Policy Gradient

Alina Vereshchaka

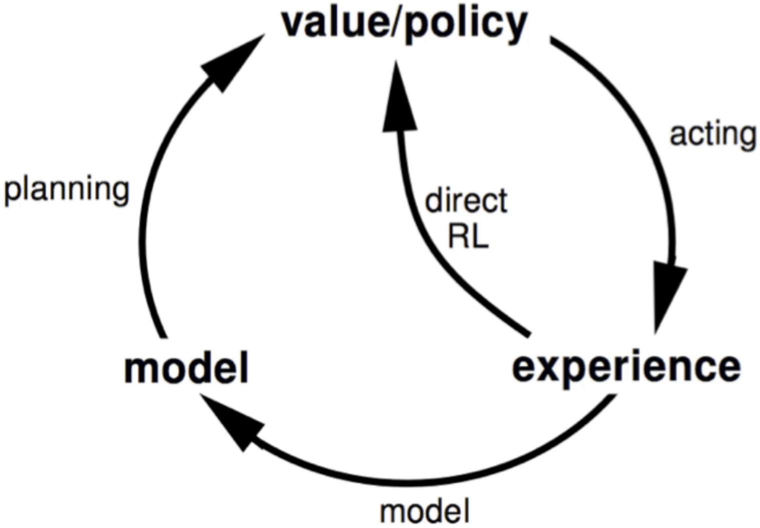
CSE4/510 Reinforcement Learning
Fall 2019

avereshc@buffalo.edu

October 17, 2019

*Slides are adopted from Policy Gradient Algorithms by Lilian Weng & David Silver's Course

Recap


value/policy
acting
planning
direct
RL
model
experience
model

Recap

- **Model-based:**

Recap

- **Model-based:** Rely on the model of the environment; either the model is known or the algorithm learns it explicitly
- **Model-free:**

Recap

- **Model-based:** Rely on the model of the environment; either the model is known or the algorithm learns it explicitly
- **Model-free:** No dependency on the model during learning
- **On-policy:**

Recap

- **Model-based:** Rely on the model of the environment; either the model is known or the algorithm learns it explicitly
- **Model-free:** No dependency on the model during learning
- **On-policy:** Use the deterministic outcomes or samples from the target policy to train the algorithm
- **Off-policy:**

Recap

- **Model-based:** Rely on the model of the environment; either the model is known or the algorithm learns it explicitly
- **Model-free:** No dependency on the model during learning
- **On-policy:** Use the deterministic outcomes or samples from the target policy to train the algorithm
- **Off-policy:** Training on a distribution of transitions or episodes produced by a different behavior policy rather than that produced by the target policy

Recap: DQN

left or **right**?

Recap: DQN

Right! Ready for next one?

DQN is trained to minimize

$$L \approx E[Q(s_t, a_t) - (r_t + \gamma \cdot max_{a'} Q(s_{t+1}, a'))]^2$$

Simple 2-state world

	True	(A)	(B)
Q(s0,a0)	*1*	*1*	*2*
Q(s0,a1)	*2*	*2*	*1*
Q(s1,a0)	*3*	*3*	*3*
Q(s1,a1)	*100*	*50*	*100*

Q: Which prediction is better (A/B)?

DQN is trained to minimize

$$L \approx E[Q(s_t, a_t) - (r_t + \gamma \cdot max_{a'} Q(s_{t+1}, a'))]^2$$

Simple 2-state world

	True	(A)	(B)
Q(s0,a0)	*1*	*1*	*2*
Q(s0,a1)	*2*	*2*	*1*
Q(s1,a0)	*3*	*3*	*3*
Q(s1,a1)	*100*	*50*	*100*
		better policy	**less MSE**

Recap: DQN

DQN is trained to minimize

$$L \approx E[Q(s_t, a_t) - (r_t + \gamma \cdot max_{a'} Q(s_{t+1}, a'))]^2$$

Simple 2-state world

	True	(A)	(B)
Q(s0,a0)	*1*	*1*	*2*
Q(s0,a1)	*2*	*2*	*1*
Q(s1,a0)	*3*	*3*	*3*
Q(s1,a1)	*100*	*50*	*100*

Q-learning will prefer worse policy (B)!

