

Reinforcement Learning

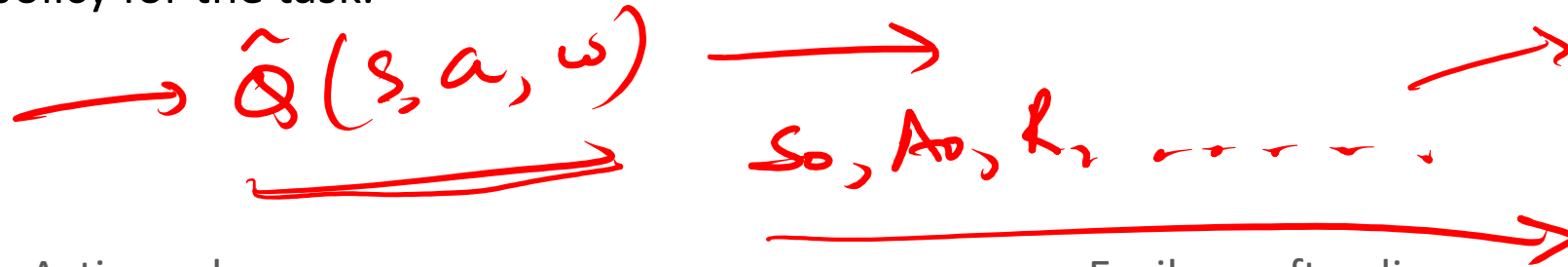
Policy Approximation

Policy as a parameterized function

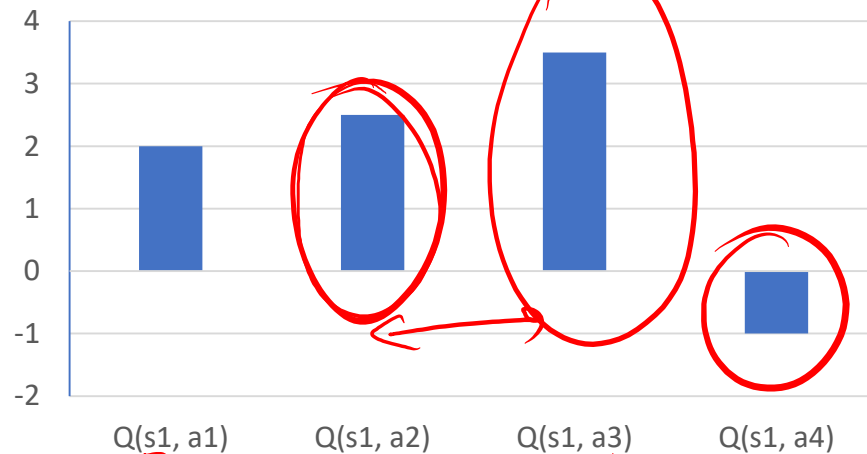
- Till now in control methods we talked about finding the true value functions and used that to generate the policy for the task.

