Policy gradient

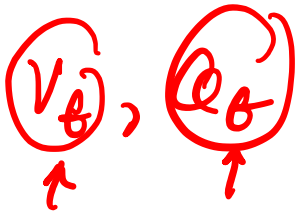
Konpat Preechakul
Chulalongkorn University
October 2019

Recap overview

Model-free	Model-based
Environment is a black box	We know environment (transitions, rewards)
Value-based	**Policy-based**
We learn value. Use value to improve policy greedily	We directly learn policy (from some value)
On-policy	**Off-policy**
Experience comes from target policy (interactive experience)	Experience comes from behavior policy (observative experience)

Recap Value-based

- **Value function approximation**
  - Monte Carlo
    - Value error
  - Temporal difference
    - Semi gradient
      - Poor convergence
      - Tricks
    - Full gradient
      - Better convergence
      - Might not good in practice



Today topic is policy-based

- Model the policy directly (not from the value function)

$\pi_\theta(a|s)$

- Value function is used as a guide

$\theta \leftarrow (Q, V)$

Motivation of policy-based

- Continuous action space

$\max_a Q(s,a) = ?$

$(-1, 1) \rightarrow 0.8$

- Optimal policy might not be "deterministic"

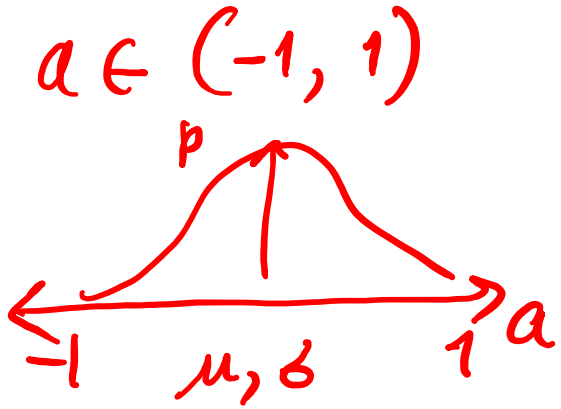
$\operatorname{argmax}_a Q \leftarrow$ Deterministic

- Value-based RL might not give a smooth improvement of the policy

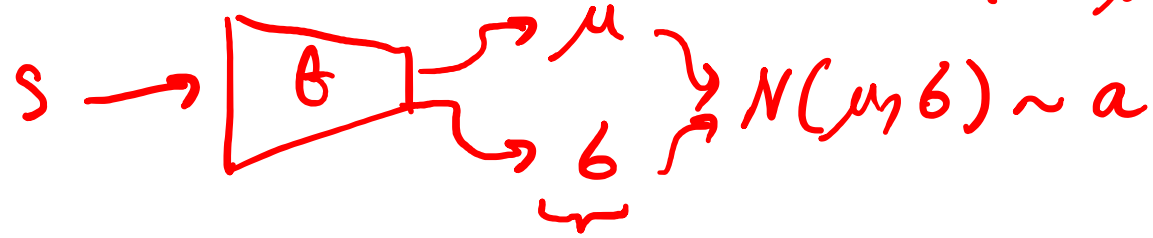
$Q \rightarrow Q'$

- Convergence problems in value based RL

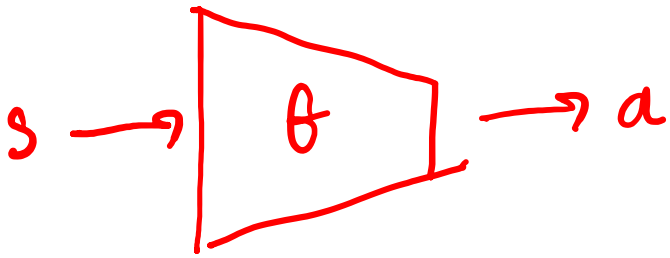
Modeling a policy



- Stochastic policy $\pi_\theta(a|s)$



- Deterministic policy $\pi_\theta(s) \rightarrow a$



How to improve the policy?

- We have neural nets, we want to optimize them with SGD
- What is the objective? $J(\theta)$ ←
- What is the gradient? w.r.t. θ ←

Policy gradient

Policy gradient (PG)

- Objective function

$$J(\theta) = \sum_s P_{s_0}(s) V^{\pi}(s)$$

- Policy gradient

$$\nabla J(\theta) = \sum_s d^{\pi}(s) \sum_a Q^{\pi}(s,a) \nabla_{\theta} \pi_{\theta}(a|s)$$

Sutton, Richard S., David Mcallester, Satinder Singh, and Yishay Mansour. 1999. "Policy Gradient Methods for Reinforcement Learning with Function Approximation." *In Advances in Neural Information Processing Systems 12*, 1057–63.

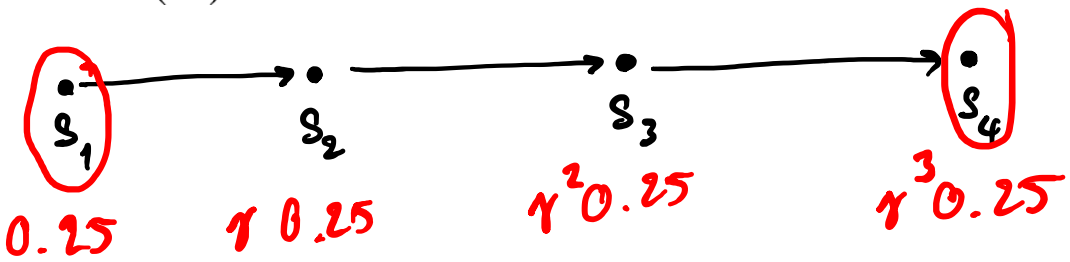
$d^\pi(s)$

Discounted state distribution

- What does it mean?

$\uparrow V \Rightarrow r + \gamma r + \gamma^2 r + \dots$ $\gamma < 1$

$d^\pi(s)$ ← discounted

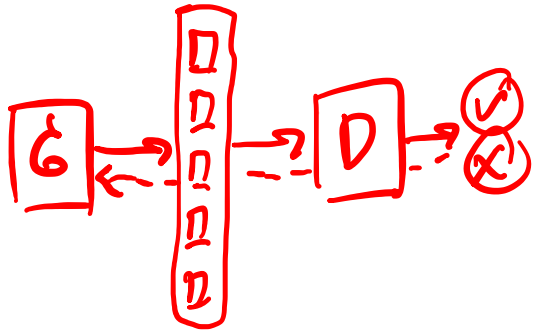


- We care less of far away states
- Why?

Policy gradient (PG)

- Another form (REINFORCE; likelihood ratio)

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



$= \mathbb{E}_{s\sim d^{\pi}} \mathbb{E}_{a\sim\pi}$

$= \sum_{s} d^{\pi}(s) \sum_{a} \pi(a|s)\, Q^{\pi} \frac{\nabla \log \pi}{\frac{\nabla \pi}{\pi}}$

$= \sum_{s} d^{\pi}(s) \sum_{a} Q^{\pi} \nabla \pi$

Williams, Ronald J. 1992. "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning." *Machine Learning* 8 (3-4): 229–56.