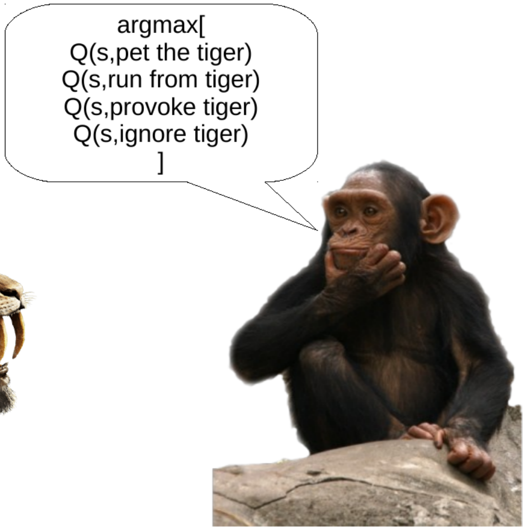
better policy

less MSE

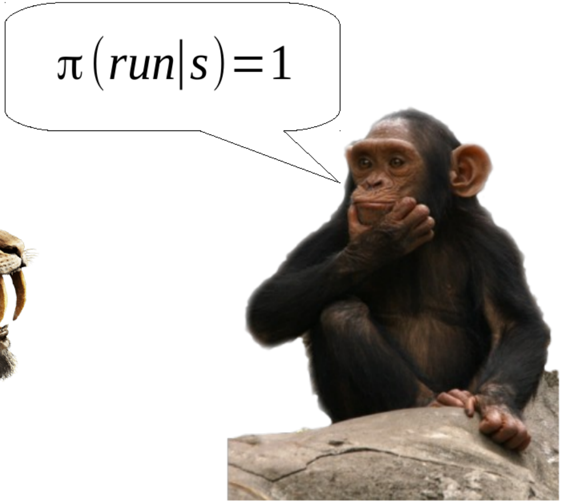
Recap: Summary

- Often computing q-values is harder than picking optimal actions
- We could avoid learning value functions by directly learning agent's policy $\pi_\theta(a|s)$

Recap: DQN


argmax[
Q(s,pet the tiger)
Q(s,run from tiger)
Q(s,provoke tiger)
Q(s,ignore tiger)
]

Recap: DQN



Recap: Value Based Reinforcement Learning

- In Value based approximations, we approximate the value or action-value function using parameters θ

$$V_\theta(s) \approx V^\pi(s)$$
$$Q_\theta(s, a) \approx Q^\pi(s, a)$$

Recap: Value Based Reinforcement Learning

- In Value based approximations, we approximate the value or action-value function using parameters θ

$$V_\theta(s) \approx V^\pi(s)$$
$$Q_\theta(s, a) \approx Q^\pi(s, a)$$

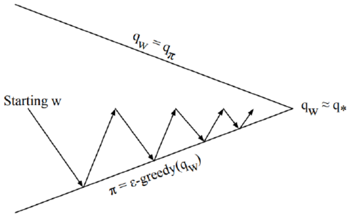
- How do we get the policy?

Recap: Value Based Reinforcement Learning

- In Value based approximations, we approximate the value or action-value function using parameters θ

$$V_\theta(s) \approx V^\pi(s)$$
$$Q_\theta(s,a) \approx Q^\pi(s,a)$$

- How do we get the policy? $\rightarrow$ A policy was generated directly from the value function (e.g. using ϵ-greedy)



Policy Gradient

- Can we directly parametrise the policy?

$$\pi_\theta(s, a) = P[a|s, \theta]$$

Policy Gradient

- Can we directly parametrise the policy?

$$\pi_\theta(s, a) = P[a|s, \theta]$$

- The policy gradient methods target at modeling and optimizing the policy directly. The policy is usually modeled with a parameterized function respect to $\theta, \pi_\theta(a|s)$.