- $V_{\pi}(s)$

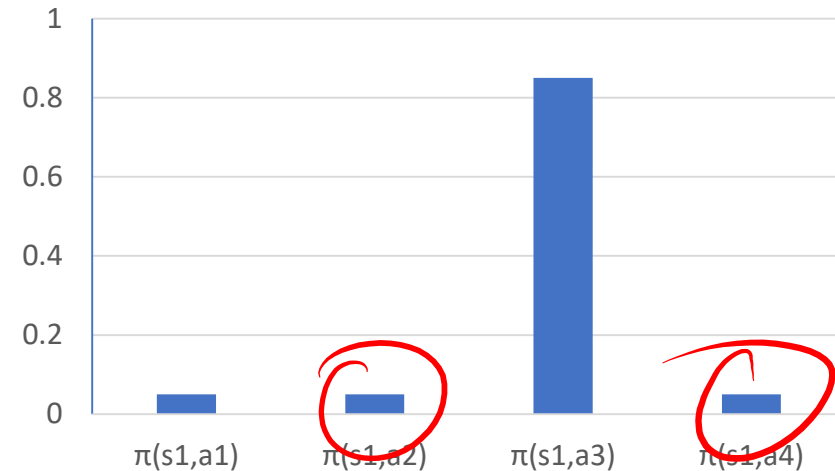
- $Q_{\pi}(s, a)$



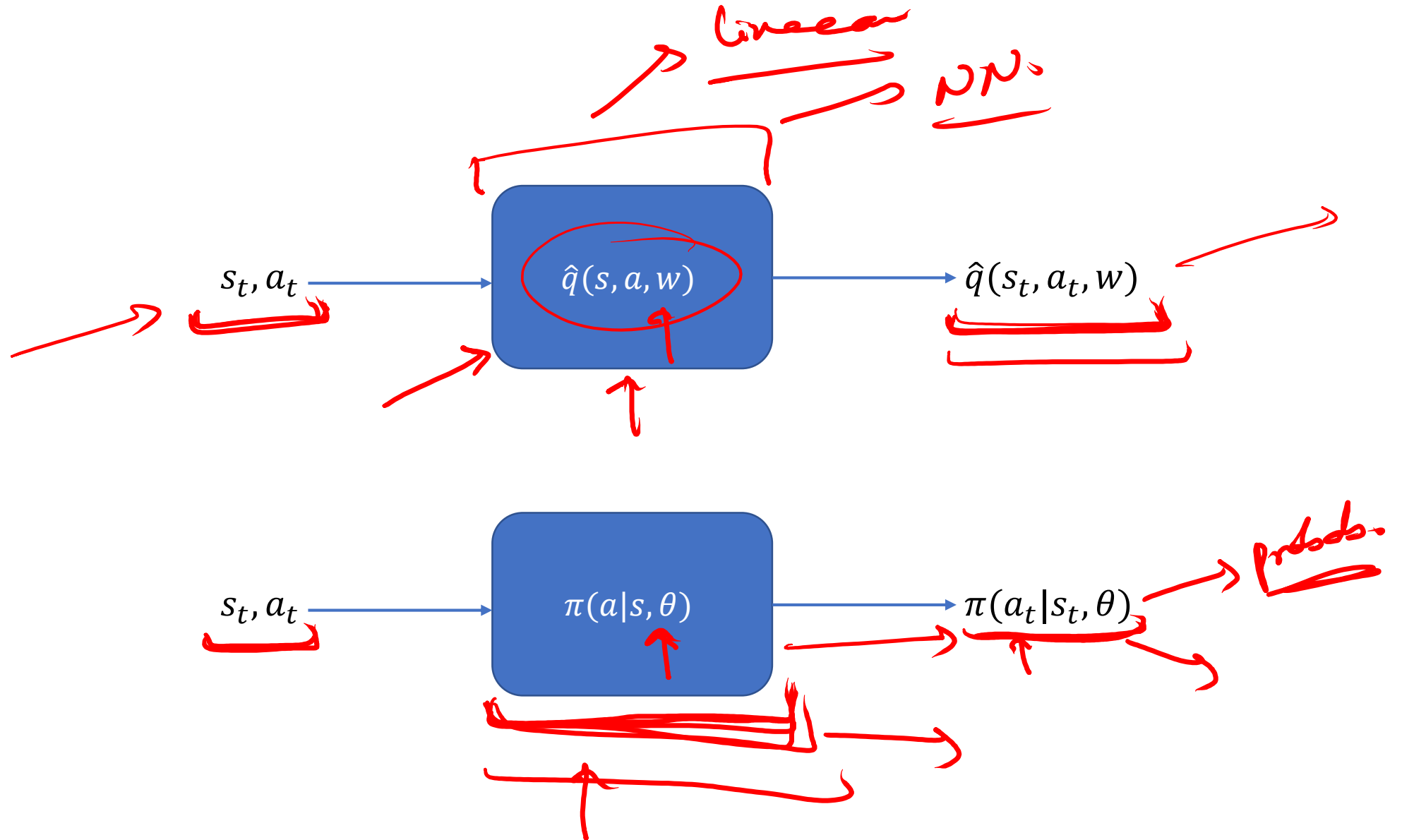
Action values



Epsilon-soft policy



Policy as a parameterized function



Policy as a parameterized function

- $\pi(a|s, \theta)$ is a probability of selecting action a in state s , according to our function approximator.

- $\pi(a|s, \theta) \geq 0 \quad \forall a \in A, s \in S$
- $\sum_a \pi(a|s, \theta) = 1 \quad \forall s \in S$

- Soft-max policy parametrization

- Action preference = $h(s, a, \theta)$

NN
linear

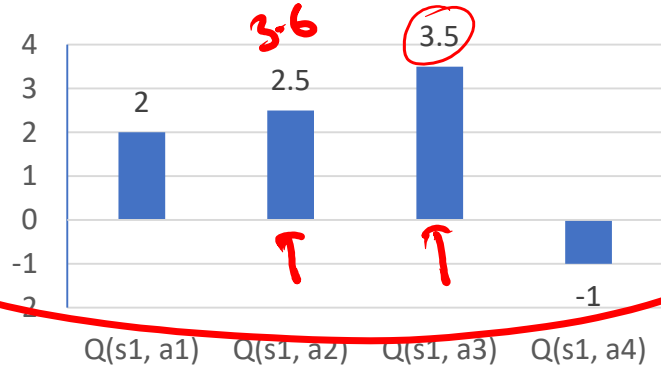
$$\pi(a|s, \theta) = \frac{e^{h(s, a, \theta)}}{\sum_b e^{h(s, b, \theta)}}$$

