



Boulder

# Reinforcement Learning; Simulated Environments



[YouTube Playlist](#)

**Maziar Raissi**

**Assistant Professor**

Department of Applied Mathematics

University of Colorado Boulder

[maziar.raissi@colorado.edu](mailto:maziar.raissi@colorado.edu)



Boulder

# Continuous Control with Deep Reinforcement Learning



[YouTube Playlist](#)

human arm  $\rightarrow$  7 degrees of freedom

$$a_i \in \{-k, 0, k\}, i = 1, \dots, 7$$

$3^7 = 2187 \rightarrow$  dimensionality of action space

## Background

$\mathcal{E}$   $\rightarrow$  environment (maybe stochastic)

$t \rightarrow$  time step

$x_t \rightarrow$  observation

$a_t \rightarrow$  action

$r_t \rightarrow$  reward

$s_t = x_1, a_1, x_2, a_2, \dots, a_{t-1}, x_t \rightarrow$  state  
 ↳ partially observed environment

$s_t = x_t \rightarrow$  fully observed environment

$\pi : \mathcal{S} \rightarrow \mathcal{P}(\mathcal{A})$  set of legal actions  
 ↳ states  
 policy

$p(s_1) \rightarrow$  initial state distribution

$\{p(s_{t+1}|s_t, a_t)\} \rightarrow$  transition dynamics

$\{r(s_t, a_t)\} \rightarrow$  reward function

MDP: Markov Decision Process

$$R_t = \sum_{i=t}^T \gamma^{i-t} r(s_i, a_i) \rightarrow \text{future discounted return}$$

$$J = \mathbb{E}_{r_i, s_i \sim \mathcal{E}, a_i \sim \pi}[R_1]$$

$\rho^\pi \rightarrow$  discounted state visitation distribution for a policy  $\pi$

$$Q^\pi(s_t, a_t) = \mathbb{E}_{r_i \geq t, s_i > t \sim \mathcal{E}, a_i > t \sim \pi}[R_t | s_t, a_t]$$

↳ action-value function  
 expected return after taking action  $a_t$  in state  $s_t$  and thereafter following policy  $\pi$

$$Q^\pi(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim \mathcal{E}} [r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi}[Q^\pi(s_{t+1}, a_{t+1})]]$$

↳ Bellman Equation

↳ if deterministic target policy

$$Q^\pi(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim \mathcal{E}} [r(s_t, a_t) + \gamma Q^\pi(s_{t+1}, a_{t+1})]$$

It is therefore possible to learn  $Q^\pi$  off-policy

$\beta \rightarrow$  stochastic behavior policy

$$\mu(s) = \arg \max_a Q(s, a) \rightarrow \text{greedy policy}$$

$$L(\theta^Q) = \mathbb{E}_{s_t \sim \rho^\beta, a_t \sim \beta, r_t \sim \mathcal{E}} [(Q(s_t, a_t | \theta^Q) - y_t)^2]$$

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1}) | \theta^Q)$$

\* replay buffer

\* separate target network

$\mu(s | \theta^\mu) \rightarrow$  actor function

$Q(s, a) \rightarrow$  critic (Q-Learning & Bellman Equation)

$$\nabla_{\theta^\mu} J \approx \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_{\theta^\mu} Q(s, a | \theta^Q) |_{s=s_t, a=\mu(s_t | \theta^\mu)}]$$

$$= \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_a Q(s, a | \theta^Q) |_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu} \mu(s | \theta^\mu) |_{s=s_t}]$$

$$L(\theta^\mu) = -\mathbb{E}_{s_t \sim \rho^\beta} [Q(s_t, \underbrace{\mu(s_t | \theta^\mu)}_{a_t} | \theta^Q)]$$

Deterministic Policy Gradient (DPG) algorithm

### Algorithm 1 DDPG algorithm

Randomly initialize critic network  $Q(s, a | \theta^Q)$  and actor  $\mu(s | \theta^\mu)$  with weights  $\theta^Q$  and  $\theta^\mu$ .