Policy Gradient

- Can we directly parametrise the policy?

$$\pi_\theta(s, a) = P[a|s, \theta]$$

- The policy gradient methods target at modeling and optimizing the policy directly. The policy is usually modeled with a parameterized function respect to $\theta, \pi_\theta(a|s)$.
- **Sidenote:** $\pi_\theta(s, a) = \pi(A_t|S_t, \theta)$

Advantages of Policy-Based RL

Advantages

- Better convergence
- Effective in high-dimensional or continuous spaces
- Can learn stochastic policies
- Can be applied to POMDP

Advantages of Policy-Based RL

Advantages

- Better convergence
- Effective in high-dimensional or continuous spaces
- Can learn stochastic policies
- Can be applied to POMDP

Disadvantages

- Typically converge to a local rather than global optimum
- Evaluating a policy is typically inefficient and high variance

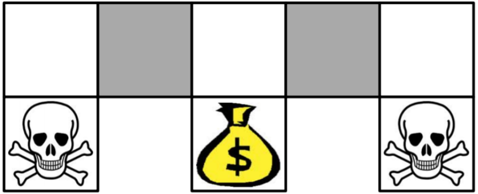
Example: Rock-Paper-Scissors

- Two-player game of rock-paper-scissors
  - Scissors beats paper
  - Rock beats scissors
  - Paper beats rock
- Consider policies for *iterated* rock-paper-scissors

Example: Rock-Paper-Scissors

- Two-player game of rock-paper-scissors
  - Scissors beats paper
  - Rock beats scissors
  - Paper beats rock
- Consider policies for *iterated* rock-paper-scissors
  - A deterministic policy is easily exploited
  - A uniform random policy is optimal (i.e. Nash equilibrium)

Example: Aliased Gridworld



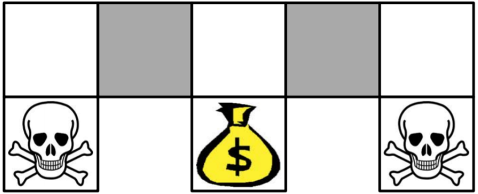
- The agent cannot differentiate the grey states
- Consider features of the following form (for all N, E, S, W)

$$\phi(s, a) = 1(\text{wall to N, a = move E})$$

- Compare value-based RL, using an approximate value function

$$Q_\theta(s, a) = f(\phi(s, a), \theta)$$

Example: Aliased Gridworld



- The agent cannot differentiate the grey states
- Consider features of the following form (for all N, E, S, W)

$$\phi(s, a) = 1(\text{wall to N, a = move E})$$

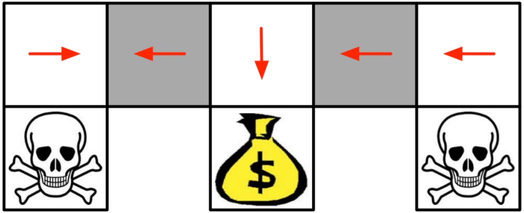
- Compare value-based RL, using an approximate value function

$$Q_\theta(s, a) = f(\phi(s, a), \theta)$$

- To policy-based RL, using a parametrised policy

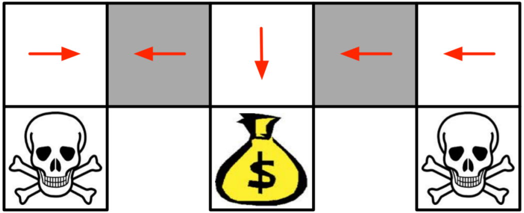
$$\pi_\theta(s, a) = g(\phi(s, a), \theta)$$

Example: Aliased Gridworld



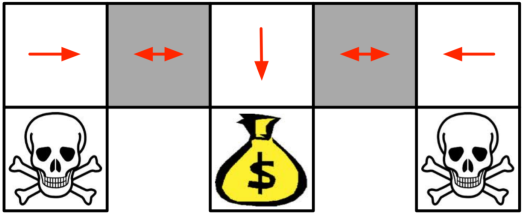
- Under aliasing, an optimal deterministic policy will either
  - move W in both grey states (shown by red arrows)
  - move E in both grey states

Example: Aliased Gridworld



- Under aliasing, an optimal deterministic policy will either
  - move W in both grey states (shown by red arrows)
  - move E in both grey states
- Either way, it can get stuck and never reach the money
- Value-based RL learns a near-deterministic policy (e.g. greedy or ϵ-greedy)
- So it will traverse the corridor for a long time

