Policy Gradient Methods

wonseok Jung

1. Two methods of choosing action

- action-value :
  - Learning the action value
  - Estimate action value을 바탕으로 action을 선택한다.
  - Policies would not even exist without the action-value estimates
- Parameterized policy :
  - select actions without consulting value function
  - Value function still be used to learn policy parameter
  - Value function이 action을 선택하는 기준으로 사용되지 않는다

1.1 Policy parameter vector

- $\theta \in R^{d'}$:Policy's parameter vector
- $\pi(a \mid s, \theta) = Pr\{At = a \mid S_t = S, \theta_t = \theta\}$
  - Probability action a is taken at time t, given that the environment is in state s at time t with parameter θ

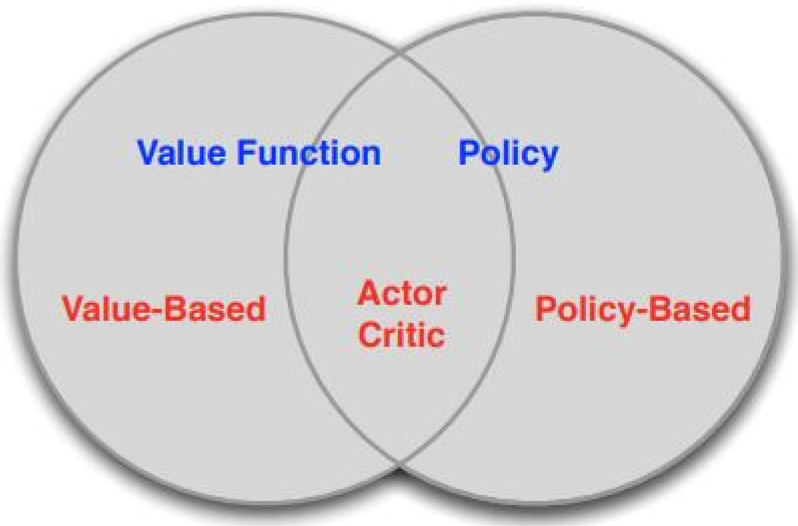
1.2 Learning policy parameter

$$\theta_{t+1} = \theta_t + \alpha \widehat{\nabla J(\theta)}$$

- $\nabla J(\theta_t)$: stochastic estimate
  - expectation approximates the gradient of the measure to its argument θ

- Policy gradient methods : optimizing parametrized policies with respect to the expected return by gradient ascent
- Actor-critic methods : Methods that learng approximations to both policy and value function

- Value Based
  - Learnt Value Function
  - Implicit policy (e.g. ϵ-greedy)
- Policy Based
  - No Value Function
  - Learnt Policy
- Actor-Critic
  - Learnt Value Function
  - Learnt Policy



2. Policy Approximation and ITS Advantages

3. The Policy gradient Theorem

- $\epsilon - greedy$ 는 작은 action value의 변화에도 action 선택이 완전히 바뀔수 있다.
- 하지만 policy parameterization 방법은 Parameter를 배우며 Policy parameterd의 action probabilities가 smoothly하게 변한다.
- 이렇게 parameter에 의해 policy가 달라진다면, policy-gradient 방법을 이용하여 gradient asecent를 approximate하는것이 가능하다.

3.1 Two type of cases

- $J(\theta)$: Performance measure
- Episodic case, Continuous case 두 가지로 나눌수 있다.
- Episodic case : the performace measure as the value o the start state of the episode,

$$J(\theta) \doteq v_{\pi\theta}(s_0)$$

- $v_{\pi\theta}(s_0)$: True value for π_θ , the policy determinded by θ

3.2 Challenging

- With function approximation, it may seem challenging to change the policy parameter in a way that esnures improvements.
  - performance depends on both the action selection and the distribution of states in which those selections are made
  - Both of these are affected by the policy parameter
- Policy gradient theorem : analytic expression for the gradient of performace with respect to the policy parameter

$$\bigtriangledown J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \bigtriangledown \pi(a \mid s, \theta)$$

3.3 Policy Gradient

$$\bigtriangledown J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \bigtriangledown \pi(a \mid s, \theta)$$

π : policy corresponding to parameter vector θ

$\propto$: propotional to

μ : on-policy distribuion under

- From chapter 10 :
  - $\mu_\pi(s) :\doteq lim_{t \to \infty} Pr\left\{S_t = s \mid A_{0:t-1} \sim \pi\right\}$
  - steady-state distribution

4. REINFORCE: Monte Carlo Policy Gradient

$$\bigtriangledown J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \bigtriangledown \pi(a \mid s, \theta)$$

$$= E_\pi[\sum_a q_\pi(S_t,a) \bigtriangledown \pi(a \mid S_t, \theta)]$$

- Policy gradiet theorem : sum over a states weight by how often the states occur under the target poicy π