Making sense of policy gradient

What does it do?

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

- Gradient for each action is "weighted" based on the goodness

s ; a_0 $Q(\cdot, a_0) = 10$

a_1 $Q(\cdot, a_1) = 5$

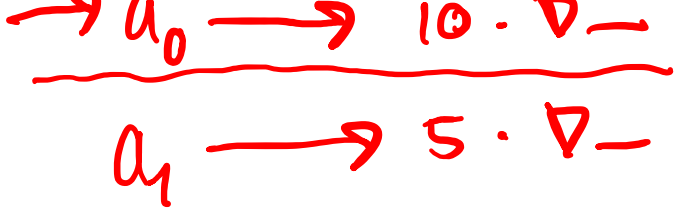
$\nabla_{a_0} \propto 10 \cdot \nabla$ —

$\nabla_{a_1} \propto 5 \cdot \nabla$ —

Efficiency of likelihood ratio

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

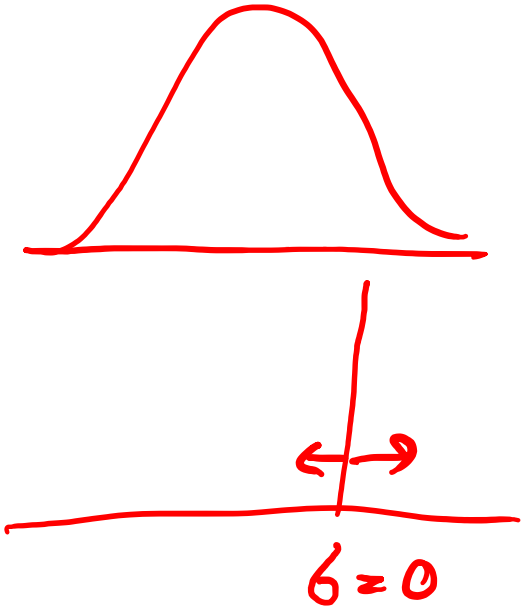
- Gradient is "relative"



$$\nabla J(\theta) = \sum_{s} P_{s_0}(s)\, \nabla V(s)$$

- To get a useful gradient, you need "many" samples
- Gradient is indirect

Policy collapse

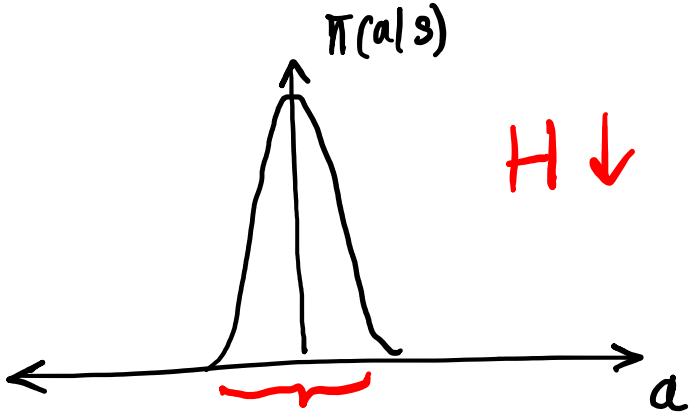


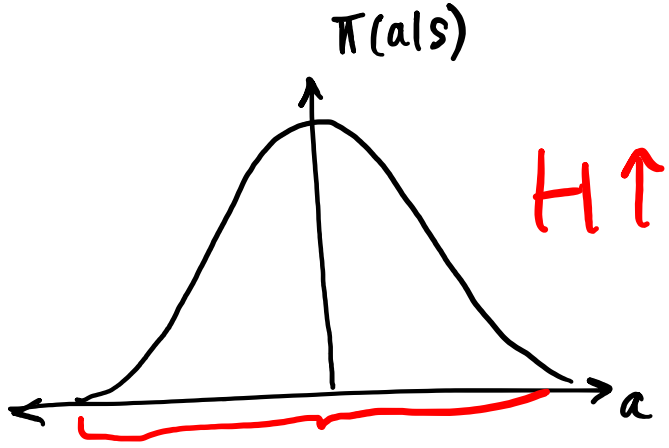
- Stochastic policy

- Policy often collapses into deterministic policy

- After which, no exploration

Exploration of policy gradient





Entropy

$$H(p) = -\sum_x p(x) \log p(x)$$

- We want to discourage policy collapse

- Revised objective

$$J(\theta) = \mathbb{E}_{s,a}\left[Q^\pi(s,a) \log \pi_\theta(a|s) + \beta H(\pi_\theta(s))\right]$$

Policy gradient pseudocode

for until satisfied **do**

collect episode $(s_0, a_0, r_1, \dots)$ using π

$J = \sum_i \gamma^i (g_i \log \pi_\theta(a_i|s_i) + \beta H(\pi_\theta(s_i)))$

$\theta \leftarrow \theta + \alpha \nabla J$

end for

g_i

$H(\cdot)$

Policy gradient is on-policy

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[Q^{\pi}(s,a)\nabla \log \pi_{\theta}(a|s)\right]$$

$$\nabla J(\theta) = \sum_{s} d^{\pi}(s) \sum_{a} \pi(a|s)\left[Q^{\pi}(s,a)\nabla \log \pi_{\theta}(a|s)\right]$$

$\mathbb{E}_{s,a\sim\pi}$

Proof of policy gradient

① $V^{\pi}(s) = \sum_a \pi(a|s)\, Q^{\pi}(s,a)$

$\nabla_\theta V^{\pi}(s) = \sum_a Q \nabla \pi + \pi \nabla Q$

$\nabla Q^{\pi}(s,a) = \nabla \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$

$= \gamma \sum_{s'} p(s'|s,a)\, \nabla V^{\pi}(s')$

$\nabla_\theta V^{\pi}(s) = \sum_a Q \nabla \pi + \gamma \sum_a \pi \sum_{s'} p(s'|s,a)\left[\nabla_\theta V^{\pi}(s')\right]$

$= \sum_a Q \nabla \pi + \gamma \sum_a \pi \sum_{s'} p(s'|s,a) \sum_{a'} Q' \nabla \pi' + \gamma^2 \sum_a \pi \sum_{s'} p(s \to s', 2) \ldots$