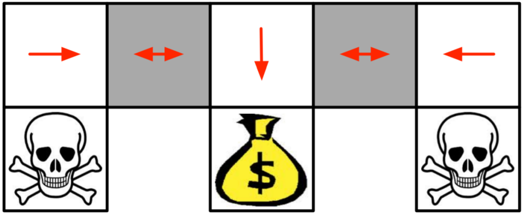
Example: Aliased Gridworld



- An optimal stochastic policy will randomly move E or W in grey states

$$\pi_\theta(\text{wall to N and S, move E}) = 0.5$$
$$\pi_\theta(\text{wall to N and S, move W}) = 0.5$$

Example: Aliased Gridworld



- An optimal stochastic policy will randomly move E or W in grey states

$$\pi_\theta(\text{wall to N and S, move E}) = 0.5$$
$$\pi_\theta(\text{wall to N and S, move W}) = 0.5$$

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

Policy Gradient Methods

We consider methods for learning the policy parameter based on the gradient of some scalar performance measure $J(\theta)$ with respect to the policy parameter. We seek to *maximize performance*, so their updates approximate gradient ascent in J:

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

Policy Approximation: Soft-max in action preferences

The policy can be parameterized in any way, as long as $\pi_\theta(a|s,\theta)$ is differential with respect to its parameters.

If the action space is discrete and not too large, then form parameterized numerical preferences $h(s,a,\theta) \in R$ for each state–action pair. The actions with the highest preferences in each state are given the highest probabilities of being selected:

$$\pi(a|s,\theta) = \frac{e^{h(s,a,\theta)}}{\sum_b e^{h(s,b,\theta)}}$$

Policy Objective Functions

Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ

Policy Objective Functions

Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ

But how do we measure the quality of a policy π_θ?

Policy Objective Functions

Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ

But how do we measure the quality of a policy π_θ?

- In episodic environments we can use the start value

$$J_1(\theta) = V_{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

Policy Objective Functions

Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ

But how do we measure the quality of a policy π_θ?

- In episodic environments we can use the start value

$$J_1(\theta) = V_{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

- In continuing environments we can use the average value

$$J_{avV}(\theta) = \sum_s d_{\pi_\theta}(s) V_{\pi_\theta}(s)$$

Policy Objective Functions

Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ

But how do we measure the quality of a policy π_θ?

- In episodic environments we can use the start value

$$J_1(\theta) = V_{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

- In continuing environments we can use the average value

$$J_{avV}(\theta) = \sum_s d_{\pi_\theta}(s) V_{\pi_\theta}(s)$$

- Or the average reward per time-step

$$J_{avR}(\theta) = \sum_s d_{\pi_\theta}(s) \sum_a \pi_\theta(s, a) R_s^a$$

where $d^{\pi_\theta}(s)$ is a stationary distribution of Markov chain for $\pi_t heta$

Policy Objective Functions

- **Goal:** given policy $\pi_\theta(s, a)$ with parameters θ, find best θ
- But how do we measure the quality of a policy π_θ?
- In continuing environments we can use the average value

$$J_{avV}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s_0)$$

- In the episodic case, $d^{\pi_\theta}(s)$ is defined to be:

  the expected number of time steps t on which $S_t = s$ in a randomly generated episode starting in s_0 and following π and the dynamics of the MDP

Episode

Episode of experience under policy π: $S_1, A_1, R_2, \dots, S_k \sim \pi$

Recap: Derivation Tricks

- Log derivative trick

$$\nabla_\theta \log p(x,\theta) = \frac{\nabla_\theta p(x,\theta)}{p(x,\theta)}$$

$$\nabla_\theta p(x,\theta) = p(x,\theta)\nabla_\theta \log p(x,\theta)$$

- Product rule

$$\nabla(fg) = f\nabla g + g\nabla f$$

Policy Gradient Theorem, Proof I

$\nabla_\theta V^\pi(s)$

How to decompose it in terms of $Q^\pi(s, a)$?

Policy Gradient Theorem, Proof I

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\nabla_\theta \Big(\sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a) \Big)
\end{aligned}
$$

How to decompose it in terms of $Q^\pi(s, a)$?

Policy Gradient Theorem, Proof I

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\nabla_\theta \Big(\sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a) \Big) \\
=&\sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s,a) + \pi_\theta(a|s) \nabla_\theta Q^\pi(s,a) \Big)
\end{aligned}
$$

How to decompose it in terms of $Q^\pi(s,a)$?

