Reinforcement Learning

Policy Gradients

Alberto Maria Metelli

13th February 2024

Book References

Richard S. Sutton, Andrew G. Barto
Reinforcement Learning: An Introduction (second edition)
Chapter 13

Csaba Szepesvári
Algorithms for Reinforcement Learning
Section 4.4

Outline

Outline

Value Functions and Policies

- So far, we have learned **parametric value functions**

$$
\begin{aligned}
v_{\mathbf{w}}(s) &\approx v_\pi(s) \text{ or } v_*(s) \\
q_{\mathbf{w}}(s,a) &\approx q_\pi(s,a) \text{ or } q_*(s,a)
\end{aligned}
$$

  where $\mathbf{w} \in \mathbb{R}^m$ is the parameter vector

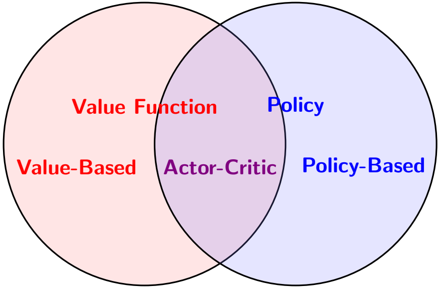
- And a policy is derived from those (e.g., greedy, ϵ-greedy, Boltzmann, ...)
- We now focus on learning **parametric policies**

$$
\pi_{\boldsymbol{\theta}}(a|s) = \Pr\left(\text{playing action } a \text{ in state } s|\boldsymbol{\theta}\right)
$$

  where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the parameter vector

Value Based and Policy–Based Reinforcement Learning

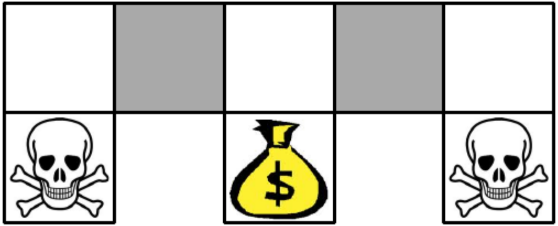
- Value-Based
  - **Learn** value function $q_{\mathbf{w}}(s, a)$
  - **Implicit** policy (e.g., greedy, ϵ-greedy, Boltzmann, ...)
- Policy-Based
  - **No** value function
  - **Learn** policy $\pi_{\boldsymbol{\theta}}(a|s)$
- Actor-Critic
  - **Learn** value function $q_{\mathbf{w}}(s, a)$
  - **Learn** policy $\pi_{\boldsymbol{\theta}}(a|s)$



Advantages of Policy–Based RL

- Advantages
  - Better **convergence** properties
  - Effective in **high–dimensional** or **continuous action** spaces
    - When $|\mathcal{A}| = \infty$ computing $\sup_{a \in \mathcal{A}} q(s, a)$ is hard!
  - **Policy subspace** can be chosen according to the **task**
    - Policies might be **simpler** than value functions
  - Can learn **stochastic policies**
    - **Exploration** can be directly controlled
    - Better tackle **partial observability** or **non-Markovianity**
  - Can benefit from **demonstrations**
- Disadvantages
  - Typically converge to a **local** rather than a **global** optimum
  - Evaluating a policy is typically **inefficient** and **high variance**

Example: Aliased Gridworld



- The agent **cannot distinguish** the gray states
- Consider **features**:

$$\mathbf{x}(s) = (\mathbf{1}\{\text{wall up}\}, \mathbf{1}\{\text{wall left}\}, \mathbf{1}\{\text{wall right}\}, \mathbf{1}\{\text{wall down}\})^{\mathsf{T}}$$

- Compare value–based RL, using a **parametrized value function**

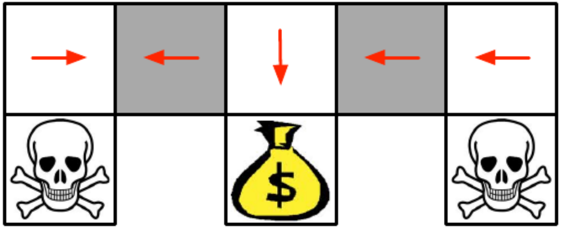
$$q_{\mathbf{w}}(s, \cdot) = f(\mathbf{x}(s), \mathbf{w})$$

- To policy–based RL, using a **parameterized policy**

$$\pi_{\boldsymbol{\theta}}(\cdot|s) = g(\mathbf{x}(s), \boldsymbol{\theta})$$

Pictures from (Silver, 2015; Hado van Hasselt, 2015)

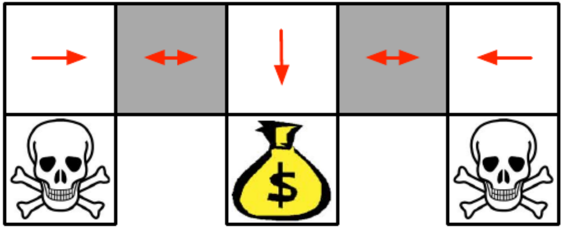
Example: Aliased Gridworld



- Under aliasing, an optimal **deterministic** policy will either
  - move left in both gray states
  - move right in both gray states
- Either way, it can get stuck and **never** reach the money
- If we add some **uniform noise** over the actions, we will reach the money sooner or later

Pictures from (Silver, 2015; Hado van Hasselt, 2015)

Example: Aliased Gridworld



- An optimal **stochastic** policy will randomly move left or right in gray states

$$\pi_{\boldsymbol{\theta}}(\text{right} \mid \text{wall up } \wedge \text{wall down}) = 0.5$$
$$\pi_{\boldsymbol{\theta}}(\text{left} \mid \text{wall up } \wedge \text{wall down}) = 0.5$$

- It will reach the goal state in a few steps with high probability
- Policy–based RL can learn the optimal stochastic policy

Pictures from (Silver, 2015; Hado van Hasselt, 2015)

Policy Optimization Objective Function

- **Goal**: given a policy $\pi_{\boldsymbol{\theta}}(a|s)$ with parameters $\boldsymbol{\theta}$, find best $\boldsymbol{\theta} \in \mathbb{R}^d$
- But how do we **measure** the quality of a policy $\pi_{\boldsymbol{\theta}}$?
- We need a **scalar objective**: the **expected return**

$$J(\boldsymbol{\theta}) = \mathbb{E}_{S_0 \sim d_0}\left[v_{\pi_{\boldsymbol{\theta}}}(S_0)\right] = \mathbb{E}\left[\sum_{t=0}^{+\infty} \gamma^t R_{t+1} | S_0 \sim d_0, \pi_{\boldsymbol{\theta}}\right]$$

where d_0 is the **initial state distribution**

Trajectory View

- If we have **trajectories** of finite length T
- We define the **probability of a trajectory** $\tau = (S_0, A_0, S_1, A_1, \dots S_{T-1}, A_{T-1}, S_T)$

$$p_{\boldsymbol{\theta}}(\tau) = d_0(S_0) \prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}}(A_t|S_t) p(S_{t+1}|S_t, A_t)$$

- and the **trajectory return**

$$G(\tau) = \sum_{t=0}^{T-1} \gamma^t r(S_t, A_t)$$

- We can rewrite at the **expected return** as

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} [G(\tau)]$$

Occupancy View

- An alternative way, that works also with **infinite-length** trajectories
- We define the γ-**discounted occupancy** (Sutton et al., 1999)

$$d_{\pi_{\boldsymbol{\theta}}}(s) = \sum_{t=0}^{+\infty} \gamma^t \Pr\left(S_t = s | S_0 \sim d_0, \pi_{\boldsymbol{\theta}}\right)$$

  - It is **not** a distribution as it integrates to $\frac{1}{1-\gamma}$
  - If the **stationary distribution** of the Markov chain induced by policy $\pi_{\boldsymbol{\theta}}$ exists, we have

$$\lim_{\gamma \to 1^-} (1-\gamma) d_{\pi_{\boldsymbol{\theta}}}(s) = \text{stationary distribution}$$

  - If the MDP has finite-horizon T, the series stops at time instant $T-1$
- We can rewrite at the **expected return** as

$$J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[r(S, A)\right]$$

Occupancy View

- How to sample from $d_{\pi_{\boldsymbol{\theta}}}$?