$s_1 \rightarrow a_1 \rightarrow h(s_1, a_1, \theta)$
 $s_1 \rightarrow a_2 \rightarrow h(s_1, a_2, \theta)$
 $s_1 \rightarrow a_3 \rightarrow h(s_1, a_3, \theta)$

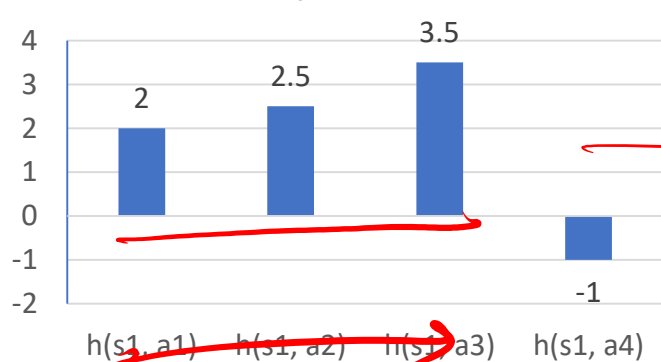
$\pi(a_2|s_1, \theta)$

Policy as a parameterized function

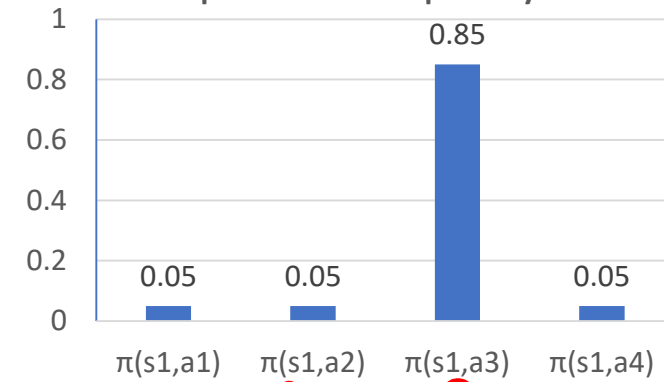
Action values



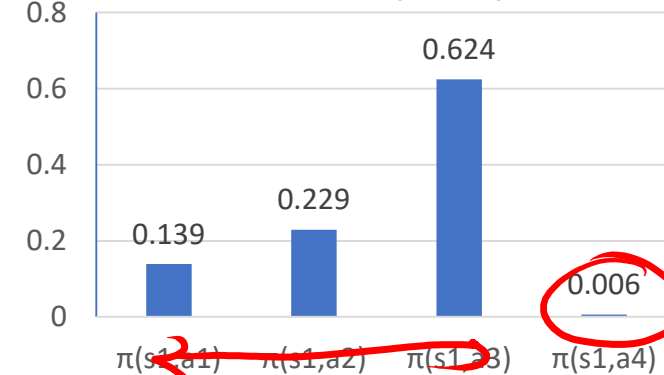
Action preferences



Epsilon-soft policy

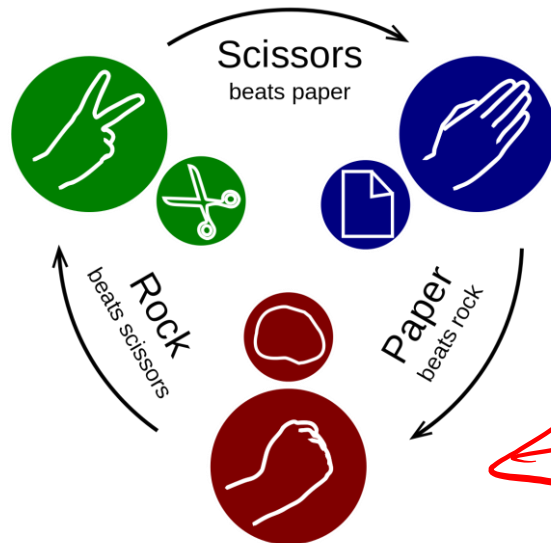


Soft-max policy



Policy as a parameterized function

In card games with imperfect information, the optimal policy is often to do two different things with specific probabilities, such as when bluffing in Poker.