; Derivative product rule.

Policy Gradient Theorem, Proof I

How to decompose it in terms of $Q^\pi(s, a)$?

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\nabla_\theta \Big(\sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a) \Big) \\
=&\sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s, a) + \pi_\theta(a|s) \nabla_\theta Q^\pi(s, a) \Big)
\end{aligned}
$$

; Derivative product rule.

; Extend Q^π with future state value.

$$
= \sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s, a) + \pi_\theta(a|s) \nabla_\theta \sum_{s', r} P(s', r|s, a)(r + V^\pi(s')) \Big)
$$

Policy Gradient Theorem, Proof I

How to decompose it in terms of $Q^\pi(s, a)$?

$$\nabla_\theta V^\pi(s)$$

$$= \nabla_\theta \Big(\sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a) \Big)$$

$$= \sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s, a) + \pi_\theta(a|s) \nabla_\theta Q^\pi(s, a) \Big)$$

; Derivative product rule.

; Extend Q^π with future state value.

$$= \sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s, a) + \pi_\theta(a|s) \nabla_\theta \sum_{s', r} P(s', r|s, a)(r + V^\pi(s')) \Big)$$

$$= \sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s, a) + \pi_\theta(a|s) \sum_{s', r} P(s', r|s, a) \nabla_\theta V^\pi(s') \Big)$$

$$\nabla_\theta V^\pi(s)$$

How to decompose it in terms of $Q^\pi(s,a)$?

$$=\nabla_\theta\Big(\sum_{a\in\mathcal{A}}\pi_\theta(a|s)Q^\pi(s,a)\Big)$$

$$=\sum_{a\in\mathcal{A}}\Big(\nabla_\theta\pi_\theta(a|s)Q^\pi(s,a)+\pi_\theta(a|s)\nabla_\theta Q^\pi(s,a)\Big)$$

; Derivative product rule.

; Extend Q^π with future state value.

$$=\sum_{a\in\mathcal{A}}\Big(\nabla_\theta\pi_\theta(a|s)Q^\pi(s,a)+\pi_\theta(a|s)\nabla_\theta\sum_{s',r}P(s',r|s,a)(r+V^\pi(s'))\Big)$$

$$=\sum_{a\in\mathcal{A}}\Big(\nabla_\theta\pi_\theta(a|s)Q^\pi(s,a)+\pi_\theta(a|s)\sum_{s',r}P(s',r|s,a)\nabla_\theta V^\pi(s')\Big)$$

$$=\sum_{a\in\mathcal{A}}\Big(\nabla_\theta\pi_\theta(a|s)Q^\pi(s,a)+\pi_\theta(a|s)\sum_{s'}P(s'|s,a)\nabla_\theta V^\pi(s')\Big)$$

; Because $P(s'|s,a)=\sum_r P(s',r|s,a)$

Policy Gradient Theorem

Now we have

$$\nabla_\theta V^\pi(s) = \sum_{a \in \mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s,a) + \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s') \Big)$$

This equation has a nice recursive form and the future state value function $V_\pi(s')$ can be repeated unrolled by following the same equation.

Policy Gradient Theorem

Let's consider the following visitation sequence.

$$s \xrightarrow{a \sim \pi_\theta(.|s)} s' \xrightarrow{a \sim \pi_\theta(.|s')} s'' \xrightarrow{a \sim \pi_\theta(.|s'')} \dots$$

Policy Gradient Theorem

Let's consider the following visitation sequence.

$$s \xrightarrow{a \sim \pi_\theta(.|s)} s' \xrightarrow{a \sim \pi_\theta(.|s')} s'' \xrightarrow{a \sim \pi_\theta(.|s'')} \dots$$

$\rho^\pi(s \rightarrow x, k)$ is the probability of transitioning from state s to state x with policy π_θ after k step.

- When k = 0: $\rho^\pi(s \rightarrow s, k = 0) = 1$

Policy Gradient Theorem

Let's consider the following visitation sequence.

$$s \xrightarrow{a \sim \pi_\theta(.|s)} s' \xrightarrow{a \sim \pi_\theta(.|s')} s'' \xrightarrow{a \sim \pi_\theta(.|s'')} \dots$$

$\rho^\pi(s \to x, k)$ is the probability of transitioning from state s to state x with policy π_θ after k step.