$S_0 \sim d_0$
for $t = 0, 1, \dots$ **do**
    Toss a coin with $1-\gamma$ head probability (i.e., a Bernoulli r.v. B_t with $p = 1-\gamma$)
    if head (i.e., $B_t = 1$) **then**
        return S_t
    end if
    $A_t \sim \pi_{\boldsymbol{\theta}}(\cdot|S_t)$
    $S_{t+1} \sim p(\cdot|S_t, A_t)$
end for

Trajectory View and Occupancy View are Equivalent

$$\mathbb{E}_{\tau\sim p_{\boldsymbol{\theta}}}[G(\tau)] = \mathbb{E}_{\substack{S\sim d_{\pi_{\boldsymbol{\theta}}}\\ A\sim\pi_{\boldsymbol{\theta}}(\cdot|S)}}[r(S,A)]$$

Proof.

$$\mathbb{E}_{\tau\sim p_{\boldsymbol{\theta}}}[G(\tau)] = \int_\tau p_{\boldsymbol{\theta}}(\tau)G(\tau)\mathrm{d}\tau = \int_\tau d_0(s_0)\prod_{l=0}^{T-1}\pi_{\boldsymbol{\theta}}(a_l|s_l)P(s_{l+1}|s_l,a_l)\sum_{t=0}^{T-1}\gamma^t r(s_t,a_t)\mathrm{d}\tau$$

$$= \sum_{t=0}^{T-1}\gamma^t\int_\tau d_0(s_0)\prod_{l=0}^{T-1}\pi_{\boldsymbol{\theta}}(a_l|s_l)P(s_{l+1}|s_l,a_l)r(s_t,a_t)\mathrm{d}\tau$$

$$= \sum_{t=0}^{T-1}\gamma^t\int_{\tau_{0:t}}\underbrace{d_0(s_0)\prod_{l=0}^{t-1}\pi_{\boldsymbol{\theta}}(a_l|s_l)P(s_{l+1}|s_l,a_l)\pi_{\boldsymbol{\theta}}(a_t|s_t)}_{\text{past}}r(s_t,a_t)$$

$$\times\int_{\tau_{t+1:T}}\underbrace{\cancelto{1}{P(s_{t+1}|s_t,a_t)\prod_{l=t+1}^{T-1}\pi_{\boldsymbol{\theta}}(a_l|s_l)P(s_{l+1}|s_l,a_l)}}_{\text{future}}\mathrm{d}\tau$$

Trajectory View and Transition View are Equivalent

Proof.

$$= \sum_{t=0}^{T-1} \gamma^t \int_{s_t, a_t} \underbrace{\int_{\tau_{0:t-1}} d_0(s_0) \prod_{l=0}^{t-1} \pi_{\boldsymbol{\theta}}(a_l|s_l) P(s_{l+1}|s_l, a_l) \mathrm{d}\tau_{0:t-1}}_{= \Pr(S_t = s_t | \pi_{\boldsymbol{\theta}}, S_0 \sim d_0)} \pi_{\boldsymbol{\theta}}(a_t|s_t) r(s_t, a_t) \mathrm{d}s_t \mathrm{d}a_t$$

$$= \sum_{t=0}^{T-1} \gamma^t \int_{s_t, a_t} \Pr\left(S_t = s_t | \pi_{\boldsymbol{\theta}}, S_0 \sim d_0\right) \pi(a_t|s_t) r(s_t, a_t) \mathrm{d}s_t \mathrm{d}a_t$$

$$= \int_{s,a} \underbrace{\sum_{t=0}^{T-1} \gamma^t \Pr\left(s_t = s | \pi_{\boldsymbol{\theta}}, s_0 \sim d_0\right)}_{d_{\pi_{\boldsymbol{\theta}}}(s)} \pi_{\boldsymbol{\theta}}(a|s) r(s, a) \mathrm{d}s_t \mathrm{d}a_t$$

$$= \int_{s,a} d_{\pi_{\boldsymbol{\theta}}}(s) \pi_{\boldsymbol{\theta}}(a|s) r(s,a) \mathrm{d}s \mathrm{d}a = \mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} [r(S, A)]$$

□

Outline

Policy Gradients

- Policy-based reinforcement learning is a **stochastic optimization** problem
- Find $\boldsymbol{\theta}$ that maximizes $J(\boldsymbol{\theta})$
- Some approaches **do not use gradient**
  - Hill climbing
  - Simplex
  - Genetic algorithms
- Greater **efficiency** often possible using gradient
  - Gradient descent
  - Conjugate gradient
  - Quasi–Newton
- We focus on **gradient descent**, many extensions possible
- And on methods that exploit **sequential structure**

Greedy vs Incremental

- **Greedy** updates

$$\boldsymbol{\theta}_{k+1} \in \arg\max_{\boldsymbol{\theta} \in \mathbb{R}^d} \mathbb{E}_{A \sim \pi_{\boldsymbol{\theta}}} [q_{\pi_{\boldsymbol{\theta}_k}}(s, A)]$$

$$v_{\pi_{\boldsymbol{\theta}_0}} \xrightarrow{\text{large change}} \pi_{\boldsymbol{\theta}_1} \xrightarrow{\text{large change}} v_{\pi_{\boldsymbol{\theta}_1}} \xrightarrow{\text{large change}} \pi_{\boldsymbol{\theta}_2} \xrightarrow{\text{large change}}$$

- Potentially **unstable** learning process with **large policy jumps**
- **Policy Gradient** updates

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \Big|_{\boldsymbol{\theta} = \boldsymbol{\theta}_k}$$

$$v_{\pi_{\boldsymbol{\theta}_0}} \xrightarrow{\text{small change}} \pi_{\boldsymbol{\theta}_1} \xrightarrow{\text{small change}} v_{\pi_{\boldsymbol{\theta}_1}} \xrightarrow{\text{small change}} \pi_{\boldsymbol{\theta}_2} \xrightarrow{\text{small change}}$$

- **Stable** learning process with **smooth policy improvement**

Policy Gradient

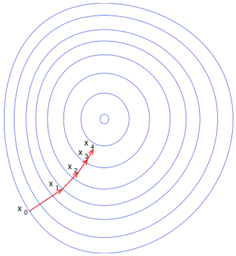
- Let $J : \mathbb{R}^d \to \mathbb{R}$ be any **policy objective function** (Peters and Schaal, 2008)
- Policy gradient algorithms search for a **local maximum** in $J(\boldsymbol{\theta})$ by **gradient ascent**

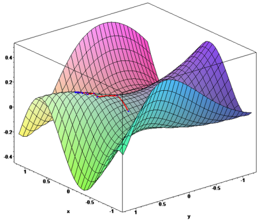
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta} \qquad\qquad \Delta\boldsymbol{\theta} = \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- where $\nabla_{\boldsymbol{\theta}} J(v\theta)$ is the **policy gradient**

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_d} \end{pmatrix}$$

- and α is a **step–size** parameter (or **learning rate**)





Policy Gradient Methods

- **Black-Box Approaches**
  - Finite-Difference Methods
- **White-Box Approaches**
  - Likelihood Ratio Methods (vanilla policy gradient, natural policy gradient)

Outline

Computing Gradients by Finite Differences

- **Black–box** approach (Sadegh and Spall, 1998)
- To **evaluate** policy gradient of $\pi_{\boldsymbol{\theta}}(a|s)$ with $\boldsymbol{\theta} \in \mathbb{R}^d$
- For each dimension $k \in \{1, \dots, d\}$
  - Estimate k–th **partial derivative** of objective function w.r.t. $\boldsymbol{\theta}$
  - By **perturbing** $\boldsymbol{\theta}$ by small amount ϵ in k–th dimension

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \theta_k} \approx \frac{J(\boldsymbol{\theta} + \epsilon \mathbf{u}_k) - J(\boldsymbol{\theta})}{\epsilon}$$

  where $\mathbf{u}_k$ is unit vector with 1 in k–th component, 0 elsewhere
- Simple, noisy, inefficient, but sometimes effective
- Works for arbitrary policies, even if policy is **not differentiable**
- Do not need to know the **functional form** of $\pi_{\boldsymbol{\theta}}(a|s)$

Outline

White–Box approach

- We now compute the gradient **analytically** (Peters and Schaal, 2008)
- Policy $\pi_{\boldsymbol{\theta}}(a|s)$ must be **stochastic** and **differentiable** in $\boldsymbol{\theta}$
- Assume we **know** the gradient $\nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s)$
- $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)$ is called **score function**
- **log trick** identity:

$$\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) \frac{\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})}{f(\boldsymbol{\theta})} = f(\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta})$$

Likelihood Ratio Gradient

- We can compute the gradient w.r.t. $\boldsymbol{\theta}$

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \int_{\mathcal{T}} p_{\boldsymbol{\theta}}(\tau) G(\tau) \mathrm{d}\tau \\
&= \int_{\mathcal{T}} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(\tau) G(\tau) \mathrm{d}\tau \\
&= \int_{\mathcal{T}} p_{\boldsymbol{\theta}}(\tau) \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) G(\tau) \mathrm{d}\tau \\
&= \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) G(\tau) \right]
\end{aligned}
$$

- We have rewritten the gradient as an **expectation** over trajectory
- so we can **estimate** it from samples!