Initialize target network  $Q'$  and  $\mu'$  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

        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  $R$

        Set  $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} | \theta^{\mu'}) | \theta^{Q'})$

        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:

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

    Update the target networks:

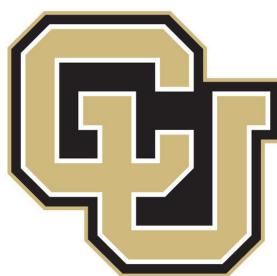
$$\theta^{Q'} \leftarrow \tau \theta^Q + (1 - \tau) \theta^{Q'}$$

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

    end for

end for

<https://www.youtube.com/watch?v=tJB1qkC1wWM&feature=youtu.be>



Boulder



[YouTube Playlist](#)

$(\mathcal{S}, \mathcal{A}, P, c, \rho_0, \gamma) \rightarrow$  infinite-horizon discounted Markov Decision Process (MDP)

$\mathcal{S} \rightarrow$  finite set of states

$\mathcal{A} \rightarrow$  finite set of actions

$P \rightarrow$  transition probability distribution

$P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$

$c \rightarrow$  cost function

$c : \mathcal{S} \rightarrow \mathbb{R}$

$\rho_0 \rightarrow$  distribution of initial state  $s_0$

$\rho_0 : \mathcal{S} \rightarrow \mathbb{R}$

$\gamma \in (0, 1) \rightarrow$  discount factor

$\pi \rightarrow$  stochastic policy

$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$

$\eta(\pi) \rightarrow$  expected discounted cost

$\eta(\pi) = \mathbb{E}_{s_0, a_0, \dots} \left[ \sum_{t=0}^{\infty} \gamma^t c(s_t) \right]$

$s_0 \sim \rho_0(s_0), a_t \sim \pi(a_t | s_t), s_{t+1} \sim P(s_{t+1} | s_t, a_t)$

$Q_\pi(s_t, a_t) \rightarrow$  state-action value function

$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \dots} \left[ \sum_{l=0}^{\infty} \gamma^l c(s_{t+l}) \right]$

$V_\pi(s_t) \rightarrow$  value function

$V_\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \dots} \left[ \sum_{l=0}^{\infty} \gamma^l c(s_{t+l}) \right]$

$A_\pi(s_t, a_t) \rightarrow$  advantage function

$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t)$

# Trust Region Policy Optimization

$\rho_\pi(s) \rightarrow$  (unnormalized) discounted visitation frequencies

$\rho_\pi(s) = (P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P(s_2 = s) + \dots)$

where  $s_0 \sim \rho_0$  and the actions are chosen according to  $\pi$

**Algorithm 1** Approximate policy iteration algorithm guaranteeing non-increasing expected cost  $\eta$

Initialize  $\pi_0$ . See the paper for proof!

**for**  $i = 0, 1, 2, \dots$  until convergence **do**

    Compute all advantage values  $A_{\pi_i}(s, a)$ .

    Solve the constrained optimization problem

$$\pi_{i+1} = \arg \min_{\pi} \left[ L_{\pi_i}(\pi) + \underbrace{\left( \frac{2\epsilon\gamma}{(1-\gamma)^2} \right)}_{C} D_{\text{KL}}^{\max}(\pi_i, \pi) \right]$$

    where  $\epsilon = \max_s \max_a |A_\pi(s, a)|$

    and  $L_{\pi_i}(\pi) = \eta(\pi_i) + \sum_s \rho_{\pi_i}(s) \sum_a \pi(a|s) A_{\pi_i}(s, a)$

**end for**  $D_{\text{KL}}^{\max}(\pi, \tilde{\pi}) = \max_s D_{\text{KL}}(\pi(\cdot|s) \parallel \tilde{\pi}(\cdot|s))$

In practice, if we used the penalty coefficient  $C$  above recommended by the theory, the step sizes would be very small.

$\eta(\theta) := \eta(\pi_\theta)$

$L_\theta(\tilde{\theta}) := L_{\pi_\theta}(\pi_{\tilde{\theta}})$ , and  $D_{\text{KL}}(\theta \parallel \tilde{\theta}) := D_{\text{KL}}(\pi_\theta \parallel \pi_{\tilde{\theta}})$

$\overline{D}_{\text{KL}}^\rho(\theta_1, \theta_2) := \mathbb{E}_{s \sim \rho} [D_{\text{KL}}(\pi_{\theta_1}(\cdot|s) \parallel \pi_{\theta_2}(\cdot|s))]$

$\underset{\theta}{\text{minimize}} L_{\theta_{\text{old}}}(\theta)$  trust region constraint

subject to  $\overline{D}_{\text{KL}}^{\rho_{\theta_{\text{old}}}}(\theta_{\text{old}}, \theta) \leq \delta$ .

$\underset{\theta}{\text{minimize}} \sum_s \rho_{\theta_{\text{old}}}(s) \sum_a \pi_\theta(a|s) A_{\theta_{\text{old}}}(s, a)$

subject to  $\overline{D}_{\text{KL}}^{\rho_{\theta_{\text{old}}}}(\theta_{\text{old}}, \theta) \leq \delta$ .