$p(s \to s', 1)$

* $J(\theta) = \sum_s P_{s_0}(s) V^{\pi}(s)$

$\nabla_\theta V(s)$

∇J

$\sum_{s'} p(s') \cdot \sum_{s''} p(s'')$

↳ $\sum_{s'} p(s \to s', 2, \pi)$

$\sum Q \nabla \pi + \gamma \sum \pi \sum p \nabla V''$

$$\nabla_\pi V^\pi(s) = \sum_a Q \nabla \pi + \gamma \sum_a \pi \sum_{s'} p(s \to s', 1, \pi) \cdot \sum_{a'} Q' \nabla \pi$$

$$+ \gamma^2 \sum_a \pi \sum_{s'} p(s \to s'', 2, \pi) \sum_{a''} Q'' \nabla \pi''$$

$$+ \gamma^3 \sum_a \sum_{s'} p(s \to s''', 3, \pi) \sum_{a'''} Q''' \nabla \pi'''$$

$$+ \dots$$

$$= \sum_{s'} \left[\sum_{k=0}^{\infty} p(s \to s', k, \pi) \gamma^k \right] \sum_a Q \nabla \pi$$

$$d^\pi(s')$$

$$= \sum_{s'} d^\pi(s') \sum_a Q(s, a) \nabla \pi(a|s)$$

$$J(\theta) = \sum_{s_0} p_{s_0}(s) \uparrow$$

Policy gradient extensions

Variance of policy gradient

- What is the variance?

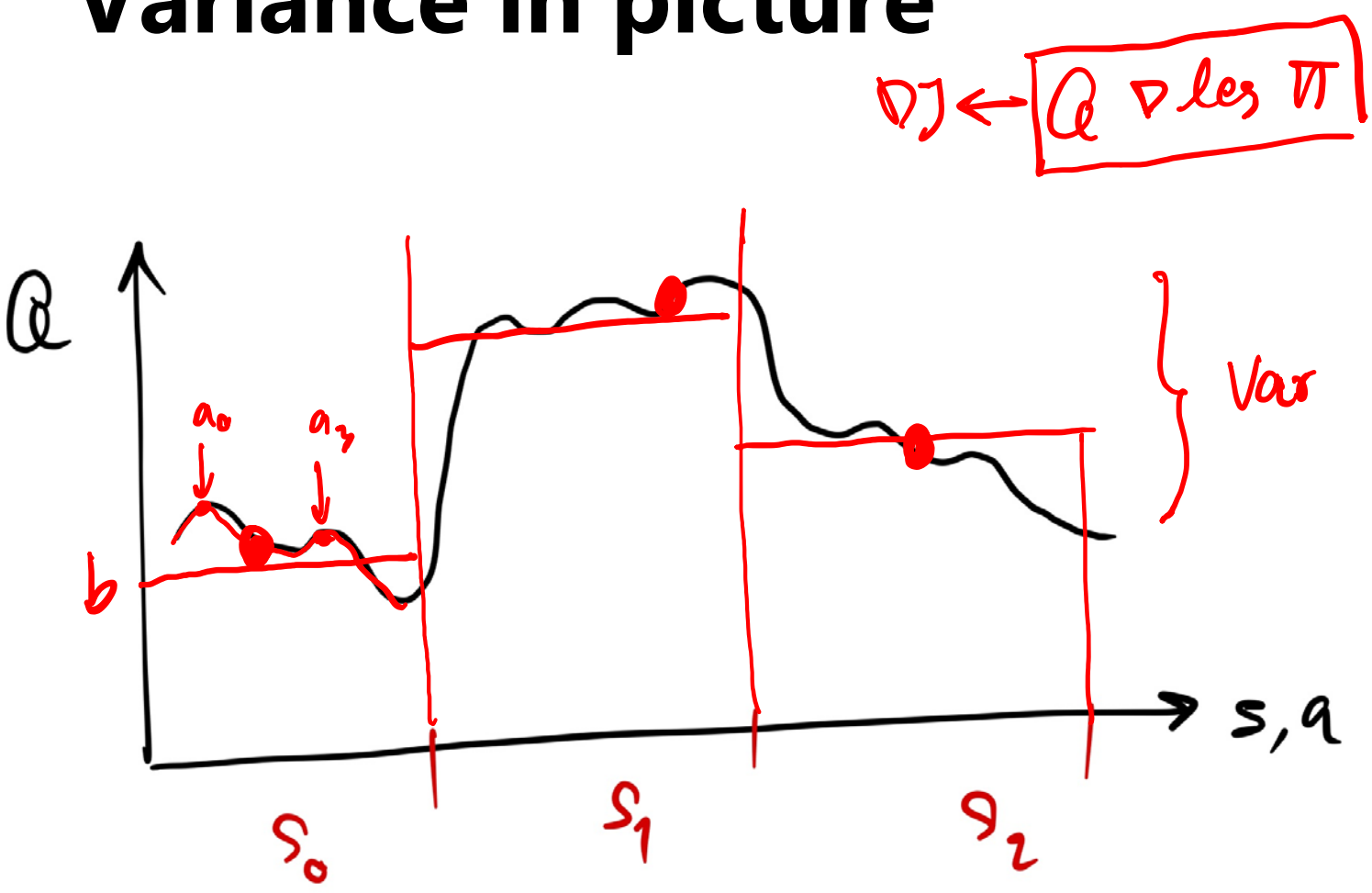
$$\text{Var}(Q) = \mathbb{E}_{s,a}\left[Q(s,a) - \overline{Q}\right]^2$$

- Policy gradient has high variance

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[Q^{\pi}(s,a)\nabla \log \pi_\theta(a|s)\right]$$

- High variance slows down learning!

Variance in picture



Variance reduction

$$f(s,a) = Q(s,a) - b(s)$$

- If b(s) correlates well with Q(s,a)
- f(s,a) could have "lower" variance
- b(s) is called a "baseline"

- **There is no change in gradient! Why?**

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[(Q^{\pi}(s,a) - b(s))\nabla \log \pi_{\theta}(a|s)\right]$$

Baseline

$$\nabla J(\theta) = \mathbb{E}_{s,a}[(Q^{\pi}(s,a) - b(s))\nabla \log \pi_\theta(a|s)]$$

$Q^\pi \nabla \log$

$E\, Q \nabla \log \; - \; E\, b \nabla \log$

$= 0$

$$\hookrightarrow E_s\Big[\sum_a \pi(a|s) \cdot b(s) \cdot \nabla \log \pi(a|s)\Big] \qquad \frac{\nabla \pi}{\pi}$$

$$= E_s\Big[\sum_a \pi \cdot b \cdot \frac{\nabla \pi}{\pi}\Big]$$

$$= E_s\Big[\sum_a b \cdot \nabla \pi\Big]$$

$$= E_s\Big[b \sum_a \nabla \pi\Big]$$

$$= E_s\Big[b \nabla \sum_a \pi(a|s)\Big] \Rightarrow E_s[b \underbrace{\nabla 1}_{0}] = 0$$

Baseline

- What if b(s) becomes b(s, a)?
- Can we use the same argument?