In the game of rock-paper-scissors, the optimal policy is a uniform random policy.

Advantages and disadvantages of policy based methods

- Advantages:
 - Better convergence properties
 - Effective in high-dimensional or continuous action spaces.
 - Can learn stochastic policy
- Disadvantages
 - Often converge to local instead of global optima.
 - High variance

Reinforcement Learning

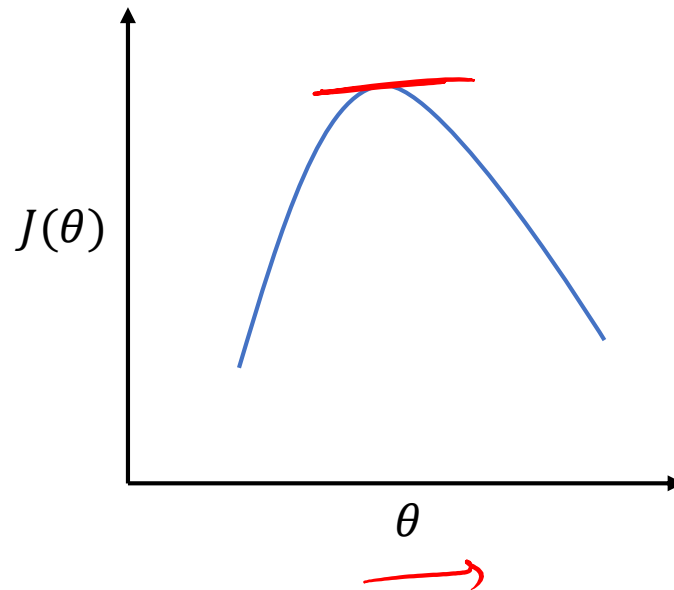
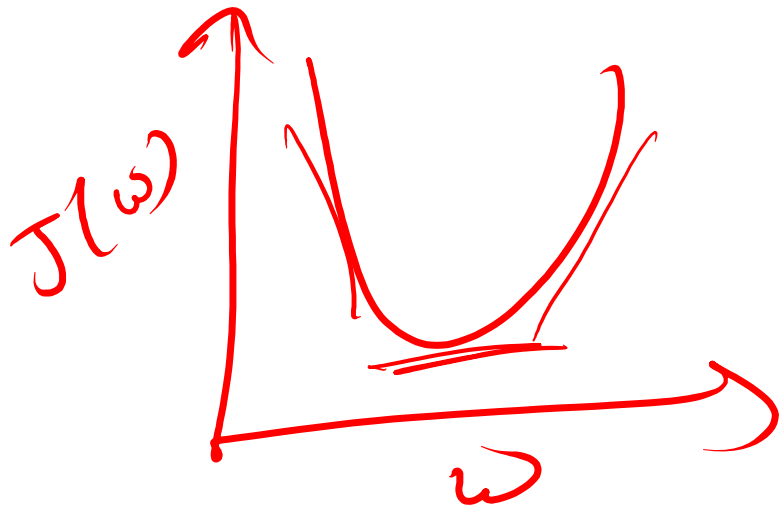
Policy Gradient Theorem



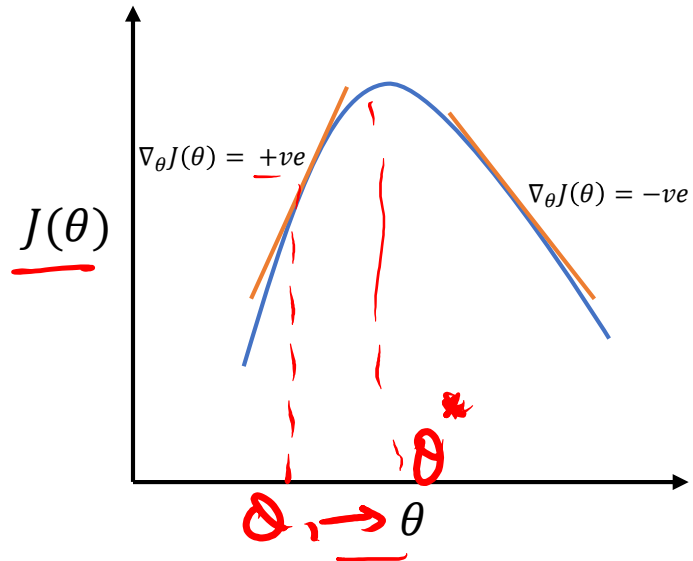
Policy gradient objective function

- For a policy $\pi(a|s, \theta)$ parameterized by θ we need to maximize the objective function $J(\theta)$.
- $J(\theta)$ is some formulation of reward, which is measure of how good a policy π_θ is.
- For episodic cases, $J(\theta)$ can be the value function at the starting state.
 - $J(\theta) = v_{\pi_\theta}(s_0)$