$\sum_s \rho_{\theta_{\text{old}}}(s) [\dots] = \frac{1}{1-\gamma} \mathbb{E}_{s \sim \rho_{\theta_{\text{old}}}} [\dots]$

$\sum_a \pi_\theta(a|s_n) A_{\theta_{\text{old}}}(s_n, a) =$

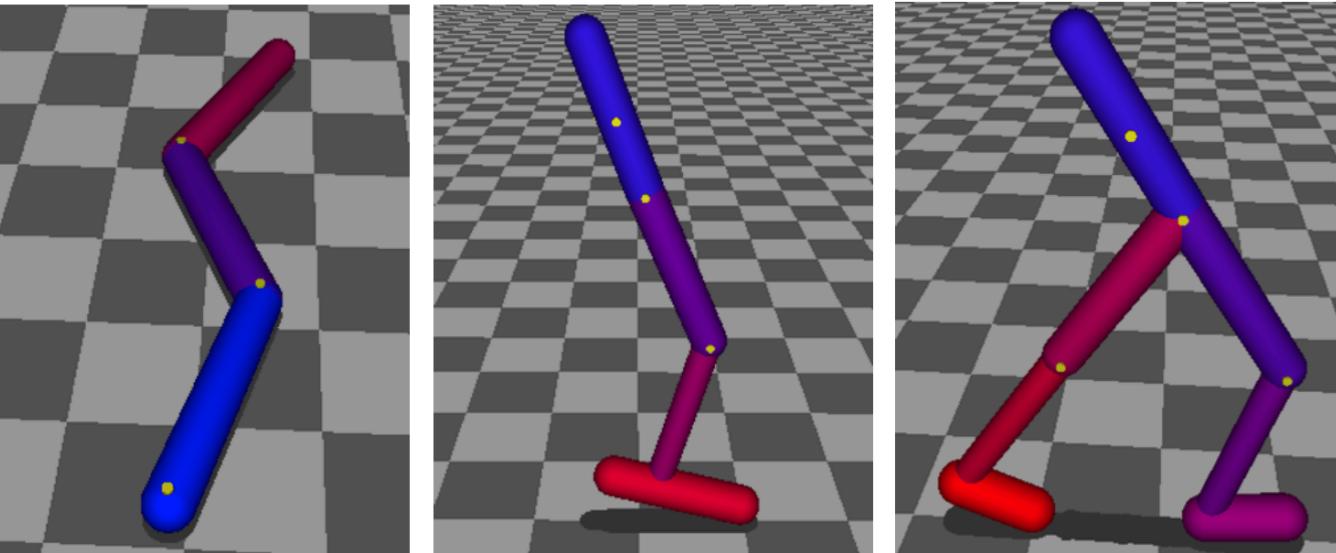
$a \text{ importance sampling } \mathbb{E}_{a \sim q} \left[ \frac{\pi_\theta(a|s_n)}{q(a|s_n)} A_{\theta_{\text{old}}}(s_n, a) \right]$

Replace advantage values by Q-values which only changes the objective by a constant.

$\underset{\theta}{\text{minimize}} \mathbb{E}_{s \sim \rho_{\theta_{\text{old}}}, a \sim q} \left[ \frac{\pi_\theta(a|s)}{q(a|s)} Q_{\theta_{\text{old}}}(s, a) \right]$

subject to  $\mathbb{E}_{s \sim \rho_{\theta_{\text{old}}}} [D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s) \parallel \pi_\theta(\cdot|s))] \leq \delta$ .

$q(a|s) = \pi_{\theta_{\text{old}}}(a|s)$  TRPO





Boulder



[YouTube Playlist](#)

# Conjugate Gradient Method

TRPO objective function

$$\pi_{k+1} = \arg \max_{\pi'} \mathcal{L}_{\pi_k}(\pi') \text{ s.t. } \bar{D}_{KL}(\pi' || \pi_k) \leq \delta$$

$$\mathcal{L}_{\theta_k}(\theta) \approx g^T (\theta - \theta_k)$$

$$\bar{D}_{KL}(\theta || \theta_k) \approx \frac{1}{2} (\theta - \theta_k)^T H (\theta - \theta_k)$$

