

Advanced Topics in Reinforcement Learning

Lecture 13: Policy Gradients I

Josiah Hanna

University of Wisconsin — Madison

Announcements

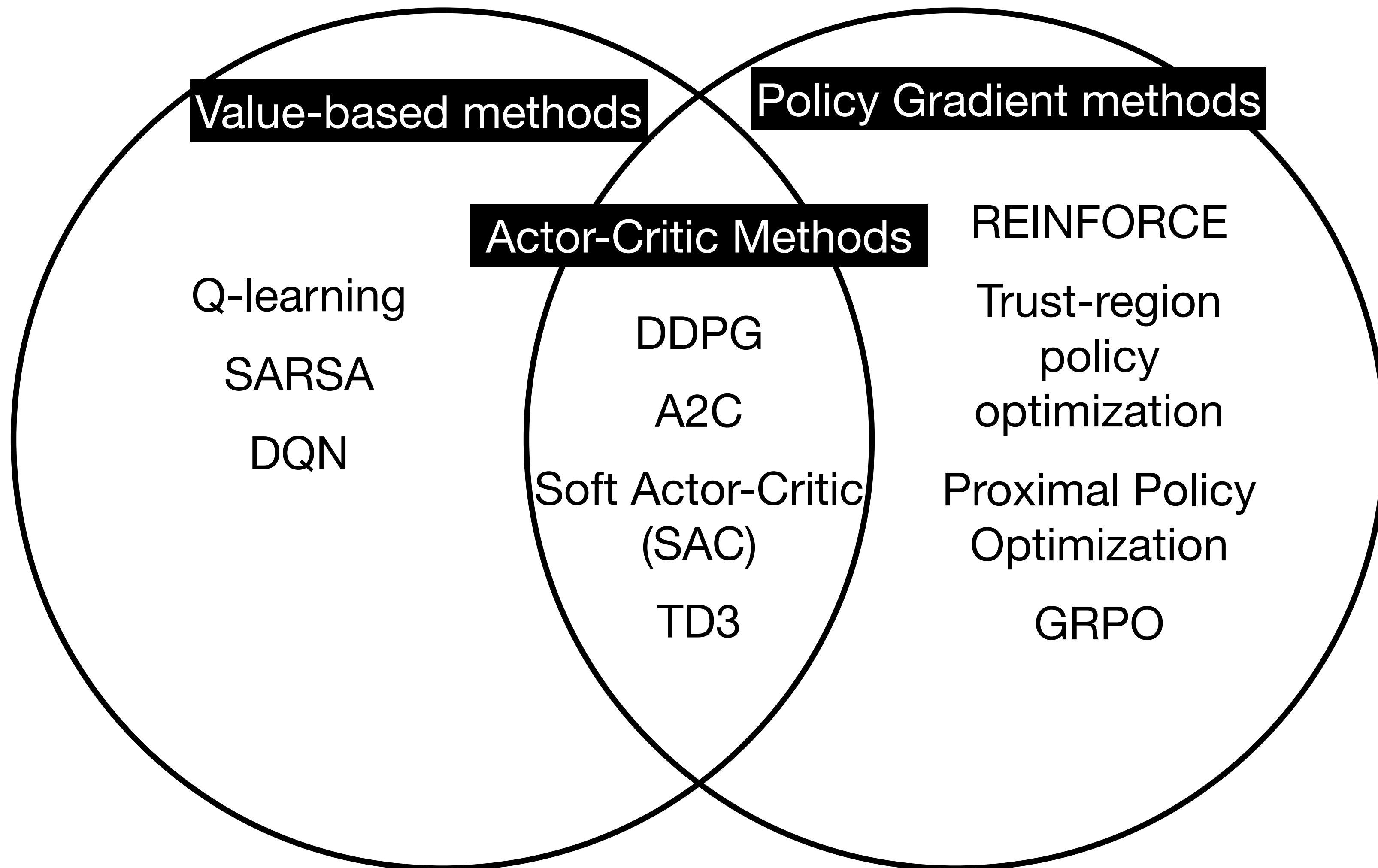
- Literature review due Friday @ 11:59pm
- Read “Between MDPs and Semi-MDPs:…” For next week
- Upcoming dates:
 - Exam: October 29 in class

Learning Outcomes

After this week, you will be able to:

1. Understand how to formulate the basic policy gradient RL approach.
2. Compare and contrast value-based, policy-based, and actor-critic model-free RL methods.
3. Identify key techniques in advanced policy-based RL methods.

