ST449 Artificial Intelligence and Deep Learning

Lecture 10

Policy Gradient Methods

Milan Vojnovic

https://github.com/lse-st449/lectures

Topics of this lecture

- Policy-based learning
- Policy learning objective functions
- Policy gradient theorem
- Actor-critic algorithms
- Compatible function approximation theorem
- Variance reduction using a baseline

Policies that we studied so far

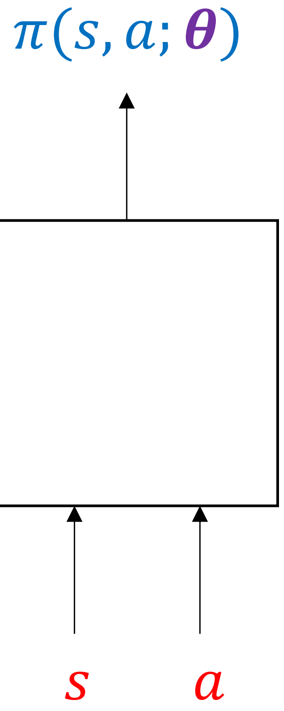
- Greedy:

$$\pi(s,a) = \frac{1}{|\arg\max_{a'} Q(s,a')|} 1_{a \in \arg\max_{a'} Q(s,a')}$$

- Soft policies, e.g. ϵ-greedy:

$$\pi(s,a) = (1-\epsilon) \frac{1}{|\arg\max_{a'} Q(s,a')|} 1_{a \in \arg\max_{a'} Q(s,a')} + \epsilon \frac{1}{|A(s)|}$$

Policy function approximation



Types of RL with function approximation

Value function approximation \ Policy function approximation	No	Yes
No		Policy-based
Yes	Value-based	Actor-critic

Value-based:
- Implicit policy, e.g. ϵ-greedy

Policy-based:
- No value function
- Learn policy

Actor-critic:
- Learn value
- Learn policy

Pros and cons of policy-based learning

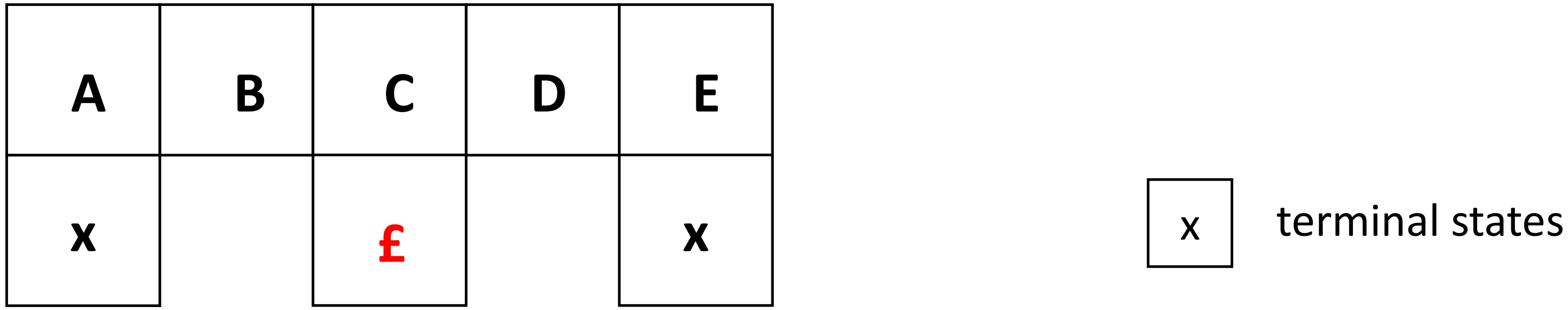
Pros:

- Better convergence properties
- Scalable for high-dimensional or continuous action spaces
- Suitable for learning stochastic policies

Cons:

- Convergence to local minima
- Policy evaluation typically inefficient and high variance

Example

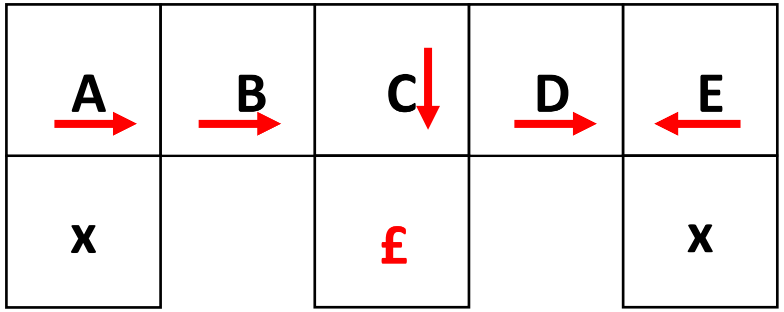


- Suppose we have features vectors:

$$\phi(s,a) = \begin{cases} 1 & \text{if } s \in \{\mathrm{A, B, C, D, E}\} \text{ and } a = \text{move to E} \\ 0 & \text{otherwise} \end{cases}$$