$$\arg \min_x f(x) = \frac{1}{2} x^T H x - x^T g$$

$$\nabla f(x) = Hx - g$$

Conjugate gradient method

$$r_0 = g - Hx_0 = -\nabla f(x_0)$$

$$p_0 = r_0$$

$$x_1 = x_0 + \alpha_0 p_0$$

$$f(x_1) = f(x_0 + \alpha_0 p_0) =: g(\alpha_0)$$

$$g'(\alpha_0) = 0 \implies \alpha_0 = \frac{p_0^T (g - Hx_0)}{p_0^T H p_0}$$

$$r_1 = g - Hx_1 = -\nabla f(x_1)$$

$$p_1 = r_1 - \frac{p_0^T H r_1}{p_0^T H p_0} p_0 \rightarrow \text{Gram-Schmidt orthonormalization}$$

$$p_0^T H p_1 = p_0^T H r_1 - \frac{p_0^T H r_1}{p_0^T H p_0} p_0^T H p_0 = 0$$

$\implies p_0$  and  $p_1$  are conjugate w.r.t.  $H$ .

$$x_2 = x_1 + \alpha_1 p_1$$

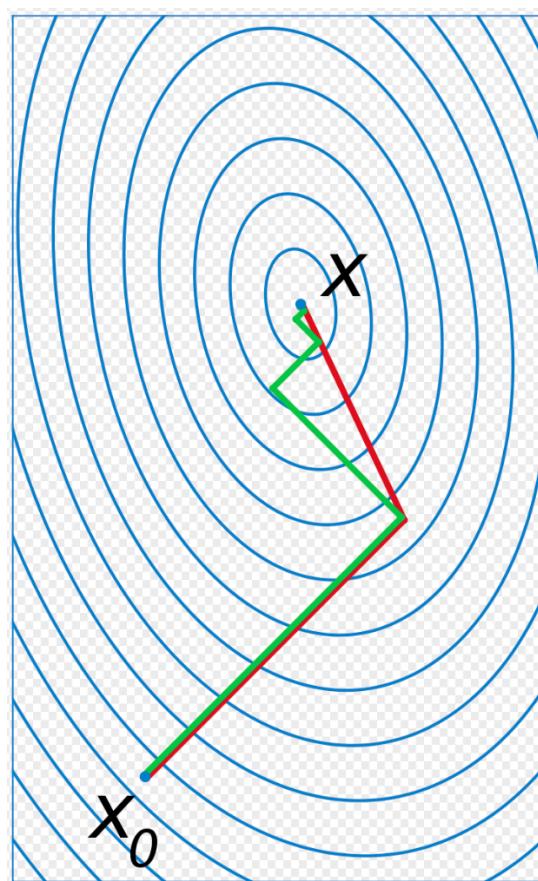
$$\alpha_1 = \frac{p_1^T (g - Hx_1)}{p_1^T H p_1}$$

Linear Approximation

$$\mathcal{L}_{\theta_k}(\theta) \approx \mathcal{L}_{\theta_k}(\theta_k) + g^T (\theta - \theta_k)$$

$$g \doteq \nabla_{\theta} \mathcal{L}_{\theta_k}(\theta) |_{\theta_k}$$

$$H \doteq \nabla_{\theta}^2 \bar{D}_{KL}(\theta || \theta_k) |_{\theta_k}$$



Quadratic Approximation

$$\bar{D}_{KL}(\theta || \theta_k) \approx \bar{D}_{KL}(\theta_k || \theta_k) + \nabla_{\theta} \bar{D}_{KL}(\theta || \theta_k) |_{\theta_k} (\theta - \theta_k) + \frac{1}{2} (\theta - \theta_k)^T H (\theta - \theta_k)$$

