Brain Inspired Artificial Intelligence 4: Policy Search and Actor-Critic Methods

Eiji Uchibe

Dept. of Brain Robot Interface

ATR Computational Neuroscience Labs.

Outline

- Policy gradient approaches to MDP problems
  - Finite Differences Method
  - REINFORCE
  - Variance reduction
- Actor-critic method
  - Stationary distribution
  - Policy Gradient Theorem
- Natural gradient
  - Kullback Leibler divergence

Policy Based Approaches

- In the last lecture, we represent the state value or state-action value function
- A policy is computed directly from the learned value function
  - e.g. using epsilon-greedy

- In this lecture, we will directly parameterize the policy

Three Types of RL

critic only (value-based RL)

learn the value function $q(s,a)$

$\pi(a \mid s)$ is derived from $q(s,a)$

Q-learning, double Q-learning SARSA, DQN

actor-critic

learn $q(s,a)$ and $\pi(a \mid s)$

$\pi(a \mid s)$ is modified by $q(s,a)$

A2C, A3C, DDPG, soft Actor-Critic

actor only (policy-based RL)

learn the policy $\pi(a \mid s)$

$q(s,a)$ is not estimated

REINFORCE, PGPE

Policy-Based Approach

- Suppose that a finite-length trajectory τ is generated by some stochastic policy $\pi(a \mid s, \boldsymbol{\theta})$

$$\tau = \{S_0, A_0, R_1, S_1, A_1, R_2, S_3, \dots, S_{T-1}, A_{T-1}, R_T, S_T\}$$

- The return of a trajectory τ is

$$G(\tau) = \sum_{t=1}^{T} \gamma^{t-1} R_t$$

random variable!

- Reminder: Under the Markovian assumption, a probability to observe τ is

$$\Pr(\tau \mid \boldsymbol{\theta}) = p_0(S_0) \prod_{t=0}^{T-1} \pi(A_t \mid S_t, \boldsymbol{\theta}) p(S_{t+1}, R_t \mid S_t, A_t)$$

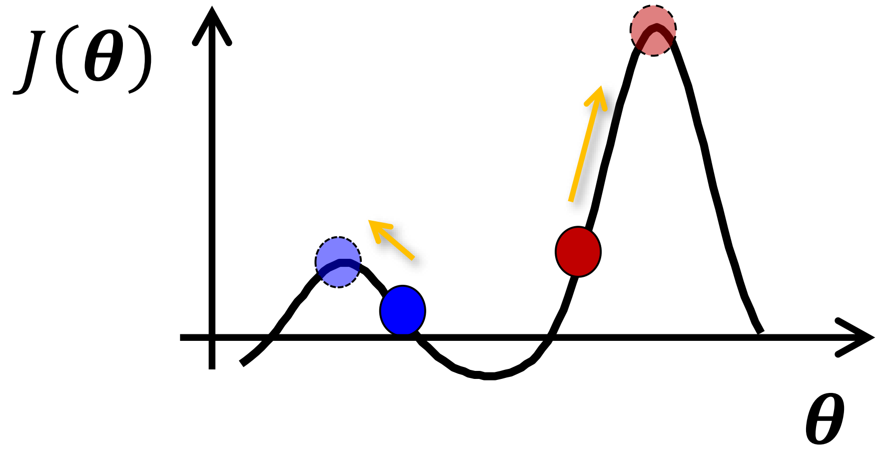
Policy-Based Approach

- The goal is to maximize the objective function

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[G(\tau)] = \sum_{\tau} \Pr(\tau \mid \boldsymbol{\theta})G(\tau)$$

- It is not feasible to compute $\sum_{\tau}$ in practice because we have to consider all possible trajectories
- In order to maximize $J(\boldsymbol{\theta})$, a gradient method is applied

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



where α is a positive step-size parameter

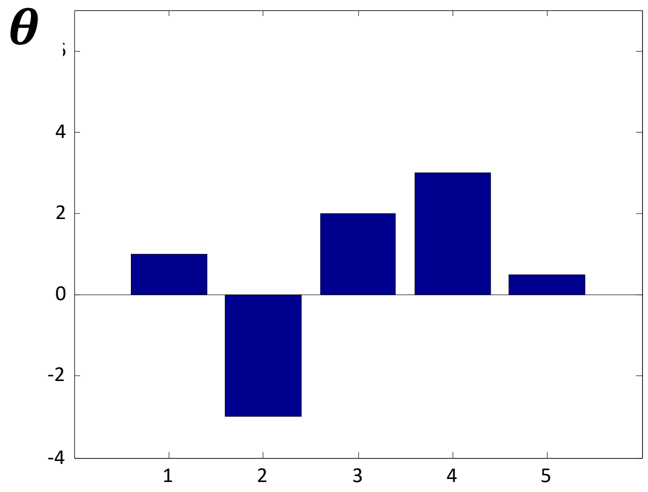
- Local optimal solution

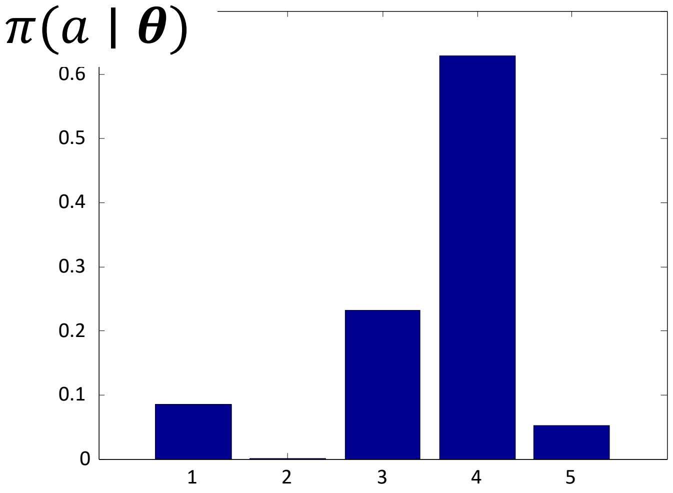
Reminder: Stochastic Policy in the Bandit Problem