- Two approaches:
  - Value-based using the action-value approximator: $\hat{Q}(s,a;\boldsymbol{w}) = f(\phi(s,a),\boldsymbol{w})$
  - Policy-based using the policy approximator: $\pi(s,a;\boldsymbol{\theta}) = g(\phi(s,a),\boldsymbol{\theta})$

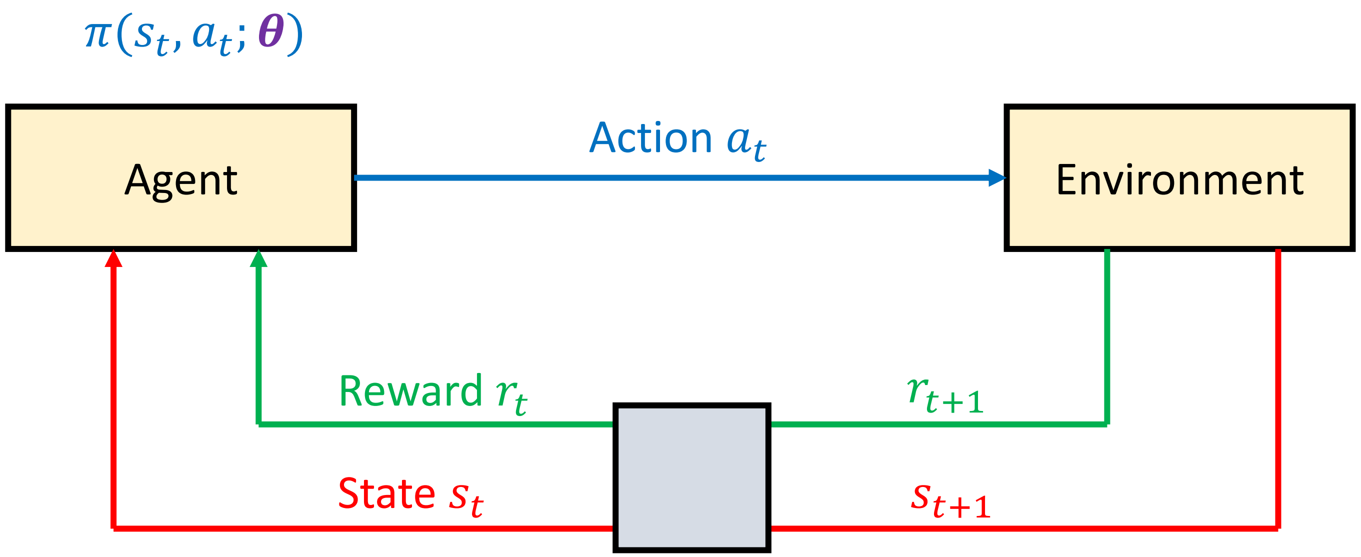
Example (cont'd)



- States **B** and **D** cannot be distinguished because of the symmetry
- A deterministic policy **can get stuck**

- Optimal stochastic policy: at states **B** and **D** move either to **E** or **W** equiprobably
- Value-based learning: learns a nearly-deterministic policy
  - Long time until reaching **£**
- Policy-based learning: can learn an optimal policy

Policy-based reinforcement learning problem



- Problem: how to fit parameters of a policy approximator?
- Solution: find parameter θ that maximizes an objective function

Policy objective functions

- Average rewards:

$$J(\boldsymbol{\theta}) = \lim_{n\to\infty} \frac{1}{n} \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[\sum_{t=1}^{n} r_t] = \sum_{s,a} \mu^{\pi(\cdot;\boldsymbol{\theta})}(s)\pi(s,a;\boldsymbol{\theta})R_s^a$$

- $R_s^a \coloneqq \mathbf{E}[r_{t+1}|s_t = s, a_t = a]$

- $\mu^{\pi(\cdot;\boldsymbol{\theta})}(\cdot)$ assumed to be the stationary distribution under policy $\pi(\cdot,\cdot;\boldsymbol{\theta})$:

$$\mu^{\pi(\cdot;\boldsymbol{\theta})}(s) = \sum_{s',a'} P_{s',s}^{a'} \mu^{\pi(\cdot;\boldsymbol{\theta})}(s')\pi(s',a';\boldsymbol{\theta}) \qquad \text{(global balance equations)}$$

Policy objective functions (cont'd)

- Discounted expected rewards: given a discount factor $\gamma \in [0,1]$ and initial state distribution μ, maximize the expected discounted rewards:

$$J(\boldsymbol{\theta}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[\sum_{t\geq 0} \gamma^t r_{t+1}]$$

or, equivalently,

$$J(\boldsymbol{\theta}) = \sum_{s^0} \mu(s^0)\, V^{\pi(\cdot;\boldsymbol{\theta})}(s^0)$$