$$b(s) = ?$$

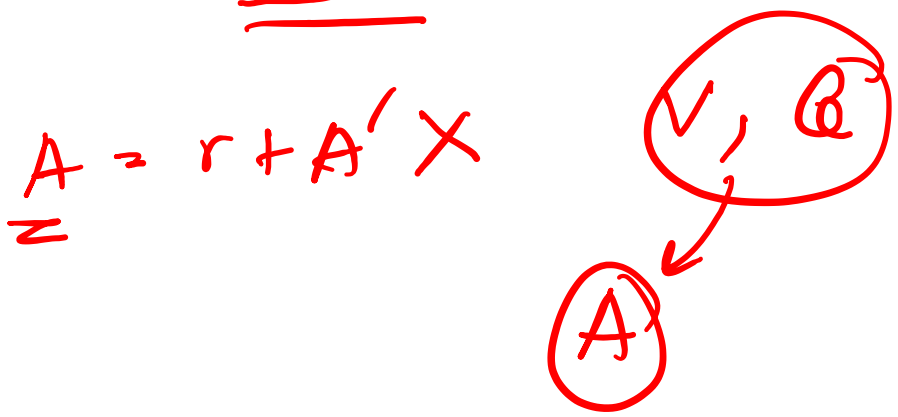
- What is a good baseline for Q(s,a)?

$$\frac{1}{N(a)} \sum_a Q(a, s) \longrightarrow \sum_a Q(s,a)\,\pi(a|s) = V(s)$$

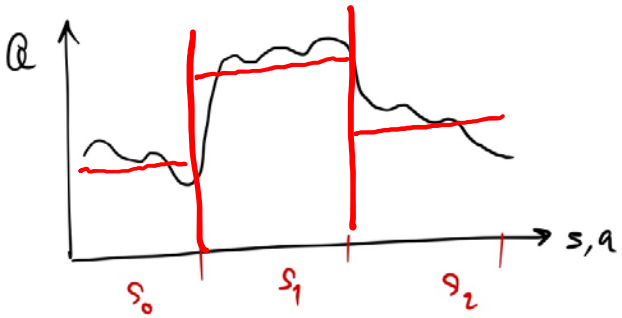
$$E\left[\underbrace{\left(Q(s,a) - V(s)\right)}_{A(s,a)} \cdot \nabla \log \pi\right] \quad \bigstar$$

Advantage function

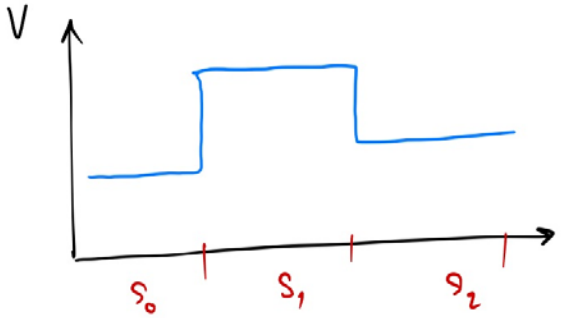
$$A(s,a) = Q(s,a) - V(s)$$

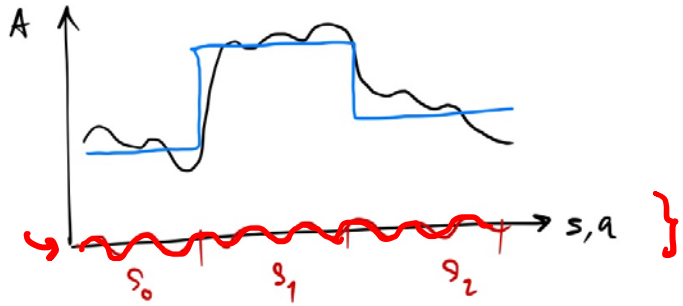


Baseline in action



$A = Q - V$



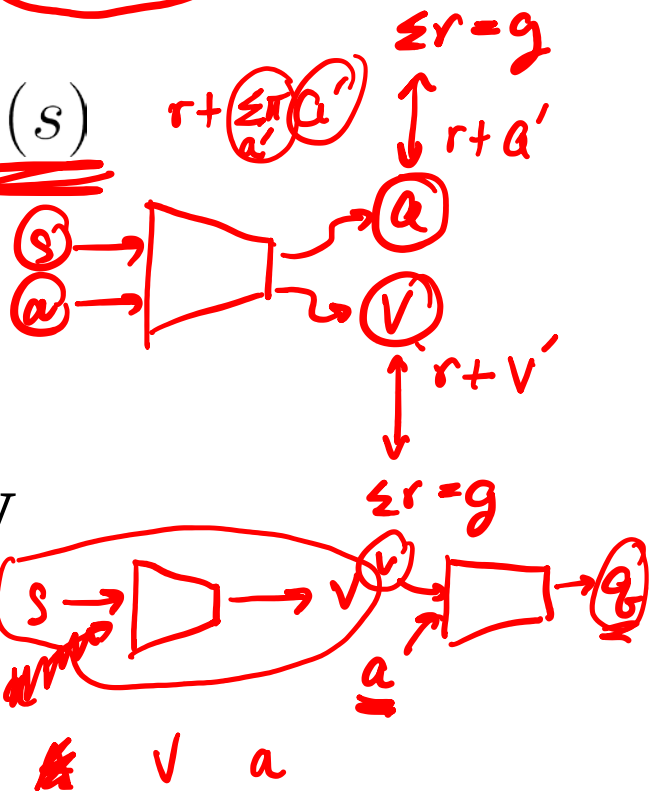


Advantage Actor-critic (A2C)

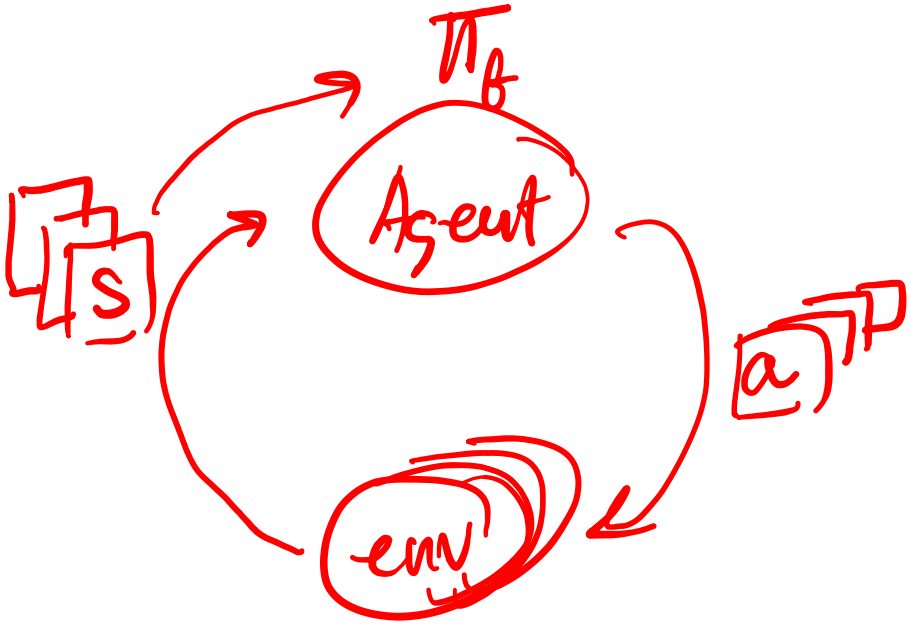
$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[A(s,a)\nabla \log \pi_\theta(a|s)\right]$$

$$A(s,a) = Q(s,a) - V(s)$$

- **Advantage = Critic**
- **Policy = Actor**
- Need to model either A or Q or V
- How to do it efficiently?