Likelihood Ratio Gradient

- What about $\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau)$?

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) &= \nabla_{\boldsymbol{\theta}} \log \left(d_0(S_0) \prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}}(A_t|S_t) P(S_{t+1}|S_t, A_t) \right) \\
&= \nabla_{\boldsymbol{\theta}} \left(\log d_0(S_0) + \sum_{t=0}^{T-1} \log \pi_{\boldsymbol{\theta}}(A_t|S_t) + \sum_{t=0}^{T-1} \log P(S_{t+1}|S_t, A_t) \right) \\
&= \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_t|S_t)
\end{aligned}
$$

Example: Softmax Policy

- For **finite action spaces** ($|\mathcal{A}| < \infty$), we can use a **softmax policy**
- Weight actions using **linear combination** of features $\mathbf{x}(s,a)^\intercal \boldsymbol{\theta}$ where $\mathbf{x} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}^d$
- Probability of action is proportional to exponential weight

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{e^{\mathbf{x}(s,a)^\intercal \boldsymbol{\theta}}}{\int_{a'} e^{\mathbf{x}(s,a')^\intercal \boldsymbol{\theta}} \mathrm{d}a'} \propto e^{\mathbf{x}(s,a)^\intercal \boldsymbol{\theta}}$$

- The **score function** is

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) = \mathbf{x}(s,a) - \mathbb{E}_{A \sim \pi_{\boldsymbol{\theta}}(\cdot|s)}[\mathbf{x}(s,A)]$$

- Different representation with state features $\mathbf{x} : \mathcal{S} \rightarrow \mathbb{R}^d$ and action weights $\boldsymbol{\theta} : \mathcal{A} \rightarrow \mathbb{R}^d$

$$\pi_{\boldsymbol{\theta}}(a|s) \propto e^{\mathbf{x}(s)^\intercal \boldsymbol{\theta}(a)}$$

Example: Gaussian Policy

- In **continuous action spaces** $\mathcal{A} = \mathbb{R}$, a Gaussian policy is natural
- **Mean** is a linear combination of state features $\mu_{\boldsymbol{\theta}}(s) = \mathbf{x}(s)^{\mathsf{T}}\boldsymbol{\theta}$, where $\boldsymbol{\theta} : \mathcal{S} \to \mathbb{R}^d$
- **Variance** may be fixed σ^2, or can also parameterized
- Policy is a **Gaussian**, $a \sim \mathcal{N}(\mu_{\boldsymbol{\theta}}(s), \sigma)$

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{a - \mu_{\boldsymbol{\theta}}(s)}{\sigma}\right)^2\right)$$

- The **score function** is

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) = \frac{(a - \mu_{\boldsymbol{\theta}}(s))\mathbf{x}(s)}{\sigma^2}$$

- Can be extended to multidimensional actions $\mathcal{A} = \mathbb{R}^p$

Outline

REINFORCE

- Recall the **policy gradient** form

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) G(\tau) \right] \\
&= \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\left(\sum_{l=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l | S_l) \right) \left(\sum_{t=0}^{T-1} \gamma^t r(S_t, A_t) \right) \right]
\end{aligned}
$$

- Simplest idea is to replace the expectation with the sample mean $\rightarrow$ **REINFORCE** (Williams, 1992)

REINFORCE (Williams, 1992)

Initialize $\boldsymbol{\theta}$ arbitrarily

for all iterations $k = 1, \dots, K$ **do**

Sample m trajectories $\tau_i = (S_0^i, A_0^i, S_1^i, A_1^i, \dots, S_{T-1}^i, A_{T-1}^i, S_T^i)$ following $\pi_{\boldsymbol{\theta}}$

Compute the REINFORCE gradient estimate

$$\widehat{\nabla}_{\boldsymbol{\theta}}^{\mathsf{RF}} J(\boldsymbol{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \widehat{g}_i$$

where $$\widehat{g}_i = \left(\sum_{l=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l^i | S_l^i) \right) \left(\sum_{t=0}^{T-1} \gamma^t r(S_t^i, A_t^i) \right)$$

Update parameters

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \widehat{\nabla}_{\boldsymbol{\theta}}^{\mathsf{RF}} J(\boldsymbol{\theta})$$

end for

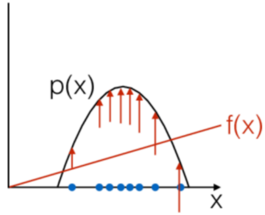
return $\boldsymbol{\theta}$

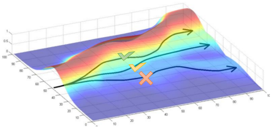
REINFORCE: Intuition

- $\widehat{g}_i$ are **unbiased** estimates of $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
- $G(\tau_i)$ measures how good is trajectory τ
- Moving in the direction of $\widehat{g}_i$ pushes up the log probability of the trajectory, in proportion to how good it is

$$\widehat{g}_i = \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau_i) G(\tau_i)$$

- **Interpretation**: uses good trajectories as supervised examples
  - Like maximum likelihood in supervised learning
  - good trajectories are made more likely while bad less
  - Trial and Error approach





Pictures from (Schulman, 2016) and (Levine, 2018)

REINFORCE

- Pros
  - Easy to compute
  - Does not use Markov property!
  - Can be used in **partially observable** MDPs without modification
- Cons
  - Use a single Monte Carlo estimate $G(\tau)$
  - It has possibly a very **large variance**: grows with T (Papini et al., 2019)

$$\mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_{\boldsymbol{\theta}}^{\mathsf{RF}} J(\boldsymbol{\theta})\right] \leq O\left(\frac{T\kappa R_{\max}}{m(1-\gamma)^2}\right) \qquad \text{where} \qquad \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\|_2 \leq \kappa$$

  - **Slow** convergence

Outline

G(PO)MDP

- We can **reduce the variance** thanks to the **causality property** (Baxter and Bartlett, 2001)
  - Reward collected in the **past** do not depend on actions played in the **future**

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\left(\sum_{l=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l | S_l) \right) \left(\sum_{t=0}^{T-1} \gamma^t r(S_t, A_t) \right) \right]$$

$$= \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T-1} \left(\gamma^t r(S_t, A_t) \sum_{l=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l | S_l) \right) \right]$$

$$= \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T-1} \left(\gamma^t r(S_t, A_t) \underbrace{\sum_{l=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l | S_l)}_{\text{past}} \right) \right] + \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T-1} \left(\gamma^t r(S_t, A_t) \underbrace{\cancel{\mathbb{E} \left[\sum_{l=t+1}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l | S_l) | \tau_{0:t} \right]}^{0}}_{\text{future}} \right) \right]$$

- This is a consequence of the **log trick**

$$\mathbb{E}_{A \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|s) \right] = \int \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) \mathrm{d}a = \nabla_{\boldsymbol{\theta}} \int \pi_{\boldsymbol{\theta}}(a|s) \mathrm{d}a = \nabla_{\boldsymbol{\theta}} 1 = 0$$

G(PO)MDP

Initialize $\boldsymbol{\theta}$ arbitrarily
for all iterations $k = 1, \dots, K$ **do**
Sample m trajectories $\tau_i = (S_0^i, A_0^i, S_1^i, A_1^i, \dots, S_{T-1}^i, A_{T-1}^i, S_T^i)$ following $\pi_{\boldsymbol{\theta}}$
Compute the G(PO)MDP gradient estimate

$$\widehat{\nabla}_{\boldsymbol{\theta}}^{\text{G(PO)MDP}} J(\boldsymbol{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \widehat{g}_i$$

$$\text{where} \qquad \widehat{g}_i = \left(\sum_{t=0}^{T-1} \gamma^t r(S_t^i, A_t^i) \sum_{l=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l^i | S_l^i) \right)$$

Update parameters

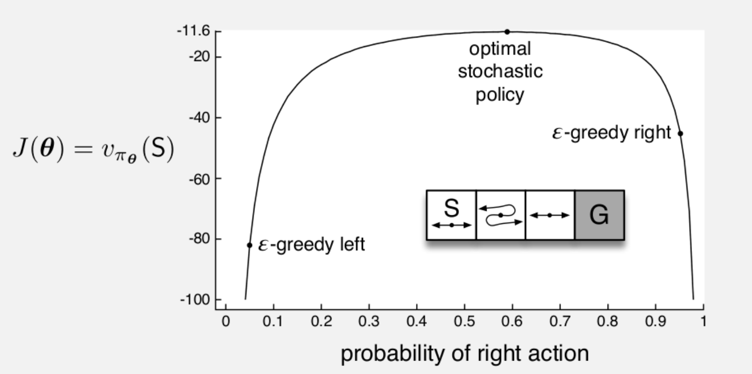
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \widehat{\nabla}_{\boldsymbol{\theta}}^{\text{G(PO)MDP}} J(\boldsymbol{\theta})$$

end for
return $\boldsymbol{\theta}$

Example: Short Corridor

- Left and Right actions have **reversed** effect in the central state
- -1 reward in every state $\neq$ G
- Start in state S
- **Softmax** policy with features

$$\mathbf{x}(s, \text{right}) = (1, 0)^{\mathsf{T}}$$
$$\mathbf{x}(s, \text{left}) = (0, 1)^{\mathsf{T}}$$