- For example, distribution μ may have all its mass on a specific initial state

- If $\gamma = 1$, the task is assumed to be episodic

Policy gradient theorem

- Thm. For any differentiable policy $\pi(s, a; \boldsymbol{\theta})$ with respect to parameter $\boldsymbol{\theta}$ the policy gradient for average reward and discounted expected rewards objective is

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{s,a} \mu^{\pi(\cdot;\boldsymbol{\theta})}(s, a)\, \nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))\, Q^{\pi(\cdot;\boldsymbol{\theta})}(s, a)$$

- For average reward objective:

$\mu^{\pi(\cdot;\boldsymbol{\theta})}$ is the stationary distribution of $\{(s_t, a_t)\}_t$ under policy $\pi(\cdot; \boldsymbol{\theta})$

- For discounted expected rewards objective:

$$\mu^{\pi(\cdot;\boldsymbol{\theta})}(s, a) = \sum_{t\geq 0} \gamma^t \sum_{s^0} \Pr_{\pi(\cdot;\boldsymbol{\theta})}[s_t = s, a_t = a | s_0 = s^0]\mu(s^0)$$

Policy scores

- The term

$$\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))$$

is often referred as the **policy score**

Comment on discounted expected rewards

- For every $\gamma \in [0,1)$, we have

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \mathbf{E}_{(s,a)\sim\tilde{\mu}^{\pi(\cdot;\boldsymbol{\theta})}}\left[\nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}))\, Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a)\right]$$

where $\tilde{\mu}^{\pi(\cdot;\boldsymbol{\theta})}(s,a;\gamma) = \frac{\mu^{\pi(\cdot;\boldsymbol{\theta})}(s,a;\gamma)}{\sum_{s',a'} \mu^{\pi(\cdot;\boldsymbol{\theta})}(s',a';\gamma)} = (1-\gamma)\mu^{\pi(\cdot;\boldsymbol{\theta})}(s,a;\gamma)$

- Abbreviated notation: $\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[\cdot] \equiv \mathbf{E}_{(s,a)\sim\tilde{\mu}^{\pi(\cdot;\boldsymbol{\theta})}}[\cdot]$

Proof for average rewards

- The action-value functions can be defined as:

$$Q^{\pi}(s,a) = \mathbf{E}_{\pi}[\sum_{t\geq 0}(r_{t+1} - J(\boldsymbol{\theta})) \mid s_0 = s, a_0 = a]$$

- $\nabla_{\boldsymbol{\theta}} V^{\pi}(s) = \nabla_{\boldsymbol{\theta}} \sum_a \pi(s,a) Q^{\pi}(s,a)$

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

$$= \sum_a \nabla_{\boldsymbol{\theta}} \pi(s,a) Q^{\pi}(s,a) + \sum_a \pi(s,a) \nabla_{\boldsymbol{\theta}} [R_s^a - J(\boldsymbol{\theta}) + \sum_{s'} P_{s,s'}^a V^{\pi}(s')]$$

$$= \sum_a \nabla_{\boldsymbol{\theta}} \pi(s,a) Q^{\pi}(s,a) + \sum_{a,s'} \pi(s,a) P_{s,s'}^a \nabla_{\boldsymbol{\theta}} V^{\pi}(s') - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

Proof for average rewards (cont'd)

- Take expectation with respect to the stationary distribution in both sides of the last equation in the previous slide:

$$\sum_s \mu^\pi(s)\nabla_{\boldsymbol{\theta}} V^\pi(s) =$$

$$= \sum_{s,a} \mu^\pi(s)\nabla_{\boldsymbol{\theta}}\pi(s,a)Q^\pi(s,a) + \sum_{s'} \underbrace{\sum_{a,s} \mu^\pi(s)\pi(s,a)P^a_{s,s'}}_{= \mu^\pi(s') \text{ (stationary distribution)}} \nabla_{\boldsymbol{\theta}} V^\pi(s') - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- $\Rightarrow \ \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{s,a} \mu^\pi(s)\nabla_{\boldsymbol{\theta}}\pi(s,a)Q^\pi(s,a)$

$$= \sum_{s,a} \mu^\pi(s)\pi(s,a)\frac{\nabla_{\boldsymbol{\theta}}\pi(s,a)}{\pi(s,a)}Q^\pi(s,a)$$

$$= \sum_{s,a} \mu^\pi(s,a)\nabla_{\boldsymbol{\theta}}\log(\pi(s,a))Q^\pi(s,a)$$

Proof for discounted expected rewards

- Basic identities:

(A) $V^{\pi}(s) = \sum_a \pi(s,a) Q^{\pi}(s,a)$

(B) $Q^{\pi}(s,a) = \sum_{s'} P^a_{s,s'}(R^a_{s,s'} + \gamma V^{\pi}(s'))$

(C) $\nabla_{\boldsymbol{\theta}} V^{\pi}(s) = \sum_a \nabla_{\boldsymbol{\theta}} \pi(s,a) Q^{\pi}(s,a) + \sum_a \pi(s,a) \nabla_{\boldsymbol{\theta}} Q^{\pi}(s,a)$

(D) $\nabla_{\boldsymbol{\theta}} Q^{\pi}(s,a) = \gamma \sum_{s'} P^a_{s,s'} \nabla_{\boldsymbol{\theta}} V^{\pi}(s')$

Proof for discounted expected rewards (cont'd)

- $\nabla_{\boldsymbol{\theta}} V^{\pi}(s) \overset{(C)}{=} \sum_a \nabla_{\boldsymbol{\theta}} \pi(s_0, a) Q^{\pi}(s, a) + \sum_a \pi(s_0, a) \nabla_{\boldsymbol{\theta}} Q^{\pi}(s, a)$

$$\overset{(D)}{=} \sum_a \pi(s, a) \nabla_{\boldsymbol{\theta}} \log(\pi(s, a))\, Q^{\pi}(s, a) + \underbrace{\sum_{a, s'} \gamma \pi(s, a) P^a_{s, s'} \nabla_{\boldsymbol{\theta}} V^{\pi}(s')}_{S}$$

- $S \overset{(C)}{=} \sum_{a, s', a'} \gamma \pi(s, a) P^a_{s, s'} \pi(s', a') \nabla_{\boldsymbol{\theta}} \log(\pi(s', a'))\, Q^{\pi}(s', a')$

$$+ \sum_{a, s', a'} \gamma \pi(s, a) P^a_{s, s'} \pi(s', a') \nabla_{\boldsymbol{\theta}} Q^{\pi}(s', a')$$

…

Proof for discounted expected rewards (cont'd)

$\Rightarrow$ the policy gradient equation:

$$\nabla_{\boldsymbol{\theta}} V^{\pi}(s) = \sum_{s',a'} \mu^{\pi}(s',a'|s;\gamma)\, \nabla_{\boldsymbol{\theta}} \log(\pi(s',a'))\, Q^{\pi}(s',a')$$

$$\mu^{\pi}(s',a'|s;\gamma) := \sum_{t\geq 0} \gamma^{t} \pi(s',a') \Pr_{\pi}[s_t = s'|s_0 = s]$$

Example 1: softmax policy gradient

- State-action pairs weighted by linear combination of features:

$$\pi(s, a; \boldsymbol{\theta}) = \frac{e^{\phi(s,a)^\top \boldsymbol{\theta}}}{\sum_{a'} e^{\phi(s,a')^\top \boldsymbol{\theta}}}$$

- The score function:

$$\frac{\partial}{\partial \theta_i} \log(\pi(s, a; \boldsymbol{\theta})) = \phi_i(s, a) - \frac{\sum_{a'} \phi_i(s, a') e^{\phi(s,a')^\top \boldsymbol{\theta}}}{\sum_{a'} e^{\phi(s,a')^\top \boldsymbol{\theta}}}$$

or, equivalently,

$$\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta})) = \phi(s, a) - \mathbf{E}_{a' \sim \pi(s, \cdot; \boldsymbol{\theta})}[\phi(s, a')]$$

Example 2: continuous action space

- Actions state space: set of real numbers $A = \mathbf{R}$
- Policy approximator:

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

where $\mu(s; \boldsymbol{\theta})$ and $\sigma(s; \boldsymbol{\theta})$ are mean and deviation function approximators

- Linear function approximators with feature vectors $\phi_\mu(s)$ and $\phi_\sigma(s)$:
  - $\mu(s; \boldsymbol{\theta}) = \phi_\mu(s)^\top \boldsymbol{\theta}_\mu$ and $\sigma(s; \theta) = \phi_\sigma(s)^\top \boldsymbol{\theta}_\sigma$
  - $\nabla_{\boldsymbol{\theta}_\mu} \log(\pi(s, a; \theta)) = \frac{a - \mu(s; \boldsymbol{\theta})}{\sigma(s; \boldsymbol{\theta})^2} \phi_\mu(s)$
  - $\nabla_{\boldsymbol{\theta}_\sigma} \log(\pi(s, a; \theta)) = \frac{(a - \mu(s; \boldsymbol{\theta}))^2 - \sigma(s; \boldsymbol{\theta})^2}{\sigma(s; \boldsymbol{\theta})^2} \phi_\sigma(s)$

(Exercise: check this)

