Fall 2025

Actor-critic Methods

Bringing together value-based and policy-based RL

Cathy Wu

6.7920: Reinforcement Learning: Foundations and Methods

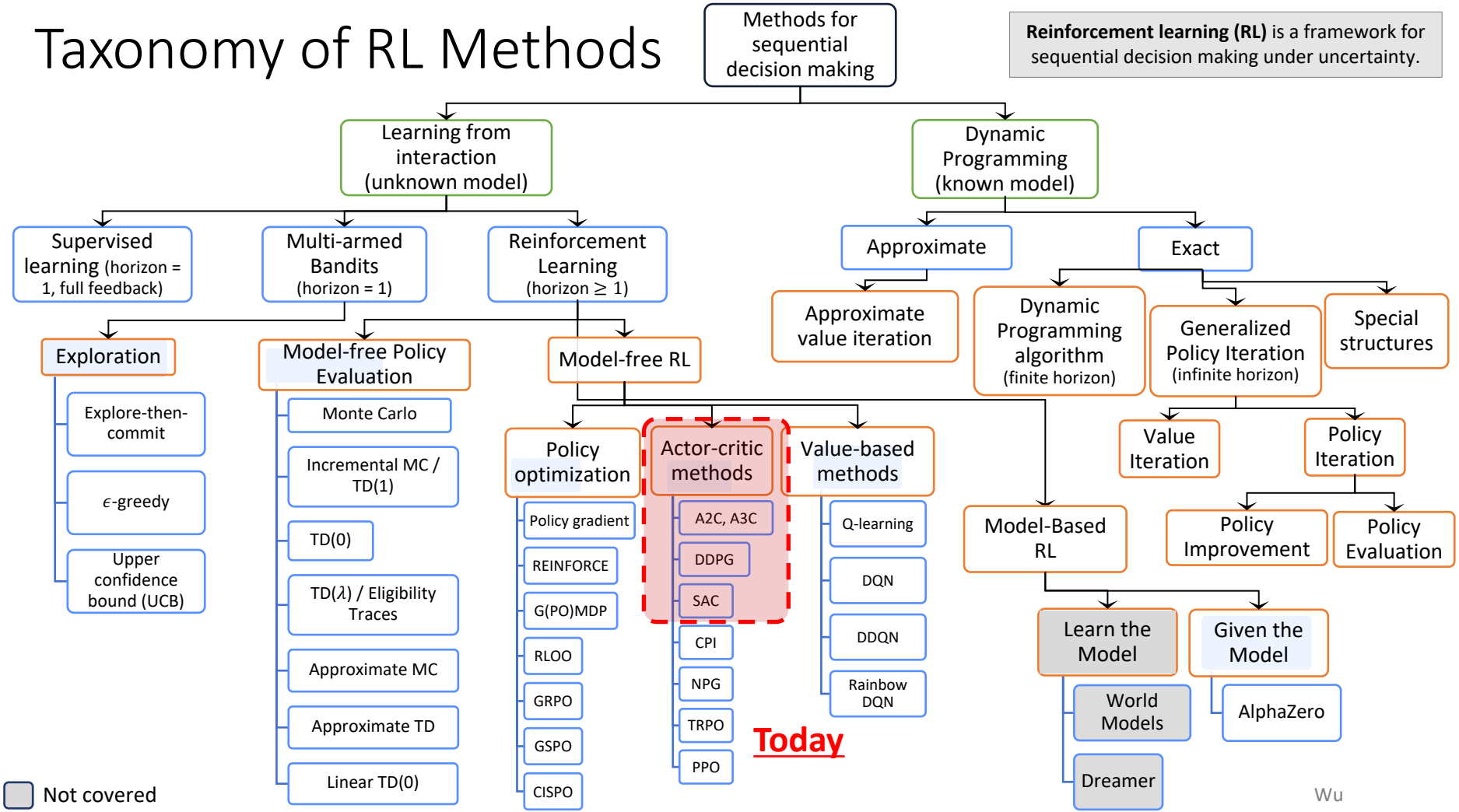
Readings

1. *Josh Achiam. Spinning Up. Part 3: Intro to Policy Optimization. OpenAI, 2018.*
2. *NDP §6.1: Generic issues – from parameters to policies*
3. *SB Chapter 13: Policy Gradient Methods*

