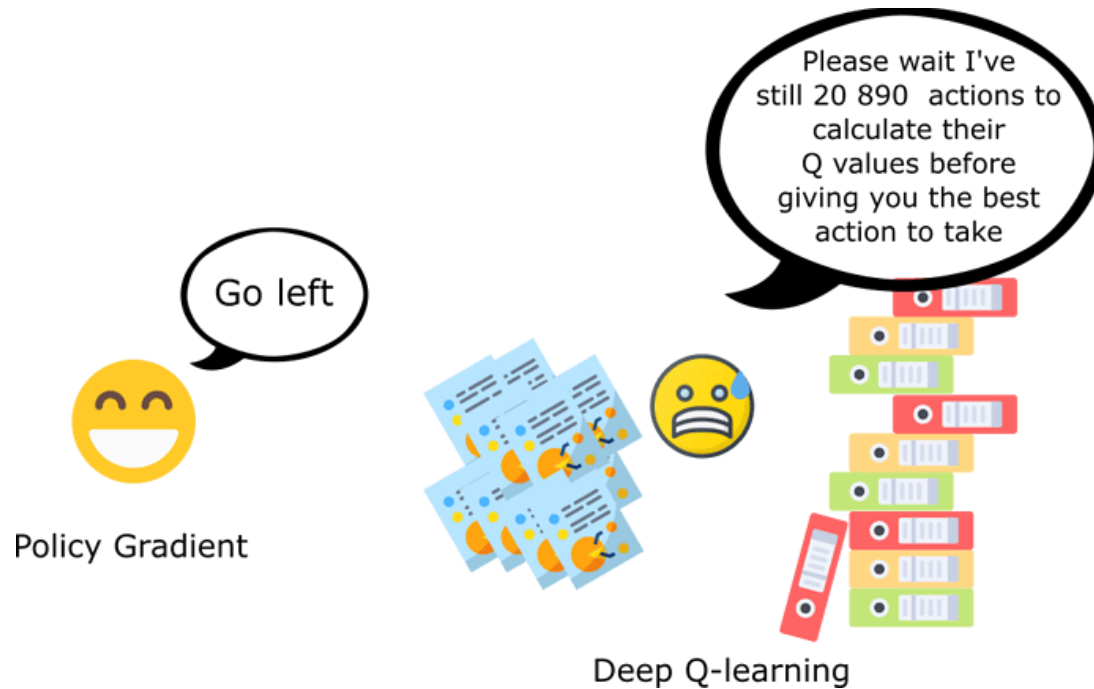


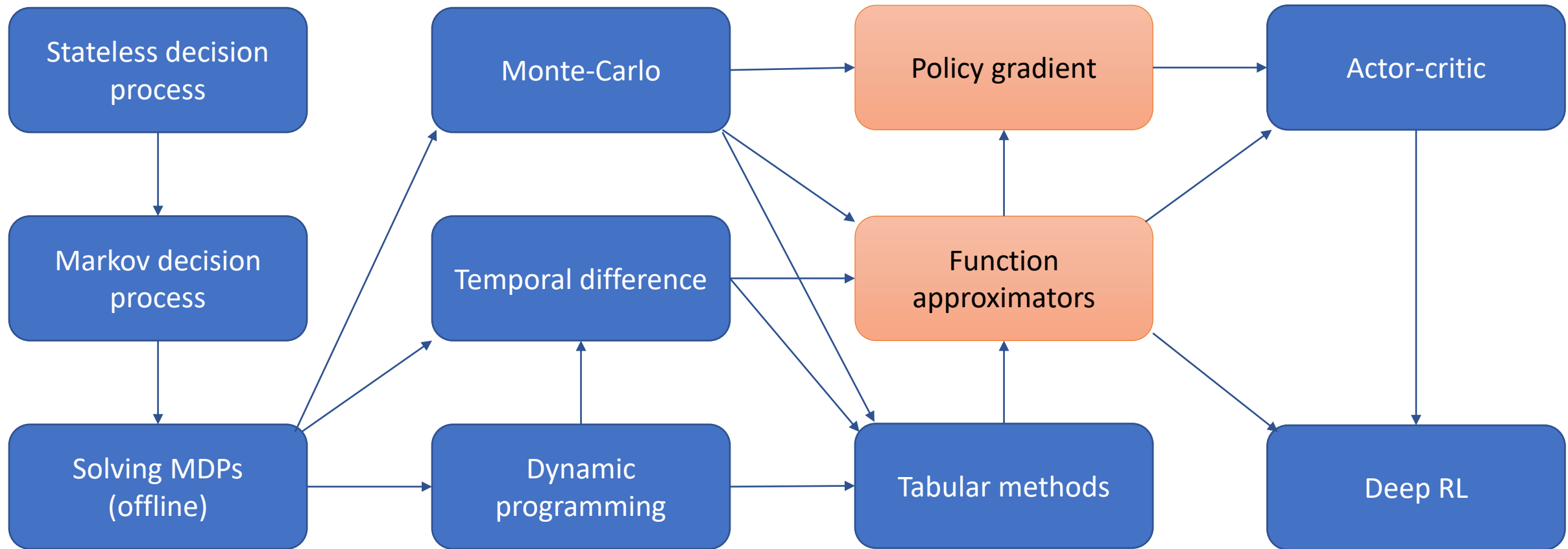
CSCE-642 Reinforcement Learning

Chapter 13: Policy Gradient



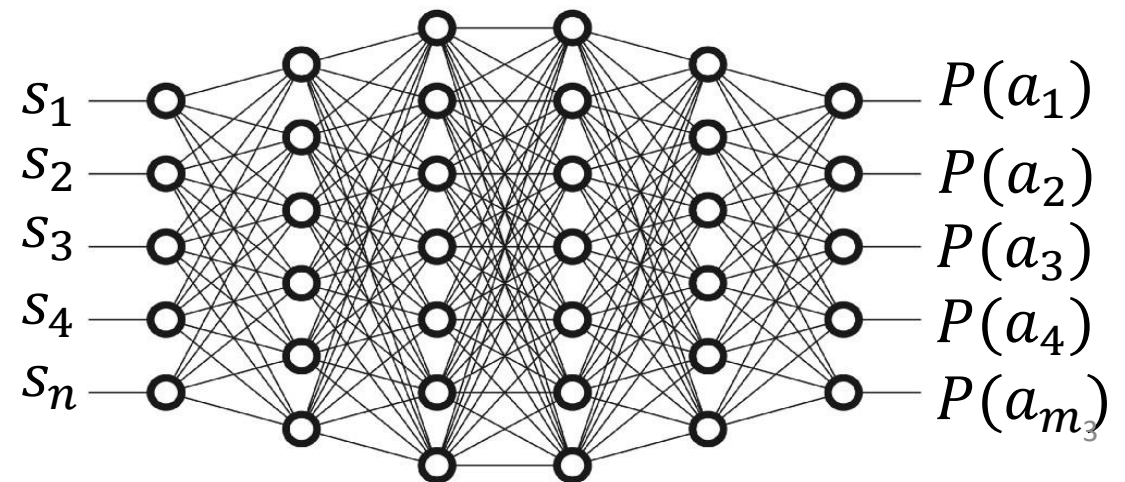
Instructor: Guni Sharon

CSCE-689, Reinforcement Learning



Notation

- Soft policy as a parametrized function: $\pi_{\theta}(a|s)$
 - For a policy gradient we require a continuous, differentiable policy... (Softmax activation can be useful in discrete action spaces)
 - $\pi(a|s)$ is assumed to be differentiable with respect to θ
 - E.g., a DNN where θ is the set of weights and biases
- $J(\theta)$ is a scalar policy performance measure (expected sum of discounted rewards) with respect to the policy params
 - $= \mathbb{E}_{\tau \sim \pi_{\theta}} [\sum_t \gamma^t R_t]$



