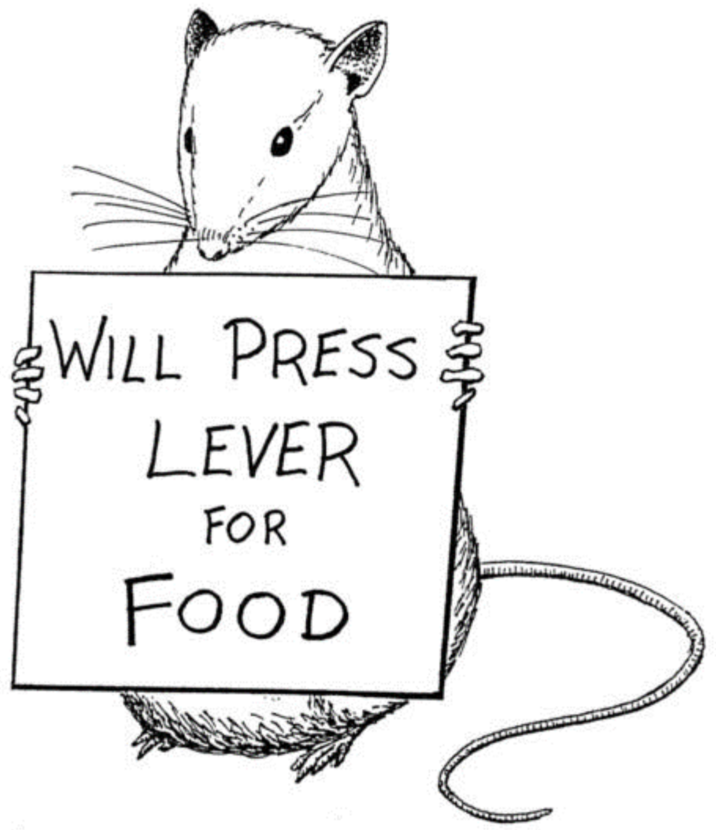
UCI University of California, Irvine

CS 277: Control and Reinforcement Learning

Winter 2021

Lecture 4: Policy-Gradient Methods

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



Logistics

assignments

- Assignment 1, practical part to be published shortly
  - Both parts due next Friday

enrollment

- Enrollment (and dropping) is now open to all

Today's lecture

DQN Tricks

Policy Gradient Methods

Variance Reduction

Deep TD reinforcement learning

- Deep Q Learning (historically called DQN):

$$\mathscr{L}_\theta(s,a,r,s') = (r + \gamma \max_{a'} Q_{\bar{\theta}}(s',a') - Q_\theta(s,a))^2$$

- This algorithm should work off-policy, so we can keep past experience
  - Replay buffer = data set of recent past experience of learner policy at that time
  - Variants differ on
    - How to add experience to the buffer
    - How to sample from the buffer

Interaction policy

- In model-free RL, we often get data by interaction with the environment
  - How should we interact?
  - Must we use current learner policy (on-policy data) or another (off-policy data)
- On-policy methods (e.g. MC): must use current policy
- Off-policy methods: can use different policy — but not too different!
  - Otherwise may have train–test distribution mismatch
- In either case, must make sure interaction policy explores well enough