Outline

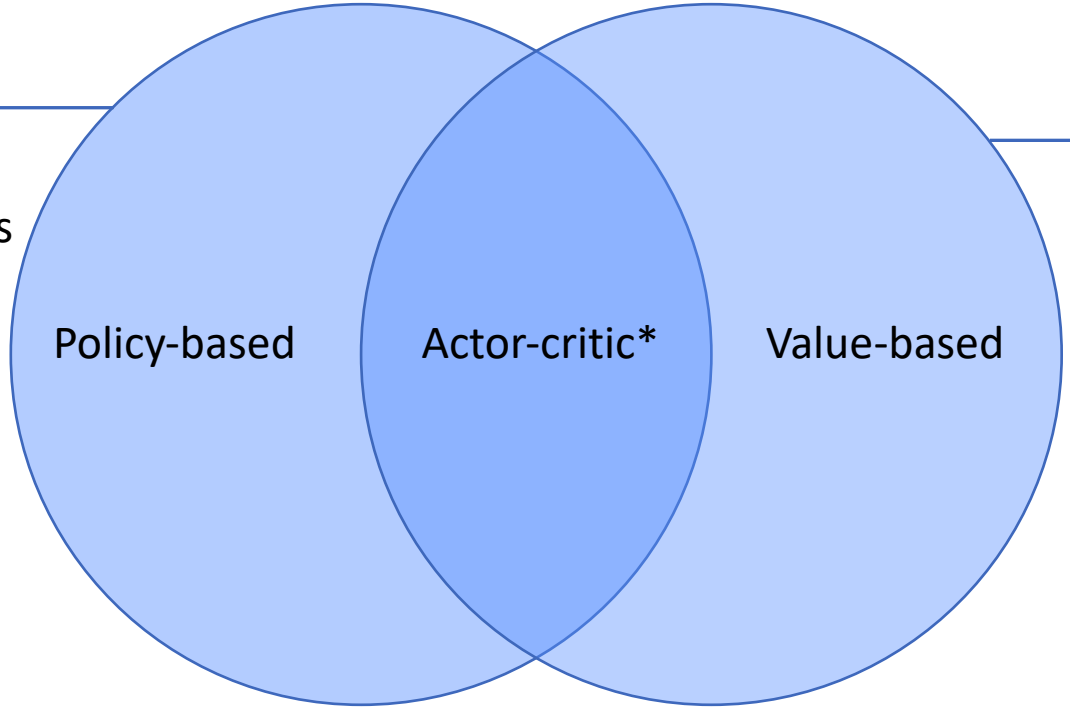
1. Compatible function approximation
2. Advantages and Advantage Actor-Critic (A2C)
3. Asynchronous A2C (A3C)
4. Deep Deterministic Policy Gradient (DDPG)
5. Soft Actor-Critic (SAC)

Taxonomy of RL Methods



Reinforcement learning (RL) is a framework for sequential decision making under uncertainty.

RL methods overview



Optimize policies
(and only policies)

Policy gradients
REINFORCE
Genetic algorithms

Optimize value functions
(and only value functions)

Value iteration
Q-learning
SARSA
DQN
Rainbow DQN

Elements of both

DDPG, SAC, PPO

The best methods use elements of both.