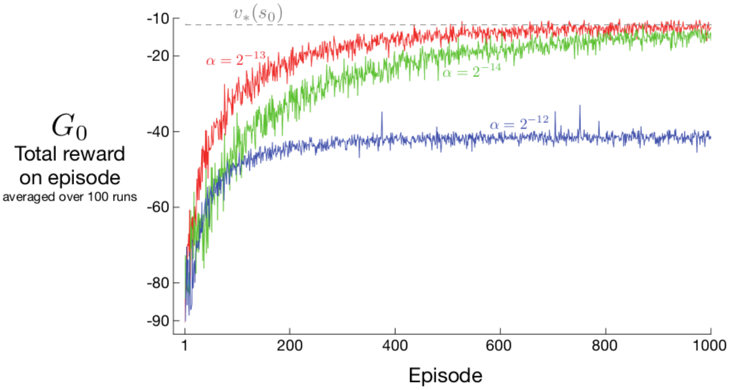


Pictures from (Sutton and Barto, 2018)

Example: Short Corridor - G(PO)MDP

- Left and Right actions have **reversed** effect in the central state
- -1 reward in every state $\neq$ G
- Start in state S
- **Softmax** policy with features

$$\mathbf{x}(s, \text{right}) = (1, 0)^{\mathsf{T}}$$
$$\mathbf{x}(s, \text{left}) = (0, 1)^{\mathsf{T}}$$



Pictures from (Sutton and Barto, 2018)

G(PO)MDP

- G(PO)MDP has **smaller** variance w.r.t. REINFORCE: no longer depends on T (Papini et al., 2019)

$$\mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_{\boldsymbol{\theta}}^{\mathsf{G(PO)MDP}} J(\boldsymbol{\theta})\right] \le O\left(\frac{\kappa R_{\max}}{m(1-\gamma)^3}\right) \qquad \text{where} \qquad \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\|_2 \le \kappa$$

- Still the variance is quite large

Policy Gradient and Baselines

- We can further reduce the variance with a **baseline** $\mathbf{b}(\tau) \in \mathbb{R}^d$

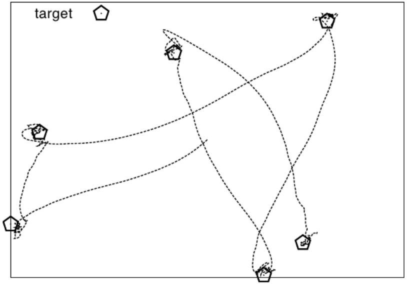
$$J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \odot (G(\tau) - \mathbf{b}(\tau)) \right] \qquad \odot = \text{element-wise product}$$

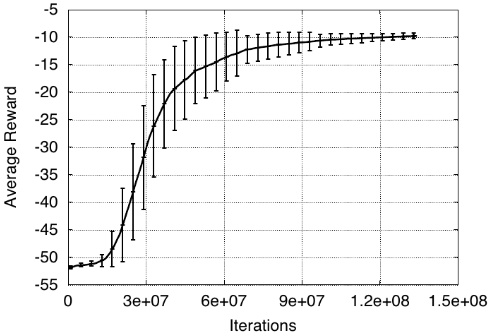
- **Unbiased** if $\mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \odot \mathbf{b}(\tau) \right] = \mathbf{0}$
- Computed to **minimize the variance** of the estimator
  - Scalar vs vectorial
  - Time-independent or time-dependent
- **Optimal** vectorial baseline for REINFORCE (Peters and Schaal, 2008)
  - The optimal one is **time-independent**

$$\mathbf{b}^{\text{RF}} = \frac{\mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\left(\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \right)^2 G(\tau) \right]}{\mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\left(\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \right)^2 \right]}$$

- **Peters** time-dependent vectorial baseline for G(PO)MDP (Peters and Schaal, 2008)

Puck World Example





- **Continuous actions** exert small force on puck
- Puck is rewarded for getting **close to target**
- Target location is **reset** every 30 seconds
- Policy is trained using variant (conjugate) of Monte–Carlo policy gradient

Pictures from (Silver, 2015)

Outline

Convergence Results

- Policy gradient is a **stochastic gradient**

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \widehat{\nabla} J(\boldsymbol{\theta}) = \boldsymbol{\theta} + \alpha \left(\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) + \text{noise} \right)$$

- $J(\boldsymbol{\theta})$ is **non-convex**
  - Converges **asymptotically** to a **local minimum** (under some technical assumptions) (Yuan et al., 2021)
- **Large variance** of stochastic gradients (growing with the length of the horizon)
- Possible **insufficient exploration**: naïve stochastic exploration
- **Global convergence** under some specific assumptions (Bhandari and Russo, 2019)

Outline

Going Beyond the Finite-Horizon

- So far, we considered **finite-length** trajectories
- What about **infinite-length** trajectories?

Theorem (Policy Gradient Theorem (Sutton et al., 1999))

For an infinite-horizon MDP, let $\pi_{\boldsymbol{\theta}}$ be a **stochastic** *policy* **differentiable** *in $\boldsymbol{\theta}$, the policy gradient is given by:*

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\pi_{\boldsymbol{\theta}}}(S, A) \right]$$

Proof of the Policy Gradient Theorem

Proof.

First of all, we observe:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int d_0(s) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(s) \mathrm{d}s$$

Consider the **Bellman equation**:

$$q_{\pi_{\boldsymbol{\theta}}}(s,a) = r(s,a) + \gamma \int p(y|s,a) v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y$$

We derive the **Bellman equation for the gradient**:

$$\begin{aligned}
\color{red}{\nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(s)} &= \nabla_{\boldsymbol{\theta}} \left(\int \pi_{\boldsymbol{\theta}}(a|s) q_{\pi_{\boldsymbol{\theta}}}(s,a) \mathrm{d}a \right) \\
&= \int \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) q_{\pi_{\boldsymbol{\theta}}}(s,a) \mathrm{d}a + \int \pi_{\boldsymbol{\theta}}(a|s) \nabla_{\boldsymbol{\theta}} q_{\pi_{\boldsymbol{\theta}}}(s,a) \mathrm{d}a \\
&= \int \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) q_{\pi_{\boldsymbol{\theta}}}(s,a) \mathrm{d}a + \underbrace{\gamma \int \pi_{\boldsymbol{\theta}}(a|s) \int p(y|s,a) \color{red}{\nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y)} \mathrm{d}y \mathrm{d}a}_{=f(s)}
\end{aligned}$$

Proof of the Policy Gradient Theorem

Proof.

Multiply by $d_{\pi_{\boldsymbol{\theta}}}(s)$ and integrate over the states

$$\begin{aligned}
\int d_{\pi_{\boldsymbol{\theta}}}(s) f(s) \mathrm{d}s &= \int d_{\pi_{\boldsymbol{\theta}}}(s) \gamma \int \pi_{\boldsymbol{\theta}}(a|s) \int p(y|s,a) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y \mathrm{d}a \mathrm{d}s \\
&= \int \sum_{t=0}^{+\infty} \gamma^t \Pr(S_t = s | \pi_{\boldsymbol{\theta}}, S_0 \sim d_0) \gamma \int \pi_{\boldsymbol{\theta}}(a|s) \int p(y|s,a) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y \mathrm{d}a \mathrm{d}s \\
&= \int \left(\sum_{t=0}^{+\infty} \gamma^{t+1} \Pr(S_{t+1} = y | \pi_{\boldsymbol{\theta}}, S_0 \sim d_0) \right) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y \\
&= \int \left(\sum_{t=0}^{+\infty} \gamma^{t+1} \Pr(S_{t+1} = y | \pi_{\boldsymbol{\theta}}, S_0 \sim d_0) + d_0(y) - d_0(y) \right) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y \\
&= \int d_{\pi_{\boldsymbol{\theta}}}(y) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y - \int d_0(y) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y
\end{aligned}$$