- We considered the policy represented by the Boltzmann distribution

$$\pi(a_i \mid \boldsymbol{\theta}) = \frac{\exp(\theta_i)}{\sum_{j=1}^{k} \exp\left(\theta_j\right)}$$

- θ_i is interpreted as a preference value of action a_i
- We are going to consider $\pi(a \mid s, \boldsymbol{\theta})$ that is a conditional probability on state s





Linear-Exponential Policy

- We introduce a state-action feature vector $\boldsymbol{\phi}(s, a)$ to represent the preference for action a in state s

$$\pi(a \mid s, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}^\top \boldsymbol{\phi}(s, a))}{\sum_{a'} \exp\left(\boldsymbol{\theta}^\top \boldsymbol{\phi}(s, b)\right)}$$

- The derivative of the log policy is given by

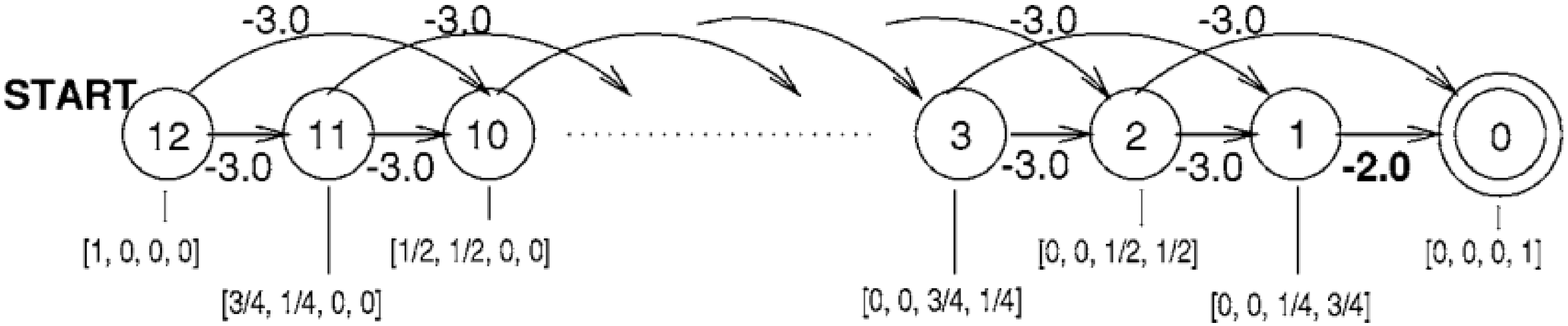
$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta}) &= \boldsymbol{\phi}(s, a) - \mathbb{E}_\pi[\boldsymbol{\phi}(s, a)] \\ &= \boldsymbol{\phi}(s, a) - \sum_{a'} \pi(\, a' \mid s, \boldsymbol{\theta}\,) \boldsymbol{\phi}(s, a')\end{aligned}$$

Policy Parameterization

- State-dependent feature vector is widely used for simplicity

$$\pi(a \mid s, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}_a^\top \boldsymbol{\phi}(s))}{\sum_{b \in \mathcal{A}} \exp(\boldsymbol{\theta}_b^\top \boldsymbol{\phi}(s))}$$

- Example of the feature vector $\boldsymbol{\phi}(s)$ [Boyan 2002]
  - 12 discrete states
  - 4 dimensional feature vector



Policy Parameterization (Continuous Actions)

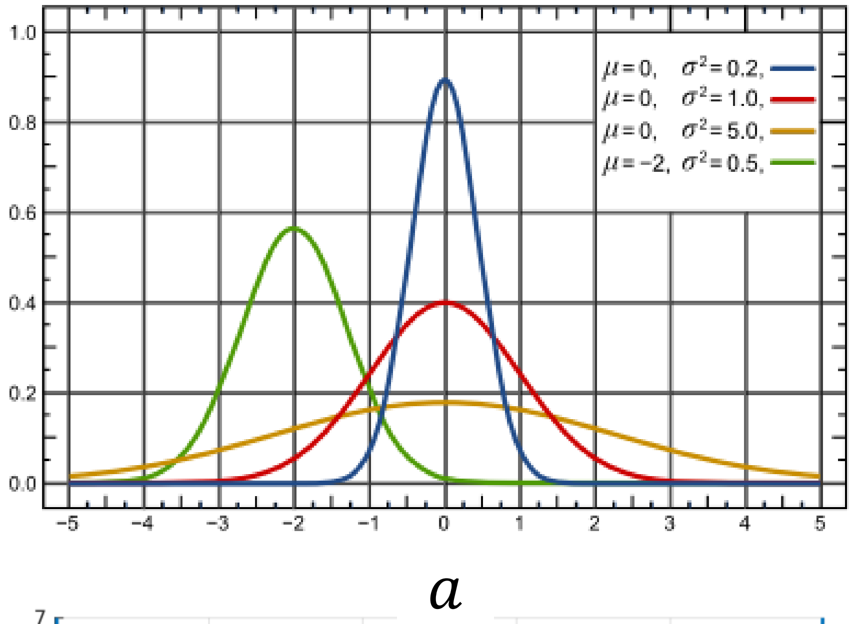
- Gaussian policy

$$\pi(\,a \mid s, \boldsymbol{\theta}\,) = \mathcal{N}\left(\,a \mid \mu(s, \boldsymbol{\theta}), \sigma^2\,\right)$$

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

where $\mu(s, \boldsymbol{\theta}) = \boldsymbol{\phi}(s)^\top \boldsymbol{\theta}$