* Warning: precise definitions may vary

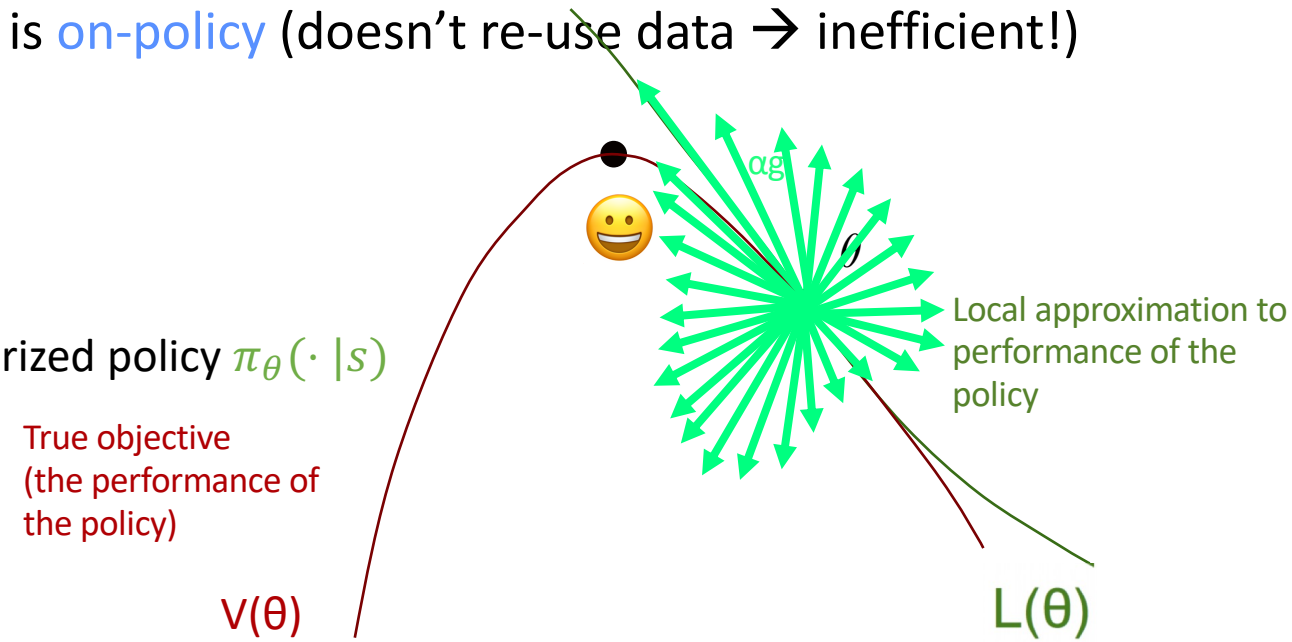
Policy gradients & high variance: the saga continues

- Monte-Carlo policy gradient is unbiased but still has high variance

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t+1}^{T} r_{t'}\right]$$

- Policy gradient is on-policy (doesn't re-use data → inefficient!)

Consider a parameterized policy $\pi_\theta(\cdot|s)$



Policy- and value-based methods → actor-critic

- Monte-Carlo policy gradient is unbiased but still has high variance

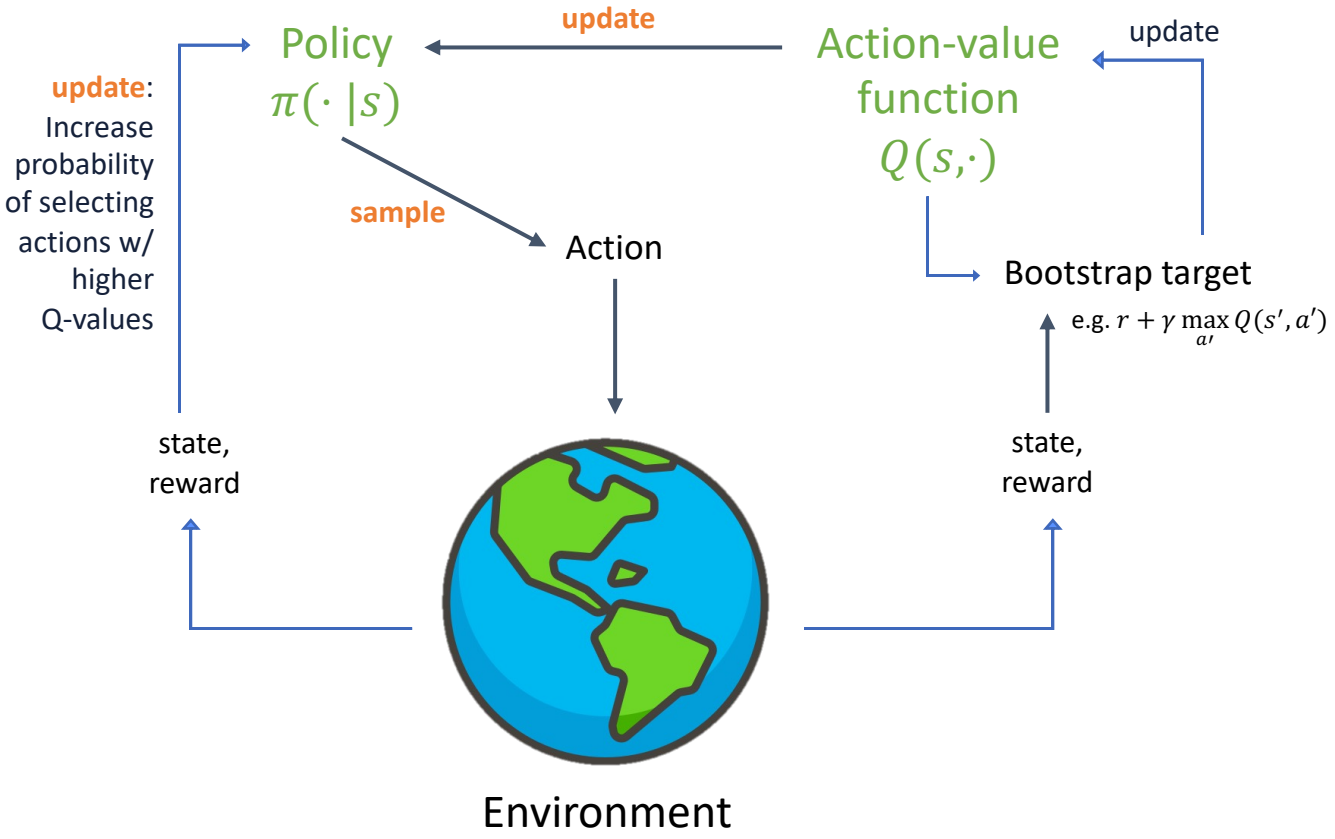
$$\nabla_\theta V(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t+1}^{T} r_{t'}\right]$$