Integrating the gradient of the value function:

$$\cancel{\int d_{\pi_{\boldsymbol{\theta}}}(s) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(s) \mathrm{d}s} = \int d_{\pi_{\boldsymbol{\theta}}}(s) \int \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) q_{\pi_{\boldsymbol{\theta}}}(s,a) \mathrm{d}a \mathrm{d}s + \cancel{\int d_{\pi_{\boldsymbol{\theta}}}(y) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y} - \underbrace{\int d_0(y) \nabla_{\boldsymbol{\theta}} v_{\pi_{\boldsymbol{\theta}}}(y) \mathrm{d}y}_{\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})}$$

□

Outline

Reducing Variance using Critic

- G(PO)MDP still has a **high variance**
- We use a **critic** to estimate the action–value function

$$q_{\mathbf{w}}(s, a) \approx q_{\pi_{\boldsymbol{\theta}}}(s, a)$$

- Actor–critic algorithms maintain **two** sets of parameters
  - **Critic**: Updates **action–value function** parameters $\mathbf{w} \in \mathbb{R}^n$
  - **Actor**: Updates **policy parameters** $\boldsymbol{\theta} \in \mathbb{R}^d$, in direction suggested by critic
- Actor–critic algorithms follow an **approximate policy gradient** via policy gradient theorem

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &\approx \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} [\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\mathbf{w}}(S, A)] \\
\Delta\boldsymbol{\theta} &= \alpha \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\mathbf{w}}(S, A)
\end{aligned}$$

Estimating the Action–Value Function

- Computing the critic is a **policy evaluation** problem
  - G(PO)MDP is equivalent to estimate $q_{\mathbf{w}}(S_t, A_t)$ with a single MC simulation

$$q_{\mathbf{w}}(S_t, A_t) \equiv G_t = \sum_{l=t}^{T-1} \gamma^{l-t} r(S_l, A_l)$$

  - Monte Carlo policy evaluation
  - Temporal–Difference learning (TD(0), TD(λ))
  - Least–Squares Policy Evaluation

Action–Value Actor–Critic

- Using linear value function approximation $q_{\mathbf{w}}(s,a) = \mathbf{x}(s,a)^{\mathsf{T}}\mathbf{w}$
  - **Critic**: Updates $\mathbf{w}$ by linear semi-gradient TD(0) and learning rate β
  - **Actor**: Updates $\boldsymbol{\theta}$ by policy gradient theorem and learning rate α

Initialize $\boldsymbol{\theta} \in \mathbb{R}^n$ and $\mathbf{w} \in \mathbb{R}^d$
loop for each episode
    Initialize S
    Sample $A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)$
    Take action A, observe reward R, and next state S'
    loop for each step of the episode
        Sample $A' \sim \pi_{\boldsymbol{\theta}}(\cdot|S')$
        $\delta \leftarrow R + \gamma q_{\mathbf{w}}(S', A') - q_{\mathbf{w}}(S, A)$
        Update critic $\mathbf{w} \leftarrow \mathbf{w} + \beta\delta\nabla_{\mathbf{w}} q_{\mathbf{w}}(S, A)$
        Update actor $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\mathbf{w}}(S, A)$
        $S \leftarrow S'$
        $A \leftarrow A'$
        Take action A, observe reward R, and next state S'
    end loop
end loop

Bias in Actor–Critic Algorithms

- $q_{\mathbf{w}}(s,a)$ is a **biased** estimate of $q_{\pi_{\boldsymbol{\theta}}}(s,a)$
- The update of $\boldsymbol{\theta}$ may not follow the gradient of $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
- A biased policy gradient may **not** find the right solution
- If we choose action–value function approximation $q_{\mathbf{w}}(s,a)$ **carefully**, we can **avoid** any bias!

Outline

Compatible Function Approximation

Theorem (Compatible Function Approximation Theorem Sutton et al. (1999))

An action–value function $q_{\mathbf{w}}(s,a)$ *is* **compatible** *with the policy space* $\pi_{\boldsymbol{\theta}}$ *if:*

1. *The following identity between gradients hold:*

$$\nabla_{\mathbf{w}} q_{\mathbf{w}}(s,a) = \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) \qquad \forall (s,a) \in \mathcal{S} \times \mathcal{A}$$

2. *Value function parameters* $\mathbf{w}$ *minimize the mean square value error under the* γ*-discounted occupancy:*

$$\mathbf{w} \in \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^d} \overline{VE}(\mathbf{w}) := \frac{1}{2} \mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\left(q_{\pi_{\boldsymbol{\theta}}}(S,A) - q_{\mathbf{w}}(S,A) \right)^2 \right]$$

Then, the policy gradient computed replacing $q_{\pi_{\boldsymbol{\theta}}}(s,a)$ *with* $q_{\mathbf{w}}(s,a)$ *is exact, i.e.,*

$$\mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\mathbf{w}}(S,A) \right] = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

Proof of Compatible Function Approximation Theorem

Proof.

If $\mathbf{w}$ is chosen to **minimize** mean square value error, gradient of $\overline{\text{VE}}(\mathbf{w})$ w.r.t. $\mathbf{w}$ must be zero:

$$\mathbf{0} = \nabla_{\mathbf{w}} \overline{\text{VE}}(\mathbf{w}) = \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} [(q_{\pi_{\boldsymbol{\theta}}}(S,A) - q_{\mathbf{w}}(S,A)) \nabla_{\mathbf{w}} q_{\mathbf{w}}(S,A)] \qquad \text{(condition (2))}$$

$$= \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} [(q_{\pi_{\boldsymbol{\theta}}}(S,A) - q_{\mathbf{w}}(S,A)) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S)] \qquad \text{(condition (1))}$$

$$\implies \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} [\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\mathbf{w}}(S,A)] = \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} [\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\pi_{\boldsymbol{\theta}}}(S,A)] = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

□

- Actually, it is necessary that $\mathbf{w}$ is just a **stationary point** of $\overline{\text{VE}}(\mathbf{w})$
- A straightforward choice:

$$q_{\mathbf{w}}(s,a) = \mathbf{w}^{\mathrm{T}} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)$$

Actor-Critic with a Baseline

- Similarly to the trajectory case, we can use a **baseline** $\mathbf{b}(s)$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) \odot (q_{\pi_{\boldsymbol{\theta}}}(S, A) - \mathbf{b}(S)) \right]$$

- This can **reduce variance**, without biasing

$$\mathbb{E}_{A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) \odot \mathbf{b}(S) \right] = \mathbf{0}$$

- A **good** (but slightly suboptimal) choice is the state value function $b(s) = v_{\pi_{\boldsymbol{\theta}}}(s)$
- So we can rewrite the policy gradient using the **advantage function** $A_{\pi_{\boldsymbol{\theta}}}(s, a)$

$$\begin{aligned} A_{\pi_{\boldsymbol{\theta}}}(s, a) &= q_{\pi_{\boldsymbol{\theta}}}(s, a) - v_{\pi_{\boldsymbol{\theta}}}(s) \\ \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \mathop{\mathbb{E}}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) A_{\pi_{\boldsymbol{\theta}}}(S, A) \right] \end{aligned}$$

- The advantage function can notably reduce the variance!

Actor-Critic with a Advantage Function

- We could estimate $v_{\mathbf{v}}(s) \approx v_\pi(s)$ and $q_{\mathbf{w}}(s,a) \approx q_\pi(s,a)$ **independently**
- $A_{\mathbf{w},\mathbf{v}}(s,a) = q_{\mathbf{w}}(s,a) - v_{\mathbf{v}}(s)$ is **biased** and **unstable**
- Instead, we consider the TD-error

$$\delta_{\pi_{\boldsymbol{\theta}}}(s,a,s') = r(s,a) + \gamma v_{\pi_{\boldsymbol{\theta}}}(s') - v_{\pi_{\boldsymbol{\theta}}}(s)$$

- $\delta_{\pi_{\boldsymbol{\theta}}}(s,a,s')$ is an **unbiased** estimate for $A_{\pi_{\boldsymbol{\theta}}}(s,a)$

$$\mathbb{E}_{S' \sim p(\cdot|S,A)}\left[r(S,A) + \gamma v_{\pi_{\boldsymbol{\theta}}}(S') - v_{\pi_{\boldsymbol{\theta}}}(S) | S, A\right] = q_{\pi_{\boldsymbol{\theta}}}(S,A) - v_{\pi_{\boldsymbol{\theta}}}(S)$$