- When k = 0: $\rho^\pi(s \to s, k = 0) = 1$
- When k = 1, we scan through all possible actions and sum up the transition probabilities to the target state: $\rho^\pi(s \to s', k = 1) = \sum_a \pi_\theta(a|s) P(s'|s, a)$

Policy Gradient Theorem

Let's consider the following visitation sequence.

$$s \xrightarrow{a \sim \pi_\theta(.|s)} s' \xrightarrow{a \sim \pi_\theta(.|s')} s'' \xrightarrow{a \sim \pi_\theta(.|s'')} \dots$$

$\rho^\pi(s \to x, k)$ is the probability of transitioning from state s to state x with policy π_θ after k step.

- When k = 0: $\rho^\pi(s \to s, k = 0) = 1$
- When k = 1, we scan through all possible actions and sum up the transition probabilities to the target state: $\rho^\pi(s \to s', k = 1) = \sum_a \pi_\theta(a|s)P(s'|s, a)$
- Imagine that the goal is to go from state s to x after $k + 1$ steps while following policy π_θ. We can first travel from s to a middle point s' (any state can be a middle point, $s' \in S$) after k steps and then go to the final state x during the last step. In this way, we are able to update the visitation probability recursively: $\rho^\pi(s \to x, k + 1) = \sum_{s'} \rho^\pi(s \to s', k)\rho^\pi(s' \to x, 1)$

Policy Gradient

Let

$$\phi(s) = \sum_{a \in \mathcal{A}} \nabla_\theta \pi_\theta(a|s) Q^\pi(s,a)$$

If we keep on extending $\nabla_\theta V^\pi(.)$ infinitely, it is easy to find out that we can transition from the starting state s to any state after any number of steps in this unrolling process and by summing up all the visitation probabilities, we get $\nabla_\theta V^\pi(.)$

$\nabla_\theta V^\pi(s)$

$$\nabla_\theta V^\pi(s)$$

$$= \phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s')$$

$$\nabla_\theta V^\pi(s)$$

$$=\phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s')$$

$$=\phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s')$$

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \nabla_\theta V^\pi(s')
\end{aligned}
$$

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) [\phi(s') + \sum_{s''} \rho^\pi(s' \to s'', 1) \nabla_\theta V^\pi(s'')]
\end{aligned}
$$

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) [\phi(s') + \sum_{s''} \rho^\pi(s' \to s'', 1) \nabla_\theta V^\pi(s'')] \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \nabla_\theta V^\pi(s'') \text{ ; Consider } s' \text{ as the middle point for } s \to s''
\end{aligned}
$$

$$
\begin{aligned}
&\nabla_\theta V^\pi(s)\\
=&\phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s')\\
=&\phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s')\\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \nabla_\theta V^\pi(s')\\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) [\phi(s') + \sum_{s''} \rho^\pi(s' \to s'', 1) \nabla_\theta V^\pi(s'')]\\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \nabla_\theta V^\pi(s'') \text{ ; Consider } s' \text{ as the middle point for } s \to s''\\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \phi(s'') + \sum_{s'''} \rho^\pi(s \to s''', 3) \nabla_\theta V^\pi(s''')
\end{aligned}
$$

$$
\begin{aligned}
&\nabla_\theta V^\pi(s) \\
=&\phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \nabla_\theta V^\pi(s') \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) [\phi(s') + \sum_{s''} \rho^\pi(s' \to s'', 1) \nabla_\theta V^\pi(s'')] \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \nabla_\theta V^\pi(s'') \text{ ; Consider } s' \text{ as the middle point for } s \to s'' \\
=&\phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \phi(s'') + \sum_{s'''} \rho^\pi(s \to s''', 3) \nabla_\theta V^\pi(s''') \\
=&\sum_{x \in \mathcal{S}} \sum_{k=0}^{\infty} \rho^\pi(s \to x, k) \phi(x)
\end{aligned}
$$

Policy Gradient

The nice rewriting above allows us to exclude the derivative of Q-value function, $\nabla_\theta Q_\pi(s, a)$. By plugging it into the objective function $J(\theta)$, we are getting the following