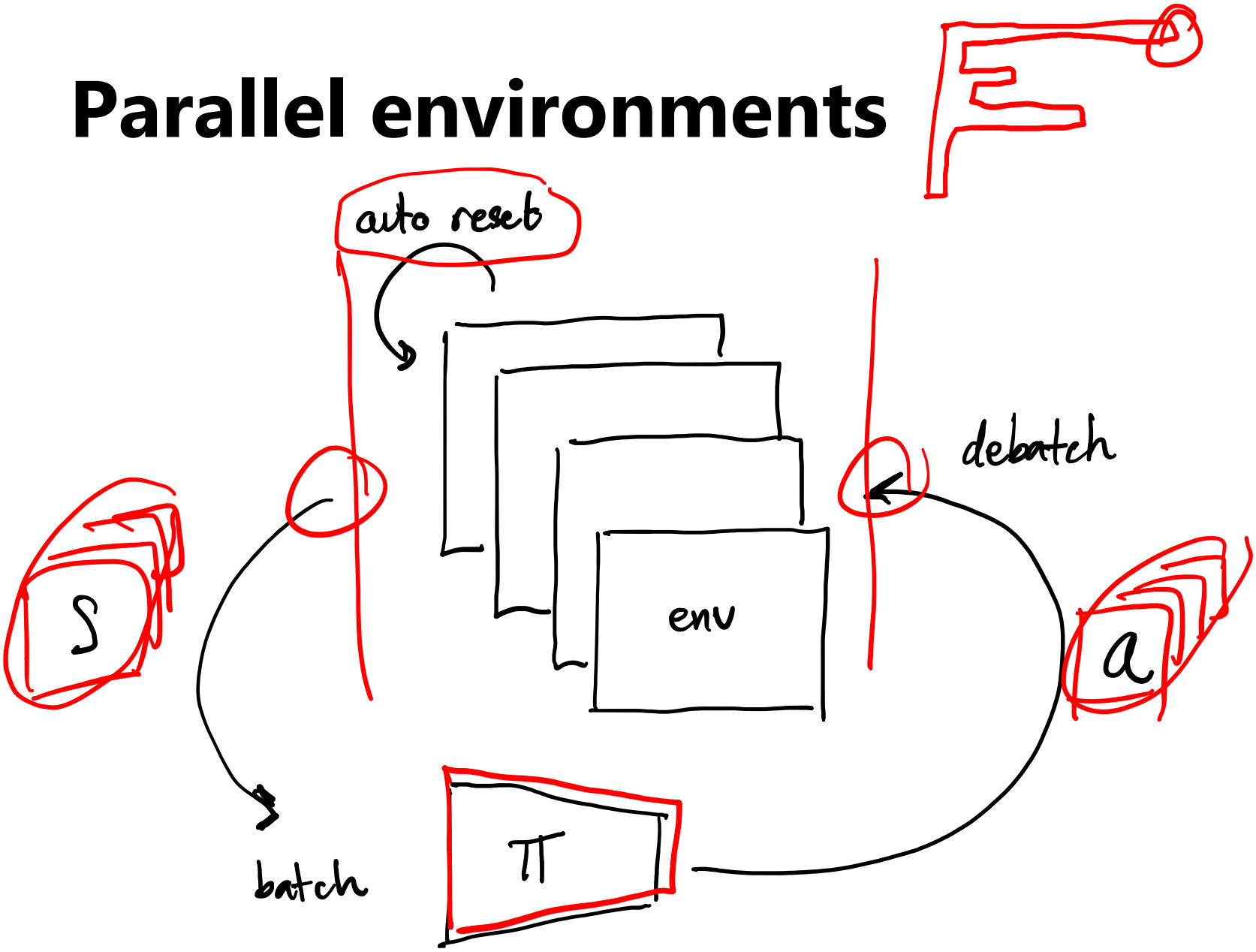


Sutton, Richard S., David Mcallester, Satinder Singh, and Yishay Mansour. 1999. "Policy Gradient Methods for Reinforcement Learning with Function Approximation." *In Advances in Neural Information Processing Systems 12*, 1057–63.

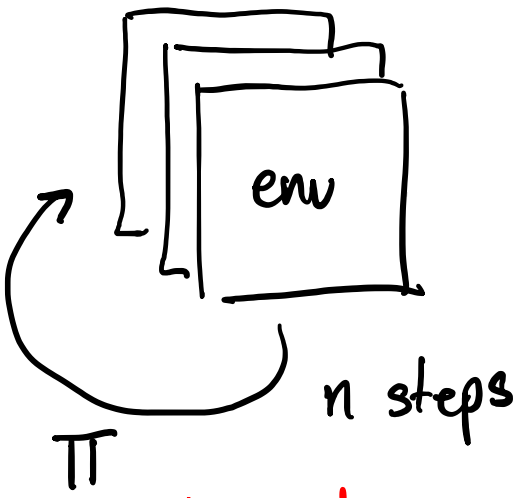


Practical considerations

Parallel environments



N-step exploration



3 step

$$\left[(s, a, r, \text{done})_0, (\dots)_1, (\dots)_2 \right]$$

$$\left[(\dots)_0, (\dots)_1, (\dots)_2 \right]$$

done → false, true

[f f f f t]

[f t f f f]

done = True → s_0

autoreset

A slew of returns

1. $\sum_{t=0}^{\infty} r_t$: total reward of the trajectory.
2. $\sum_{t'=t}^{\infty} r_{t'}$: reward following action a_t.
3. $\sum_{t'=t}^{\infty} r_{t'} - b(s_t)$: baselined version of previous formula.
4. $Q^{\pi}(s_t, a_t)$: state-action value function.
5. $A^{\pi}(s_t, a_t)$: advantage function.
6. $r_t + V^{\pi}(s_{t+1}) - V^{\pi}(s_t)$: TD residual.

N-step TD Residual

$$A_t \approx r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^n V(s_{t+n}) - V(s_t)$$

Mnih, Volodymyr, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. "Asynchronous Methods for Deep Reinforcement Learning." *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1602.01783.