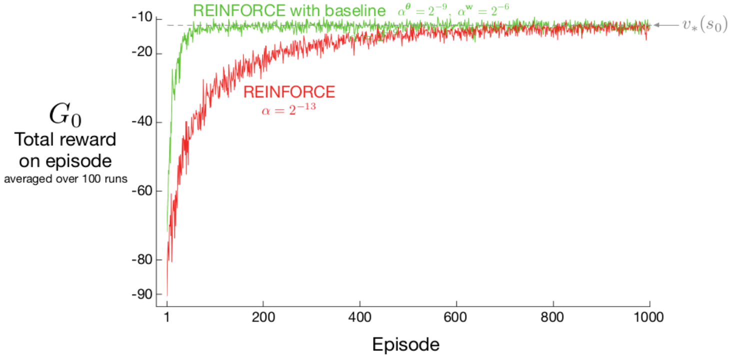
- In practice, we estimate just $v_{\mathbf{v}}(s) \approx v_{\pi_{\boldsymbol{\theta}}}(s)$ and approximate:

$$A_{\pi_{\boldsymbol{\theta}}}(s,a) \approx r(s,a) + \gamma v_{\mathbf{v}}(s') - v_{\mathbf{v}}(s)$$

Example: Short Corridor

- Left and Right actions have **reversed** effect in the central state
- -1 reward in every state $\neq$ G
- Start in state S
- Advantage function estimated as:

$$A_{\pi_{\boldsymbol{\theta}}}(S_t, A_t) \approx G_t - v_{\mathbf{v}}(S_t)$$



Pictures from (Sutton and Barto, 2018)

Outline

Outline

Alternative Policy Gradient Directions

- Gradient ascent algorithms can follow **any** ascent direction $\mathbf{d} \in \mathbb{R}^d$, i.e.

$$\mathbf{d}^{\mathsf{T}} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) > 0$$

- The **steepest ascent direction**, i.e., the **vanilla gradient** $\mathbf{d} = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ yields the **most increase** of $f(\boldsymbol{\theta})$ per "unit" of change in $\boldsymbol{\theta}$

$$\frac{\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})}{\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})\|_2} = \lim_{\epsilon \to 0} \operatorname*{arg\,max}_{\mathbf{d} \in \mathbb{R}^d : \|\mathbf{d}\|_2 \leq \epsilon} \underbrace{\frac{f(\boldsymbol{\theta} + \mathbf{d}) - f(\boldsymbol{\theta})}{\epsilon}}_{\approx \text{ incremental ratio}}$$

- The "unit" of change in $\boldsymbol{\theta}$ is measured in **Euclidean** distance $\|\cdot\|_2$
- Distance should be chosen based on the **manifold** and not based on **coordinates** (Amari, 1998)
- What if we change the **distance**?

Natural Gradient

- In a **Riemannian** space, the distance is defined as (Amari, 1998)

$$d(\boldsymbol{\theta}, \boldsymbol{\theta} + \Delta\boldsymbol{\theta}) = \|\Delta\boldsymbol{\theta}\|^2_{\mathbf{G}(\boldsymbol{\theta})} := \Delta\boldsymbol{\theta}^T \mathbf{G}(\boldsymbol{\theta}) \Delta\boldsymbol{\theta}$$

  where $\mathbf{G}(\boldsymbol{\theta})$ is the **metric tensor** (positive definite matrix)
  - In the Euclidian space we have $\mathbf{G}(\boldsymbol{\theta}) = \mathbf{Id}$
- The **steepest ascent in a Riemannian space** is given by Ollivier et al. (2017)

$$\frac{\mathbf{G}(\boldsymbol{\theta})^{-1} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})}{\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})\|_{\mathbf{G}(\boldsymbol{\theta})^{-1}}} = \lim_{\epsilon \to 0} \operatorname*{arg\,max}_{\mathbf{d} \in \mathbb{R}^d : \|\mathbf{d}\|_{\mathbf{G}(\boldsymbol{\theta})} \leq \epsilon} \frac{f(\boldsymbol{\theta} + \mathbf{d}) - f(\boldsymbol{\theta})}{\epsilon}$$

- The corresponding direction is called **natural gradient**

$$\widetilde{\nabla}_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \mathbf{G}(\boldsymbol{\theta})^{-1} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$$

- This is an **ascent direction** as $\mathbf{G}(\boldsymbol{\theta})$ is positive definite
- How to select the metric tensor $\mathbf{G}(\boldsymbol{\theta})$?

Fisher Information Matrix

- A common choice in ML is the **Fisher Information Matrix** (FIM) as metric tensor

$$\mathbf{F}(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau)^{\mathsf{T}} \right] = -\mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\mathcal{H}_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \right]$$

- The FIM has some remarkable properties
  - It is the Hessian of the **KL-divergence** between two distributions

$$\lim_{\|\Delta\boldsymbol{\theta}\|_2 \to 0} \mathcal{H}_{\boldsymbol{\theta}} D_{\mathsf{KL}} \left(p_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}} \right) = \mathbf{F}(\boldsymbol{\theta}) \qquad D_{\mathsf{KL}} \left(p_{\boldsymbol{\theta}'} \| p_{\boldsymbol{\theta}} \right) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}'}} \left[\log \frac{p_{\boldsymbol{\theta}'}(\tau)}{p_{\boldsymbol{\theta}}(\tau)} \right]$$

  - The second-order **Taylor expansion** of the KL-divergence

$$D_{\mathsf{KL}} \left(p_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}, p_{\boldsymbol{\theta}} \right) = \frac{1}{2} \Delta\boldsymbol{\theta}^{\mathsf{T}} \mathbf{F}(\boldsymbol{\theta}) \Delta\boldsymbol{\theta} + o(\|\Delta\boldsymbol{\theta}\|^3)$$

Natural Gradient

- A step of **vanilla gradient** $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ controls the Euclidean distance $\|\boldsymbol{\theta}' - \boldsymbol{\theta}\|_2$

$$\Delta\boldsymbol{\theta} = \alpha \frac{\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})}{\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})\|_2} \quad \Longrightarrow \quad D_{\mathsf{KL}}\left(p_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}, p_{\boldsymbol{\theta}}\right) \simeq \frac{\alpha^2}{2} \frac{\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})\|_{\mathbf{F}(\boldsymbol{\theta})}}{\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})\|_2}$$

- A step of **natural gradient** $\widetilde{\nabla}_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \mathbf{F}(\boldsymbol{\theta})^{-1} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ controls the KL-divergence $D_{\mathsf{KL}}\left(p_{\boldsymbol{\theta}'} \| p_{\boldsymbol{\theta}}\right)$

$$\Delta\boldsymbol{\theta} = \alpha \frac{\mathbf{F}(\boldsymbol{\theta})^{-1} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})}{\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})\|_{\mathbf{F}(\boldsymbol{\theta})^{-1}}} \quad \Longrightarrow \quad D_{\mathsf{KL}}\left(p_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}, p_{\boldsymbol{\theta}}\right) \simeq \frac{\alpha^2}{2}$$

- Thus, natural gradient is **invariant** to the parametrization $p_{\boldsymbol{\theta}}$

Example of Invariance to Parametrization

- Two parametrizations:

$$f_1(\theta) = \mathbb{E}_{Y \sim \mathcal{N}(\cdot|\theta,\sigma^2)}[r(Y)] \qquad\qquad f_2(\rho) = \mathbb{E}_{Y \sim \mathcal{N}(\cdot|2\rho,\sigma^2)}[r(Y)]$$

- We have that $f_1(\theta) = f_2(\rho)$ when $2\rho = \theta$
- Vanilla gradients

$$\nabla_\theta f_1(\theta) = \mathbb{E}_{Y \sim \mathcal{N}(\cdot|\theta,\sigma^2)}\left[\frac{Y-\theta}{\sigma^2} r(Y)\right] \qquad\qquad \nabla_\rho f_2(\rho) = \mathbb{E}_{Y \sim \mathcal{N}(\cdot|2\rho,\sigma^2)}\left[\frac{2(Y-2\rho)}{\sigma^2} r(Y)\right]$$

- When $2\rho = \theta$ we have that $2\nabla_\theta f_1(\theta) = \nabla_\rho f_2(\rho)$
- Suppose we update the corresponding parameters with gradient ascent and learning rate α:

$$\theta' = \theta + \alpha \nabla_\theta f_1(\theta) \qquad\qquad \rho' = \rho + \alpha \nabla_\rho f_2(\rho)$$

- But $\theta' \neq 2\rho' \rightarrow$ **parametrization dependence**

$$\theta' = \theta + \alpha \nabla_\theta f_1(\theta) = 2\rho + \frac{\alpha}{2}\nabla_\rho f_2(\rho) \neq 2\rho + 2\alpha\nabla_\rho f_2(\rho) = 2\rho'$$