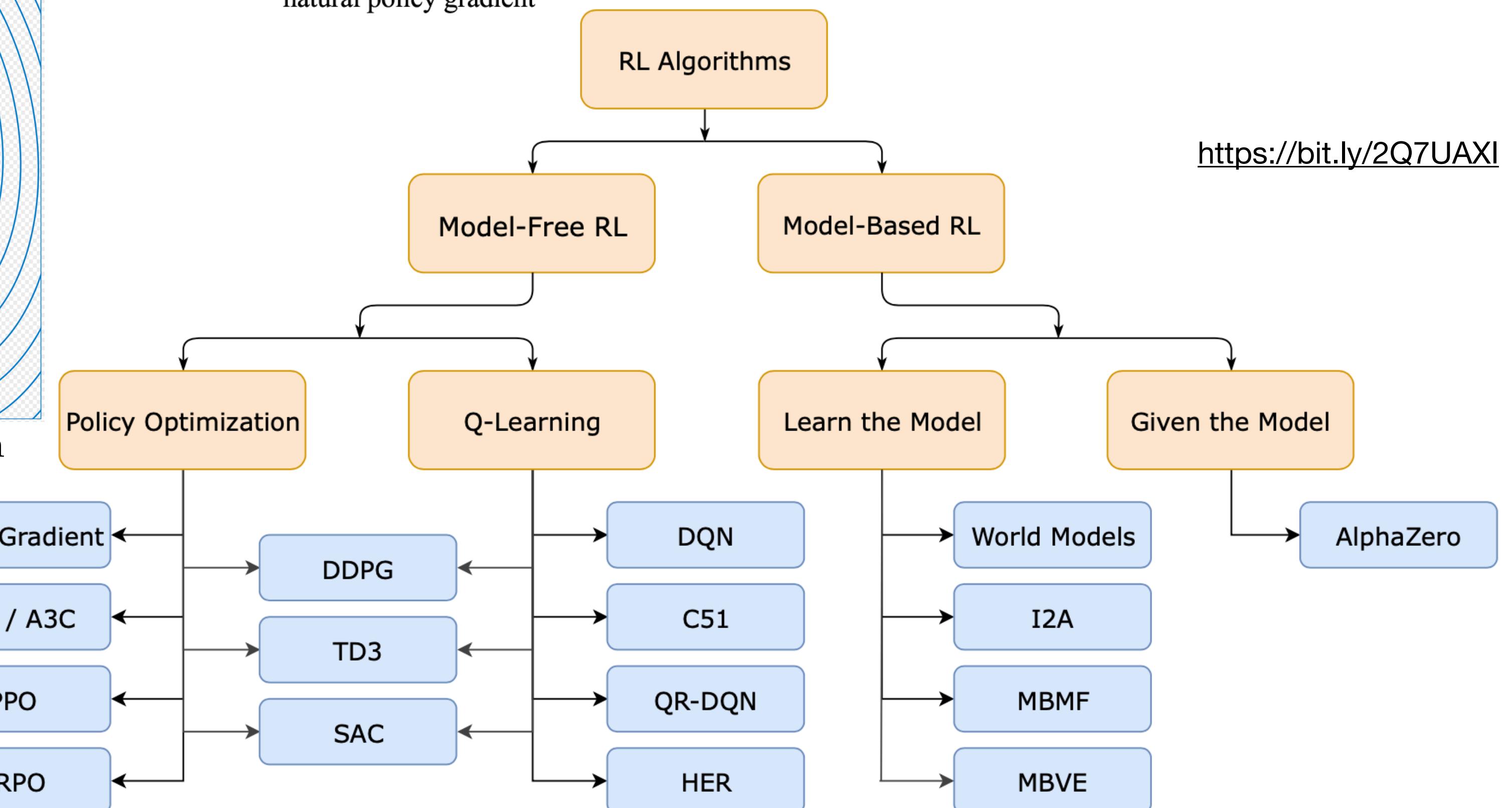
$$\theta_{k+1} = \arg \max_{\theta} g^T (\theta - \theta_k) \text{ s.t. } \frac{1}{2} (\theta - \theta_k)^T H (\theta - \theta_k) \leq \delta$$

$$H^{-1} g = x \implies Hx = g$$

$$Hx = \nabla_{\theta} \left( (\nabla_{\theta} \bar{D}_{KL}(\theta || \theta_k))^T x \right)$$

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g$$

natural policy gradient



<https://bit.ly/2Q7UAXI>



Boulder

# Asynchronous Methods for Deep Reinforcement Learning



[YouTube Video](#)

value-based model-free methods

$Q(s, a; \theta) \rightarrow$  approximate action-value function

$Q^*(s, a) \approx Q(s, a; \theta) \rightarrow$  Q-Learning

$$L_i(\theta_i) = \mathbb{E}[(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i))^2]$$

one-step return

$$r_t + \gamma r_{t+1} + \dots + \gamma^{n-1} r_{t+n-1} + \max_a \gamma^n Q(s_{t+n}, a)$$

n-step return

A single reward directly affects the values of  $n$  preceding state action pairs

policy-based model-free methods

$$\pi(a|s; \theta)$$

gradient ascent on  $\mathbb{E}[R_t]$

$$\underbrace{\nabla_\theta \log \pi(a_t|s_t; \theta) R_t}_{\text{as an unbiased estimate of } \nabla_\theta \mathbb{E}[R_t]} \rightarrow \text{REINFORCE}$$

as an unbiased estimate of  $\nabla_\theta \mathbb{E}[R_t]$

$$\nabla_\theta \log \pi(a_t|s_t; \theta) (R_t - \underbrace{b_t(s_t)}_{\text{baseline}})$$

$$b_t(s_t) \approx V^\pi(s_t)$$

$R_t - b_t \approx$  advantage of action  $a_t$  in state  $s_t$

$$A(a_t, s_t) = \underbrace{Q(a_t, s_t)}_{\approx R_t} - \underbrace{V(s_t)}_{\approx b_t}$$

$\underbrace{\pi}_{\text{actor}} - \underbrace{b}_{\text{critic}}$

Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." *International conference on machine learning*. 2016.