Schulman, John, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. "High-Dimensional Continuous Control Using Generalized Advantage Estimation." *arXiv Preprint arXiv:1506.02438*, 1–14.

Generalized Advantage

- Lambda-weighted infinite sum of many-step advantage functions
- Lambda for advantage functions

$$A_t^{(\gamma,\lambda)} = \sum_{i=0}^{\infty} (\gamma\lambda)^i \delta_{t+i}$$

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$

- We only sum to “n” (n >= 5)

Schulman, John, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. “High-Dimensional Continuous Control Using Generalized Advantage Estimation.” *arXiv Preprint arXiv:1506.02438*, 1–14.

A2C pseudocode + V_ϕ + n-step A

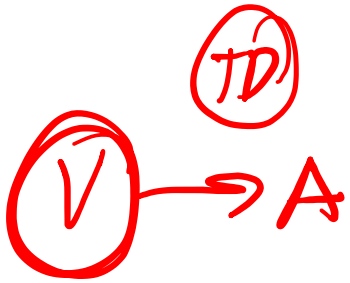
for until satisfied **do**

1. collect n step $(s_0, a_0, r_1, \ldots, s_{n-1}, a_{n-1}, r_n, s_n)$ using π
2. $q_i = \sum_{j=0}^{n} \gamma^j r_{i+j+1} + \gamma^n V_\phi(s_n)$
4. $\nabla J(\theta) = \sum_{i=0}^{n} \gamma^i \left[q_i - V_\phi(s_i)\right] \nabla \log \pi_\theta(a_i|s_i)$
3. $\nabla L(\phi) = \sum_{i=0}^{n} \left[V_\phi(s_i) - q_i\right] \nabla V_\phi(s_i)$
5. $\theta \leftarrow \theta + \alpha_1 \nabla J$
4. $\phi \leftarrow \phi + \alpha_2 \nabla L$

end for

$\alpha_2 > \alpha_1$

Stability of critics



- **Semi-gradient could diverge**
- One-step return might be unstable
- You might need to use target network in such cases
- Usually N > 5, you don't need (but using might give you even more stable training)

Sample efficiency

- Not able to reuse data
- Efficiency is not great

Off-policy is much needed

Off-policy actor and critic

Off-policy actor-critic

Off-policy critic

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[Q_\phi(s,a)\nabla \log \pi_\theta(a|s)\right]$$

- If Q learns from TD (one or many steps)

$Q \leftarrow$ n-step + IS

- How to learn Q off-policy?

$$[r + \mathbb{E}_{a\sim\pi} Q'] - Q$$

- If one step, we could use expected SARSA

Off-policy policy gradient

- Objective function (proposed by the paper)

$$J(\theta) = \sum_s d^b(s) V^\pi(s)$$

- It is not the same as on-policy one!

$$J(\theta) = \sum_s P_{s_0}(s) V^\pi(s)$$

Degris, Thomas, Martha White, and Richard S. Sutton. 2012. “Off-Policy Actor-Critic.” *arXiv [cs.LG]*. arXiv. https://doi.org/10.1.1.385.4471.

Motivation

Replay

$$J(\theta) = \sum_s d^b(s) V^\pi(s)$$

- We have the distribution already ✓
- Reweighting is not that bad

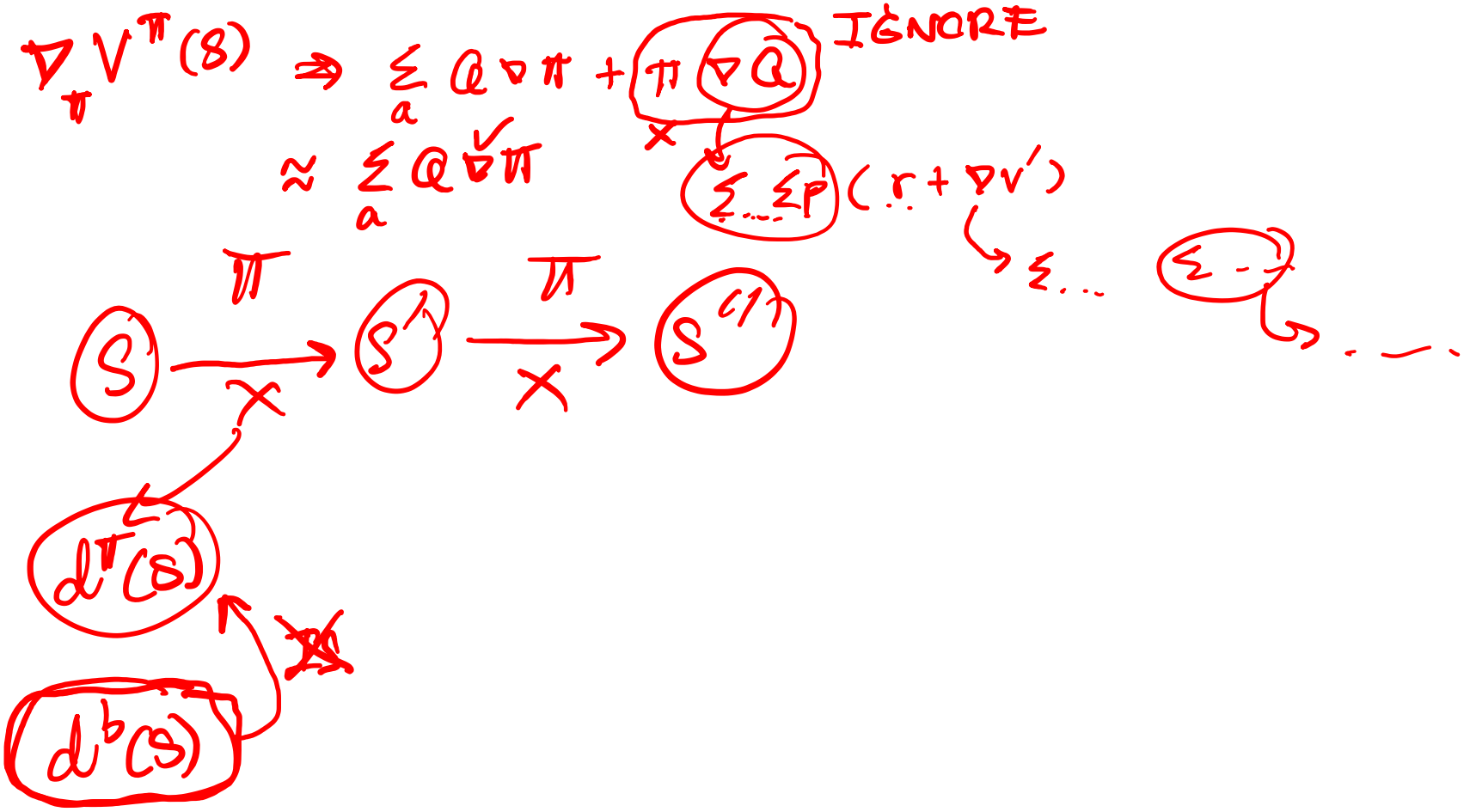
↑ capacity

Off-policy policy gradient

$$\nabla_\theta J(\theta) \approx \sum_s d^b(s) \sum_a Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s)$$

- Why approximation?