- Incorporate an estimate of $Q^\pi(s,a) \Longrightarrow$ actor-critic
  - Critic: estimate the value function
  - Actor: update the policy in the direction suggested by the critic
- Actor-critic

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\, Q^{\pi_\theta}(s_t,a_t)\right]$$

- These are equivalent (see HW).

Actor-critic methods



Actor-Critic

- Algorithm maintains two sets of parameters: $\theta \longmapsto \pi_\theta, \omega \longmapsto Q_\omega$
- Critic can use $TD(0)$

---

for $t = 0, \dots, T-1$ **do**

$a_t \sim \pi_\theta(s_t, \cdot)$ and observe r_{t+1} and s_{t+1}

Compute temporal difference

$$\delta_t = r_{t+1} + \gamma Q_\omega(s_{t+1}, a_{t+1}) - Q_\omega(s_t, a_t)$$

Update Q estimate

$$\omega = \omega + \beta \delta_t \nabla_\omega Q_\omega(s_t, a_t)$$

Update policy

$$\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(a_t | s_t) Q_\omega(s_t, a_t)$$

end

Actor-Critic

Issues:

- $Q_\omega(s,a)$ is a biased estimate of $Q^{\pi_\theta}(s,a)$
- The update of θ may not follow the gradient of $\nabla_\theta V(\pi_\theta)$

Solution:

- Choose the approximation space $Q_\omega(s,a)$ carefully
  $\Longrightarrow$ compatible function approximation between Q_ω and π_θ

Compatible Function Approximation

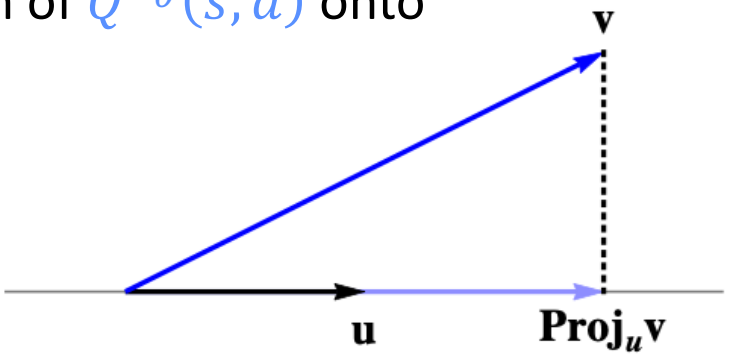
- Actor-critic

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\, Q^{\pi_\theta}(s_t, a_t)\right]$$

- Re-write using occupancy measures

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}_{s\sim d^{\pi_\theta}} E_{a\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s)\, Q^{\pi_\theta}(s, a)]$$