Improving the policy

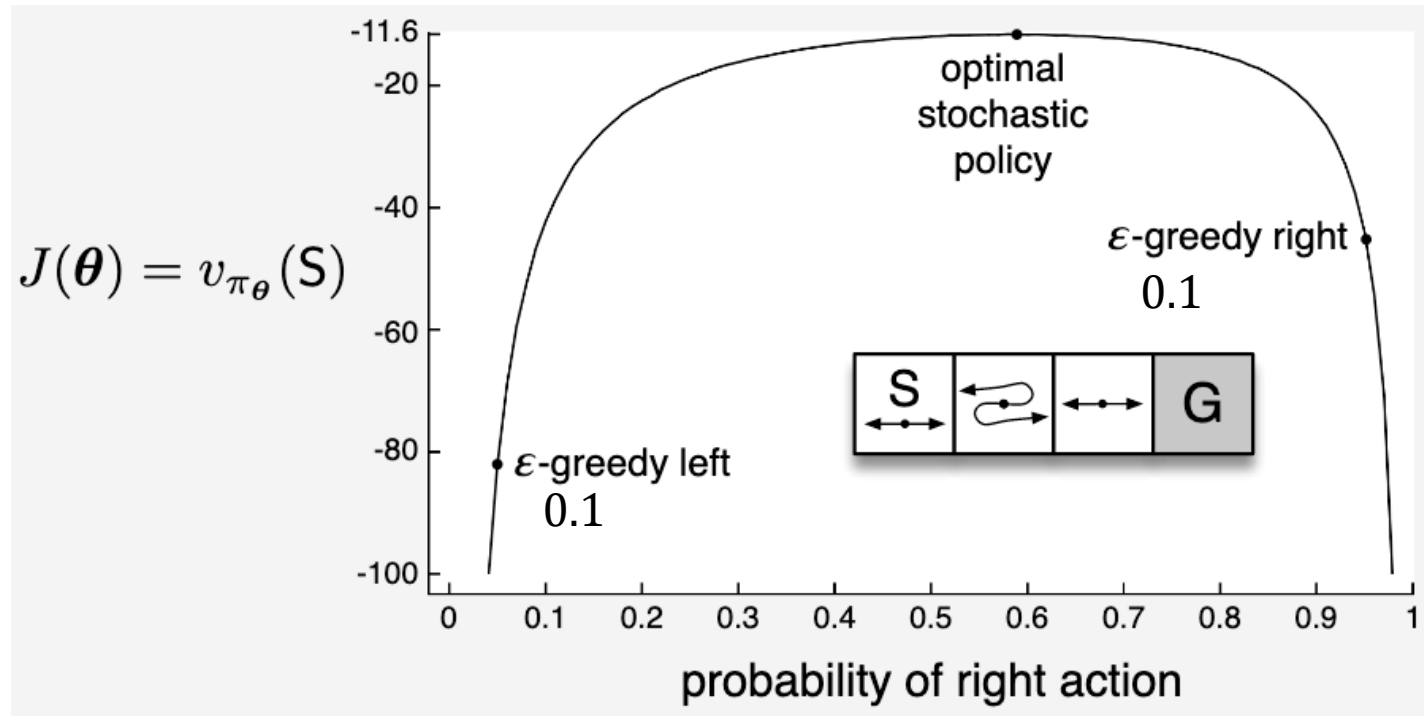
- SGD: $\theta_{t+1} = \theta_t + \alpha \widehat{\nabla J(\theta_t)}$
- Where $\widehat{\nabla J(\theta_t)}$ is a stochastic estimate, whose expectation approximates the gradient of the performance measure
- All methods that follow this general schema we call policy gradient methods
 - Might also include a value approximation for normalization (baseline)
- Methods that use approximations of both policy and value functions for computing the policy's gradient are called actor–critic methods (more on this later)

Advantages of PG

- The policy convergence over time as opposed to an epsilon-greedy value-based approach
- Naturally applies to continuous action space as opposed to a Q-learning approach
- In many domains the policy is a simpler function to approximate
 - Though this is not always the case
- Choice of policy parameterization is sometimes a good way of injecting prior knowledge
 - E.g., initial cycle assignment for a signal controller
- Can converge on stochastic optimal policies, as opposed to value-based approaches
 - Useful in games with imperfect information where the optimal play is often inherently random, e.g., bluffing in Poker

Stochastic policy

- Reward = -1 per step
- $\mathcal{A} = \{left, right\}$
- Left goes left and right goes right except in the middle state where they are reversed
- States are identical from the policy's perspective
- $\pi^* = [0.41 \quad 0.59]$



Estimating the policy gradient

- $\widehat{\nabla_{\theta} J(\theta)} = ?$
- J depends on both the action selections and the distribution of states in which those selections are made
 - Both of these are affected by the policy parameter θ
- Seems impossible to solve without knowing the transition function (or the distribution of visited states)
 - $p(s'|s, a)$ is unknown in model free RL
- The PG theorem allows us to evaluate $\widehat{\nabla_{\theta} J(\theta)}$ without the need for $p(s'|s, a)$

Policy Gradient Theorem

$$\nabla_{\theta} J(\theta) = \nabla v_{\pi}(s_0) \nabla_{\theta} \pi$$

- $\nabla v_{\pi}(s) = \nabla [\sum_a \pi(a|s) q_{\pi}(s, a)]$, for all $s \in \mathcal{S}$