Example of Invariance to Parametrization

- Fisher information matrices

$$F_1(\theta) = \mathbb{E}_{Y\sim\mathcal{N}(\cdot|\theta,\sigma^2)}\left[\left(\frac{Y-\theta}{\sigma^2}\right)^2\right] = \frac{1}{\sigma^2} \qquad F_2(\rho) = \mathbb{E}_{Y\sim\mathcal{N}(\cdot|2\rho,\sigma^2)}\left[\left(\frac{2(Y-2\rho)}{\sigma^2}\right)^2\right] = \frac{4}{\sigma^2}$$

- Natural gradients

$$\widetilde{\nabla}_\theta f_1(\theta) = F_1(\theta)^{-1}\nabla_\theta f_1(\theta) = \mathbb{E}_{Y\sim\mathcal{N}(\cdot|\theta,\sigma^2)}\left[(Y-\theta)r(Y)\right]$$

$$\widetilde{\nabla}_\rho f_2(\rho) = F_2(\rho)^{-1}\nabla_\rho f_2(\rho) = \mathbb{E}_{Y\sim\mathcal{N}(\cdot|2\rho,\sigma^2)}\left[\frac{1}{2}(Y-2\rho)r(Y)\right]$$

- When $2\rho = \theta$ we have that $\frac{1}{2}\nabla_\theta f_1(\theta) = \nabla_\rho f_2(\rho)$
- Suppose we update the corresponding parameters with gradient ascent and learning rate α:

$$\theta' = \theta + \alpha\nabla_\theta f_1(\theta) \qquad \rho' = \rho + \alpha\nabla_\rho f_2(\rho)$$

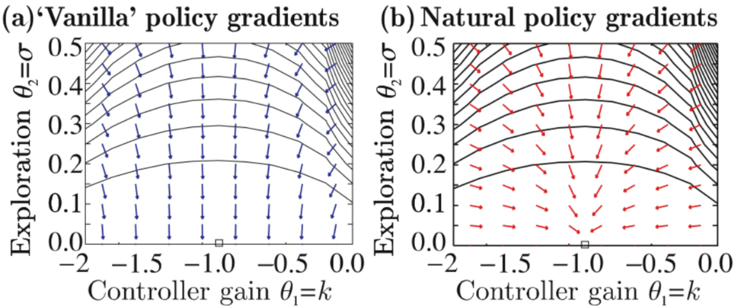
- Now $\theta' = 2\rho' \rightarrow$ **parametrization invariance**

$$\theta' = \theta + \alpha\nabla_\theta f_1(\theta) = 2\rho + 2\alpha\nabla_\rho f_2(\rho) = 2\rho'$$

Natural Policy Gradient

- In the policy gradient case, the FIM can be simplified as (Kakade, 2001)

$$\mathbf{F}(\boldsymbol{\theta}) = \mathbb{E}_{\substack{S\sim d_{\pi_{\boldsymbol{\theta}}} \\ A\sim\pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S)^{\mathsf{T}}\right]$$



Pictures from (Peters et al., 2003)

Outline

Natural Actor Critic

- Using **compatible** function approximation (Peters et al., 2005)

$$q_{\mathbf{w}}(s,a) = \mathbf{w}^{\mathsf{T}} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)$$

- So the natural policy gradient surprisingly **simplifies**

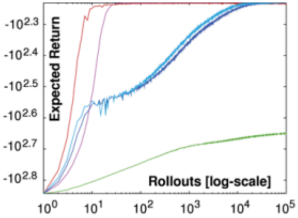
$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) q_{\mathbf{w}}(A|S) \right] \\
&= \mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S)^{\mathsf{T}} \mathbf{w} \right] \\
&= \mathbf{F}(\boldsymbol{\theta}) \mathbf{w} \\
\widetilde{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \mathbf{F}(\boldsymbol{\theta})^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbf{w}
\end{aligned}$$

- i.e., update actor parameters in direction of critic parameters

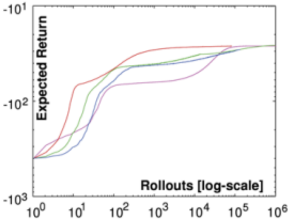
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \mathbf{w}$$

- Episodic versions with time-invariant and time-variant baselines (Peters et al., 2005)

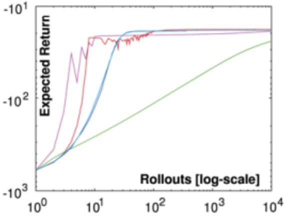
Example



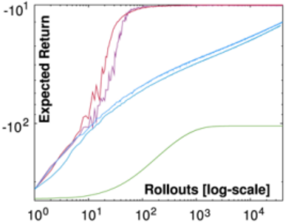
(a) Minimum motor command with splines



(c) Passing through a point with splines

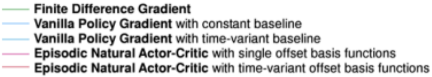


(b) Minimum motor command with motor primitives



(d) Passing through a point with motor primitives

- Learn motor plans with policy gradients
  - spline-based trajectory plans
  - nonlinear dynamic motor primitives



Pictures from (Peters and Schaal, 2008)

Outline

Outline

Off-Policy Policy Gradient

- Can we estimate the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}')$ having samples collected with $\pi_{\boldsymbol{\theta}}$?
  - $\pi_{\boldsymbol{\theta}'}$ **target** policy
  - $\pi_{\boldsymbol{\theta}}$ **behavioral** policy
- **Importance weighted** policy gradient

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}') = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\frac{p_{\boldsymbol{\theta}'}(\tau)}{p_{\boldsymbol{\theta}}(\tau)} \sum_{t=0}^{T-1} \gamma^t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}'}(A_t|S_t) q_{\pi_{\boldsymbol{\theta}}}(S_t, A_t) \right]$$

$$\approx \frac{1}{m} \sum_{i=1}^{m} \frac{p_{\boldsymbol{\theta}'}(\tau_i)}{p_{\boldsymbol{\theta}}(\tau_i)} \sum_{t=0}^{T-1} \gamma^t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}'}(A_t^i|S_t^i) q_{\pi_{\boldsymbol{\theta}}}(S_t^i, A_t^i) =: \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}'/\boldsymbol{\theta})$$

where $$\frac{p_{\boldsymbol{\theta}'}(\tau)}{p_{\boldsymbol{\theta}}(\tau)} = \frac{d_0(S_0) \prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}'}(A_t|S_t) P(S_{t+1}|S_t, A_t)}{d_0(S_0) \prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}}(A_t|S_t) P(S_{t+1}|S_t, A_t)} = \frac{\prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}'}(A_t|S_t)}{\prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}}(A_t|S_t)}$$

- The estimator is **unbiased** but...

Curse of Horizon

- Unfortunately the **variance** can explode (Metelli et al., 2018)
- The **variance** grows **exponentially** with the horizon T

$$\mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}'/\boldsymbol{\theta})\right] \lessapprox \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}}\left[\left(\frac{p_{\boldsymbol{\theta}'}(\tau)}{p_{\boldsymbol{\theta}}(\tau)}\right)^2\right] \leq \sup_{s \in \mathcal{S}} \underbrace{\mathbb{E}_{A \sim \pi_{\boldsymbol{\theta}}(\cdot|s)}\left[\left(\frac{\pi_{\boldsymbol{\theta}'}(A|s)}{\pi_{\boldsymbol{\theta}}(A|s)}\right)^2\right]}_{\approx \text{policy distance}}{}^{T}$$

- Can even be **infinite**! $\rightarrow$ **Curse of Horizon** (Liu et al., 2020)
- Several **general-purpose** fixes: clipping (Ionides, 2008), normalizations (Kuzborskij et al., 2021), smoothing (Metelli et al., 2021)...
- Some fixes specific for MDPs
  - **Per-decision** importance weighting (Precup et al., 2000)
  - **Stationary** importance weighting (Liu et al., 2018)

Per–Decision Importance Weighting

- Use **causality property** like when deriving G(PO)MDP from REINFORCE

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}') = \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T-1} \prod_{l=0}^{t} \frac{\pi_{\boldsymbol{\theta}'}(A_l|S_l)}{\pi_{\boldsymbol{\theta}}(A_l|S_l)} \gamma^t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}'}(A_t|S_t) q_{\pi_{\boldsymbol{\theta}}}(S_t, A_t) \right]$$

$$\approx \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T-1} \prod_{l=0}^{t} \frac{\pi_{\boldsymbol{\theta}'}(A_l^i|S_l^i)}{\pi_{\boldsymbol{\theta}}(A_l^i|S_l^i)} \gamma^t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}'}(A_t^i|S_t^i) q_{\pi_{\boldsymbol{\theta}}}(S_t^i, A_t^i) =: \widehat{\nabla}_{\boldsymbol{\theta}} J^{\mathsf{PD}}(\boldsymbol{\theta}'/\boldsymbol{\theta})$$