- Interpretation (inner product): projection of $Q^{\pi_\theta}(s, a)$ onto subspace spanned by $\nabla_\theta \log \pi_\theta(a|s)$
- Let $Q_\omega(s, a) = \sum_i \alpha_i [\nabla_\theta \log \pi_\theta(s, a)]_i$ where $\omega = (\alpha_i)_{|\theta|}$



Compatible Function Approximation

Theorem (Silver, 2014)

An action value function space Q_ω is compatible with a policy space π_θ if:

1. **[Feature Selection]** $\nabla_\omega Q_\omega(s,a) = \nabla_\theta \log \pi_\theta(s,a)$
2. **[Least Squares Fitting]** And if ω minimizes the squared error

$$\omega = \arg\min_\omega \mathbb{E}_{s\sim d^{\pi_\theta}} \left[\sum_a \pi_\theta(a|s) \left(Q^{\pi_\theta}(s,a) - Q_\omega(s,a) \right)^2 \right]$$

Then:

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}_{s\sim d^{\pi_\theta}} E_{a\sim\pi_\theta} [\nabla_\theta \log \pi_\theta(a|s) Q_\omega(s,a)]$$

- Remark 1: conditions for which the policy gradient is exact.
- Remark 2: approximately satisfied by linear function approximation.

Sample Efficiency in Actor-Critic

Issues:

- Sample efficiency is pretty poor
- All samples need to be generated by the current policy (on-policy learning)
- Samples are discarded after a single update

Solutions:

- Variance reduction techniques
- Asynchronous training (A3C)
- Use samples from other policies via importance sampling (not very stable) (next time)
- Conservative approaches (next time)
- Newton for Quasi-newton methods

Actor-Critic with a Baseline

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}_{s\sim d^{\pi_\theta}}\left[\sum_a \nabla_\theta \log \pi_\theta(s,a)\left(Q^{\pi_\theta}(s,a) - b(s)\right)\right]$$

- $b(s)$ minimizes the variance
- $V^\pi(s)$ is a good choice as baseline
  - It minimizes the variance in average reward [Bhatnagar et al., 2009]
- $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ is the advantage function

Actor-Critic with Advantage Function (A2C)

- It is possible to estimate V^{π} and Q^{π} independently (e.g. by $TD(0)$)
- $A^{\pi} = Q_{\omega} - V_{\nu}$ is a biased and unstable estimate

Solution:

- Consider the temporal difference error

$$\delta^{\pi_\theta} = r(s,a) + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

- δ^{π_θ} is an unbiased estimate of the advantage

$$\begin{aligned}\mathbb{E}[\delta^{\pi_\theta}|s,a] &= \mathbb{E}[r(s,a) + \gamma V^{\pi_\theta}(s')|s,a] - V^{\pi_\theta}(s) \\ &= Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)\end{aligned}$$

Actor-Critic with Advantage Function (A2C)

- Estimate only $V_v \longmapsto \delta_v = r + \gamma V_v(s') - V_v(s)$

☞ Convergence results with compatible function approximation [Bhatnagar et al., 2009]

---

for $t = 0, \dots, T-1$ **do**

$a_t \sim \pi_\theta(s_t, \cdot)$ and observer r_{t+1} and s_{t+1}

Compute temporal difference

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

Update V estimate

$$v = v + \beta \delta_t \nabla_v V_v(s_t)$$

Update policy

$$\theta = \theta + \alpha \delta_t \nabla_\theta \log \pi_\theta(a_t|s_t)$$

end

Compare (actor-critic):

$$\delta_t = r_{t+1} + \gamma Q_\omega(s_{t+1}, a_{t+1}) - Q_\omega(s_t, a_t)$$
$$\omega = \omega + \beta \delta_t \nabla_\omega Q_\omega(s_t, a_t)$$
$$\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(a_t|s_t) Q_\omega(s_t, a_t)$$

Generalized advantage estimation (GAE) (2016)

- Compute advantages in manner analogous to TD(λ)
- Compare (A2C): analogous to TD(0)

---

for $t = 0, \dots, T-1$ **do**

| $a_t \sim \pi_\theta(s_t, \cdot)$ and observer r_{t+1} and s_{t+1}

End

Compute temporal differences $\delta_0, \dots, \delta_{T-1}$

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

$$A_t^{\mathrm{GAE}(\lambda,\gamma)} = \sum_{l=0}^{T-t-1} (\lambda\gamma)^l \delta_{t+l}$$