True



Gradient Ascent



- Stochastic gradient-ascent update equation:

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

Policy Gradient Theorem

$$\underline{J(\theta)} = \underline{v_{\pi_\theta}(s_0)}$$

$$\underline{\nabla_\theta J(\theta)} = \underline{\nabla_\theta v_{\pi_\theta}(s_0)}$$

$$\underline{\nabla_\theta J(\theta)} \propto \sum_s \underline{\mu(s)} \sum_a q_{\pi_\theta}(s, a) \underline{\nabla_\theta \pi(a|s, \theta)}$$

$$\nabla_\theta J(\theta) \propto \sum_s \mu(s) \sum_a q_{\pi_\theta}(s, a) \underline{\pi(a|s, \theta) \nabla_\theta \log(\pi(a|s, \theta))}$$

$$\nabla_\theta J(\theta) \propto \sum_s \mu(s) \sum_a \pi(a|s, \theta) \nabla_\theta \log(\pi(a|s, \theta)) q_{\pi_\theta}(s, a)$$

$$\underline{\nabla_\theta J(\theta)} = \mathbb{E}_{\pi_\theta} [\underline{\nabla_\theta \log(\pi(A_t|S_t, \theta))} \underline{q_{\pi_\theta}(S_t, A_t)}]$$

↑
↓
score function

$$\underline{\nabla f(x)} = \underline{\frac{\nabla f(x)}{f(x)} f(x)} = f(x) \underline{\nabla \log(f(x))}$$

Score function for soft-max policy

- Soft-max policy:

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

- Let h be a linear function approximator,

$$h(s, a, \theta) = x(s, a) \cdot \theta^T$$

$$\nabla_{\theta} \log(\pi(a|s, \theta)) = \nabla_{\theta} \left[\log \left(\frac{e^{x(s,a)\theta^T}}{\sum_b e^{x(s,b)\theta^T}} \right) \right]$$

$$\nabla_{\theta} \log(\pi(a|s, \theta)) = x(s, a) - \sum_b \pi(b|s, \theta) x(s, b)$$

Reinforcement Learning

Monte Carlo Policy Gradient : REINFORCE

REINFORCE

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}} [\nabla_{\theta} \log(\pi(A_t | S_t, \theta)) q_{\pi_{\theta}}(S_t, A_t)]$$

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}} [\nabla_{\theta} \log(\pi(A_t | S_t, \theta)) G_t]$$

$$, \mathbb{E}_{\pi} [G_t | S_t, A_t] = q_{\pi}(S_t, A_t)$$

Reinforce update equation:

$$\theta \leftarrow \theta + \alpha G_t \nabla_{\theta} \log(\pi(A_t | S_t, \theta))$$

$\nabla_{\theta} J(\theta)$

$S_0, A_0, R_1, S_1, A_1, R_2, \dots, R_T$

G_t

REINFORCE

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

Input: a differentiable policy parameterization $\pi(a|s, \theta)$ \rightarrow *soft-max policy.*

Algorithm parameter: step size $\alpha > 0$

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

Loop forever (for each episode):

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

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

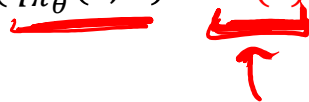
\rightarrow $G \leftarrow \sum_{k=t+1}^T \gamma^{k-t-1} R_k$

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

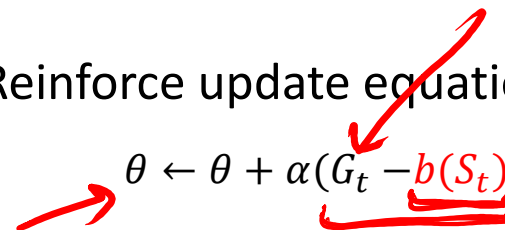
(G_t)

Reducing variance using baseline

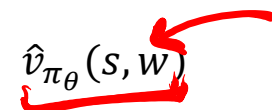
- Baseline can be any function, as long as it does not vary with a .

$$\nabla_{\theta} J(\theta) \propto \sum_s \mu(s) \sum_a (q_{\pi_{\theta}}(s, a) - \underbrace{b(s)}) \nabla_{\theta} \pi(a|s, \theta)$$