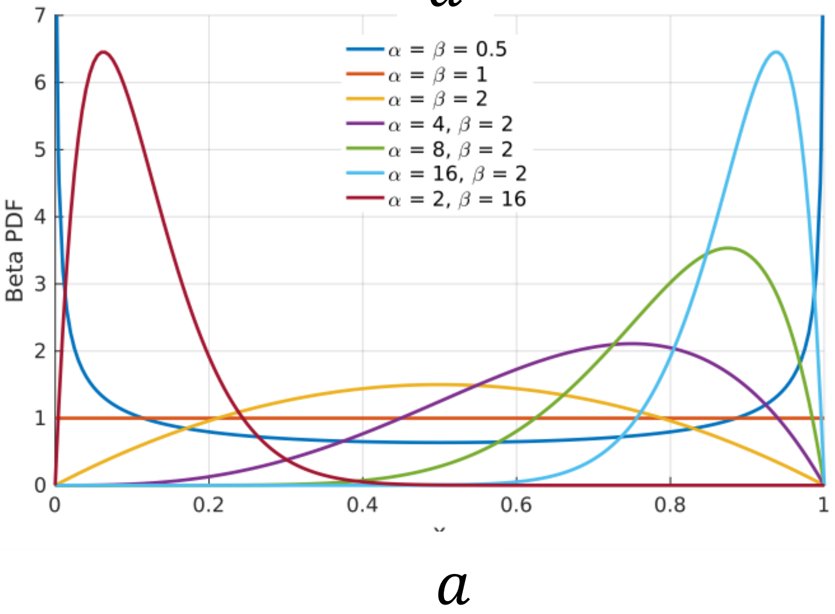
  – infinite support $a \in (-\infty, +\infty)$



- Beta policy

$$\pi(\,a \mid s, \boldsymbol{\theta}\,) = \mathcal{B}\left(\frac{a+h}{2h} \,\middle|\, \alpha(s, \boldsymbol{\theta}), \beta(s, \boldsymbol{\theta})\right)$$

  – finite support $a \in [-h, h]$



Finite Differences Method

- For each dimension
  - Estimate k-th partial derivative of objective function with respect to $\boldsymbol{\theta}$
  - By perturbing $\boldsymbol{\theta}$ by small amount ϵ in k-th dimension

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

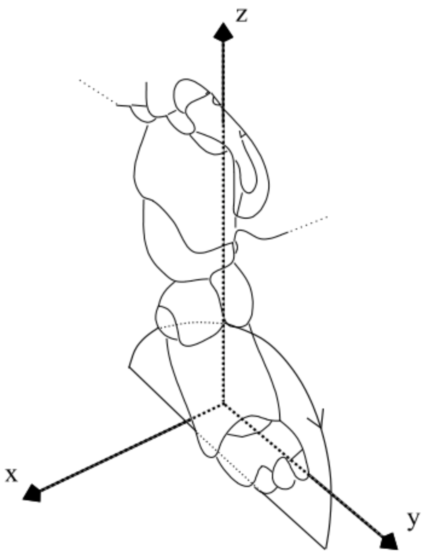
k-th element

where $\boldsymbol{u}_k = [0, \dots, 0, 1, 0, \dots, 0]^\top$

- Use n evaluations to compute policy gradient in n dimensions
- Sometimes effective

Training AIBO to Walk

- J is given by field traversal time



Parameter	Initial Value	ϵ	Best Value
Front locus:			
(height)	4.2	0.35	4.081
(x offset)	2.8	0.35	0.574
(y offset)	4.9	0.35	5.152
Rear locus:			
(height)	5.6	0.35	6.02
(x offset)	0.0	0.35	0.217
(y offset)	-2.8	0.35	-2.982
Locus length	4.893	0.35	5.285
Locus skew multiplier	0.035	0.175	0.049
Front height	7.7	0.35	7.483
Rear height	11.2	0.35	10.843
Time to move through locus	0.704	0.016	0.679
Time on ground	0.5	0.05	0.430

[Kohn and Stone, 2004]

Training AIBO to Walk

initial behavior

during training

learned behaviors

[Kohn and Stone, 2004]

Computing Policy Gradient Analytically

- Finite differences method works for arbitrary policies even if a policy is not differentiable
- However, it often estimates noisy policy gradient
- We now compute the policy gradient analytically
- $$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \left(\sum_{\tau} G(\tau) \Pr(\tau \mid \boldsymbol{\theta}) \right)$$
$$= \sum_{\tau} G(\tau) \frac{\partial \Pr(\tau \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{\tau} \Pr(\tau \mid \boldsymbol{\theta}) \, G(\tau) \frac{\partial \ln \Pr(\tau \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$
$$= \mathbb{E}_{\Pr(\tau \mid \boldsymbol{\theta})} [G(\tau) \nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})]$$

Derivation

- $\ln \Pr(\tau \mid \boldsymbol{\theta}) = \ln p_0(S_0) + \sum_{t=0}^{T-1} \ln \pi(A_t \mid S_t, \boldsymbol{\theta})$
$$+ \sum_{t=0}^{T-1} \ln p(S_{t+1}, R_t \mid S_t, A_t)$$

- Since the first and third terms do not depend on $\boldsymbol{\theta}$, the derivative is given by

$$\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta}) = \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \ln \pi(A_t \mid S_t, \boldsymbol{\theta})$$

- We do not need to know the environmental dynamics
  ➡ model free

Baseline

- As we studied in the bandit problems, the policy gradient remains unchanged even if a constant baseline is subtracted

$$\mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[G(\tau)\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})]$$
$$= \mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[(G(\tau) - b)\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})]$$

where b is a constant baseline

Reminder: Proof Sketch

- $\mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[(G(\tau) - b)\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})]$

  $= \mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[G(\tau)\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})] - b\mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})]$

- $\mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})] = \sum_{\tau} \Pr(\tau \mid \boldsymbol{\theta})\nabla_{\boldsymbol{\theta}} \ln \Pr(\tau \mid \boldsymbol{\theta})$

  $$= \sum_{\tau} \frac{\partial \Pr(\tau \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial}{\partial \boldsymbol{\theta}}\left[\sum_{\tau} \Pr(\tau \mid \boldsymbol{\theta})\right]$$

  $$= \frac{\partial}{\partial \boldsymbol{\theta}}[1] = \mathbf{0}$$

Variance Reduction

- We can tune the baseline b to reduce the variance of the policy gradient

$$\nabla_{\boldsymbol{\theta}}\, J(\boldsymbol{\theta}) = \mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}\left[(G(\tau) - b)\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \ln \pi(A_t \mid S_t, \boldsymbol{\theta})\right]$$