Update V estimate

$$v = v + \beta \delta_t \nabla_v V_v(s_t)$$

Update policy

$$\theta = \theta + \alpha A_t^{\mathrm{GAE}} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Compare (A2C):

$$\delta_t = r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t)$$
$$v = v + \beta \delta_t \nabla_v V_v(s_t)$$
$$\theta = \theta + \alpha \delta_t \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Asynchronous Advantage Actor-Critic (A3C)

- Multiple independent agents (networks) with their own weights, who interact with a different copy of the environment in parallel.
- The agents (or workers) train in parallel using a global network θ. They periodically update the global network with their $d\theta$.
- Remark: In practice, θ denotes the shared weights for the value function and the policy (multi-headed network)

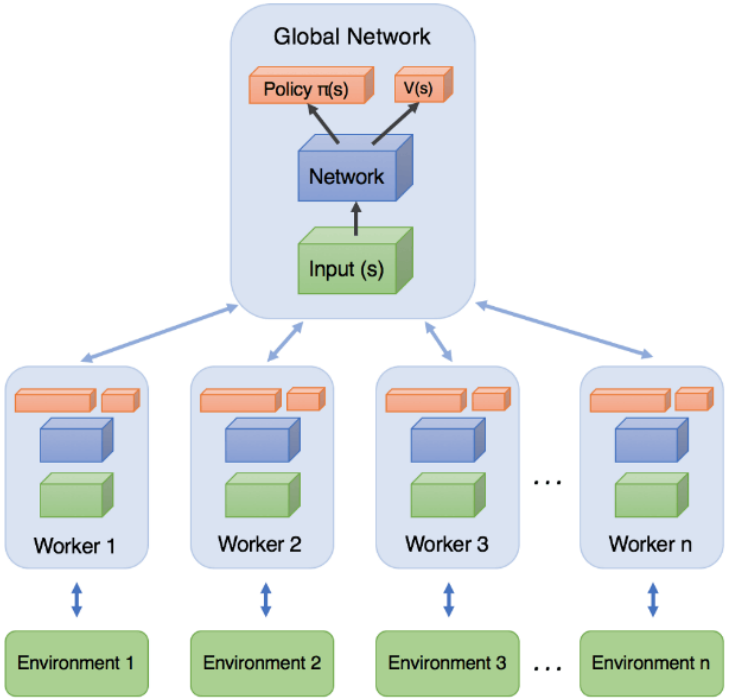
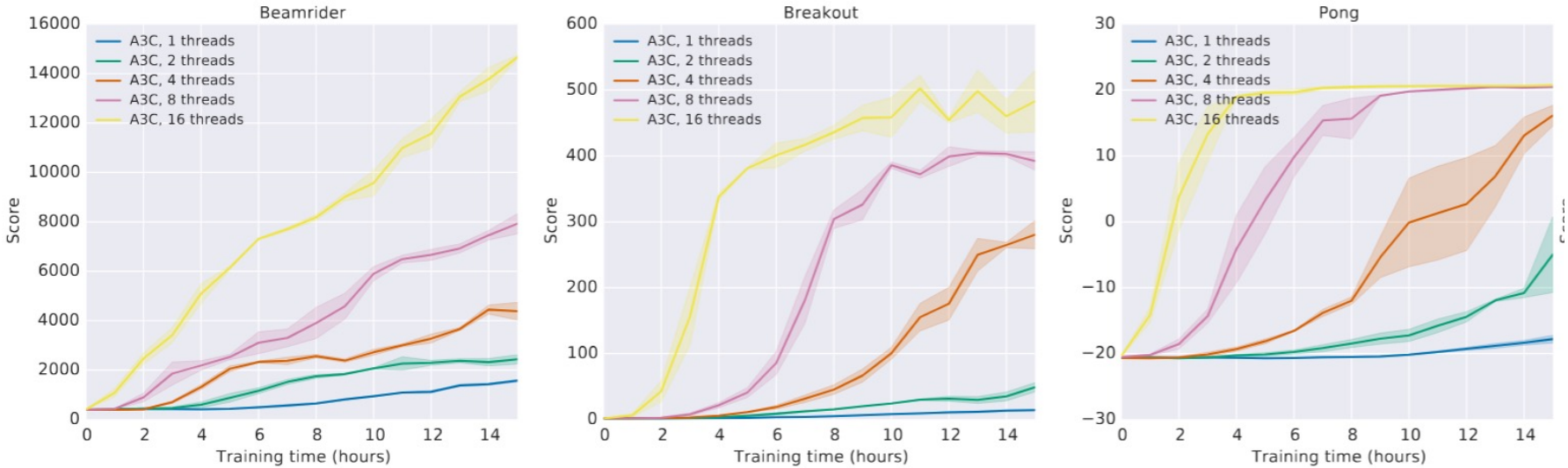