Example 3: Bernoulli, logistic example

- Action state space: $A = \{0,1\}$
- Policy approximator:

$$\pi(a = 1, s; \boldsymbol{\theta}) = 1 - \pi(a = 0, s; \boldsymbol{\theta}) := p(s; \boldsymbol{\theta})$$

where $p(s; \boldsymbol{\theta})$ is a function approximator

- Linear function approximator with feature vectors $\phi(s)$
  - Each state has action preference scores $h_0(s; \boldsymbol{\theta})$ and $h_1(s; \boldsymbol{\theta})$ such that

$$h_1(s; \boldsymbol{\theta}) - h_0(s; \boldsymbol{\theta}) = \phi(s)^\top \boldsymbol{\theta}$$

  - **Fact 1**: for exponential soft-max policy $p(s; \boldsymbol{\theta}) = \sigma(\phi(s)^\top \boldsymbol{\theta})$
  - **Fact 2**: $\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta})) = \left(a - \sigma(\phi(s)^\top \boldsymbol{\theta})\right)\phi(s)$

logistic function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

REINFORCE: MC policy gradient algorithm

- Use return R_t as an estimate of $Q^{\pi(\cdot;\boldsymbol{\theta})}(s_t, a_t)$

- **Initialization**: $\boldsymbol{\theta}$ arbitrary

- **For each** episode $(s_1, a_1, r_2, \ldots, s_{T-1}, a_{T-1}, r_T)$ generated using policy $\pi(\cdot; \boldsymbol{\theta})$:

  For $t = 1, 2, \ldots, T-1$ do:

  $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \, \nabla_{\boldsymbol{\theta}} \log(\pi(s_t, a_t; \boldsymbol{\theta}))R_t$$

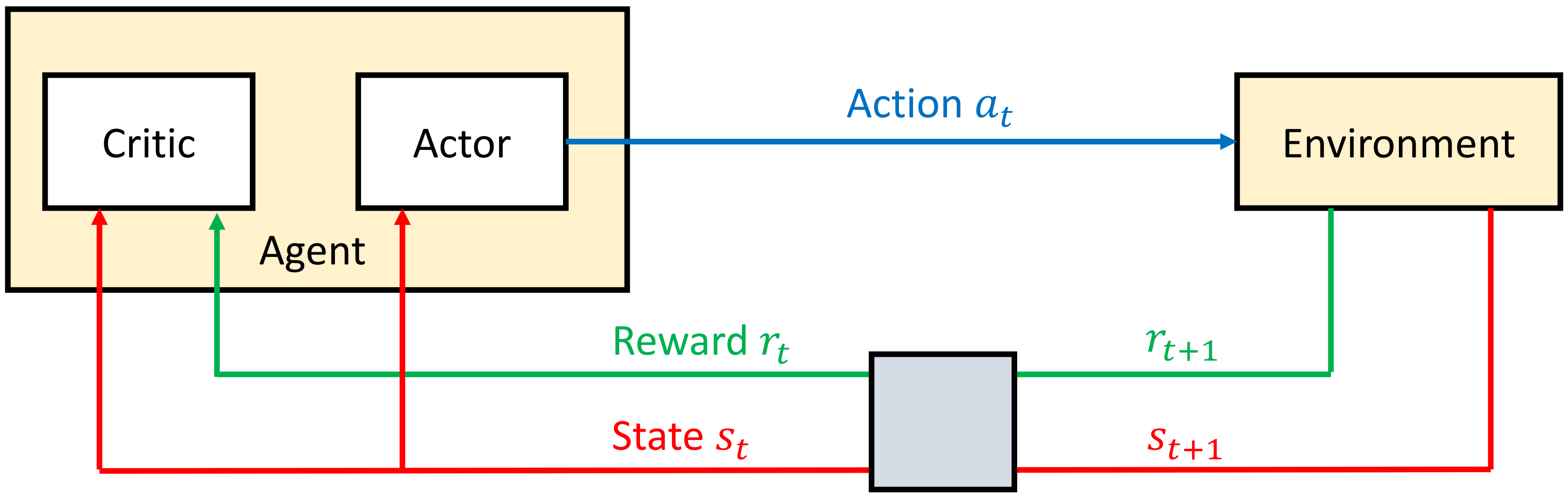
  end for

  return $\boldsymbol{\theta}$

Actor-critic algorithms

- MC policy gradient algorithm may have a high variance
- Solution sought by using actor-critic algorithms
- Actor-critic algorithms combine policy gradient with value function estimation

Actor-critic algorithms



- Critic uses an approximator to learn a value function
- Actor updates policy approximator in a direction of performance improvement

Actor-critic control

- Critic: estimates $Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a)$ by an approximator $\hat{Q}(s,a;\boldsymbol{w})$
  - The critic performs policy evaluation
  - Standard methods can be applied: MC, temporal difference learning, TD(λ), gradient-based least-square estimation methods, …
- Actor: updates policy parameter $\boldsymbol{\theta}$ in direction suggested by the critic
  - The actor performs control using approximate policy gradient:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx \mathbf{E}\left[\nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}))\hat{Q}(s,a;\boldsymbol{w})\right]$$