- Usually **smaller variance** than vanilla importance weighting (Metelli et al., 2020)

$$\mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_{\boldsymbol{\theta}}^{\mathsf{PD}} J(\boldsymbol{\theta}'/\boldsymbol{\theta})\right] \lessapprox \sum_{t=0}^{T-1} \gamma^{2t} D^t \qquad\qquad D := \sup_{s \in \mathcal{S}} \mathbb{E}_{A \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[\left(\frac{\pi_{\boldsymbol{\theta}'}(A|s)}{\pi_{\boldsymbol{\theta}}(A|s)} \right)^2 \right]$$

Stationary Importance Weighting

- Exploit the **occupancy view** of policy gradient

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}') = \mathbb{E}_{\substack{S \sim d_{\pi_{\boldsymbol{\theta}}} \\ A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)}} \left[\frac{d_{\pi_{\boldsymbol{\theta}'}}(S) \pi_{\boldsymbol{\theta}'}(A|S)}{d_{\pi_{\boldsymbol{\theta}}}(S) \pi_{\boldsymbol{\theta}}(A|S)} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}'}(A|S) q_{\pi_{\boldsymbol{\theta}}}(S, A) \right]$$

$$\approx \frac{1}{m} \sum_{i=1}^{m} \frac{d_{\pi_{\boldsymbol{\theta}'}}(S^i) \pi_{\boldsymbol{\theta}'}(A^i|S^i)}{d_{\pi_{\boldsymbol{\theta}}}(S^i) \pi_{\boldsymbol{\theta}}(A^i|S^i)} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}'}(A^i|S^i) q_{\pi_{\boldsymbol{\theta}}}(S^i, A^i) =: \widehat{\nabla}^{\mathsf{S}}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}'/\boldsymbol{\theta})$$

- The estimator is **unbiased** and mitigates the **curse of horizon**, but...
- Requires samples from the distribution $d_{\pi_{\boldsymbol{\theta}}} \rightarrow$ possibly inefficient sampling
- The functional form of $d_{\pi_{\boldsymbol{\theta}}}$ is **unknown** (depends on the transition model p) (Liu et al., 2018)

Outline

Trust Regions

- Samples collected with $\pi_{\boldsymbol{\theta}_{\text{old}}}$ can estimate **accurately** performance of policies $\pi_{\boldsymbol{\theta}}$ "close" to $\pi_{\boldsymbol{\theta}_{\text{old}}}$
- **Stability** can be improved by limiting the updates between subsequent policies
- We use a **divergence** D between policies (e.g., KL-divergence, f-divergence, Wasserstein, ...)
  - **Penalized** approach

$$\max_{\boldsymbol{\theta} \in \mathbb{R}^d} J(\boldsymbol{\theta}) - cD\left(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_{\text{old}}}\right)$$

  - **Constraint** approach

$$\max_{\boldsymbol{\theta} \in \mathbb{R}^d} J(\boldsymbol{\theta})$$
$$\text{s.t. } D\left(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_{\text{old}}}\right) \leq c$$

  where c is a hyperparameter
- Many examples: TRPO (Schulman et al., 2015), PPO (Schulman et al., 2017), POIS (Metelli et al., 2018), ...
- We will see in the next lectures...

Outline

REINFORCE Baseline

Exercise 1

Consider the REINFORCE estimator with baseline:

$$\widehat{\nabla}_{\boldsymbol{\theta}}^{\mathsf{RF}} J(\boldsymbol{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \left(\sum_{t=0}^{T-1} \gamma^t r(S_t^i, A_t^i) - \mathbf{b} \right) \odot \left(\sum_{l=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l^i | S_l^i) \right)$$

1. Compute the expression of $\mathbf{b}$ that minimizes the variance of the estimator assuming that:
   1. the baseline is scalar $b \in \mathbb{R}$;
   2. the baseline is vectorial $\mathbf{b} \in \mathbb{R}^d$.
2. Propose an estimator for such a baseline and discuss its biasedness.

References I

S. Amari. Natural gradient works efficiently in learning. *Neural Comput.*, 10(2):251–276, 1998.

L. Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *J. Artif. Intell. Res.*, 15:319–350, 2001. doi: 10.1613/jair.806.

J. Bhandari and D. Russo. Global optimality guarantees for policy gradient methods. *CoRR*, abs/1906.01786, 2019. URL http://arxiv.org/abs/1906.01786.

M. H. Hado van Hasselt, Diana Borsa. Reinforcement learning lecture series 2021. https://deepmind.com/learning-resources/reinforcement-learning-series-2021, 2015.

E. L. Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17(2):295–311, 2008.

S. M. Kakade. A natural policy gradient. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 1531–1538. MIT Press, 2001.

I. Kuzborskij, C. Vernade, A. Gyorgy, and C. Szepesvári. Confident off-policy evaluation and selection through self-normalized importance weighting. In *International Conference on Artificial Intelligence and Statistics*, pages 640–648. PMLR, 2021.

M. Lee and C. W. Anderson. Convergent reinforcement learning control with neural networks and continuous action search. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–8. IEEE, 2014.

S. Levine. Cs 294-112: Deep reinforcement learning, policy gradients. http://rail.eecs.berkeley.edu/deeprlcourse-fa18/, 2018.

Q. Liu, L. Li, Z. Tang, and D. Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

Y. Liu, P.-L. Bacon, and E. Brunskill. Understanding the curse of horizon in off-policy evaluation via conditional importance sampling. In *International Conference on Machine Learning*, pages 6184–6193. PMLR, 2020.

H. R. Maei. Gradient temporal-difference learning algorithms. 2011.

References II

H. R. Maei and R. S. Sutton. Gq (λ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. In *Proceedings of the third conference on artificial general intelligence*, volume 1, pages 91–96. Citeseer, 2010.

H. R. Maei, C. Szepesvári, S. Bhatnagar, D. Precup, D. Silver, and R. S. Sutton. Convergent temporal-difference learning with arbitrary smooth function approximation. In *Advances in Neural Information Processing Systems 22 (NIPS)*, pages 1204–1212, 2009.

H. R. Maei, C. Szepesvári, S. Bhatnagar, and R. S. Sutton. Toward off-policy learning control with function approximation. In *International Conference on Machine Learning (ICML)*, 2010.

A. M. Metelli, M. Papini, F. Faccio, and M. Restelli. Policy optimization via importance sampling. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pages 5447–5459, 2018.

A. M. Metelli, M. Papini, N. Montali, and M. Restelli. Importance sampling techniques for policy optimization. *Journal of Machine Learning Research*, 21(141):1–75, 2020.

A. M. Metelli, A. Russo, and M. Restelli. Subgaussian and differentiable importance sampling for off-policy evaluation and learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *J. Mach. Learn. Res.*, 18:18:1–18:65, 2017.

M. Papini, M. Pirotta, and M. Restelli. Smoothing policies and safe policy gradients. *arXiv preprint arXiv:1905.03231*, 2019.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008. doi: 10.1016/j.neunet.2008.02.003.

J. Peters, S. Vijayakumar, and S. Schaal. Reinforcement learning for humanoid robotics. In *Proceedings of the third IEEE-RAS international conference on humanoid robots*, pages 1–20, 2003.

J. Peters, S. Vijayakumar, and S. Schaal. Natural actor-critic. In J. Gama, R. Camacho, P. Brazdil, A. Jorge, and L. Torgo, editors, *Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005, Proceedings*, volume 3720 of *Lecture Notes in Computer Science*, pages 280–291. Springer, 2005. doi: 10.1007/11564096_29.

References III

D. Precup, R. S. Sutton, and S. P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pages 759–766. Morgan Kaufmann, 2000.

P. Sadegh and J. C. Spall. Optimal random perturbations for stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 43(10):1480–1484, 1998.

J. Schulman. Deep reinforcement learning: Policy gradients and q-learning. Technical report, Bay Area Deep Learning School, 2016.

J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, volume 37, pages 1889–1897. JMLR.org, 2015.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

D. Silver. Introduction to reinforcement learning. `https://deepmind.com/learning-resources/-introduction-reinforcement-learning-david-silver`, 2015.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12 (NIPS)*, pages 1057–1063. The MIT Press, 1999.

R. S. Sutton, C. Szepesvári, and H. R. Maei. A convergent o(n) temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pages 1609–1616. Curran Associates, Inc., 2008.

R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, volume 382, pages 993–1000, 2009.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696.

R. Yuan, R. M. Gower, and A. Lazaric. A general sample complexity analysis of vanilla policy gradient. *arXiv preprint arXiv:2107.11433*, 2021.