4.1 Replacing a with the sample action A_t

-

$$
\begin{aligned}
\nabla J(\boldsymbol{\theta}) &= \mathbb{E}_\pi\left[\sum_a \pi(a|S_t,\boldsymbol{\theta}) q_\pi(S_t,a) \frac{\nabla \pi(a|S_t,\boldsymbol{\theta})}{\pi(a|S_t,\boldsymbol{\theta})}\right] \\
&= \mathbb{E}_\pi\left[q_\pi(S_t,A_t) \frac{\nabla \pi(A_t|S_t,\boldsymbol{\theta})}{\pi(A_t|S_t,\boldsymbol{\theta})}\right] && \text{(replacing } a \text{ by the sample } A_t \sim \pi) \\
&= \mathbb{E}_\pi\left[G_t \frac{\nabla \pi(A_t|S_t,\boldsymbol{\theta})}{\pi(A_t|S_t,\boldsymbol{\theta})}\right], && \text{(because } \mathbb{E}_\pi[G_t|S_t,A_t] = q_\pi(S_t,A_t))
\end{aligned}
$$

- G_t : Return

4.2 REINFORCE algorithm

- Updte Rule :

$$\theta_{t+1} \doteq \theta_t + \alpha G_t \frac{\nabla \pi(A_t \mid S_t, \theta_t)}{\pi(A_t \mid S_t, \theta_t)}$$

- REINFORCE uses the comple te return from time t.
- All future rewards update until the end of the epsiode.
- REINFORCE는 MonteCalo 알고리즘을 사용한다. 모든 업데이트는 episode가 끝난뒤 이루어진다.

4.3 REINFORCE-Monte Carlo pseudocode

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

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ $\qquad (G_t)$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

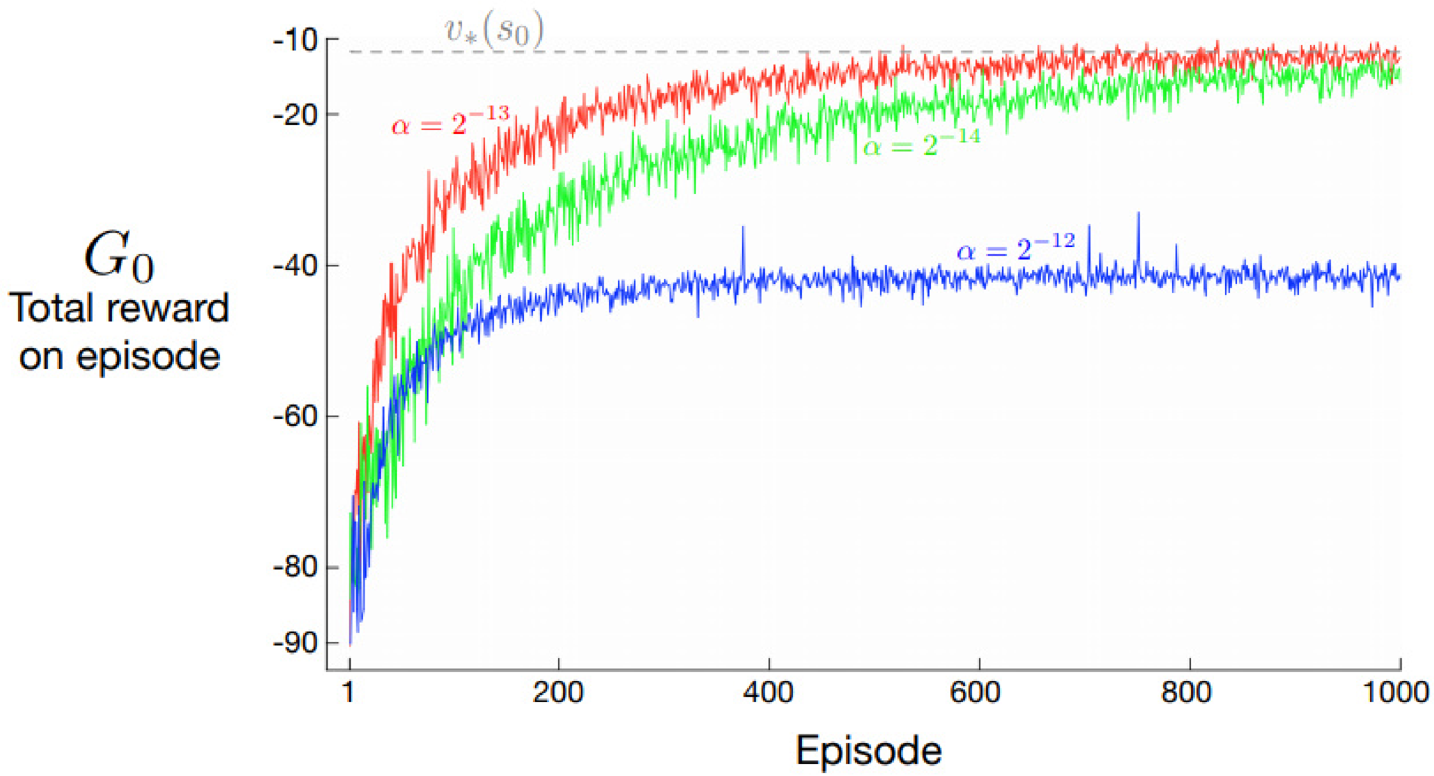


Figure 13.1: REINFORCE on the short-corridor gridworld (Example 13.1). With a good step size, the total reward per episode approaches the optimal value of the start state.

