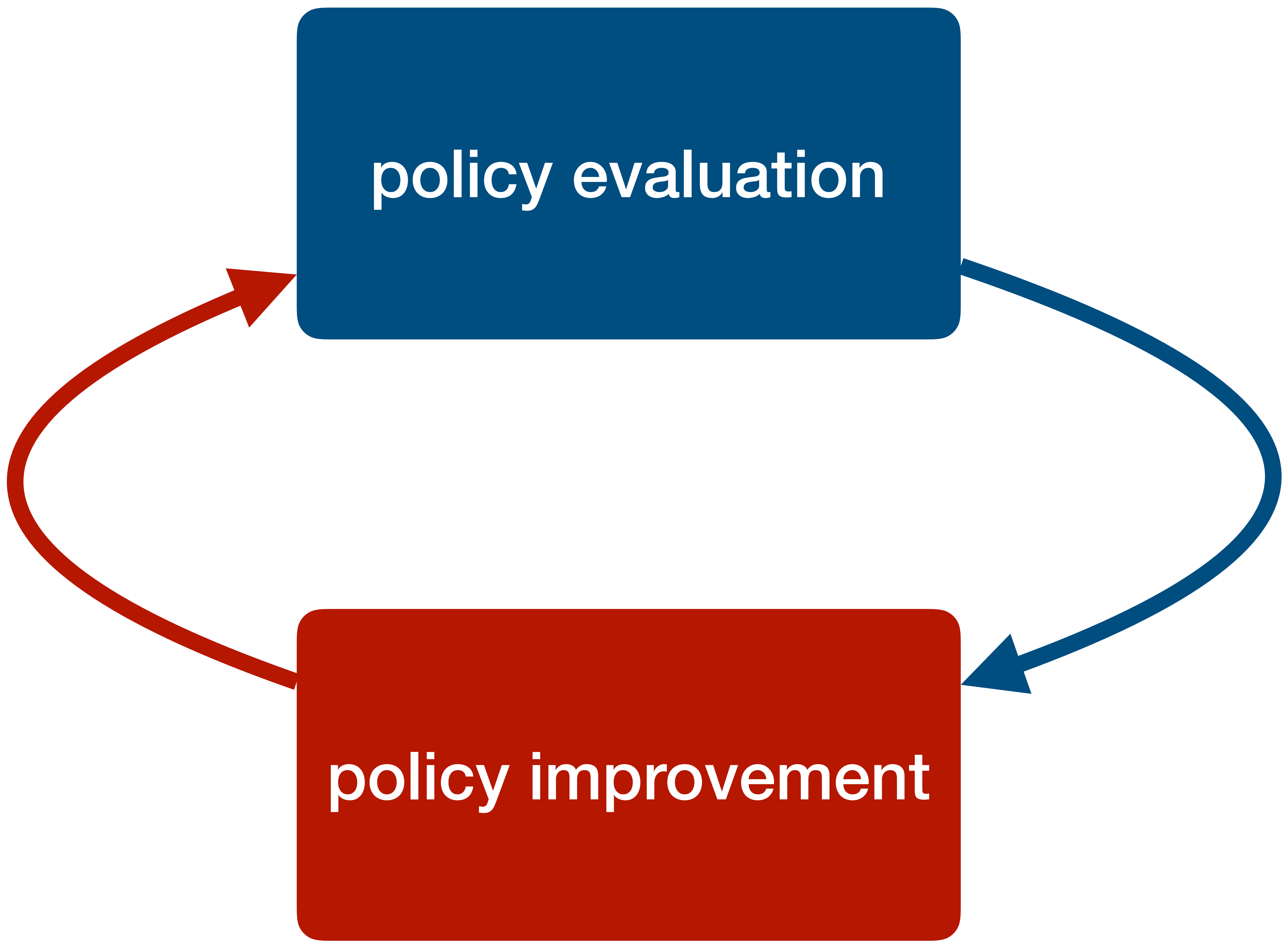
Value-based methods (e.g. DQN)



differentiable value function approximation

greedy policy

Policy-based methods



e.g. MC

differentiable policy

Policy Gradient (PG)

- Unlike minimizing $\mathcal{L}_\theta(\mathcal{D})$ in general ML, in RL we maximize $\mathcal{J}_\theta = \mathbb{E}_{\xi \sim p_{\pi_\theta}}[R]$
- This is harder since the "data" distribution depends on θ
- But there's a trick: $\nabla_\theta \log p_\theta(\xi) = \frac{1}{p_\theta(\xi)} \nabla_\theta p_\theta(\xi)$
- And so:

$$
\begin{aligned}
\nabla_\theta \mathcal{J}_\theta &= \nabla_\theta \int p_\theta(\xi) R(\xi) d\xi \\
&= \int p_\theta(\xi) \nabla_\theta \log p_\theta(\xi) R(\xi) d\xi \\
&= \mathbb{E}_{\xi \sim p_\theta}[\nabla_\theta \log p_\theta(\xi) R]
\end{aligned}
$$

REINFORCE (1992 !)