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s)q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a)\nabla\pi(a|s) + \pi(a|s)\nabla q_\pi(s, a)]$, product rule

$$\frac{d(uv)}{dx} = u \frac{dv}{dx} + v \frac{du}{dx}$$

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s) q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla \sum_{s', r} p(s', r|s, a) (r + v_\pi(s'))]$

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s) q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla \sum_{s', r} p(s', r|s, a) (r + v_\pi(s'))]$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \nabla v_\pi(s')]$
 - $p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_r p(s', r|s, a)$, r is not a function of the policy

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s) q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla \sum_{s', r} p(s', r|s, a) (r + v_\pi(s'))]$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \nabla v_\pi(s')]$
 - $p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_r p(s', r|s, a)$, r is not a function of the policy

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s) q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla \sum_{s', r} p(s', r|s, a) (r + v_\pi(s'))]$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \nabla v_\pi(s')]$
 - $p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_r p(s', r|s, a)$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \sum_{a'} [q_\pi(s', a') \nabla \pi(a'|s') + \pi(a'|s') \sum_{s''} p(s''|s', a') \nabla v_\pi(s'')]]$

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s) q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla \sum_{s', r} p(s', r|s, a) (r + v_\pi(s'))]$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \nabla v_\pi(s')]$
 - $p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_r p(s', r|s, a)$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \sum_{a'} [q_\pi(s', a') \nabla \pi(a'|s') + \pi(a'|s') \sum_{s''} p(s''|s', a') \nabla v_\pi(s'')]]$
- One step $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_{s'} \Pr(s \rightarrow s' | \pi, 1 \text{ step}) \sum_{a'} q_\pi(s', a') \nabla \pi(a'|s')$


Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s) q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s) + \pi(a|s) \nabla \sum_{s', r} p(s', r|s, a) (r + v_\pi(s'))]$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \nabla v_\pi(s')]$
 - $p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_r p(s', r|s, a)$
- $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \sum_{a'} [q_\pi(s', a') \nabla \pi(a'|s') + \pi(a'|s') \sum_{s''} p(s''|s', a') \nabla v_\pi(s'')]]$
- One step $= \sum_a [q_\pi(s, a) \nabla \pi(a|s)] + \sum_{s'} \Pr(s \rightarrow s' | \pi, 1 \text{ step}) \sum_{a'} q_\pi(s', a') \nabla \pi(a'|s')$
- Two steps $= \sum_{s'} \left((\Pr(s \rightarrow s' | \pi, 0 \text{ step}) + \Pr(s \rightarrow s' | \pi, 1 \text{ step}) + \Pr(s \rightarrow s' | \pi, 2 \text{ step})) \sum_{a'} q_\pi(s', a') \nabla \pi(a'|s') \right)$

Policy Gradient Theorem

- $\nabla v_\pi(s) = \nabla[\sum_a \pi(a|s)q_\pi(s, a)]$, for all $s \in \mathcal{S}$
- $= \sum_a [q_\pi(s, a)\nabla\pi(a|s) + \pi(a|s)\nabla q_\pi(s, a)]$, product rule
- $= \sum_a [q_\pi(s, a)\nabla\pi(a|s) + \pi(a|s)\nabla \sum_{s',r} p(s', r|s, a)(r + v_\pi(s'))]$
- $= \sum_a [q_\pi(s, a)\nabla\pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a)\nabla v_\pi(s')]$
 - $p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_r p(s', r|s, a)$
- $= \sum_a [q_\pi(s, a)\nabla\pi(a|s)] + \sum_a [\pi(a|s) \sum_{s'} p(s'|s, a) \sum_{a'} [q_\pi(s', a')\nabla\pi(a'|s') + \pi(a'|s') \sum_{s''} p(s''|s', a')\nabla v_\pi(s'')]]$
- General case: $= \sum_{x \in \mathcal{S}} (\sum_{k=0}^{\infty} \Pr(s \rightarrow x, k | \pi)) \sum_a q_\pi(x, a) \nabla\pi(a|x)$

Update the policy so it will lead us (with high likelihood) to the best k-reachable state (maximal state value)

 Number of steps

Policy Gradient Theorem

- Assume that all episodes start from s_0
 - $J(\theta) = v_{\pi_\theta}(s_0)$
- $\nabla J(\theta) = \nabla v_{\pi}(S_0) = \sum_s (\sum_{k=0}^{\infty} \Pr(s_0 \rightarrow s, k | \pi)) \sum_a q_{\pi}(s, a) \nabla \pi(a | s)$
- How can we evaluate: $\sum_{k=0}^{\infty} \Pr(s_0 \rightarrow s, k | \pi)$ for any s ?
 - Just count how many times s was visited in a given episode (\approx expected value)
 - Denoted: $\eta(s)$
- $\nabla J(\theta) = \sum_s \eta(s) \sum_a q_{\pi}(s, a) \nabla \pi(a | s)$
- **Problem:** $\eta(s)$ is sensitive to the episode length
- **Solution:** normalize $\eta(s)$ to be a probability distribution