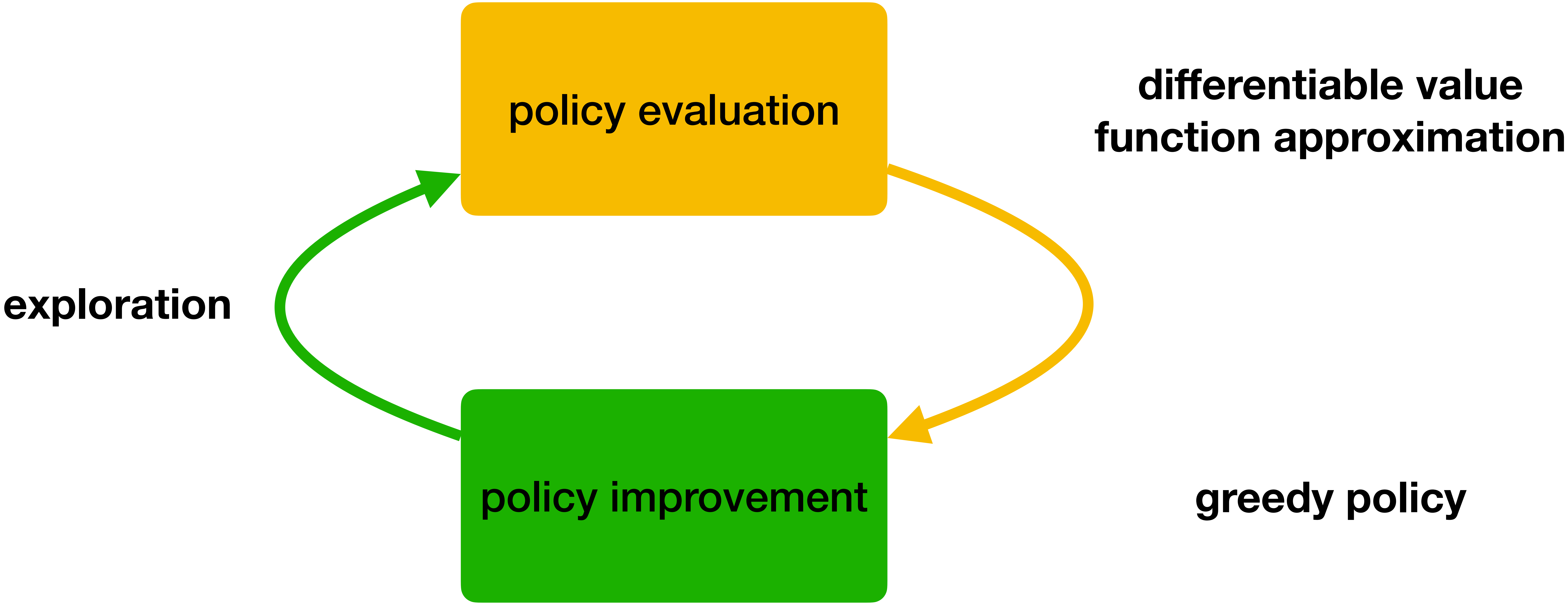
Exploration policies

- ϵ-greedy exploration: select uniform action w.p. ϵ, otherwise greedy
- Boltzmann exploration:

$$\pi(a \mid s) = \mathrm{sm}_a(Q(s,a);\beta) = \frac{\exp(\beta Q(s,a))}{\sum_{\bar{a}} \exp(\beta Q(s,\bar{a}))}$$

  - Becomes uniform as $\beta \to 0$, greedy as $\beta \to \infty$

Putting it all together: DQN



Recap

- RL is a (policy evaluation $\leftrightarrow$ policy improvement) loop
- Temporal-Difference methods exploit the dynamical-programming structure
- Off-policy methods throw out data much less often when policy changes
- Many approaches can be made differentiable for Deep RL

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: $\mathbb{E}_f[\max_a f(a)] \geq \max_a \mathbb{E}_f[f(a)]$ ($\mathbb{E}$ over randomness of f)
- While there's uncertainty in $Q_{\bar{\theta}}$, $\max_{a'} Q_{\bar{\theta}}(s', a')$ is positively biased
- 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 $Q_i(s, a)$:

$$y_i = r + \gamma Q_{1-i}(s', \arg\max_{a'} Q_i(s', a'))$$

- How to use this with DQN?