- Roll out π_θ to sample $\xi \sim p_\theta$
- Compute R and

$$\nabla_\theta \log p_\theta(\xi) = \nabla_\theta(\log p(s_0) + \sum_t (\log \pi_\theta(a_t|s_t) + \log p(s_{t+1}|s_t, a_t)))$$

- Take a gradient step with $\nabla_\theta \log p_\theta(\xi) R$
- Repeat

- This is model-free! but on-policy, + high variance of the gradient estimator

PG with Gaussian policy

- As an example in continuous action spaces: $\pi_\theta(a|s) = \mathcal{N}(\mu_\theta(s), \Sigma)$
- So that

$$\log p_\theta(\xi) = \sum_t \log \pi_\theta(a_t|s_t) + \text{const} = -\frac{1}{2}\sum_t \|a_t - \mu_\theta(s_t))\|^2_{\Sigma^{-1}} + \text{const}$$

  - Where $\|x\|^2_P = x^T P x$

- Then

$$\nabla_\theta \log p_\theta(\xi) = \sum_t \Sigma^{-1}(a_t - \mu_\theta(s_t))R\,\partial_\theta\,\mu_\theta(s_t)$$

PG: the good and the bad

$$\nabla_\theta \mathcal{J}_\theta = \mathbb{E}_{\xi \sim p_\theta}\left[\left(\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t)\right) R\right]$$

- $-\log \pi_\theta(a|s)$ is sometimes called "surprisal"
- We update θ towards being less surprised by high return
- But surprisal can get very large for unlikely actions
  - Gradient estimator has high variance when unlikely actions can have high return
  - Particularly if our policy tries to converge to deterministic

Baselines

- Constant shifts in return shouldn't matter for optimal policy

$$0 = \nabla_\theta \mathbb{E}_{\xi \sim p_\theta}[b] = \mathbb{E}_{\xi \sim p_\theta}[\nabla_\theta \log p_\theta(\xi) b]$$

- Can we use that to reduce variance **without adding bias**?

- Using the average return works pretty well in practice

$$\nabla_\theta \mathcal{J}_\theta \approx \frac{1}{N} \sum_i \nabla_\theta \log p_\theta(\xi_i)(R_i - b)$$

- With $b = \frac{1}{N} \sum_i R_i$

Optimal baseline

- Denote $g(\xi) = \nabla_\theta \log p_\theta(\xi)$
- Then $\partial_b \operatorname{Var}(\nabla_\theta \log p_\theta(\xi)(R-b))$

$$= \partial_b(\mathbb{E}[g^2(R-b)^2] - \mathbb{E}[g(R-b)]^2)$$

$$= \partial_b(\mathbb{E}[g^2R^2] - \mathbb{E}[gR]^2 - 2b\,\mathbb{E}[g^2R] + b^2\,\mathbb{E}[g^2])$$

$$= -2\,\mathbb{E}[g^2R] + 2b\,\mathbb{E}[g^2]$$

- Optimally: $b = \dfrac{\mathbb{E}[g^2R]}{\mathbb{E}[g^2]}$

Don't let the past distract you

$$\nabla_\theta \mathcal{J}_\theta = \mathbb{E}_{\xi \sim p_\theta}\left[\left(\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t)\right) R\right] = \sum_t \mathbb{E}_{s_t \sim p_\theta}[\nabla_\theta \, \mathbb{E}_{a_t|s_t \sim \pi_\theta}[R]]$$

- In our case

$$R_{\geqslant t} = \sum_{t' \geqslant t} \gamma^{t'} r(s_{t'}, a_{t'})$$

is a sufficient statistic of R

- Therefore, a lower-variance gradient estimator:

$$\sum_t \gamma^t \, \mathbb{E}_{s_t \sim p_\theta}[\mathbb{E}_{a_t|s_t \sim \pi_\theta}[\nabla_\theta \log \pi_\theta(a_t|s_t) R_{\geqslant t}]]$$

Recap

- Practical RL algorithms add tricks and heuristics to the theory
- We can take the gradient of our objective w.r.t. the policy parameters
- This often leads to high variance
- Variance can be reduced by baselines and other tricks