Figure from Atrisha Sarkar

Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." *ICML*, 2016.

Asynchronous Advantage Actor-Critic (A3C)

- Improved training exploration & stability for A2C.



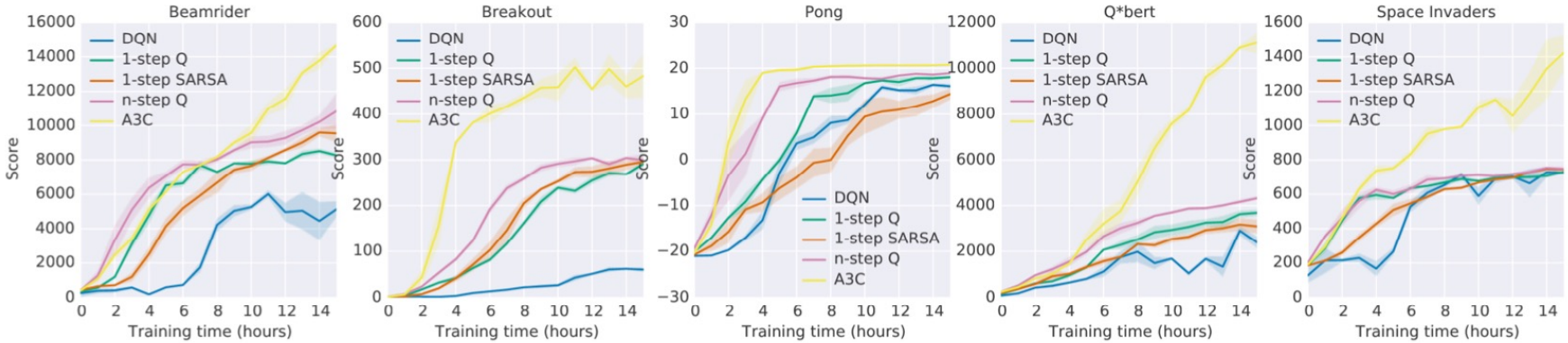
Notes

- “A3C, 1 thread” = A2C

Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." *ICML*, 2016.

Asynchronous Advantage Actor-Critic (A3C)

- Improved training exploration & stability for DQN.