- One idea: use target network as the other estimate

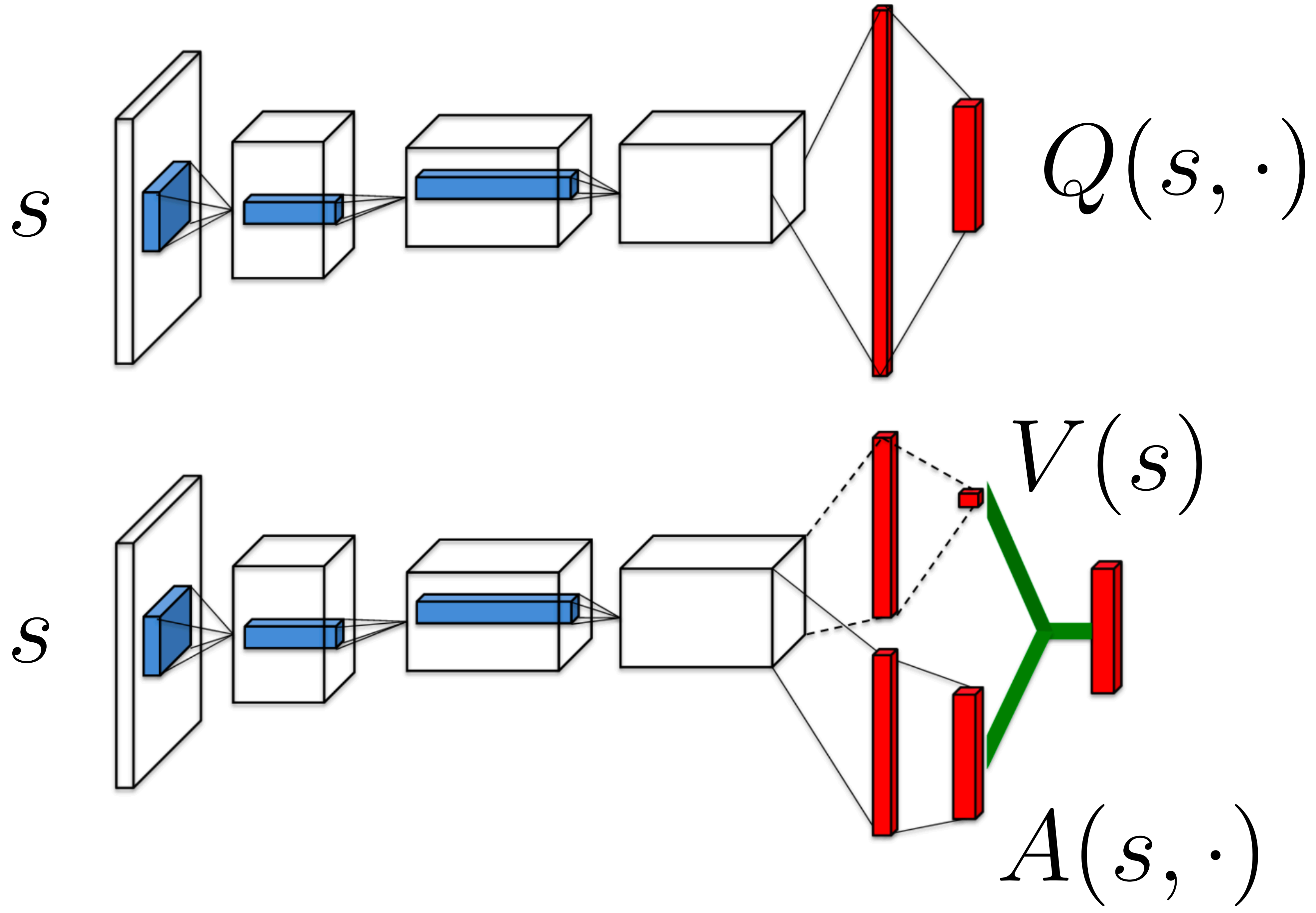
$$y = r + \gamma Q_{\bar{\theta}}(s', \arg\max_{a'} Q_\theta(s', a'))$$