Model-Free RL



Policy-based RL

- So far the policy is implicit. It is derived from a parameterized action-value function and we learn the action-value function parameters.
- Policy gradient methods use a parameterized policy and learn policy parameters with gradient ascent.
- $\pi_{\theta}(a | s) = \Pr(A_t = a | S_t = s, \theta_t = \theta)$
- $J(\theta) = v_{\pi_{\theta}}(s_0)$
- $\theta_{t+1} \leftarrow \theta_t + \alpha \nabla_{\theta} \widehat{J(\theta_t)}$

Why Policy-based?

- What are advantages to policy-based methods?
 - More easily handle continuous actions.
 - Policy gradient theorem provides stronger convergence guarantees under function approximation.
 - Useful for partial observability.
 - In some cases, the policy may be simpler to approximate.
 - Can inject prior knowledge into policy class.
- Disadvantages?
 - In other cases, action-values are easier to approximate.
 - The optimal policy is a simple function of the action-values.

Policy Parameterizations

- Policy can be **any** parameterized and differentiable distribution.
- Need $\pi_\theta(A_t = a | s)$ and $\nabla_\theta \pi_\theta(A_t = a | s)$ exists.

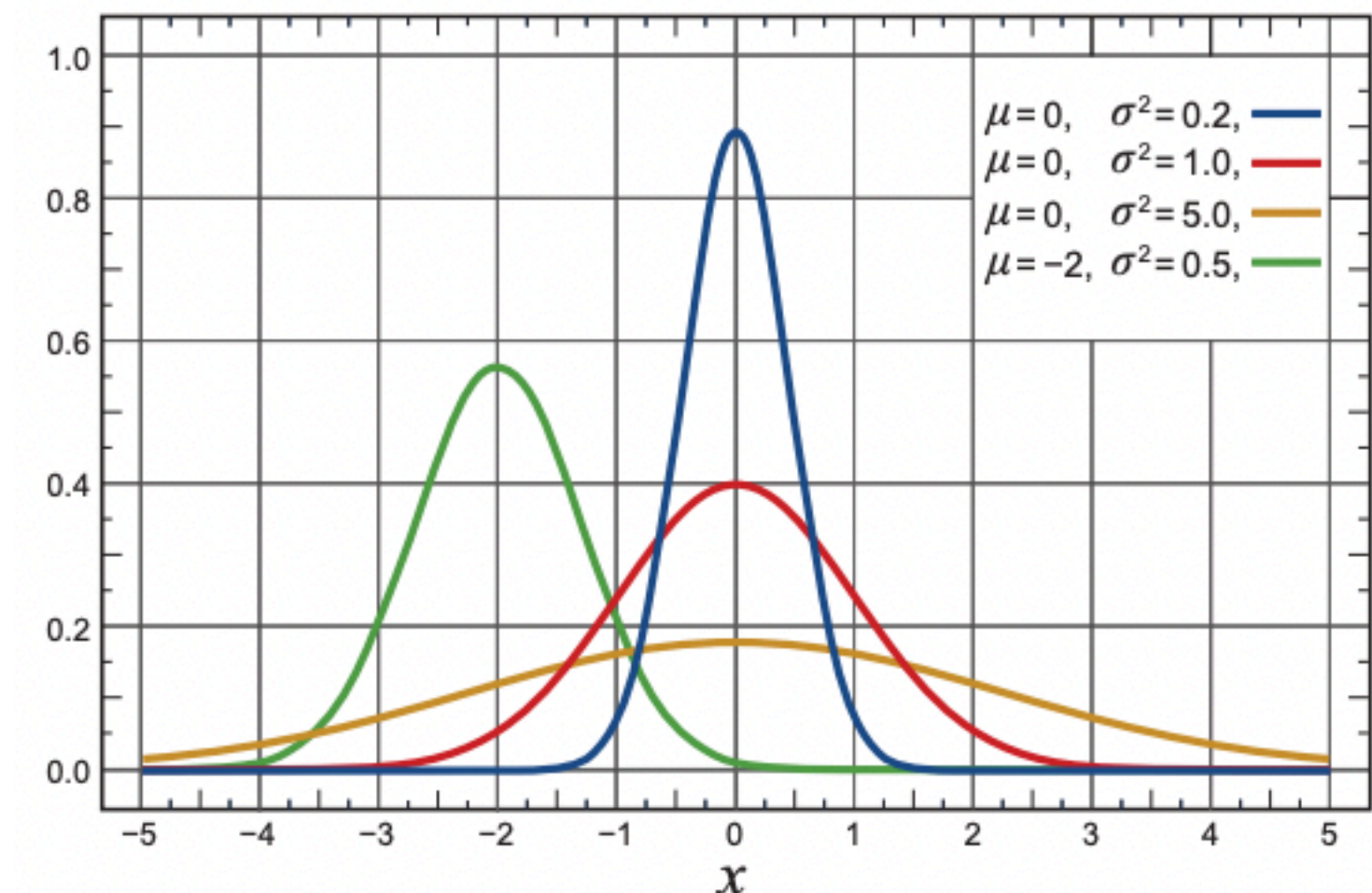
Discrete Action Example

- $\Pr(A_t = a | S_t = s) \propto \exp(h(s, a, \theta))$
- $\nabla_\theta \ln \pi_\theta(a | s, \theta) = \nabla_\theta h(s, a, \theta) - \mathbf{E}_{\pi_\theta}[\nabla_\theta h(s, A, \theta)]$

E.g., h is a neural network with $|A|$ outputs

Continuous Action Example

- $\Pr(A_t = a | S_t = s) = \mathcal{N}(\mu(s), \sigma(s))$
- $\mu(s) = f_\theta(s); \log \sigma(s) = f_\sigma(s)$
- $\nabla_\theta \ln \pi_\theta(a | s, \theta) = \frac{1}{\sigma(s)^2} (a - \mu(s)) \nabla_\theta f_\theta(s)$



Policy Gradient Theorem

- $$J(\theta) := v_{\pi_\theta}(s_0) = \sum_a \pi_\theta(a | s_0) \underbrace{\sum_{s', r} p(s', r | s_0, a) [r + v_{\pi_\theta}(s')]}_{q_{\pi_\theta}(s_0, a)}$$
- $$\nabla_\theta J(\theta) \propto \sum_s \sum_a \mu_\theta(s) q_{\pi_\theta}(s, a) \nabla_\theta \pi_\theta(a | s)$$

On-policy distribution on states
- The direction in which an infinitesimally small change to θ produces the maximum increase in $J(\theta)$.