Asynchronous Advantage Actor-Critic (A3C)

**Algorithm** Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

// Assume global shared parameter vectors  $\theta$  and  $\theta_v$ , and global shared counter  $T = 0$

// Assume thread-specific parameter vectors  $\theta'$  and  $\theta'_v$

Initialize thread step counter  $t \leftarrow 1$

<https://www.youtube.com/watch?v=0xo1Ldx3L5Q&feature=youtu.be>

**repeat**

    Reset gradients:  $d\theta \leftarrow 0$  and  $d\theta_v \leftarrow 0$ .

    Synchronize thread-specific parameters  $\theta' = \theta$  and  $\theta'_v = \theta_v$

$t_{start} = t$

    Get state  $s_t$

<https://www.youtube.com/watch?v=Ajjc08-iPx8&feature=youtu.be>

**repeat**

        Perform  $a_t$  according to policy  $\pi(a_t|s_t; \theta')$

        Receive reward  $r_t$  and new state  $s_{t+1}$

$t \leftarrow t + 1$

$T \leftarrow T + 1$

<https://www.youtube.com/watch?v=nMR5mjCFZCw&feature=youtu.be>

**until** terminal  $s_t$  **or**  $t - t_{start} == t_{max}$

$$R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{for non-terminal } s_t // \text{Bootstrap from last state} \end{cases}$$

**for**  $i \in \{t - 1, \dots, t_{start}\}$  **do**

$$R \leftarrow r_i + \gamma R$$

Accumulate gradients wrt  $\theta'$ :  $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i; \theta') (R - V(s_i; \theta'_v))$

Accumulate gradients wrt  $\theta'_v$ :  $d\theta_v \leftarrow d\theta_v + \partial (R - V(s_i; \theta'_v))^2 / \partial \theta'_v$

**end for**

Perform asynchronous update of  $\theta$  using  $d\theta$  and of  $\theta_v$  using  $d\theta_v$ .

**until**  $T > T_{max}$

$$\nabla_\theta \log \pi(a_t|s_t; \theta) \quad \underbrace{A(s_t, a_t; \theta, \theta_v)}_{\sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k}; \theta_v) - V(s_t; \theta_v)} \quad + \beta \nabla_\theta \underbrace{H(\pi(s_t; \theta))}_{\text{entropy}}$$



Boulder

# Proximal Policy Optimization Algorithms


[YouTube Video](#)

## Policy Gradient Methods

$$\hat{g} = \hat{\mathbb{E}}_t [\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t] \rightarrow \text{policy gradient estimator}$$

policy
estimator of the advantage function at timestep  $t$   
empirical average over a finite batch of samples

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t [\log \pi_{\theta}(a_t | s_t) \hat{A}_t]$$

## Trust Region Methods

$$\max_{\theta} \hat{\mathbb{E}}_t \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t \right] \rightarrow \text{linear approximation}$$

conjugate gradient algorithm

s.t.  $\hat{\mathbb{E}}_t [\text{KL}(\pi_{\theta_{\text{old}}}(.|s_t) \| \pi_{\theta}(.|s_t))] \leq \delta \rightarrow \text{quadratic approximation}$

$$\max_{\theta} \hat{\mathbb{E}}_t \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t - \beta \text{KL}(\pi_{\theta_{\text{old}}}(.|s_t) \| \pi_{\theta}(.|s_t)) \right]$$

## Clipped Surrogate Objective

$$r_t(\theta) := \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \implies r_t(\theta_{\text{old}}) = 1$$

prob. ratio

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t \right] = \hat{\mathbb{E}}_t [r_t(\theta) \hat{A}_t]$$

CPI: Conservative Policy Iteration

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[ \min \underbrace{(r_t(\theta) \hat{A}_t)}_{L^{CPI}}, \underbrace{\text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t}_{\text{removes the incentive for moving } r_t \text{ outside of the interval } [1 - \epsilon, 1 + \epsilon]} \right]$$

pessimistic lower bound

$$L_t^{CLIP+VF+S}(\theta) = \hat{\mathbb{E}}_t [L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_{\theta}](s_t)]$$