- For each θ_i, the baseline can be optimized
  - For example, the value-based baseline is used

$$b = \mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[G(\tau)]$$

  - Note that this does not minimize the variance of the gradient

REINFORCE algorithm: Monte Carlo Policy Gradient

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

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 $\pi(\cdot|\cdot, \boldsymbol{\theta})$

  $b \leftarrow T^{-1} \sum_{t=1}^{T} R_t$

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

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

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

[Williams 1992]

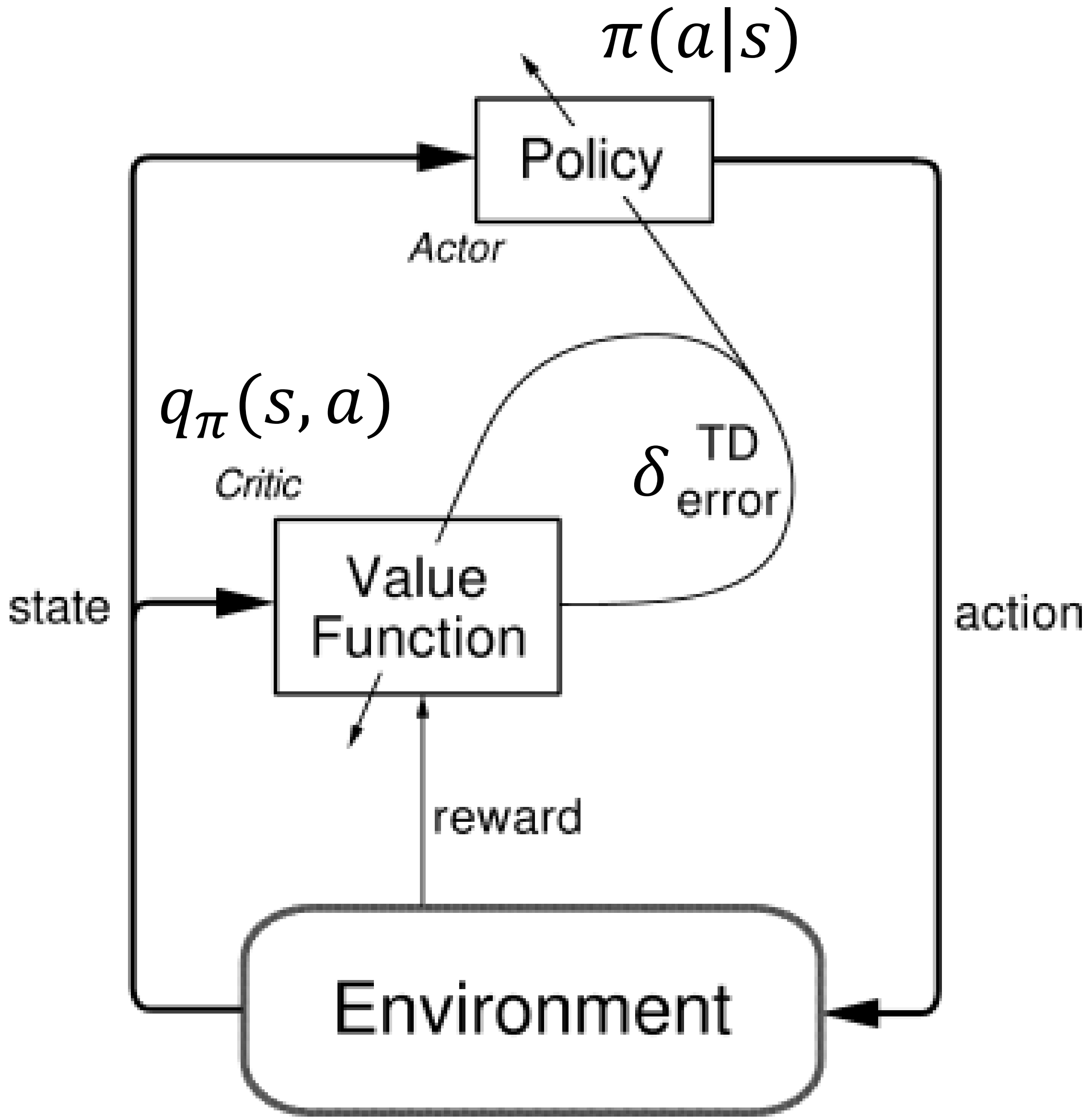
Pros and cons

	Value-based	Pure Policy-based
learning speed	slow	much slower
convergence	global optimal	local optimal
Markovian	necessary	not necessary
action	discrete	discrete/continuous

- Actor-critic approaches combines the value- and the policy-based approaches

Actor-Critic Methods

- Two learning components
  - actor: $\pi(a|s)$
  - critic: $v_\pi(s)$ or $q_\pi(s,a)$
- TD error is used to train $\pi(a|s)$ as well as $v_\pi(s)$
- Appropriate if the action is continuous



Policy-Gradient theorem

- Reminder: the update rule of the policy gradient method

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\Pr(\tau|\boldsymbol{\theta})}[(G(\tau) - b)\nabla_\theta \ln \Pr(\tau \mid \boldsymbol{\theta})]$$

- Since $G(\tau)$ is an actual return, we have to wait until an episode ends in order to know $G(\tau)$
- We will replace the actual return $G(\tau)$ with a state-action value function

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{d_\pi,\pi}[(q_\pi(s,a) - b(s))\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})]$$

$$= \sum_s d_\pi(s) \sum_a \pi(a \mid s, \boldsymbol{\theta}) \big(q_\pi(s,a) - b(s)\big)\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})$$

What is $d_\pi(s)$

- For the average-reward formulation,

$$d_\pi(s) \triangleq \lim_{t\to\infty} \Pr(S_t = s \mid A_{0:t} \sim \pi)$$

  - $d_\pi(s)$ is called the stationary distribution
  - Under some assumptions, it converges to the stationary distribution of states under π
  - The stationary distribution does not depend on the starting state (ergodicity)
- The following equation satisfies