  - Parameter update:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}))\hat{Q}(s,a;\boldsymbol{w})$$

Example actor-critic algorithm

Linear value function approximator:

$$\hat{Q}(s, a; \boldsymbol{w}) = \phi(s, a)^\top \boldsymbol{w}$$

Critic: updates $\boldsymbol{w}$ by linear TD(0)

Actor: updates $\boldsymbol{\theta}$ by policy gradient

- **Initialization**: $s, \boldsymbol{\theta}, \boldsymbol{w}$
- **For each** episode:

  Sample action a from $\pi(s, \cdot; \boldsymbol{\theta})$

  Repeat until s is terminal:

  Receive reward r and next state s'

  Sample action a' from $\pi(s', \cdot; \boldsymbol{\theta})$

  $$\delta \leftarrow r + \gamma \hat{Q}(s', a'; \boldsymbol{w}) - \hat{Q}(s, a; \boldsymbol{w})$$

  $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla_{\boldsymbol{\theta}} \log\big(\pi(s, a; \boldsymbol{\theta})\big) \hat{Q}(s, a; \boldsymbol{w})$$

  $$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \delta \phi(s, a)$$

  $$a \leftarrow a', s \leftarrow s'$$

Bias issue of approximate policy gradient

- Using approximate policy gradient introduces a bias
- A biased policy gradient may fail to converge to a globally optimal solution
- Unbiased policy gradient can be ensured under certain conditions (next slide)

Compatible function approximation theorem

- Thm. Assume the following two conditions:
  - (C1) compatibility of value function approximator and the policy:

$$\nabla_{\boldsymbol{w}} \hat{Q}(s, a; \boldsymbol{w}) = \nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))$$

  - (C2) value function approximator minimizes the mean-squared error:

$$\mathrm{MSE}(\boldsymbol{w}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\left(Q^{\pi(\cdot;\boldsymbol{\theta})}(s, a) - \hat{Q}(s, a; \boldsymbol{w})\right)^2\right]$$

- Then, the policy gradient is exact:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))\, \hat{Q}(s, a; \boldsymbol{w})]$$

Proof sketch

- First, note that by (C2), $\nabla_{\boldsymbol{w}} \mathrm{MSE}(\boldsymbol{w}) = 0$, i.e.

$$\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\left(Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a) - \hat{Q}(s,a;\boldsymbol{w})\right)\nabla_{\boldsymbol{w}}\hat{Q}(s,a;\boldsymbol{w})\right] = 0$$

- By (C1),

$$\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\left(Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a) - \hat{Q}(s,a;\boldsymbol{w})\right)\nabla_{\boldsymbol{\theta}}\log(\pi(s,a;\boldsymbol{\theta}))\right] = 0$$

which is equivalent to

$$\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\hat{Q}(s,a;\boldsymbol{w})\nabla_{\boldsymbol{\theta}}\log(\pi(s,a;\theta))\right] = \underbrace{\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a)\nabla_{\boldsymbol{\theta}}\log(\pi(s,a;\boldsymbol{\theta}))\right]}_{\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta})}$$

Compatible function approximation in action

- Consider the soft-max policy, for given state-action feature vectors $\phi(s, a)$:

$$\pi(s, a; \boldsymbol{\theta}) = \frac{e^{\phi(s,a)^\top \boldsymbol{\theta}}}{\sum_{a'} e^{\phi(s,a')^\top \boldsymbol{\theta}}}$$

- Compatibility condition requires that

$$\nabla_{\boldsymbol{w}} \hat{Q}(s, a; \boldsymbol{w}) = \nabla_\theta \log\big(\pi(s, a; \boldsymbol{\theta})\big) = \phi(s, a) - \sum_{a'} \phi(s, a')\pi(s, a'; \boldsymbol{\theta})$$

which leads to a linear approximator for the value function:

$$\hat{Q}(s, a; \boldsymbol{w}) = \underbrace{\left(\phi(s, a) - \sum_{a'} \phi(s, a')\pi(s, a'; \boldsymbol{\theta})\right)^\top}_{\text{centered state-action feature vectors}} \boldsymbol{w}$$