Policy Gradient

$\nabla_\theta J(\theta) = \nabla_\theta V^\pi(s_0)$; Starting from a random state s_0

Policy Gradient

$$\nabla_\theta J(\theta) = \nabla_\theta V^\pi(s_0)$$; Starting from a random state s_0

$$= \sum_s \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)\phi(s)$$

Policy Gradient

$$\nabla_\theta J(\theta) = \nabla_\theta V^\pi(s_0)$$; Starting from a random state s_0

$$= \sum_s \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)\phi(s)$$; Let $\eta(s) = \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)$

Policy Gradient

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta V^\pi(s_0) && \text{; Starting from a random state } s_0 \\
&= \sum_s \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)\phi(s) && \text{; Let } \eta(s) = \textstyle\sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k) \\
&= \sum_s \eta(s)\phi(s)
\end{aligned}
$$

Policy Gradient

$$\nabla_\theta J(\theta) = \nabla_\theta V^\pi(s_0)$$; Starting from a random state s_0

$$= \sum_s \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)\phi(s)$$; Let $\eta(s) = \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)$

$$= \sum_s \eta(s)\phi(s)$$

$$= \left(\sum_s \eta(s)\right) \sum_s \frac{\eta(s)}{\sum_s \eta(s)} \phi(s)$$; Normalize $\eta(s), s \in \mathcal{S}$ to be a probability distribution.

Policy Gradient

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta V^\pi(s_0) && \text{; Starting from a random state } s_0 \\
&= \sum_s \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)\phi(s) && \text{; Let } \eta(s) = \textstyle\sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k) \\
&= \sum_s \eta(s)\phi(s) \\
&= \left(\sum_s \eta(s)\right) \sum_s \frac{\eta(s)}{\sum_s \eta(s)} \phi(s) && \text{; Normalize } \eta(s), s \in \mathcal{S} \text{ to be a probability distribution.} \\
&\propto \sum_s \frac{\eta(s)}{\sum_s \eta(s)} \phi(s) && \textstyle\sum_s \eta(s) \text{ is a constant}
\end{aligned}
$$

Policy Gradient

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta V^\pi(s_0) && \text{; Starting from a random state } s_0 \\
&= \sum_s \sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k)\phi(s) && \text{; Let } \eta(s) = \textstyle\sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k) \\
&= \sum_s \eta(s)\phi(s) \\
&= \left(\sum_s \eta(s)\right) \sum_s \frac{\eta(s)}{\sum_s \eta(s)} \phi(s) && \text{; Normalize } \eta(s), s \in \mathcal{S} \text{ to be a probability distribution.} \\
&\propto \sum_s \frac{\eta(s)}{\sum_s \eta(s)} \phi(s) && \textstyle\sum_s \eta(s) \text{ is a constant} \\
&= \sum_s d^\pi(s) \sum_a \nabla_\theta \pi_\theta(a|s) Q^\pi(s,a) && d^\pi(s) = \tfrac{\eta(s)}{\sum_s \eta(s)} \text{ is stationary distribution.}
\end{aligned}
$$

Policy Gradient

Gradient can be further written as

$$\nabla_\theta J(\theta) \propto \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s)$$

Gradient can be further written as

$$\nabla_\theta J(\theta) \propto \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s)$$
$$= \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a) \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)}$$

Policy Gradient

Gradient can be further written as

$$
\begin{aligned}
\nabla_\theta J(\theta) &\propto \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s, a) \nabla_\theta \pi_\theta(a|s) \\
&= \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a) \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)} \\
&= \mathbb{E}_\pi[Q^\pi(s, a) \nabla_\theta \ln \pi_\theta(a|s)] && \text{; Because } (\ln x)' = 1/x
\end{aligned}
$$

Where $\mathbb{E}_\pi$ refers to $\mathbb{E}_{s \sim d_\pi, a \sim \pi_\theta}$ when both state and action distributions follow the policy π_θ (on policy).

Policy Gradient

This vanilla policy gradient update has no bias but high variance. Many following algorithms were proposed to reduce the variance while keeping the bias unchanged.

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi[Q^\pi(s, a)\nabla_\theta \ln \pi_\theta(a|s)]$$