- $\nabla_\theta J(\theta)$ does not depend on any gradients of p !

Note: assuming a fixed initial state

Note: this is for episodic case, $\gamma = 1$

REINFORCE

- Stochastic gradient ascent instead of true gradient ascent. Why?

- $\nabla_{\theta} J(\theta)$ can only be estimated.

- $$\nabla_{\theta} J(\theta) \propto \mathbf{E}_{\pi} \left[\sum_a \nabla_{\theta} \pi(a | S_t) q_{\pi}(S_t, a) \right] = \mathbf{E}_{\pi} [q_{\pi}(A_t | S_t) \nabla_{\theta} \log \pi(A_t | S_t)]$$

- Approximate r.h.s. with experience at time t: $q_{\pi}(S_t, A_t) \nabla \log \pi_{\theta}(A_t | S_t)$

- Then, replace $q_{\pi}(s, a)$ with G_t to get final gradient approximation. $\nabla_{\theta} J(\theta) \approx G_t \nabla_{\theta} \log \pi_{\theta}(A_t | S_t)$

- $$\theta_{t+1} \leftarrow \theta_t + \alpha G_t \nabla_{\theta} \log \pi(A_t | S_t)$$

- On- or off-policy?

REINFORCE

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

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

Algorithm parameter: step size $\alpha > 0$

Initialize policy parameter $\theta \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):

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

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

$$G \leftarrow \sum_{k=t+1}^T \gamma^{k-t-1} R_k \quad (G_t)$$

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

Usually dropped in practice

Is the policy gradient a gradient? Nota and Thomas. 2020.

Bias in Natural Actor-Critic Algorithms. Thomas. 2014.

Baselines

- REINFORCE has high-variance updates.
 - On average, updates improve the policy but any single update could make it worse.
 - Reinforce actions in proportion to the return that follows.
- Replacing G_t with $G_t - b(S_t)$, where $b(S_t)$ is constant w.r.t. action A_t ; can substantially lower variance.

$$\bullet \quad \mathbf{E}[G_t \nabla_{\theta} \log \pi(A | S)] = \mathbf{E}[G_t \nabla_{\theta} \log \pi(A | S)] - \underbrace{\mathbf{E}[b(S_t) \nabla_{\theta} \log \pi(A | S)]}_{=0} = \mathbf{E}[G_t - b(S_t) \nabla_{\theta} \log \pi(A | S)]$$

- Reinforce actions in proportion to how much better than average they are.