Policy Gradient Theorem

- $\nabla J(\theta) = \sum_s \eta(s) \sum_a q_\pi(s, a) \nabla \pi(a|s)$
- Normalize $\eta(s)$ to be a probability distribution
- $\eta(s) \propto \frac{\eta(s)}{\sum_{s'} \eta(s')} \doteq \mu(s)$
- $\nabla J(\theta) \propto \sum_s [\mu(s) \sum_a q_\pi(s, a) \nabla \pi(a|s)]$
- The gradients are column vectors of partial derivatives with respect to the components of θ

Monte-Carlo Policy Gradient

- Sample-based gradient estimation
- $\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s, a) \nabla \pi(a|s)$
- $= \mathbb{E}_\pi [\sum_a q_\pi(S_t, a) \nabla_\theta \pi(a|S_t)]$
- We can now use this gradient estimation for PG steps?
 - $\theta_{t+1} = \theta_t + \alpha \sum_a \widehat{q}_\pi(S_t, a) \nabla_\theta \pi(a|S_t; \theta)$
 - Requires an approximation of q_π , denoted \hat{q} , for any action (also those not taken)
 - Can we avoid this approximation?

REINFORCE [Williams, 1992]

1. $\theta_{t+1} = \theta_t + \alpha \sum_a \widehat{q_\pi}(S_t, a) \nabla_\theta \pi(a|S_t)$
 - Can we avoid this approximation? YES!
2. $\nabla J(\theta) \propto \mathbb{E}_\pi [\sum_a q_\pi(S_t, a) \nabla_\theta \pi(a|S_t)]$
3. $= \mathbb{E}_\pi \left[\sum_a \pi(a|S_t) q_\pi(S_t, a) \frac{\nabla_\theta \pi(a|S_t)}{\pi(a|S_t)} \right]$ (multiply and divide by the same number)
4. $= \mathbb{E}_\pi \left[q_\pi(S_t, A_t) \frac{\nabla_\theta \pi(A_t|S_t)}{\pi(A_t|S_t)} \right]$ ($\sum_x \text{Pr}(x) f(x) = \mathbb{E}_{x \sim \text{Pr}(x)}[f(x)]$)
5. $= \mathbb{E}_\pi \left[G_t \frac{\nabla_\theta \pi(A_t|S_t)}{\pi(A_t|S_t)} \right]$ ($\mathbb{E}_\pi[G_t|S_t, A_t] = q_\pi(S_t, A_t)$)
- We can now use this gradient estimation for PG steps
 - $\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla_\theta \pi(A_t|S_t)}{\pi(A_t|S_t)}$

REINFORCE [Williams, 1992]

- $\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla_{\theta} \pi(A_t|S_t;\theta)}{\pi(A_t|S_t;\theta)}$
- Each increment is proportional to the product of a return G_t and a vector
 - The gradient of the probability of taking the action actually taken over the (current) probability of taking that action
 - The vector is the direction in parameter space that most increases the probability of repeating the action A_t on future visits to state S_t
- The update increases the parameter vector (*) proportional to the return, and (**) inversely proportional to the action probability
 - (*) makes sense because it causes the parameter to move most in the directions that favor actions that yield the highest return
 - (**) makes sense because otherwise actions that are selected frequently are at an advantage (the updates will be more often in their direction) and might win out even if they do not yield the highest return

REINFORCE [Williams, 1992]

- $\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla_{\theta} \pi(A_t | S_t; \theta)}{\pi(A_t | S_t; \theta)}$
- REINFORCE uses G_t : the complete return from time t until the end of the episode
- In this sense REINFORCE is a Monte Carlo algorithm and is well defined only for the episodic case with all updates made in retrospect after the episode is completed

REINFORCE [Williams, 1992]

* In the literature you might encounter “grad log pi” instead of “grad ln pi”. That’s not a problem since: $\nabla \ln(f(x)) \propto \nabla \log(f(x))$

Specifically: $\nabla \ln(f(x)) = \ln(a) \nabla \log_a(f(x))$

REINFORCE, A Monte-Carlo Policy-Gradient Method (episodic)