Convergence theorem

- Thm. Assume:
  - $\pi(\cdot;\boldsymbol{\theta})$ and $\hat{Q}(\cdot;\boldsymbol{w})$ are differentiable functions
  - Compatibility condition holds
  - The Hessian matrix $\nabla^2_{\boldsymbol{\theta}}\pi(s,a;\boldsymbol{\theta})$ satisfies $\left\|\nabla^2_{\boldsymbol{\theta}}\pi(s,a;\boldsymbol{\theta})\right\|_\infty \leq B < \infty$
  - Step sizes are such that $\lim\limits_{t\to\infty} \eta_t = 0$ and $\sum_{t>0}\eta_t = \infty$
- Then, for any MDP with bounded rewards and the parameter updates:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta_t \sum_{s,a} \mu^{\pi(\cdot;\boldsymbol{\theta}_t)}(s)\pi(s,a;\boldsymbol{\theta}_t)\,\nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}_t))\hat{Q}(s,a;\boldsymbol{w}_t)$$

where $\boldsymbol{w}_t$ is a solution of the equation:

$$\sum_{s,a} \mu^{\pi(\cdot;\boldsymbol{\theta}_t)}(s)\pi(s,a;\boldsymbol{\theta}_t)\left[Q^{\pi(\cdot;\theta_t)}(s,a) - \hat{Q}\,(s,a;\boldsymbol{w})\right]\nabla_{\boldsymbol{w}}\,\hat{Q}\,(s,a;\boldsymbol{w}) = 0$$

are convergent in the sense that $\lim_{t\to\infty}\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t)\| = 0$

Separation of timescales

- In the last theorem, $\boldsymbol{w}_t$ is defined as a solution of a fixed point equation which has the policy's parameter vector $\boldsymbol{\theta}_t$ as a parameter

- An incremental implementation would update $\boldsymbol{w}_t$ incrementally in a similar manner as for the parameter vector $\boldsymbol{\theta}_t$ but with a larger step size parameter

- This can be seen as a separation of timescales:
  - Critic updates the value function approximator at a faster timescale trying to evaluate the current policy chosen by the actor
  - Actor varies the policy's parameter more slowly to allow the critic to evaluate the current policy

- Separation of timescales assumption is common in convergence proofs of actor-critic algorithms

Variance reduction using a baseline

- Subtracting a baseline function $B(s)$ from the policy gradient

$$\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))\,(Q^{\pi(\cdot;\boldsymbol{\theta})}(s, a) - B(s))$$

does not change the expected policy gradient:

$$\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))\,(Q^{\pi(\cdot;\boldsymbol{\theta})}(s, a) - B(s))\right]$$

$$= \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))\, Q^{\pi(\cdot;\boldsymbol{\theta})}(s, a)\right]$$

which is because

$$\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))\, B(s)] = 0 \qquad \text{(check this)}$$

Policy gradient using the advantage function

- Choose the baseline $B(s) = V^{\pi(\cdot;\boldsymbol{\theta})}(s)$

- The advantage function: $A^{\pi(\cdot;\boldsymbol{\theta})}(s,a) := Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a) - V^{\pi(\cdot;\boldsymbol{\theta})}(s)$

- Policy gradient using the advantage function:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}))\, A^{\pi(\cdot;\boldsymbol{\theta})}(s,a)\right]$$

- The advantage function can reduce variance of policy gradient

An approach for estimating the advantage function

- The critic may compute estimators of both value functions:

  $$\hat{Q}(s, a; \boldsymbol{w}) \text{ for } Q^{\pi(\cdot;\boldsymbol{\theta})}(s, a)$$

  and

  $$\hat{V}(s, a; \boldsymbol{v}) \text{ for } V^{\pi(\cdot;\boldsymbol{\theta})}(s)$$

  which can be done by standard methods such as TD learning

- The estimator of the advantage function:

  $$\hat{A}(s, a) = \hat{Q}(s, a; \boldsymbol{w}) - \hat{V}(s, a; \boldsymbol{v})$$

Another approach for estimating the advantage function

- The TD error $\delta^{\pi(\cdot;\boldsymbol{\theta})}(s,s',r) = r + \gamma V^{\pi(\cdot;\boldsymbol{\theta})}(s') - V^{\pi(\cdot;\boldsymbol{\theta})}(s)$ is an **unbiased estimate** of the advantage function:

$$\mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\delta^{\pi(\cdot;\boldsymbol{\theta})}(s_t,s_{t+1},r_{t+1})\middle|s_t = s, a_t = a\right]$$

$$= \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[r_{t+1} + \gamma V^{\pi(\cdot;\boldsymbol{\theta})}(s_{t+1})|s_t = s, a_t = a] - V^{\pi(\cdot;\boldsymbol{\theta})}(s)$$

$$= Q^{\pi(\cdot;\boldsymbol{\theta})}(s,a) - V^{\pi(\cdot;\boldsymbol{\theta})}(s)$$

$\Rightarrow$ we may use the TD error in the policy gradient:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\nabla_{\boldsymbol{\theta}} \log\big(\pi(s,a;\boldsymbol{\theta})\big)\delta^{\pi(\cdot;\boldsymbol{\theta})}\right]$$

- In practice, we will use an approximate TD error:

$$\delta^{\pi(\cdot;\boldsymbol{\theta})}(s,s',r) = r + \gamma \hat{V}(s';\boldsymbol{v}) - \hat{V}(s;\boldsymbol{v})$$