Baselines

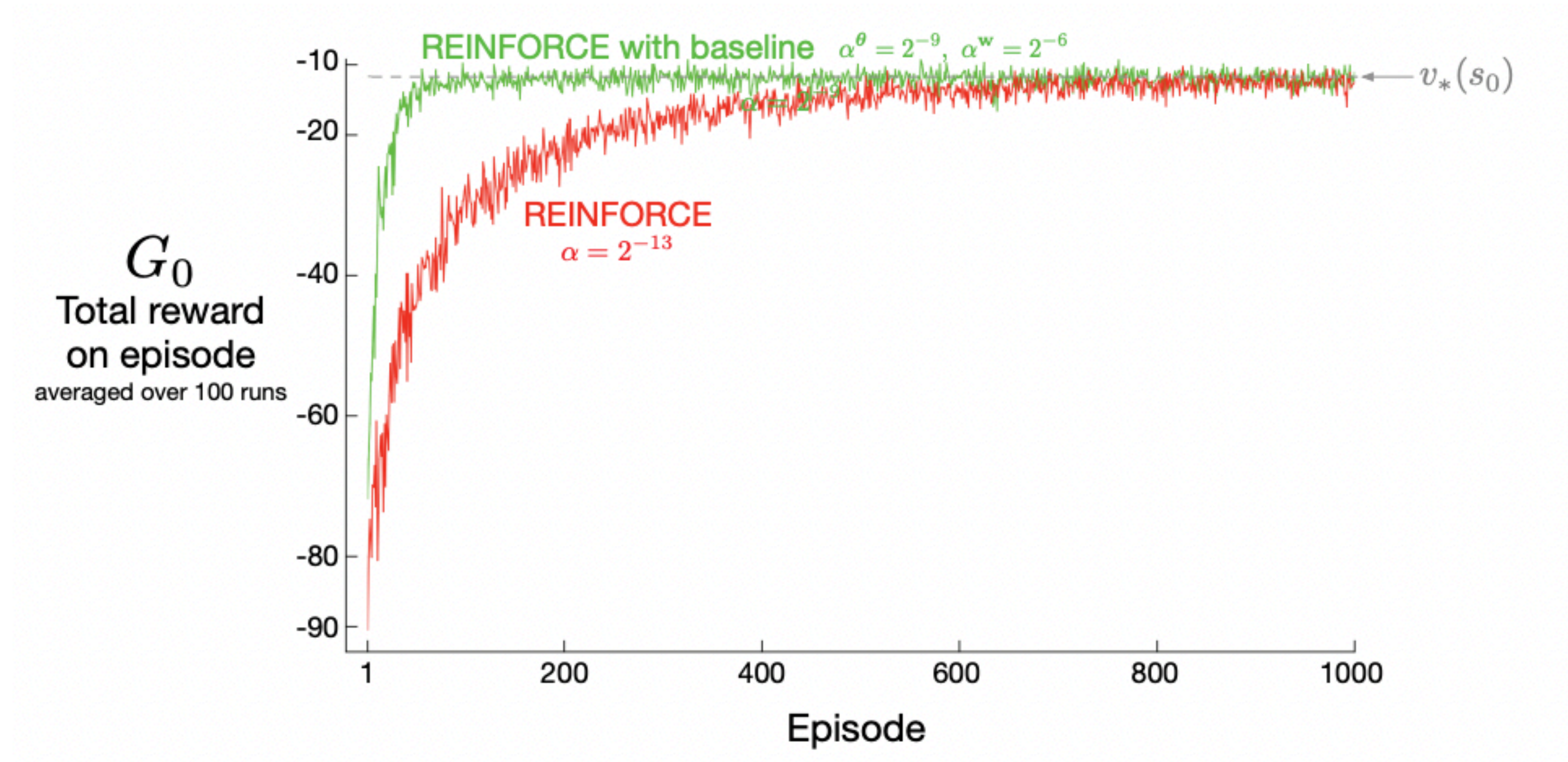


Figure 13.2 in textbook.