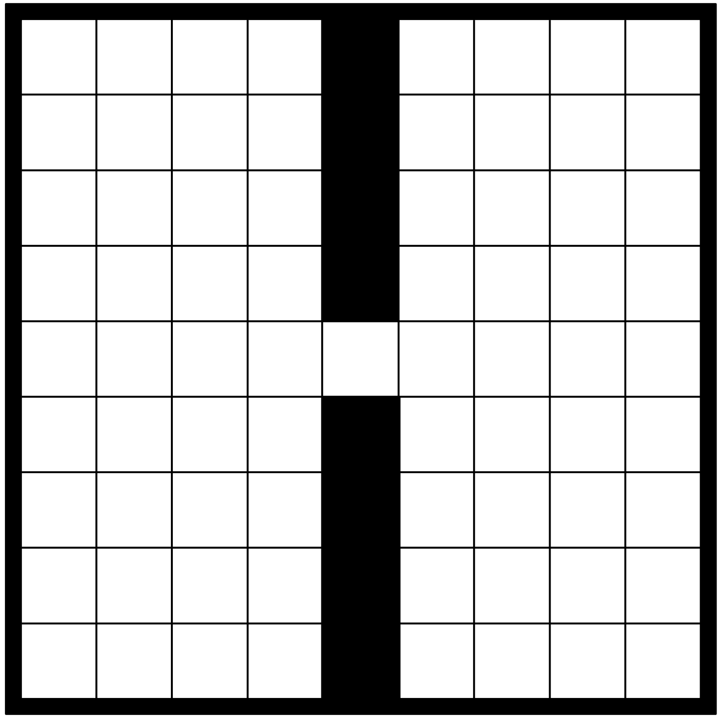
$$\sum_s d_\pi(s) \sum_a \pi(a \mid s)\, p(s' \mid s, a) = d_\pi(s')$$

Example: Stationary Distribution

- Consider the 9x9 grid world
- 5 actions (stop, left, right, up, and down)
- Random policy

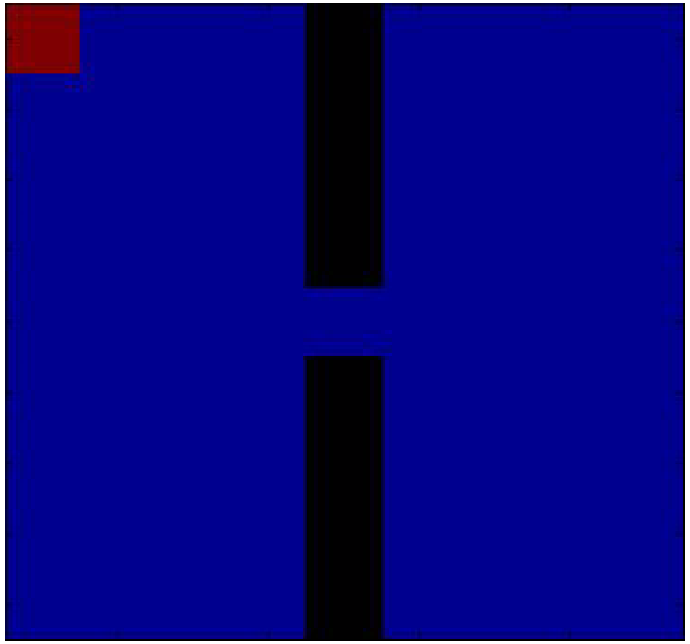
$$\pi(a \mid s) = 1/5$$

- Move to the intended direction with probability 0.9.



Example: Stationary Distribution

- Converge to the same stationary distribution



- For the infinite horizon average reward problem, the stationary distribution plays an important role to define the objective function

Average reward objective function

- So far, we considered the discounted sum of rewards to formulate the objective function
- Here, we consider the average rate of reward per time step because Policy Gradient Theorem for the discounted problems is complicated to derive
- The new objective function is given by

$$
\begin{aligned}
J(\boldsymbol{\theta}) &\triangleq \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[R_t \mid A_{0:t-1} \sim \pi] \\
&= \sum_{s} d_\pi(s) \sum_{a} \pi(a \mid s) \sum_{s',r} r p(s', r \mid s, a)
\end{aligned}
$$

Value functions for the new setting

- Value functions are re-defined as follows

$$v_\pi(s) \triangleq \mathbb{E}_\pi[G_t \mid S_t = s]$$

$$q_\pi(s, a) \triangleq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$

- G_t is also re-defined as the differential return

$$G_t \triangleq \underbrace{R_{t+1} - J(\boldsymbol{\theta})}_{\text{deviation from the average}} + R_{t+2} - J(\boldsymbol{\theta}) + R_{t+3} - J(\boldsymbol{\theta}) + \cdots$$

Derivation 1/5 (Modified Bellman equation)

- Bellman expectation equation for average reward formulation is

$$q_\pi(s,a) = \sum_{s',r} p(s',r \mid s,a)[r - J(\boldsymbol{\theta}) + v_\pi(s')]$$

where $v_\pi(s) = \sum_a \pi(a \mid s) q_\pi(s,a)$

- Reminder: Bellman expectation equation for discounted problems

$$q_\pi(s,a) = \sum_{s',r} p(s',r \mid s,a)[r + \gamma + v_\pi(s')]$$

Derivation 2/5 (Derivative w.r.t. policy parameter)

- $\nabla_{\boldsymbol{\theta}} v_\pi(s) = \nabla_{\boldsymbol{\theta}}\left(\sum_a \pi(a \mid s) q_\pi(s,a)\right)$

$$= \sum_a [\nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s,a) + \pi(a \mid s) \nabla_{\boldsymbol{\theta}} q_\pi(s,a)]$$

Bellman expectation equation

$$= \sum_a \left[\nabla_\theta \pi(a \mid s) q_\pi(s,a) + \pi(a \mid s) \nabla_{\boldsymbol{\theta}}\left(\sum_{s',r} p(s',r \mid s,a)(r - J(\boldsymbol{\theta}) + v_\pi(s'))\right)\right]$$

r does not depend on $\boldsymbol{\theta}$

$$= \sum_a \left\{\nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s,a) + \pi(a \mid s)\left[-\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) + \sum_{s'} p(s' \mid s,a) \nabla_\theta v_\pi(s')\right]\right\}$$