- Reinforce update equation with baseline:

$$\theta \leftarrow \theta + \alpha (G_t - \underbrace{b(S_t)}) \nabla_{\theta} \log(\pi(A_t|S_t, \theta))$$


- One natural choice for baseline is the state value function,

$$b(s) = \hat{v}_{\pi_{\theta}}(s, w)$$


REINFORCE – with baseline

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

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

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

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

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

Loop forever (for each episode):

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

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

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

→ $\delta \leftarrow G - \hat{v}(S_t, \mathbf{w})$

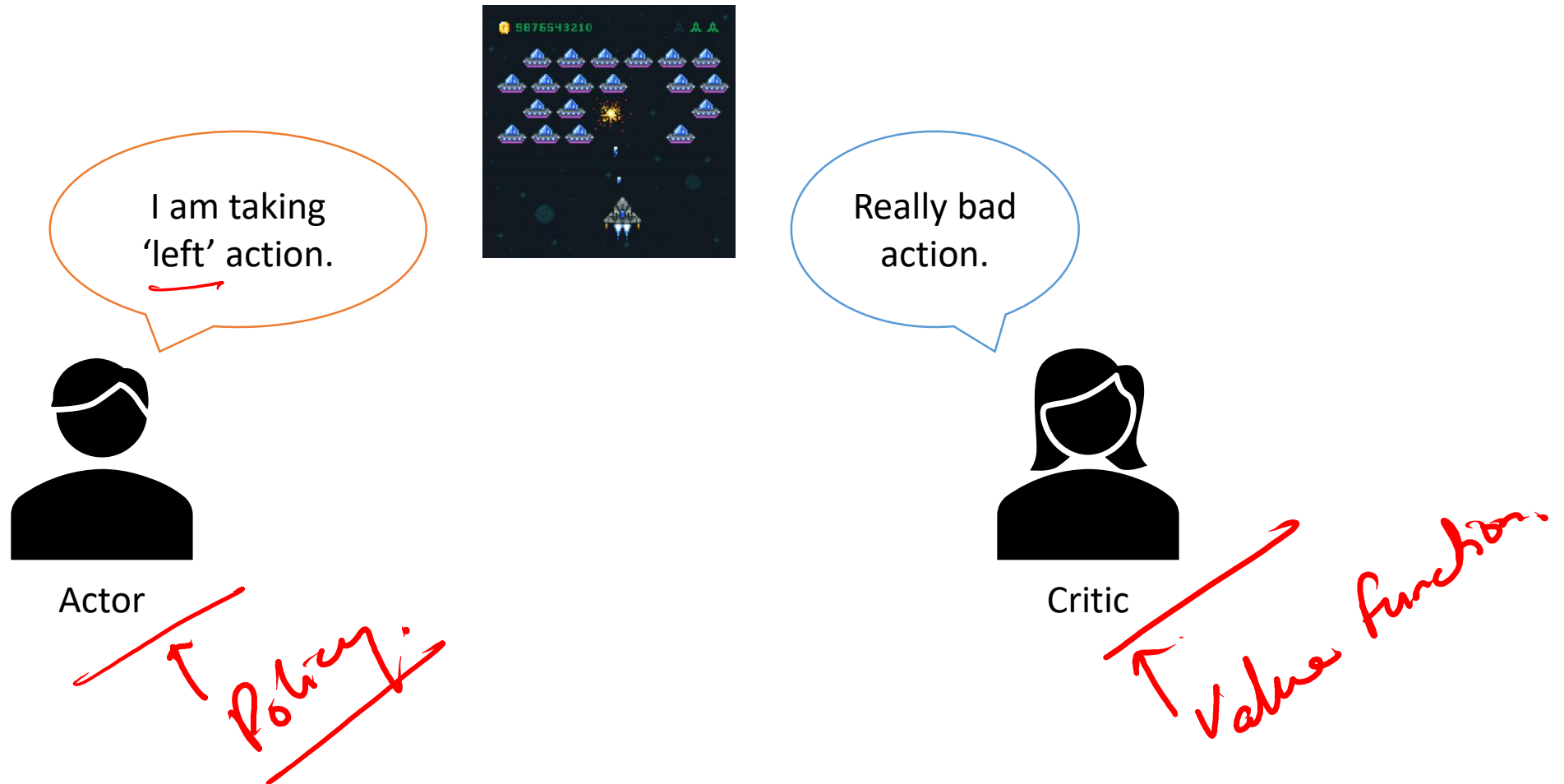
→ $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t, \mathbf{w})$