Optimal Baselines

- In practice, use an approximation for $v_\pi(S_t)$ as the baseline $b(S_t)$.
- $G_t - v_\pi(S_t) \approx q_\pi(S_t, A_t) - v_\pi(S_t) = A_\pi(S_t, A_t)$, i.e., the advantage function.
- Removes randomness due to $S_t \sim \mu(s)$.
- Better choices exist in theory but seldom used in practice.

Actor-Critic Methods

- REINFORCE uses a learned value function only to lower variance.
 - Monte Carlo return still drives which actions are reinforced.
- Actor-critic methods use learned value functions to drive policy changes.
 - Actor: the policy.
 - Critic: value function.
- Can use state-values: or action-value functions:
 - $\theta_{t+1} \leftarrow \theta_t + \alpha \delta_t \nabla_{\theta} \log \pi(A_t | S_t)$ $\delta_t \leftarrow R_{t+1} + \gamma \hat{v}(S_{t+1}) - \hat{v}(S_t)$
- Or can use action-value functions:
 - $\theta_{t+1} \leftarrow \theta_t + \alpha \hat{q}(S_t, A_t) \nabla_{\theta} \log \pi(A_t | S_t)$

Actor-Critic Methods

One-step Actor-Critic (episodic), for estimating $\pi_{\theta} \approx \pi_*$

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

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

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

Initialize policy parameter $\theta \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):

 Initialize S (first state of episode)

$I \leftarrow 1$

 Loop while S is not terminal (for each time step):

$A \sim \pi(\cdot|S, \theta)$

 Take action A , observe S', R

$\delta \leftarrow R + \gamma \hat{v}(S', \mathbf{w}) - \hat{v}(S, \mathbf{w})$ (if S' is terminal, then $\hat{v}(S', \mathbf{w}) \doteq 0$)

$\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S, \mathbf{w})$

$\theta \leftarrow \theta + \alpha^{\theta} I \delta \nabla \ln \pi(A|S, \theta)$

~~$I \leftarrow \gamma I$~~

$S \leftarrow S'$

Deterministic Policy Gradients

- Actor-critic limitations: **Random sampling for both states and actions.**
 - Can still have high variance (like REINFORCE) and also introduce bias into gradient estimates.
 - On-policy or require importance sampling to be off-policy.
- Deterministic policy gradient methods overcome these limitations in continuous action problems:
 - Learn a deterministic policy $A_t \leftarrow \pi_\theta(S_t)$.
 - Approximate $q_\pi(s, a)$ with a function approximator, $\hat{q}$, that is differentiable w.r.t. the action.
 - $\nabla_\theta J(\theta) \propto \mathbf{E}[\nabla_a \hat{q}(S_t, A_t) \nabla_\theta \pi_\theta(S_t) \mid S_t \sim d_b, A_t \sim b]$ **Can interpret as approximating Q-learning for continuous actions**