Proof of off-policy gradient



Approximation vs Oracle

$$\nabla_\theta J(\theta) = \sum_s d^b(s) \left[\sum_a Q^\pi(s,a) \nabla \pi(a|s) + \pi(a|s) \nabla Q^\pi(s,a) \right]$$

- **Oracle:**
  - Start from S
  - Update for S
  - Unroll, continue on-policy
- **Approximation:**
  - Start from S
  - Update for S

Off-policy gradient

- It gives a “good enough” gradient
- It guarantees to improve the policy
- But the fixed point for policy might not be as good (a local minima is not as good)

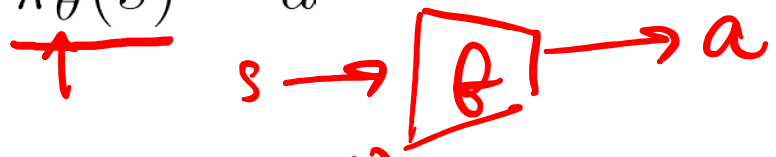
Deterministic policy gradient (DPG)

Motivation

- Deterministic policy simplifies much of the policy gradient
- **Gradient has very low variance!**
- Off-policy is trivial
- Potential efficiency gain

Deterministic Policy Gradient

- Deterministic policy $\pi_\theta(s) = a$



$V = Q(s, \pi(s))$

- Objective function

$$J(\theta) = \sum_s P_{s_0}(s) V^\pi(s) \Rightarrow \sum_s P_{s_0}(s) Q^\pi(s, a)|_{a=\pi(s)}$$

- Deterministic policy gradient

$$\nabla J(\theta) = \sum_s d^\pi(s) \nabla_a Q^\pi(s, a)|_{a=\pi(s)} \nabla_\theta \pi_\theta(s)$$

Policy improvement = backprop on critic

$$\nabla_\theta J(\theta) \approx \sum_s d^\pi(s) \nabla_a Q_\phi(s,a)|_{a=\pi(s)} \nabla_\theta \pi_\theta(s)$$

d^b

$\nabla_a Q(s,\hat{a})|_{a=\pi(s)}$

- We use a neural net as Q (critic)
- Critic "tells" what is a better action
- ✓ Very low variance
  - Single sample can make progress
- ✱ Easy overfit, critic needs to be "ahead"

Needs exploration

- Policy during exploration

$$a = \pi_\theta(s) + \epsilon$$

$\epsilon \sim N(0, \sigma)$

Connection with DQN

- DQN = Explicit max over discrete actions

$$\pi(s) = \underset{a}{\operatorname{argmax}} Q_\phi(s, a)$$

- DPG = Climb to the local max action

$$J(\theta) = \sum_s d^\pi(s) \nabla_a Q_\phi(s, a)|_{a=\pi(s)} \nabla_\theta \pi_\theta(s)$$

Off-policy DPG

- Objective function

$$J(\theta) = \sum_s d^b(s) Q^\pi(s,a)|_{a=\pi(s)}$$

- Gradient

$$J(\theta) \approx \sum_s d^b(s) \nabla_a Q^\pi(s,a)|_{a=\pi(s)} \nabla_\theta \pi_\theta(s)$$

- Same argument with off-policy gradient

Off-policy critic

$r + Q'(s', a)$

- Deterministic policy could evaluate from off-policy using SARSA-like
- (s, a, r, s')

$$\delta = \left(r + \gamma Q_\phi(s', \pi(s'))\right) - Q_\phi(s, a)$$

$$L(\phi) = \frac{1}{2}\delta^2$$

$$\nabla_\phi L(\phi) = -\delta \nabla_\phi Q_\phi(s, a)$$

Off-policy

Proof of DPG

$$J(\theta) = \sum_s d^\pi(s) \nabla_a Q^\pi(s,a)|_{a=\pi(s)} \nabla_\theta \pi_\theta(s)$$

Deterministic policy gradient extensions

Deep Deterministic Policy Gradient (DDPG) ← DQN

- DPG uses one-step bootstrapping which is unstable
- DDPG introduces:
  - ✓ Target network both actor and critic
  - ✓ Replay
- DDPG = DQN for continuous control

Lillicrap, Timothy P., Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. "Continuous Control with Deep Reinforcement Learning." *arXiv [cs.LG]*. arXiv. https://doi.org/10.1561/2200000006.

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
1. Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
  Execute action a_t and observe reward r_t and observe new state s_{t+1}
  Store transition (s_t, a_t, r_t, s_{t+1}) in R
  Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
2. Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
3. Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
  Update the actor policy using the sampled policy gradient:

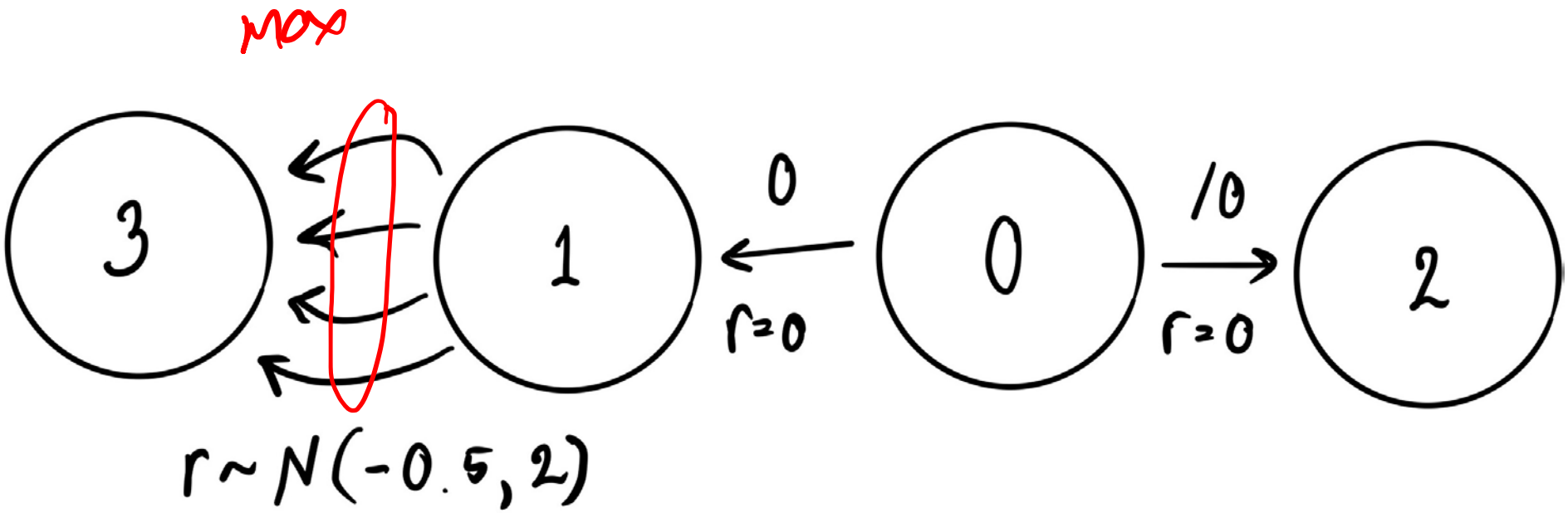
4.
$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$$