Input: a differentiable policy parameterization $\pi(a|s, \theta)$

Initialize policy parameter $\theta \in \mathbb{R}^{d'}$

Repeat forever:

Generate an episode $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \theta)$

For each step of the episode $t = 0, \dots, T - 1$:

$G \leftarrow$ return from step t

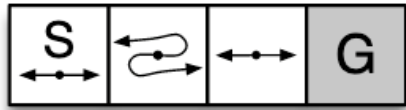
$\theta \leftarrow \theta + \alpha \gamma^t G \nabla_{\theta} \ln \pi(A_t|S_t, \theta)$

How did we get here
from $\frac{\nabla_{\theta} \pi(A_t|S_t; \theta)}{\pi(A_t|S_t; \theta)}$?

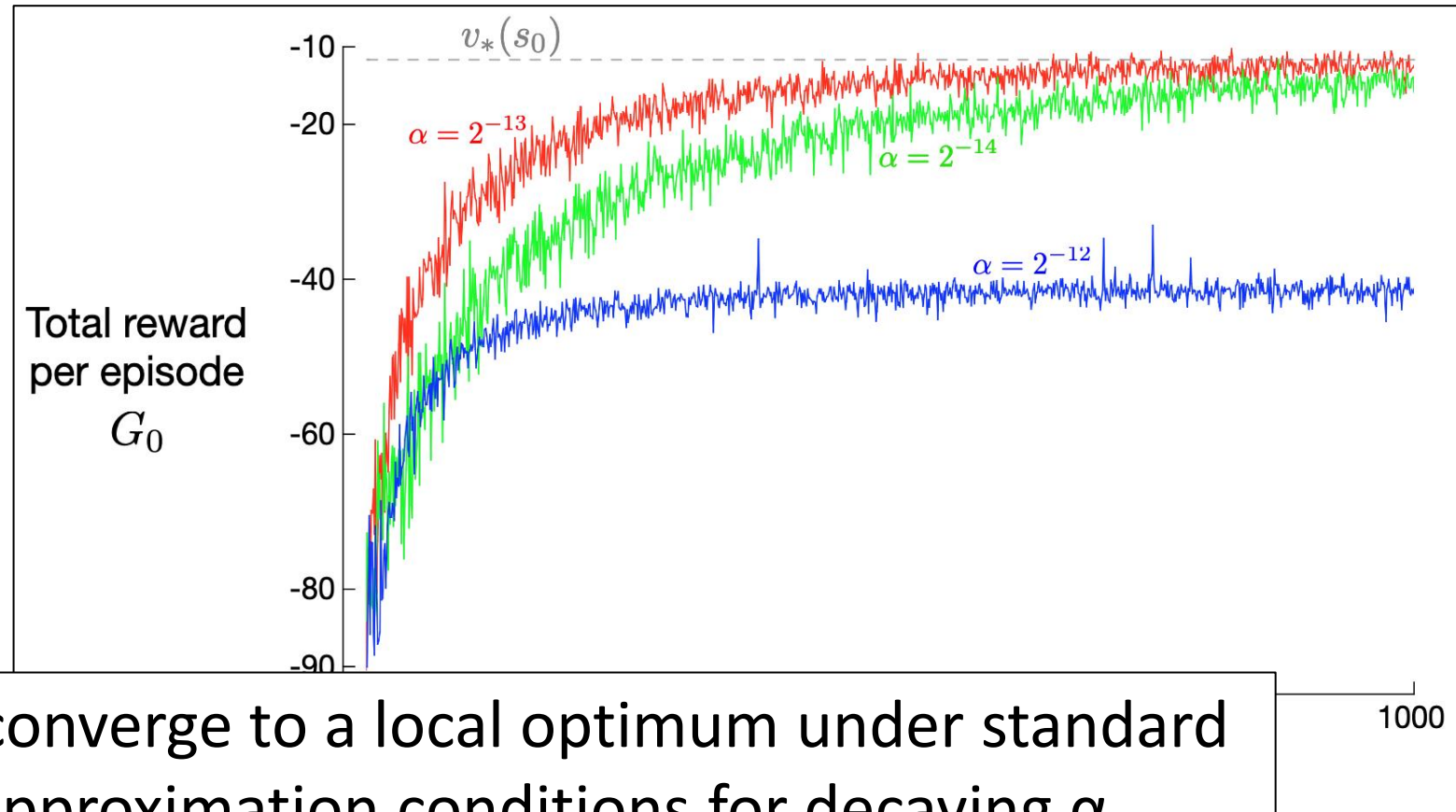
$$\nabla \ln(f(x)) = \frac{\nabla f(x)}{f(x)}$$