- α : step size에 따른 Total reward의 차이 비교

5. REINFORCE with Baseline

$$\triangledown J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \triangledown \pi(a \mid s, \theta)$$

- 위의 policy gradient theorem 은 다음의 식과 같이 어떠한 baseline $b(s)$에 의해 action value의 비교 식으로 바꿀수 있다.

$$\triangledown J(\theta) \propto \sum_s \mu(s) \sum_a (q_\pi(s,a) - b(s)) \triangledown \pi(a \mid s, \theta)$$

5.1 REINFORCE update rule with baseline

- $\theta + t + 1 \doteq \theta_t + \alpha(G_t - b(S_t))\frac{\nabla\pi(A_| S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)}$
- Baseline을 사용하므로 variance를 줄일수 있다.
- 그러므로 learning 속도가 빨라진다.
- For MDP, baseline should vary with state
  - some states all actions have high values
  - soma states all actions have low values

5.3 Baseline

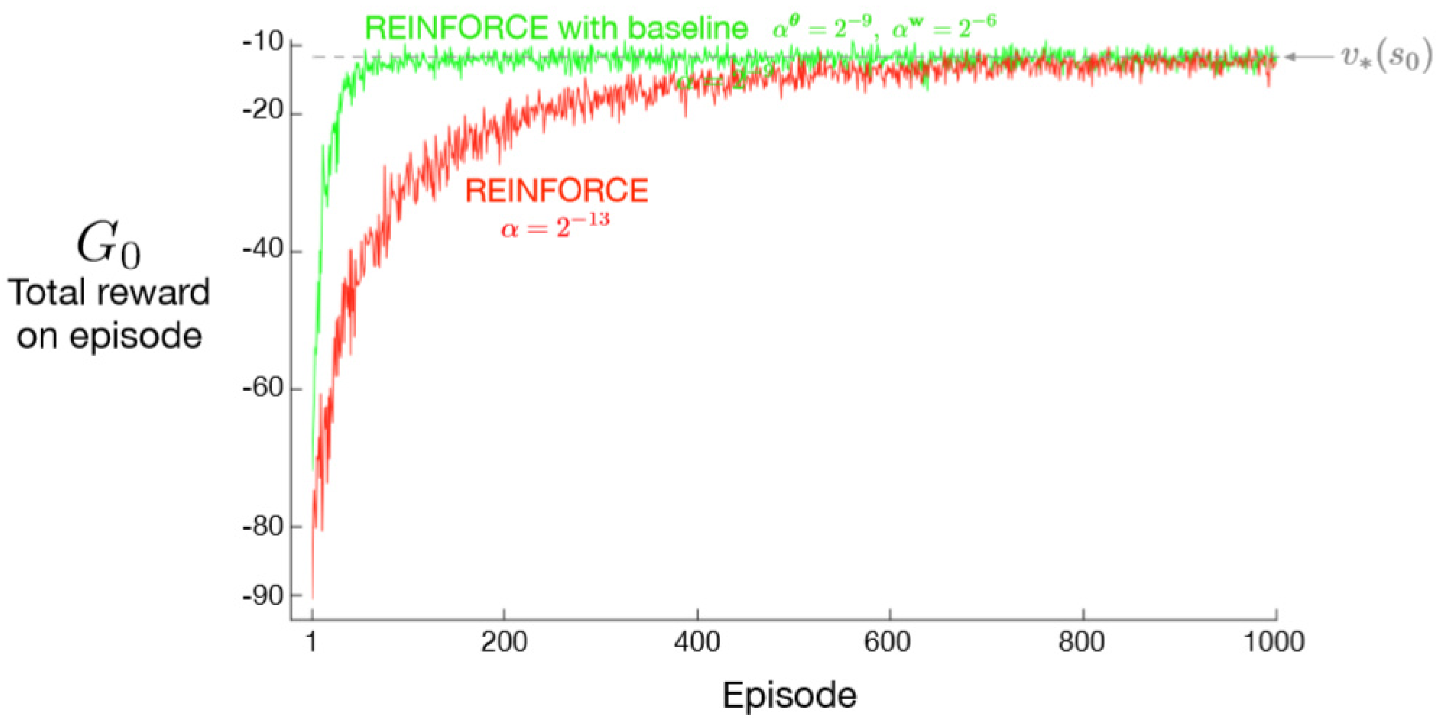


Figure 13.2: Adding a baseline to REINFORCE can make it learn much faster, as illustrated here on the short-corridor gridworld (Example 13.1). The step size used here for plain REINFORCE is that at which it performs best (to the nearest power of two; see Figure 13.1). Each line is an average over 100 independent runs.

5.2 Reinforcement with baseline pseudocode

REINFORCE with Baseline (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Algorithm parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
        $\delta \leftarrow G - \hat{v}(S_t,\mathbf{w})$
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \gamma^t \delta \nabla \hat{v}(S_t,\mathbf{w})$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$