Derivation 3/5

- $$\nabla_{\boldsymbol{\theta}} v_\pi(s) = \sum_a \Big\{ \nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s,a) + \pi(a \mid s) \Big[-\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) + \sum_{s'} p(s' \mid s,a) \nabla_\theta v_\pi(s') \Big] \Big\}$$

- $$\sum_a \pi(a \mid s) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_a \Big\{ \nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s,a) + \pi(a \mid s) \Big[\sum_{s'} p(s' \mid s,a) \nabla_{\boldsymbol{\theta}} v_\pi(s') \Big] \Big\} - \nabla_{\boldsymbol{\theta}} v_\pi(s)$$

- Note that the left hand side can be simplified as follows

$$\sum_a \pi(a \mid s) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

Derivation 4/5

- $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_a \Big\{ \nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s, a)$
$$+\pi(a \mid s) \left[\sum_{s'} p(s' \mid s, a) \nabla_{\boldsymbol{\theta}} v_\pi(s') \right] \Big\} - \nabla_{\boldsymbol{\theta}} v_\pi(s)$$
- The left hand side of the obtained equation does not depend on state s
- Thus, the right hand side does not depend on s either, and we can safely sum it over all $s \in \mathcal{S}$ weighted by $d_\pi(s)$

$$\sum_s d_\pi(s) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_s d_\pi(s) \sum_a \Big\{ \nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s, a)$$
$$+\pi(a \mid s) \left[\sum_{s'} p(s' \mid s, a) \nabla_{\boldsymbol{\theta}} v_\pi(s') \right] \Big\} - \sum_s d_\pi(s) \nabla_{\boldsymbol{\theta}} v_\pi(s)$$

Derivation 5/5

- $\sum_s d_\pi(s) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_s d_\pi(s) \sum_a \Big\{ \nabla_{\boldsymbol{\theta}} \pi(a \mid s) q_\pi(s, a)$

$$+\pi(a \mid s)\left[\sum_{s'} p(s' \mid s, a) \nabla_{\boldsymbol{\theta}} v_\pi(s')\right]\Big\} - \sum_s d_\pi(s) \nabla_{\boldsymbol{\theta}} v_\pi(s)$$

- Left hand side

$$\sum_s d_\pi(s) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- Second term of the right hand side

$$\sum_s d_\pi(s) \sum_a \pi(a \mid s)\left[\sum_{s'} p(s' \mid s, a) \nabla_{\boldsymbol{\theta}} v_\pi(s')\right] = \sum_{s'} d_\pi(s') \nabla_{\boldsymbol{\theta}} v_\pi(s')$$

  - This is identical to the third term

Policy gradient theorem

- Finally the policy gradient is given by

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \sum\nolimits_s d_\pi(s) \sum\nolimits_a q_\pi(s,a) \nabla_{\boldsymbol{\theta}} \pi(\, a \mid s, \boldsymbol{\theta}\,) \\ &= \sum\nolimits_s d_\pi(s) \sum\nolimits_a \pi(\, a \mid s, \boldsymbol{\theta}\,)\, q_\pi(s,a) \nabla_{\boldsymbol{\theta}} \ln \pi(\, a \mid s, \boldsymbol{\theta}\,) \\ &= \mathbb{E}_{d_\pi,\pi}[q_\pi(s,a) \nabla_{\boldsymbol{\theta}} \ln \pi(\, a \mid s, \boldsymbol{\theta}\,)]\end{aligned}$$

- This is called the policy gradient theorem (Sutton et al., 2000; Konda and Tsitsiklis, 2000)
- Reminder: Gradient of the bandit problems

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_\pi[(r-b) \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid \boldsymbol{\theta})]$$

Action-value actor-critic

- Prepare a critic $\hat{q}(s, a, \boldsymbol{w})$ and an actor $\pi(a \mid s, \boldsymbol{\theta})$
- Sample $A \sim \pi(a \mid S, \boldsymbol{\theta})$
- For each step do
  - Sample reward r and transition $S', R \sim p(s, r \mid S, A)$
  - Sample action $A' \sim \pi(\cdot \mid S', \boldsymbol{\theta})$
  - Compute the TD error $\delta = r + \gamma \hat{q}(S', A', \boldsymbol{w}) - \hat{q}(S, A, \boldsymbol{w})$
  - Update the policy parameter
  $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})\, \hat{q}(s, a, \boldsymbol{w})$$
  - Update the value parameter
  $$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha_{\boldsymbol{w}} \delta \nabla_{\boldsymbol{w}} \hat{q}(s, a, \boldsymbol{w})$$
  - $s \leftarrow s', a \leftarrow a'$
- end for

Approximating $\boldsymbol{q_\pi}$

- Let $\hat{q}(s, a, \boldsymbol{w})$ be an approximation of $q_\pi(s, a)$, parameterized by $\boldsymbol{w}$
- How should we approximate?
- The policy gradient is exact when the following two conditions are satisfied

1. Value function approximator is compatible to the policy

$$\nabla_{\boldsymbol{w}} \hat{q}(s, a, \boldsymbol{w}) = \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})$$

2. The parameter $\boldsymbol{w}$ minimizes the mean-squared error ε

$$\varepsilon = \mathbb{E}_{d_\pi, \pi}\left[\left(q_\pi(s, a) - \hat{q}(s, a, \boldsymbol{w})\right)^2\right]$$

Approximating $\boldsymbol{q_\pi}$

- Since $\boldsymbol{w}$ minimizes ε, $\nabla_w \varepsilon = \mathbf{0}$
- $$\nabla_{\boldsymbol{w}} \varepsilon = -\mathbb{E}_{d_\pi,\pi}\left[\big(q_\pi(s,a) - \hat{q}(s,a,\boldsymbol{w})\big)\nabla_{\boldsymbol{w}}\hat{q}(s,a,\boldsymbol{w})\right]$$
- Using the first assumption, we obtain

$$\mathbb{E}_{d_\pi,\pi}\left[\big(q_\pi(s,a) - \hat{q}(s,a,\boldsymbol{w})\big)\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})\right] = \mathbf{0}$$