REINFORCE [Williams, 1992]

- The crazy corridor domain



- With the right step size, the total reward per episode approaches the optimum
- start s



Guaranteed to converge to a local optimum under standard stochastic approximation conditions for decaying α

Calibrating REINFORCE

- Imagine a domain with two possible episode trajectories (rollouts)
 - $\tau_1 = \{S_0, A_0, R_1, S_1, \dots, A_{T-1}, R_T, S_T\}$ with $G = 1001$
 - $\tau_2 = \{S'_0, A'_0, R'_1, S'_1, \dots, A'_{T-1}, R'_T, S'_T\}$ with $G' = 1000$
 - Further assume that $R_{i < T} = R'_i = 0$ and $R_T = G, R'_T = G'$
- Following REINFORCE: $\theta_{t+1} = \theta_t + \alpha G_t \nabla_{\theta} \ln \pi(A_t | S_t; \theta)$
- How **will** the policy be updated after sampling each of the trajectories?
 - $\tau_1 ++, \tau_2 ++$
- How **should** the policy be updated after sampling each of the trajectories?
 - $\tau_1 +, \tau_2 -$
- How can we address this gap?

PG with baseline

- Normalize the returns with a baseline!
- $\nabla J(\theta) \propto \sum_s \mu(s) \sum_a (q_\pi(s, a) - \mathbf{b}(s)) \nabla \pi(a|s)$
- Are we allowed to do that???
- Can we subtract $\sum_s \mu(s) \sum_a b(s) \nabla \pi(a|s)$ from $\nabla J(\theta)$?
- **Yes!** As long as $b(s)$ isn't a function of a
- $\sum_a b(s) \nabla \pi(a|s) = b(s) \nabla \sum_a \pi(a|s) = b(s) \nabla 1 = 0$ (actions' probabilities sum to 1)
- REINFORCE with baseline: $\theta_{t+1} = \theta_t + \alpha (G_t - b(S_t)) \nabla_\theta \ln \pi_\theta(A_t|S_t)$

PG with baseline

- In the bandit algorithms the baseline was the average of the rewards seen so far
- For MDPs the baseline should be different for each states
 - In some states all actions have high (q) values, and we need a high baseline to differentiate the higher valued actions from the less highly valued ones
 - In other states all actions will have low values and a low baseline is appropriate
- What would be the equivalent of average reward (bandits) for a given state (MDP) ?
 - $v_{\pi}(s)$

PG with baseline

- $\nabla J(\theta) \propto \sum_s \mu(s) \sum_a (q_\pi(s, a) - v_\pi(s)) \nabla \pi(a|s)$
- Actions that improve on the current policy are encouraged
 - $q_\pi(s, a) - v_\pi(s) > 0$
- Actions that damage the current policy are discouraged
 - $q_\pi(s, a) - v_\pi(s) < 0$
- Also known as the **advantage** of action a in state s
- How can we learn $v_\pi(s)$?
- Another function approximator $\hat{v}(s; w)$
 - On top of the policy

REINFORCE with baseline

REINFORCE with Baseline (episodic)

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$

Input: a differentiable state-value parameterization $\hat{v}(s, \mathbf{w})$

Parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$

Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$

Repeat forever:

Generate an episode $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$

For each step of the episode $t = 0, \dots, T - 1$:

$G_t \leftarrow$ return from step t

$\delta \leftarrow G_t - \hat{v}(S_t, \mathbf{w})$

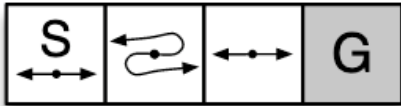
$\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \gamma^t \delta \nabla_{\mathbf{w}} \hat{v}(S_t, \mathbf{w})$

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla_{\boldsymbol{\theta}} \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Each approximator has its
unique learning rate