Critic policy evaluation methods

- The critic can use different targets to evaluate:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha\left(v_t - \hat{V}(s;\boldsymbol{w})\right)\phi(s)$$

- The target v_t is defined differently for different methods:
  - MC: $v_t = R_t$
  - TD(0): $v_t = r_{t+1} + \gamma\hat{V}(s_{t+1};\boldsymbol{w})$
  - Forward-view $\mathrm{TD}(\lambda)$: $v_t = R_t^{\lambda}$ (λ-return)
  - Backward-view $\mathrm{TD}(\lambda)$:

$$\delta_t = r_{t+1} + \gamma\hat{V}(s_{t+1};\boldsymbol{w}) - \hat{V}(s_t;\boldsymbol{w})$$
$$e_t = \gamma\lambda e_{t-1} + \phi(s_t)$$
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha\delta_t e_t$$

Actor policy gradient methods

- The policy gradient:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}\left[\nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}))\, A^{\pi(\cdot;\boldsymbol{\theta})}(s,a)\right]$$

- Gradient ascent method:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla_{\boldsymbol{\theta}} \log(\pi(s,a;\boldsymbol{\theta}))\hat{A}(s,a;\boldsymbol{v})$$

- Examples:
    - MC: $\hat{A}(s,a;\boldsymbol{w}) = R_t - \hat{V}(s;\boldsymbol{w})$
    - One-step TD error: $\hat{A}(s,a;\boldsymbol{w}) = r + \gamma\hat{V}(s';\boldsymbol{w}) - \hat{V}(s;\boldsymbol{w})$

Summary

- Equivalent form of policy gradient: $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbf{E}_{\pi(\cdot;\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \log(\pi(s, a; \boldsymbol{\theta}))v]$

Method	v
MC (REINFORCE)	R_s^a
Q actor-critic	$\hat{Q}(s, a; \boldsymbol{w})$
Advantage actor-critic	$\hat{A}(s, a; \boldsymbol{w})$
TD actor-critic	$\delta^{\pi(\cdot;\boldsymbol{\theta})}$
TD(λ) actor-critic	$\delta^{\pi(\cdot;\boldsymbol{\theta})} e^{\pi(\cdot;\boldsymbol{\theta})}$

- Use stochastic gradient descent algorithm
- Critic variants that use policy evaluation to estimate $Q^{\pi(\cdot;\boldsymbol{\theta})}$, $V^{\pi(\cdot;\boldsymbol{\theta})}$, or $A^{\pi(\cdot;\boldsymbol{\theta})}$

References

- Sutton and Barto, Reinforcement Learning, The MIT Press, 2nd edition, 2018, Chapter 13: Policy gradient methods
- Sutton et al, Policy Gradient Methods for Reinforcement Learning with Function Approximation, NIPS 2000
- Silver, Lecture 7: [Policy Gradient]

Seminar exercises

- Solving Atari game Breakout using policy gradient method

Extras

Reshaping rewards

- Consider an MDP(P, R) with transition probabilities and expected rewards

$$P = (P^a_{s,s'}) \text{ and } R = (R^a_{s,s'})$$

- Consider a transformation of expected rewards as follows

$$\tilde{R}^a_{s,s'} = R^a_{s,s'} + \gamma B(s') - B(s)$$

for an arbitrarily given function $B: S \to \mathbf{R}$

- For example, the transformation of expected rewards holds if instantaneous rewards are transformed as $\tilde{r}_{t+1} = r_{t+1} + \gamma B(s_{t+1}) - B(s_t)$

Reshaping rewards theorem

- Thm: MDP(P, R) and MDP$(P, \tilde{R})$ have the same optimal policies.

- Exercise: show this

Proof sketch

- Let Q^* be the optimal action value function and π^* be an optimal policy for MDP(P, R), and let $\tilde{Q}^*$ be the optimal value function for MDP$(P, \tilde{R})$

- By the Bellman's optimality equation:

$Q^*(s,a) = \sum_{s'} P_{s,s'}^a [R_{s,s'}^a + \gamma \max_{a'} Q^*(s',a')]$ for all state action pairs (s,a)

- Since $\sum_{s'} P_{s,s'}^a \left[R_{s,s'}^a + \gamma \max_{a'} Q^*(s',a')\right]$

$$= \sum_{s'} P_{s,s'}^a [\tilde{R}_{s,s'}^a + B(s) + \gamma \max_{a'} [Q^*(s',a') - B(s')]]$$

we have $\underbrace{[Q^*(s,a) - B(s)]}_{\tilde{Q}^*(s,a)} = \sum_{s'} P_{s,s'}^a [\tilde{R}_{s,s'}^a + \gamma \max_{a'} \underbrace{[Q^*(s',a') - B(s)]}_{\tilde{Q}^*(s',a')}]$