- Another idea: Clipped Double Q-Learning

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

Dueling Networks

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



- Issue: $Q = (V + c) + (A - c)$ is underdetermined

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

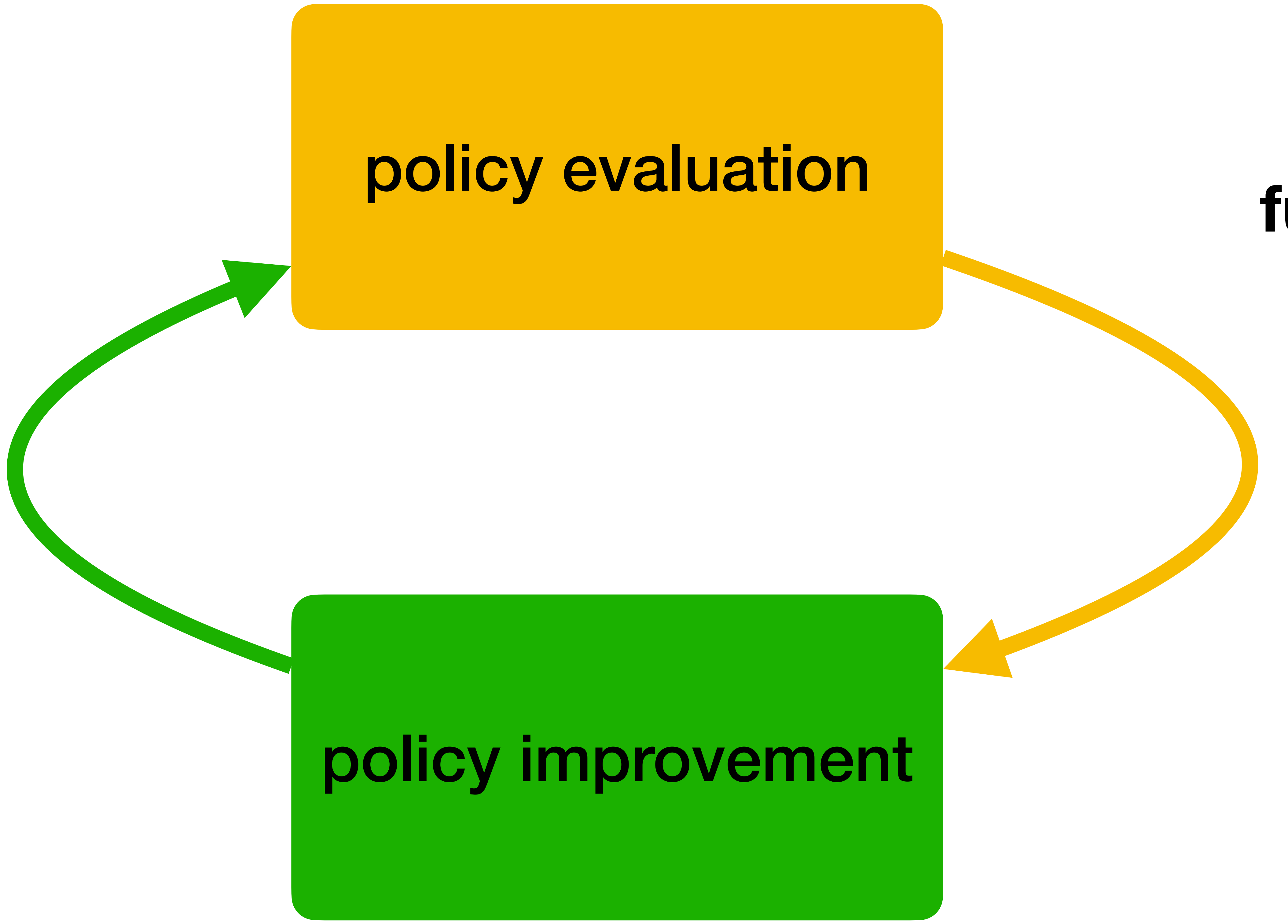
Today's lecture

DQN Tricks

Policy Gradient Methods

Variance Reduction

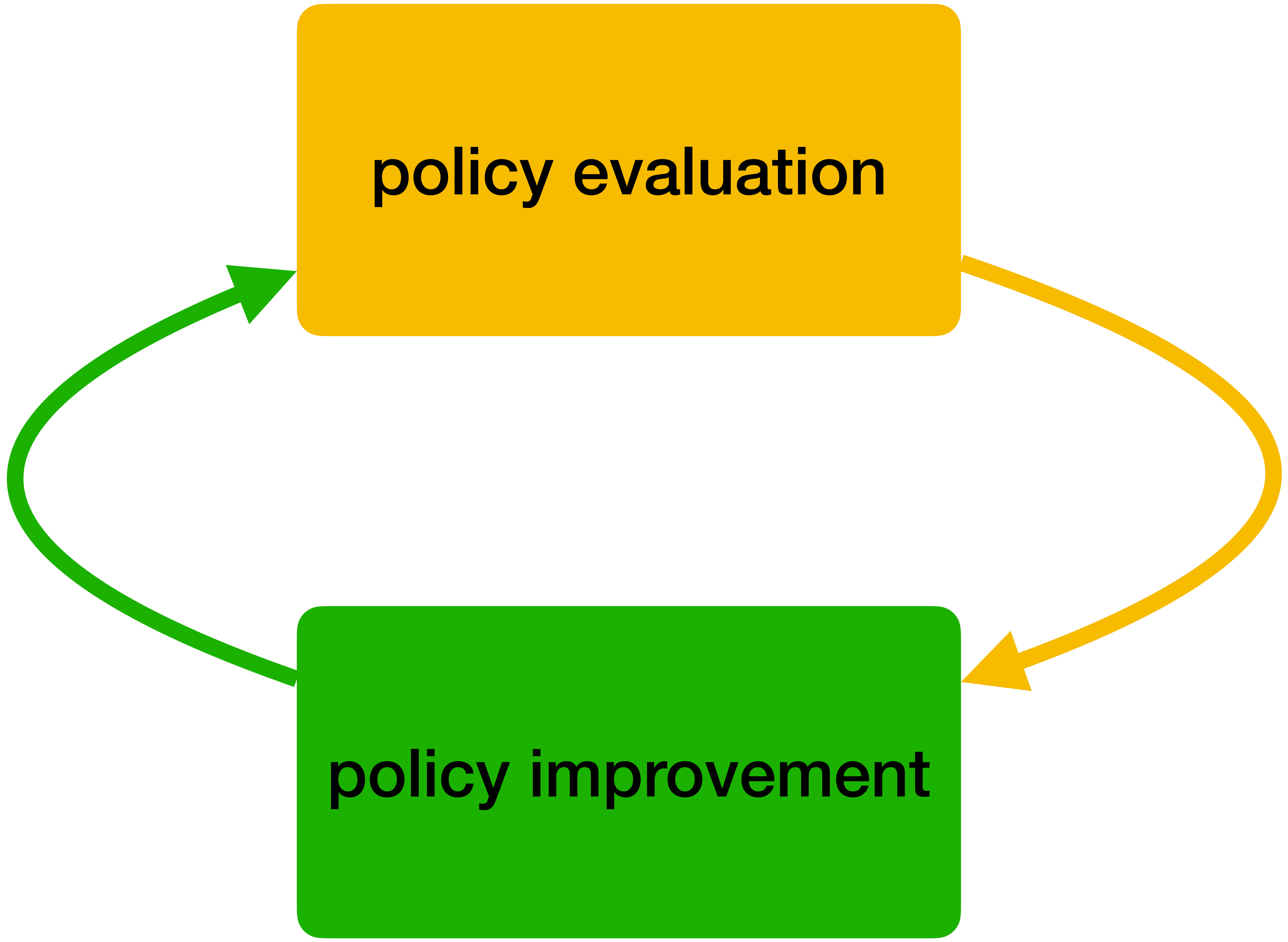
Value-based methods (e.g. DQN)



differentiable value function approximation

greedy policy

Policy-based methods



e.g. MC

differentiable policy

Policy Gradient (PG)

- Unlike minimizing $\mathscr{L}_\theta(\mathscr{D})$ 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: $\nabla_\theta \log p_\theta(\xi) = \dfrac{1}{p_\theta(\xi)} \nabla_\theta p_\theta(\xi)$
- Log-derivative / score-function / REINFORCE trick: estimate gradient using samples of $p_\theta(\xi)$

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

REINFORCE (1992 !)

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

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

- 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: $\pi_\theta(a \mid s) = \mathcal{N}(\mu_\theta(s), \Sigma)$
- So that

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

  - Where $\|x\|^2_P = x^\intercal P x$ is the Mahalanobis norm

- Then

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

PG: the good and the bad