→ $\theta \leftarrow \theta + \alpha^{\theta} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \theta)$

Reinforcement Learning

Actor-Critic

Actor-Critic based methods



Actor-Critic based methods

- From policy gradient theorem,

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} [\nabla_{\theta} \log(\pi(A_t | S_t, \theta)) q_{\pi_{\theta}}(S_t, A_t)]$$

- We use a critic to estimate action value function.

$$q_{\pi_{\theta}}(s, a) \approx \hat{q}_{\pi_{\theta}}(s, a, w)$$

Actor-Critic based methods

- For Q-actor critic,

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} [\nabla_{\theta} \log(\pi(A_t|S_t, \theta)) \hat{q}_{\pi_{\theta}}(S_t, A_t, w)]$$

$$\theta \leftarrow \theta + \alpha \hat{q}_{\pi_{\theta}}(S_t, A_t, w) \nabla_{\theta} \log(\pi(A_t|S_t, \theta))$$

- Adding baseline (advantageous actor critic),

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} [\nabla_{\theta} \log(\pi(A_t|S_t, \theta)) (\hat{q}_{\pi_{\theta}}(S_t, A_t, w) - \hat{v}_{\pi_{\theta}}(S_t, w))]$$

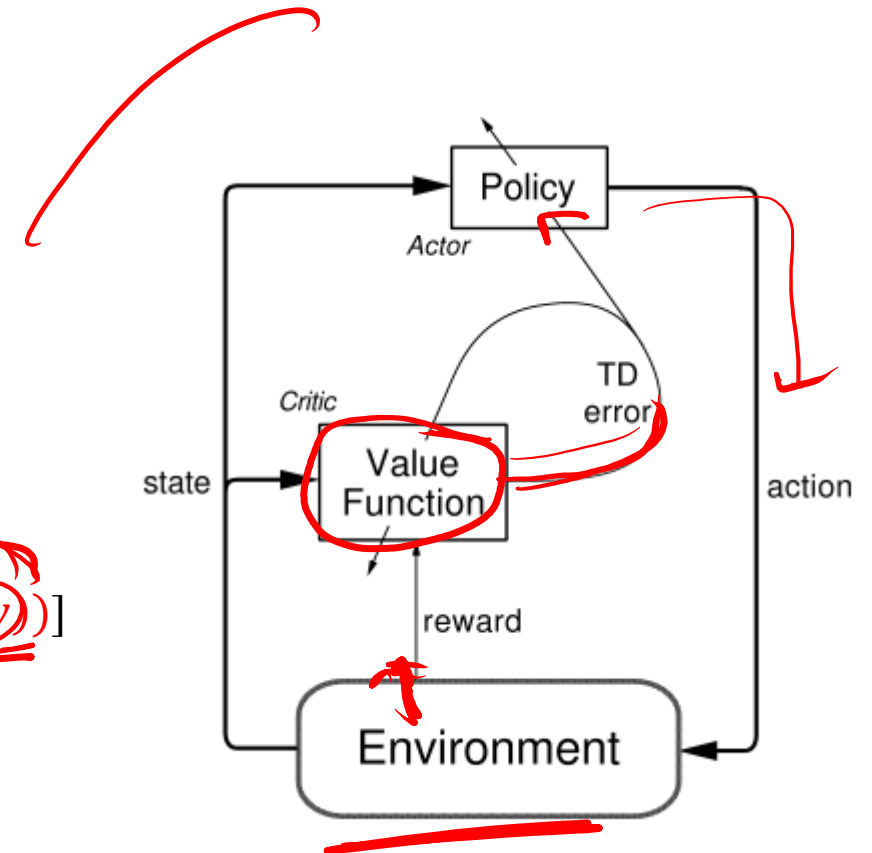
$$\theta \leftarrow \theta + \alpha (\hat{q}_{\pi_{\theta}}(S_t, A_t, w) - \hat{v}_{\pi_{\theta}}(S_t, w)) \nabla_{\theta} \log(\pi(A_t|S_t, \theta))$$

- TD(0) actor critic,

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} [\nabla_{\theta} \log(\pi(A_t|S_t, \theta)) (R_{t+1} + \gamma \hat{v}_{\pi_{\theta}}(S_{t+1}, w) - \hat{v}_{\pi_{\theta}}(S_t, w))]$$

$$\delta = (R_{t+1} + \gamma \hat{v}_{\pi_{\theta}}(S_{t+1}, w) - \hat{v}_{\pi_{\theta}}(S_t, w))$$

$$\theta \leftarrow \theta + \alpha \delta \nabla_{\theta} \log(\pi(A_t|S_t, \theta))$$