$(V_{\theta}(s_t) - V_t^{\text{targ}})^2$  entropy bonus

$$\hat{A}_t = -V(s_t) + r_t + \gamma r_{t+1} + \dots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(s_T)$$

$$\hat{A}_t = \delta_t + (\gamma \lambda) \delta_{t+1} + \dots + (\gamma \lambda)^{T-t+1} \delta_{T-1},$$

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

same as above if  $\lambda = 1$

## Algorithm 1 PPO, Actor-Critic Style

```

for iteration=1,2,... do
    for actor=1,2,...,N do
        Run policy  $\pi_{\theta_{\text{old}}}$  in environment for  $T$  timesteps
        Compute advantage estimates  $\hat{A}_1, \dots, \hat{A}_T$ 
    end for
    Optimize surrogate  $L$  wrt  $\theta$ , with  $K$  epochs and minibatch size  $M \leq NT$ 
     $\theta_{\text{old}} \leftarrow \theta$ 
end for

```

# Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor


[YouTube Video](#)

Model-free Deep Reinforcement Learning

- very high sample complexity
- brittle convergence properties (meticulous hyper-parameter tuning)

**Continuous state and action spaces**

$(\mathcal{S}, \mathcal{A}, p, r) \rightarrow$  infinite-horizon Markov decision process (MDP)

$\mathcal{S} \rightarrow$  state space (continuous)

$\mathcal{A} \rightarrow$  action space (continuous)

$p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, \infty)$

unknown state transition probability

probability density of the next state  $s_{t+1} \in \mathcal{S}$

given the current state  $s_t \in \mathcal{S}$  and action  $a_t \in \mathcal{A}$

$r : \mathcal{S} \times \mathcal{A} \rightarrow [r_{\min}, r_{\max}]$

reward emitted on each transition

$\rho_\pi(s_t) \rightarrow$  state marginal of the trajectory distribution induced by a policy  $\pi(a_t|s_t)$

$\rho_\pi(s_t, a_t) \rightarrow$  state-action marginal

**Maximum Entropy Reinforcement Learning**

$$J(\pi) = \sum_{t=0}^T \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_\pi} [r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi(\cdot | \mathbf{s}_t))]$$

$Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$

soft  $Q$ -value

$\mathcal{T}^\pi \rightarrow$  Bellman backup operator

$$\mathcal{T}^\pi Q(\mathbf{s}_t, \mathbf{a}_t) \triangleq r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{\mathbf{s}_{t+1} \sim p} [V(\mathbf{s}_{t+1})]$$

$$V(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi} [Q(\mathbf{s}_t, \mathbf{a}_t) - \log \pi(\mathbf{a}_t | \mathbf{s}_t)]$$

soft state value function

**Lemma (Soft Policy Evaluation)**

$$Q^{k+1} = \mathcal{T}^\pi Q^k \implies Q^k \rightarrow \text{soft } Q\text{-value as } k \rightarrow \infty$$

**Lemma (Soft Policy Improvement)**

$$\begin{aligned} \pi_{\text{new}} &= \arg \min_{\pi' \in \Pi} D_{\text{KL}} \left( \pi'(\cdot | \mathbf{s}_t) \parallel \frac{\exp(Q^{\pi_{\text{old}}}(\mathbf{s}_t, \cdot))}{Z^{\pi_{\text{old}}}(\mathbf{s}_t)} \right) \\ &\implies Q^{\pi_{\text{new}}}(\mathbf{s}_t, \mathbf{a}_t) \geq Q^{\pi_{\text{old}}}(\mathbf{s}_t, \mathbf{a}_t), \forall (\mathbf{s}_t, \mathbf{a}_t) \in \mathcal{S} \times \mathcal{A} \end{aligned}$$

**Theorem (Soft Policy Iteration)**

Repeated application of soft policy evaluation and soft policy improvement converges to a policy  $\pi^*$  such that  $Q^{\pi^*}(\mathbf{s}_t, \mathbf{a}_t) \geq Q^\pi(\mathbf{s}_t, \mathbf{a}_t), \forall \pi \in \Pi$ .