- $$\begin{aligned}&\mathbb{E}_{d_\pi,\pi}[\hat{q}(s,a,\boldsymbol{w})\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})] \\ &\quad = \mathbb{E}_{d_\pi,\pi}[q_\pi(s,a)\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})] = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\end{aligned}$$

Theorem

- Policy gradient $\nabla_{\boldsymbol{\theta}} J$ is calculated by

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{d_\pi,\pi}[\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})\, \hat{q}(s, a, \boldsymbol{w})]$$

where $$\nabla_{\boldsymbol{w}} \hat{q}(s, a, \boldsymbol{w}) = \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})$$

- The equation of the first assumption implies that $\hat{q}$ is represented by

$$\hat{q}(s, a, \boldsymbol{w}) = \boldsymbol{w}^\top \underbrace{\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})}_{\text{basis function vector}}$$

- Then, we can write

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{d_\pi,\pi}[\nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})\, \nabla_{\theta} \ln \pi(a \mid s, \boldsymbol{\theta})^\top \boldsymbol{w}]$$

Note on $\hat{q}(s, a, w)$

- Since the projected value function is given by

$$\hat{q}(s, a, \boldsymbol{w}) = \boldsymbol{w}^{\top} \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s, \boldsymbol{\theta})$$

- It satisfies

$$\sum_{a} \pi(a \mid s; \boldsymbol{\theta}) \hat{q}(s, a, \boldsymbol{w}) = \sum_{a} \boldsymbol{w}^{\top} \nabla_{\boldsymbol{\theta}} \pi(a \mid s, \boldsymbol{\theta})$$

$$= \boldsymbol{w}^{\top} \sum_{a} \nabla_{\boldsymbol{\theta}} \pi(a \mid s, \boldsymbol{\theta}) = \boldsymbol{w}^{\top} \nabla_{\boldsymbol{\theta}} \left[\sum_{a} \pi(a \mid s, \boldsymbol{\theta}) \right]$$

$$= \boldsymbol{w}^{\top} \nabla_{\boldsymbol{\theta}}(1) = 0$$

- It suggests that $\mathbb{E}_{\pi}[\hat{q}(s, a, \boldsymbol{w})] = 0$

Note on $\hat{q}(s, a, w)$

- Interestingly, it is enough to use a linear function approximator to update the actor

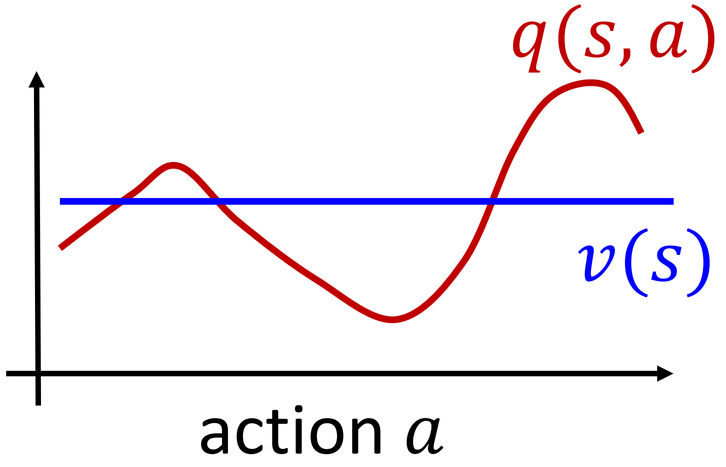
$$q(s, a; \boldsymbol{w}) = \boldsymbol{w}^\top \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s; \boldsymbol{\theta})$$

- Choice of $b(s)$ is important. When $b(s) = V(s)$,

$$q(s, a) - v(s) = A(s, a)$$

  is usually called the advantage function

- The TD error can be used as a target value to train the advantage function [Morimura et al., 2005]



Summary

- REINFORCE algorithm is the most classical policy gradient, and it is applicable even though the environment is not Markovian
- The Actor-Critic has many equivalent forms:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}[\nabla_\theta \ln \pi(\, a \mid s, \theta\,)\, Q^w(s,a)] \qquad \text{Q Actor-Critic}$$

$$= \mathbb{E}[\nabla_\theta \ln \pi(\, a \mid s, \theta\,)\, A^w(s,a)] \qquad \text{Advantage Actor-Critic}$$

$$= \mathbb{E}[\nabla_\theta \ln \pi(\, a \mid s, \theta\,)\, \delta] \qquad \text{TD Actor-Critic}$$

- Each leads a stochastic gradient ascent algorithm
- Critic uses policy evaluation to estimate q_π, A_π, or v_π

References

- Choromanski, K., Iscen, A., Sindhwani, V., Tan, J. and Coumans, E. Optimizing Simulations with Noise-Tolerant Structured Exploration. In Proc. of ICRA 2018.
- Deisenroth, M., Neumann, G., and Peters, J. (2013). A survey on policy search for robotics. Foundations and Trends in Robotics, 2(1-2): 1-142.
- Kober, J. and Peters, J. Policy Search for Motor Primitives in Robotics. NIPS 2009.
- Kohl, N. and Stone, P. (2004). Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion. In Proc. of IEEE International Conference on Robotics and Automation, 2619-2624.

References

- Konda, V. and Tsitsiklis, J. (2000). Actor-Critic Algorithms. NIPS 12.
- Lagoudakis, M. and Parr, R. (2003). Least-squares policy iteration. Machine Learning, 4, 1107-1149.
- Peters, J. and Schaal, S. (2008). Natural Actor-Critic. Neurocomputing, 71(7-9), 1180-1190.
- Peters, J. and Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. Neural Networks, 21(4), 682-697.
- Sehnke, F. et al. Parameter-exploring policy gradients. Neural Networks, 23. 551-559, 2010.
- Sutton, R., McAllester, D. Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. NIPS 12.

References

- Theodorou, E. A. et al. A generalized path integral control approach to reinforcement learning. Journal of Machine Learning Research. 11, 3153-3197, 2010.
- Vlassis, N. et al. Learning model-free robot control by a Monte Carlo EM. Autonomous Robotics. 27, 123-130, 2009.
- Williams, R. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning. 8(3): 229-256, 1992.