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

- $-\log \pi_\theta(a \mid s)$ 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 / lower-support

Today's lecture

DQN Tricks

Policy Gradient Methods

Variance Reduction

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

$$
\begin{aligned}
\partial_b \mathrm{var}(\nabla_\theta \log p_\theta(\xi)(R-b))& \\
&= \partial_b(\mathbb{E}[g^2(R-b)^2] - \mathbb{E}[g(R-b)]^2) \\
&= \partial_b(\mathbb{E}[g^2R^2] - \mathbb{E}[gR] - 2b\mathbb{E}[g^2R] + b^2\mathbb{E}[g^2]) \\
&= -2\mathbb{E}[g^2R] + 2b\mathbb{E}[g^2]
\end{aligned}
$$

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

Rao–Blackwell theory

- Suppose we use data x to generate an estimate $\hat{\theta}(x)$ of parameter θ
- Let y be a sufficient statistic of x for θ
  - That is, there's nothing more, on top of y, that x can tell us about θ
- Consider the estimator $\hat{\theta}(y) = \mathbb{E}[\hat{\theta}(x) \,|\, y]$ of θ
  - It has the same bias as $\hat{\theta}(x)$, and lower variance
  - Which also means it has lower MSE

Don't let the past distract you

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

- In our case, $R_{\geq t} = \sum_{t' \geq t} \gamma^{t'} r(s_{t'}, a_{t'})$ is a sufficient statistic of R for $\mathcal{J}_\theta$
- Therefore, a lower-variance gradient estimator:

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

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