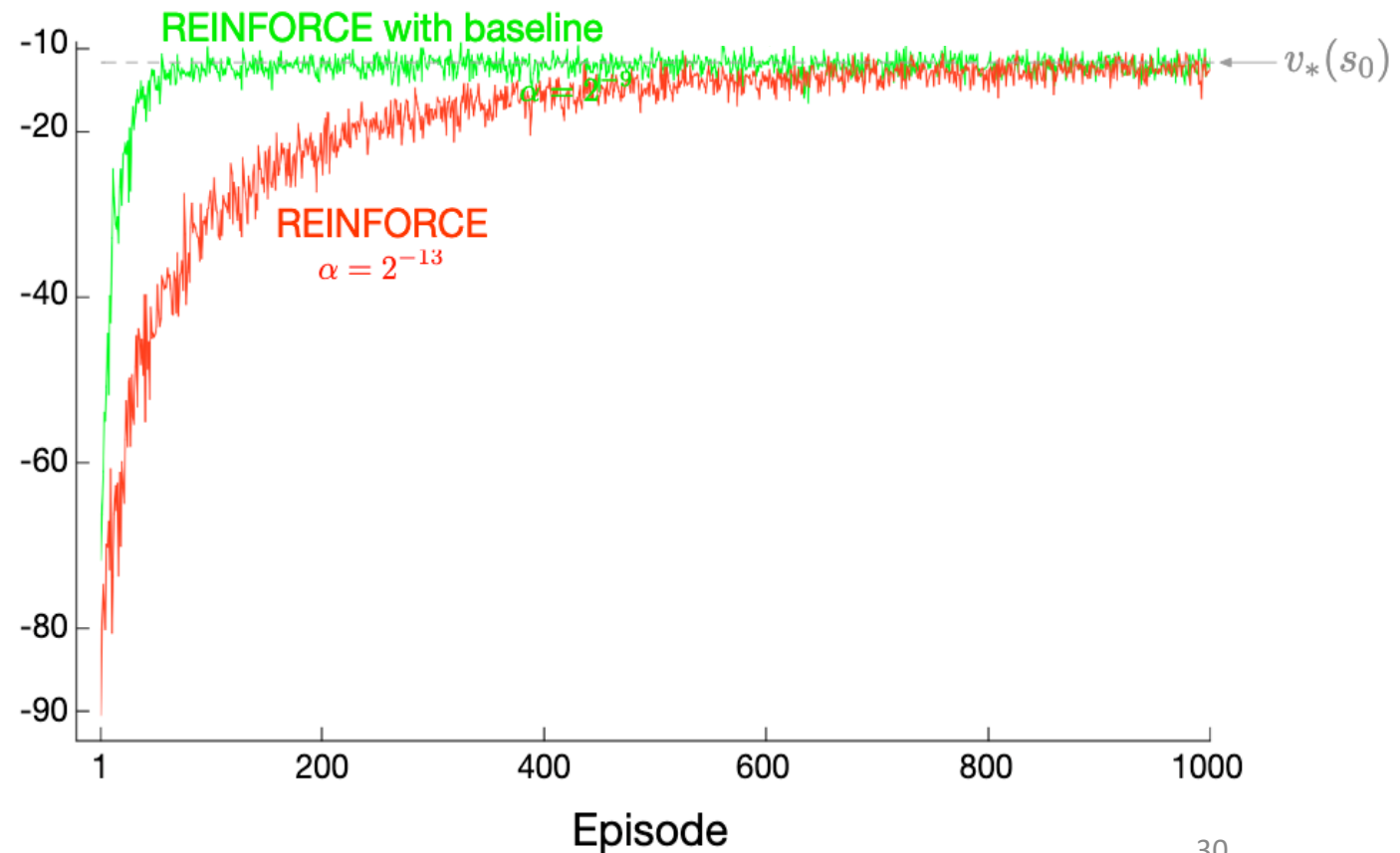
REINFORCE with baseline

- The crazy corridor domain



- $\alpha^\theta = 2^{-9}$
- $\alpha^w = 2^{-6}$

Total reward
per episode
 G_0



What did we learn?

- In many domains it's more efficient to learn the policy directly
 - Instead of deriving one from state or action values
- Applicable for continuous action spaces and stochastic policies
- Approximate the change to the policy that results in the highest increase in the expected return: $\nabla_{\theta} \widehat{J}(\theta)$
- Such approximation is made possible by the policy gradient theorem
- Should we reinforce policies that result in high return?
 - Not always, a different policy might yield higher return
 - We need to use a baseline to determine if a policy is **relatively** good

What next?

- **Lecture:** Actor-Critic
- **Assignments:**
 - REINFORCE, by Nov. 11, EOD
 - Deep Q-Learning, by Nov. 4, EOD
- **Quiz (on Canvas):**
 - Policy Gradient, by Oct. 30, EOD
 - Deep Q-Learning, by Oct. 28, EOD
- **Project:**
 - Literature survey, by Nov. 4, EOD