What is $d_\pi(s)$

- For the discounted-reward formulation, $d^\pi(s)$ is defined by

$$d_\pi(s) = \sum_{t=0}^{\infty} \gamma^t \Pr(S_t = s \mid S_0, \pi)$$

  - Dependent on the starting state
- For the average-reward formulation,

$$d_\pi(s) = \lim_{t \to \infty} \Pr(S_t = s \mid S_0, \pi)$$

  - Under some assumptions, it converges to the stationary distribution of states under π
  - The stationary distribution does not depend on the starting state S_0

Function approximation

- Consider a linear function approximator

$$G(\tau) \approx \boldsymbol{w}^\top \nabla_{\boldsymbol{\theta}} \ln p(\, \tau \mid \boldsymbol{\theta} \,) + \boldsymbol{v}^\top \boldsymbol{\varphi}(S_0)$$

where $\boldsymbol{w} \in \mathbb{R}^n$ is the parameter to be tuned.

- $\boldsymbol{\varphi}(S_0) \in \mathbb{R}^{n'}$ and $\boldsymbol{v} \in \mathbb{R}^{n'}$ denote the basis function and the weight vector, respectively.
- The parameter vectors $\boldsymbol{w}$ and $\boldsymbol{v}$ can be estimated by the least squares method from samples
- $\boldsymbol{v}^\top \boldsymbol{\varphi}(S_0)$ can be interpreted as an approximator of the baseline b

How to estimate $\boldsymbol{w}$?

- The dataset $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$ is given, where

$$\tau_i = \left\{S_0^i, A_0^i, R_1^i, \dots, S_{T-1}^i, A_{T-1}^i, R_{T-1}^i, S_T^i\right\}$$

- The error of the i-th data is computed by

$$e_i = \left[\nabla_{\boldsymbol{\theta}} \ln p(\tau_i \mid \boldsymbol{\theta})^\top \quad \boldsymbol{\varphi}^\top(S_0^i)\right] \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{v} \end{bmatrix} - G_i$$

  where $G_i = \sum_{t=0}^{T-1} R_t^i$

- Since this is the ordinary least squares problem, the objective function is constructed by

$$E = \sum_{i=1}^{N} e_i^2$$

Algorithm of Episodic NAC

Algorithm 8 Episodic Natural Actor Critic

Input: Policy parametrization $\boldsymbol{\theta}$,

data set $\mathcal{D} = \left\{ \boldsymbol{x}_{1:T}^{[i]}, \boldsymbol{u}_{1:T-1}^{[i]}, r_{1:T}^{[i]} \right\}_{i=1\ldots N}$

for each sample $i = 1 \ldots N$ **do**

Compute returns: $R^{[i]} = \sum_{t=0}^{T} r_t^{[i]}$

Compute features: $\boldsymbol{\psi}^{[i]} = \begin{bmatrix} \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}} \left(\boldsymbol{u}_t^{[i]} \middle| \boldsymbol{x}_t^{[i]}, t \right) \\ \boldsymbol{\varphi}(\boldsymbol{x}_0^{[i]}) \end{bmatrix}$

end for

Fit advantage function and initial value function

$$\boldsymbol{R} = \left[R^{[1]}, \ldots, R^{[N]} \right]^T, \quad \boldsymbol{\Psi} = \left[\boldsymbol{\psi}^{[1]}, \ldots, \boldsymbol{\psi}^{[N]} \right]^T$$

$$\begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{v} \end{bmatrix} = (\boldsymbol{\Psi}^T \boldsymbol{\Psi})^{-1} \boldsymbol{\Psi}^T \boldsymbol{R}$$

return $\nabla_{\boldsymbol{\theta}}^{\mathrm{eNAC}} J_{\boldsymbol{\theta}} = \boldsymbol{w}$

note

$\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}_t^i \mid \boldsymbol{x}_t^i)$

$= \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\tau}_i)$

[Deisenroth et al. 2013]

Reminder: Policy Gradient Theorem

- The vanilla gradient for the infinite horizon can be derived from the Policy Gradient Theorem:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{d_\pi,\pi}\left[\nabla_{\boldsymbol{\theta}} \ln \pi(\, a \mid s, \boldsymbol{\theta}\,)\left(\hat{q}(s, a, \boldsymbol{w}) - \hat{b}(s)\right)\right]$$

- This theorem is independently proved in 2000 by Sutton et al. and Konda and Tsitsiklis
- Then, a linear function estimator is constructed to approximate $\hat{q}(s, a, \boldsymbol{w}) - \hat{b}(s)$

[Sutton et al. 2000; Konda and Tsitsiklis 2000]

Advantage function

- If $\hat{b}(s) = \hat{v}(s, \boldsymbol{v})$, the following function is called the advantage function

$$A^{\pi}(s, a) = \hat{q}(s, a, \boldsymbol{w}) - \hat{v}(s, \boldsymbol{v})$$

where

$$\hat{v}(s, \boldsymbol{v}) \approx \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^{k} R_{t+k+1} \mid S_t = s\right]$$

$$\hat{q}(s, a, \boldsymbol{w}) \approx \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^{k} R_{t+k+1} \mid S_t = s, A_t = a\right]$$

- Then, consider the following approximators

$$\hat{A}(s, a, \boldsymbol{w}) \triangleq \boldsymbol{w}^{\top} \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s)$$

$$\hat{v}(s, \boldsymbol{v}) \triangleq \boldsymbol{v}^{\top} \boldsymbol{\varphi}(s)$$

- Important features

$$\mathbb{E}_\pi\{A^\pi(s,a)\} = \sum_a \pi(\, a \mid s\,) A^\pi(s,a) = 0$$

$$\mathbb{E}_\pi\{\boldsymbol{w}^\top \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s)\} = 0$$

- Using the approximators in the previous slide, $\hat{q}(s, a, \boldsymbol{w}, \boldsymbol{v})$ can be approximated by

$$\hat{q}(s, a, \boldsymbol{w}, \boldsymbol{v}) = \boldsymbol{w}^\top \nabla_{\boldsymbol{\theta}} \ln \pi(a \mid s) + \boldsymbol{v}^\top \varphi(s)$$

- We can apply any policy evaluation algorithm