**Soft Actor-Critic**

$V_\psi(\mathbf{s}_t) \rightarrow$  parametrized state value function

$Q_\theta(\mathbf{s}_t, \mathbf{a}_t) \rightarrow$  parametrized soft  $Q$ -function

$\pi_\phi(\mathbf{a}_t | \mathbf{s}_t) \rightarrow$  tractable policy

$$J_V(\psi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}} \left[ \frac{1}{2} (V_\psi(\mathbf{s}_t) - \mathbb{E}_{\mathbf{a}_t \sim \pi_\phi} [Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \log \pi_\phi(\mathbf{a}_t | \mathbf{s}_t)])^2 \right]$$

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \mathcal{D}} \left[ \frac{1}{2} (Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \hat{Q}(\mathbf{s}_t, \mathbf{a}_t))^2 \right]$$

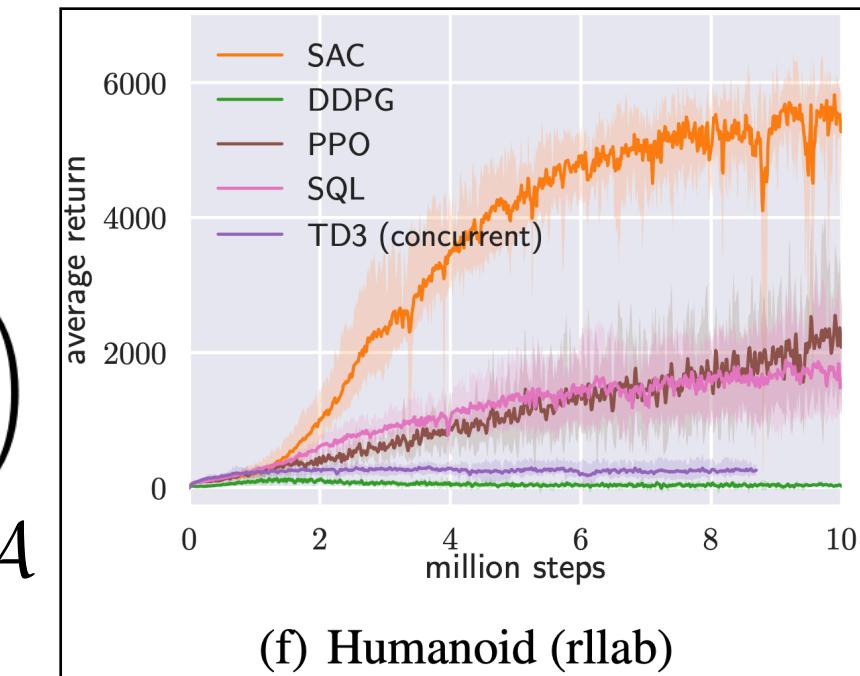
$$\hat{Q}(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{\mathbf{s}_{t+1} \sim p} [V_{\bar{\psi}}(\mathbf{s}_{t+1})]$$

$\bar{\psi}$  can be an exponentially moving average of the value network weights

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}} \left[ D_{\text{KL}} \left( \pi_\phi(\cdot | \mathbf{s}_t) \parallel \frac{\exp(Q_\theta(\mathbf{s}_t, \cdot))}{Z_\theta(\mathbf{s}_t)} \right) \right]$$

$\mathbf{a}_t = f_\phi(\epsilon_t; \mathbf{s}_t) \rightarrow$  reparameterization trick

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}, \epsilon_t \sim \mathcal{N}} [\log \pi_\phi(f_\phi(\epsilon_t; \mathbf{s}_t) | \mathbf{s}_t) - Q_\theta(\mathbf{s}_t, f_\phi(\epsilon_t; \mathbf{s}_t))]$$




---

**Algorithm 1** Soft Actor-Critic

Initialize parameter vectors  $\psi, \bar{\psi}, \theta, \phi$ .

for each iteration do

for each environment step do

$\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t | \mathbf{s}_t)$

$\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t)$

$\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$

end for

for each gradient step do

$\psi \leftarrow \psi - \lambda_V \hat{\nabla}_\psi J_V(\psi)$

$\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$  for  $i \in \{1, 2\}$

$\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$

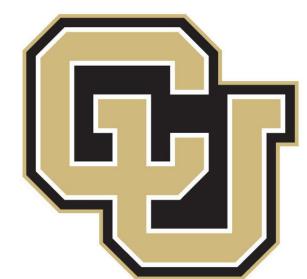
$\bar{\psi} \leftarrow \tau \psi + (1 - \tau) \bar{\psi}$

end for

end for

---

defined implicitly in terms of  $f_\phi$   
can be differentiated through!



Boulder



# Questions?

[YouTube Playlist](#)

---