Deterministic Policy Gradients

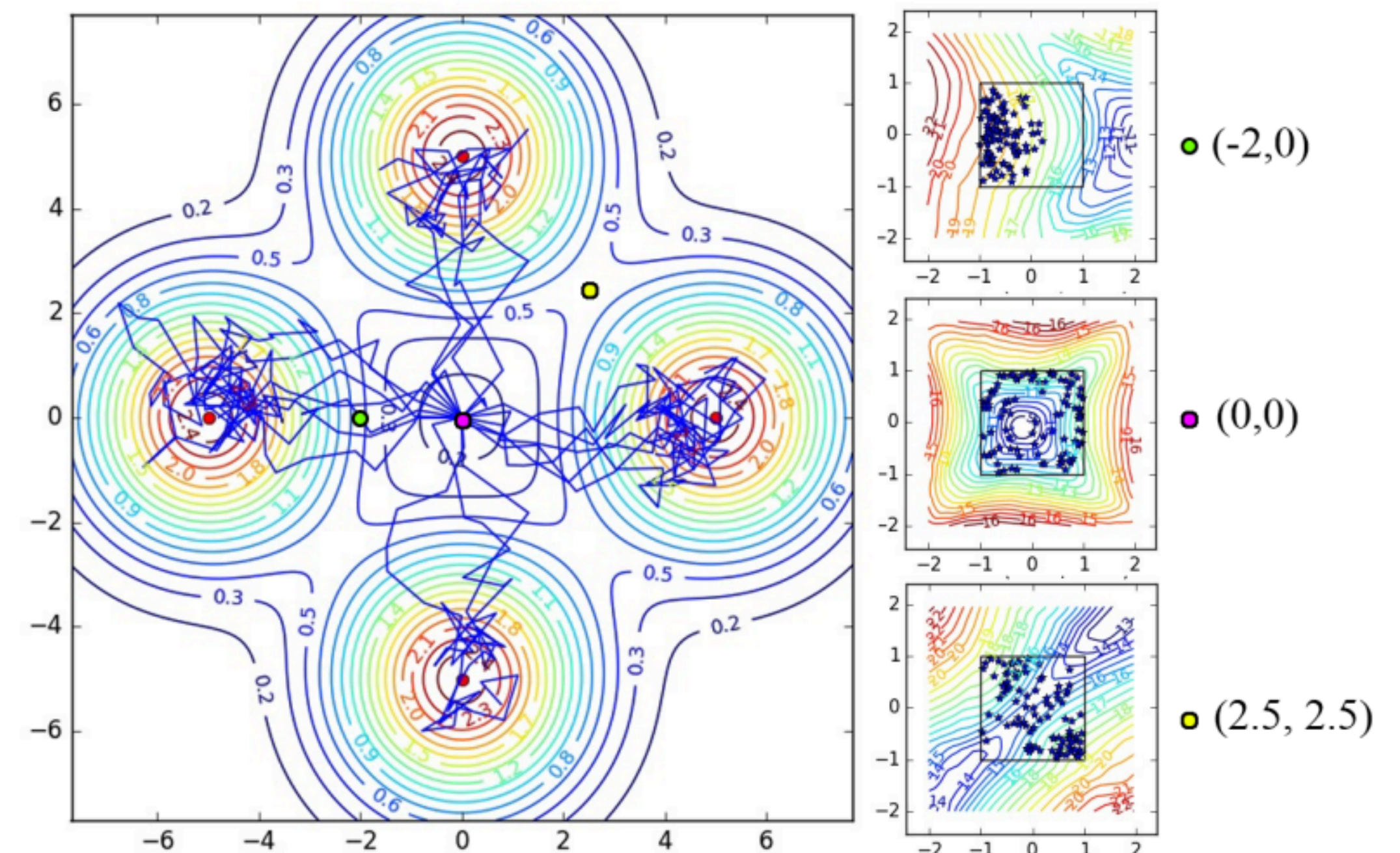
- Basis for several state-of-the-art off-policy deep RL algorithms:
 - Deep Deterministic Policy Gradient (DDPG). Lilicrap et al. 2015.
 - Twin Delayed DDPG (TD3). Fujimoto et al. 2018.
 - Trains two critics and uses minimum like double Q-learning.
 - Soft Actor-Critic (SAC). Haarnoja et al. 2018.
 - Trains stochastic actor using reparameterization trick to lower variance; also double Q-learning technique.

Max-Entropy RL

- Max Entropy RL: maximize return while maintaining stochasticity in the policy

- $$J(\theta) := \mathbf{E}_{\pi} \left[\sum_{t=0}^T R_{t+1} + \underbrace{H(\pi_{\theta}(\cdot | S_t))}_{\text{Entropy}} \right]$$

- Intuition: if multiple ways to maximize return, policy should learn them all.



Soft Actor-Critic

- Actor-critic method for the MaxEnt setting.
- Train two critics with learning target:

$$Y_t = R_{t+1} + \gamma \min(\tilde{Q}_1(S_{t+1}, a), \tilde{Q}_2(S_{t+1}, a)) - \alpha \log \pi(A_{t+1} | S_{t+1})$$

- Train one actor by minimizing loss:

$$L(\theta) := \mathbf{E}_{\pi}[\alpha \log \pi_{\theta}(a | S_t) - \min(\tilde{Q}_1(S_t, a), \tilde{Q}_2(S_t, a))]$$

- Also: adapt α over-time, use weighted average of past network parameters to provide slow critic targets, experience replay for data-efficiency.

Summary

- Policy improvement theorem provides theoretical foundation for guaranteed policy improvement even with function approximation.
- Policy gradient methods can learn with or without learned value functions.
- Actor-critic methods use a learned value function as a replacement for the return in basic policy gradient methods.

Action Items

- Complete literature review.
- Begin studying for midterm exam
- Read “Between MDPs and semi-MDPs:...”