TD Error

TD(0) Actor Critic Algorithm

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

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

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

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

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

Loop forever (for each episode):

Initialize S (first state of episode)

$I \leftarrow 1$

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

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

Take action A , observe S', R

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

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

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

$I \leftarrow \gamma I$

$S \leftarrow S'$

γ^t

Difference between Value based and Policy Based RL

- Value Based

- Learn Value Function
- Implicit policy (e.g. ϵ -greedy policy)

- Policy Based

- No value function
- Learnt policy

- Actor-Critic

- Learnt Value Function
- Learnt Policy

Reinforcement Learning

Policy Gradient in continuous action space

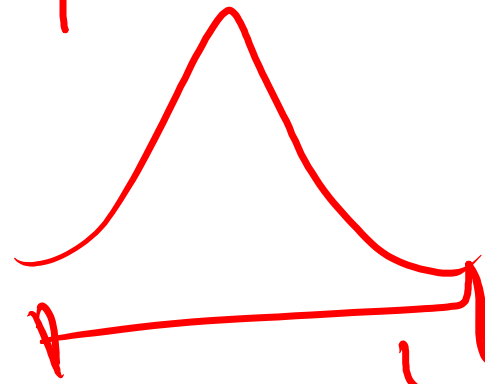
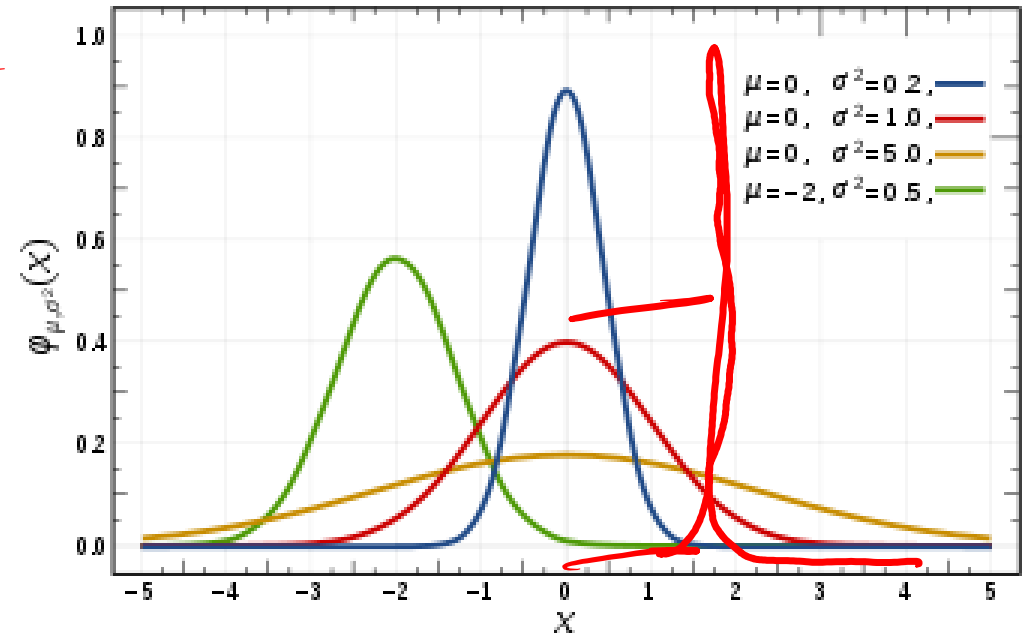
Policy for continuous action space

- Probability density function for normal distribution is,

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- Policy parameterized as a normal distribution,

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



$\sigma(s, \theta)$
 $\mu(s, \theta)$

Policy for continuous action space

- Using linear function approximator for μ and σ ,

→ • $\theta = [\theta_\mu, \theta_\sigma]$

→ • $\mu(s, \theta) = \theta_\mu^T \cdot x_\mu(s)$

→ • $\sigma(s, \theta) = \exp(\theta_\sigma^T \cdot x_\sigma(s))$

- Gradient of log policy has two components,

→ • $\nabla_{\theta_\mu} \log(\pi(a|s, \theta)) = \frac{1}{\sigma(s, \theta)^2} (a - \mu(s, \theta)) x_\mu(s)$

→ • $\nabla_{\theta_\sigma} \log(\pi(a|s, \theta)) = \left(\frac{(a - \mu(s, \theta))^2}{\sigma(s, \theta)^2} - 1 \right) x_\sigma(s)$