  Update the target networks:

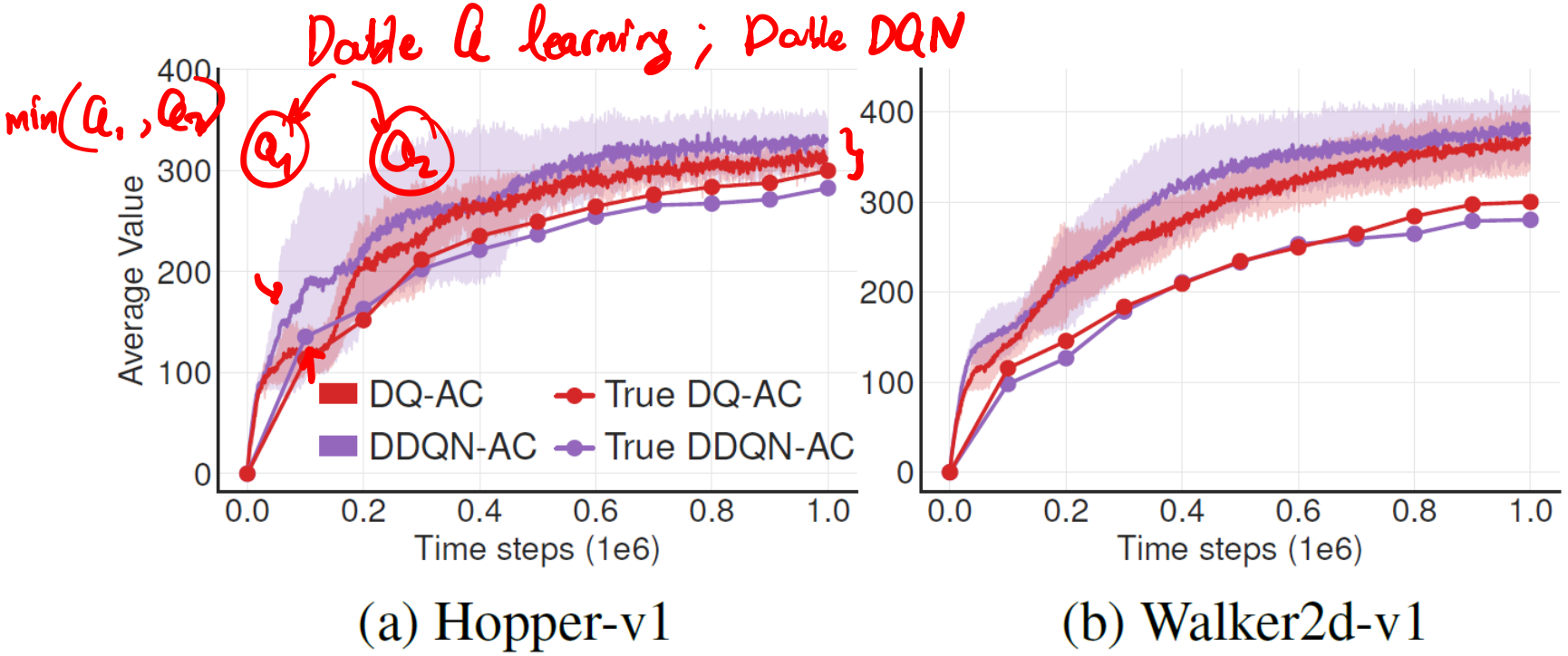
5.
$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

Maximization bias



Maximization bias in DDPG



(a) Hopper-v1 (b) Walker2d-v1

- Double DQN doesn't work as well
- Because the policy slowly changes, the value functions are not "independent" enough

Fujimoto, Scott, Herke van Hoof, and David Meger. 2018. "Addressing Function Approximation Error in Actor-Critic Methods." *arXiv [cs.AI]*. arXiv. https://doi.org/10.1089/lrb.2012.0022.

MISC.

Action dependent baseline

- PG has high variance because it uses “indirect gradient”

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

- DPG has lower variance because it can “backprop”

$$\nabla J(\theta) = \sum_{s} d^{\pi}(s)\nabla_{a}Q_{\phi}(s,a)|_{a=\pi(s)}\nabla_{\theta}\pi_{\theta}(s)$$

Action dependent baseline

- Can we combine the two?
- Q-Prop

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[(Q^{\pi}(s,a) - \overline{Q}(s,a))\nabla_{\theta}\log\pi_{\theta}(a|s)\right] + \mathbb{E}_{s}\left[\nabla_{a}Q(s,a)|_{a=\pi(s)}\nabla_{\theta}\pi_{\theta}(s)\right]$$

- Taylor expansion (first order)

$$\overline{Q}(s,a) = Q(s,\pi(s)) + \nabla_{a}Q(s,\pi(s))(a-\pi(s))$$

Gu, Shixiang, Timothy Lillicrap, Zoubin Ghahramani, Richard E. Turner, and Sergey Levine. 2016. “Q-Prop: Sample-Efficient Policy Gradient with An Off-Policy Critic.” *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1611.02247.

Policy gradient from minimizing KL

- If we look at Q as "unnormalized" policy
  - A little bit sharper of Q is exp(Q)
  - This is our target policy
- We could use a KL:

$$\pi = \underset{\pi \in \Pi}{\operatorname{argmin}} \, \mathrm{D}_{KL} \left(\pi(\cdot|s) \middle\| \frac{\exp(Q(\cdot, s))}{Z} \right)$$

- Minimizing KL is an optimization task

Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor." *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1801.01290.

Policy gradient from minimizing KL

- KL policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_s \left[\nabla_\theta \mathrm{D}_{KL} \left(\pi(\cdot|s) \middle\| \frac{\exp(Q(\cdot, s))}{Z} \right) \right]$$

- Z is a constant, ignored
- Policy improve to Q
- Policy eval: Q gets even sharper
- Repeat

Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor.” *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1801.01290.

Assignments

- Pull chula_rl (beware of conflicts; back up)
- A2C with continuous Cartpole
- DDPG with continuous Cartpole