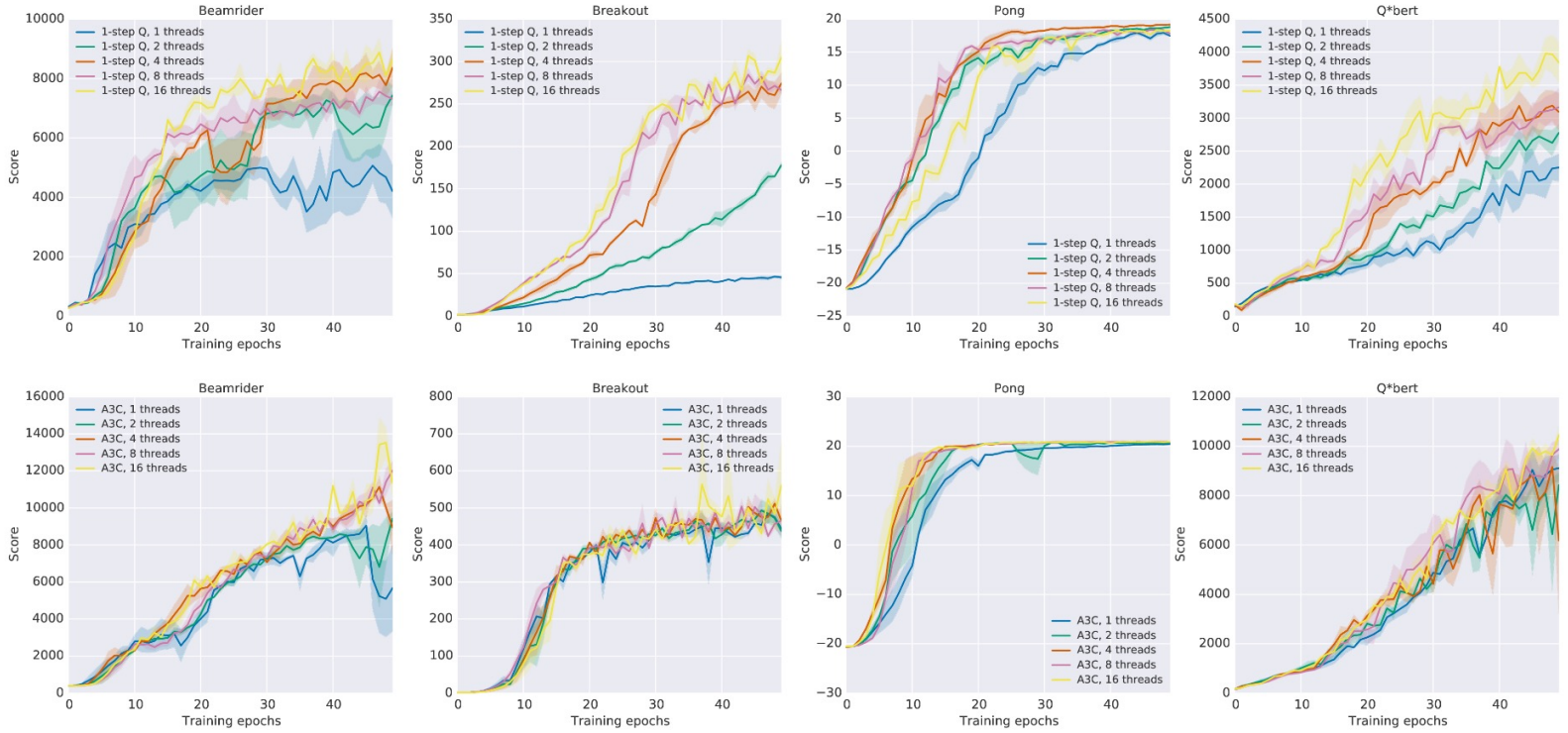
Notes

- "1-step Q", "1-step SARSA", and "n-step Q" are asynchronous variants of DQN (not tabular Q-learning)
- Caveat: y-axis is wall-clock time, not environment samples!

Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." *ICML*, 2016.

Asynchronous Advantage Actor-Critic (A3C)

- Sample efficiency improvements are less pronounced



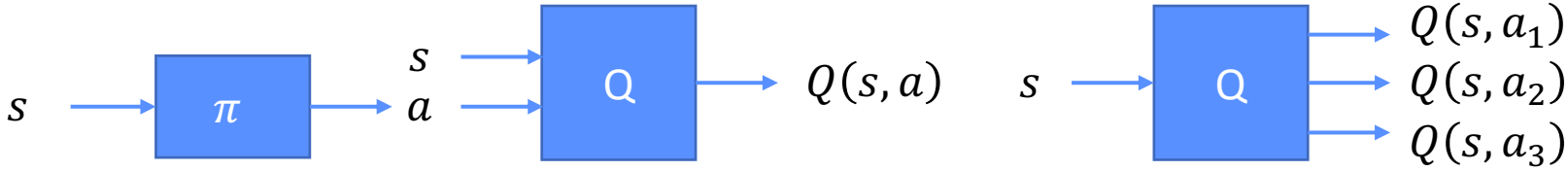
Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." *ICML*, 2016.

Outline

1. Compatible function approximation
2. Advantages and Advantage Actor-Critic (A2C)
3. Asynchronous A2C (A3C)
4. **Deep Deterministic Policy Gradient (DDPG)**
5. Soft Actor-Critic (SAC)

Deep Deterministic Policy Gradient (DDPG)

- Recall: value-based methods have trouble handling continuous actions/large action spaces
- Key idea: simplify Q using deterministic policies



- Recall: $V_D(\pi) = \mathbb{E}_{s\sim d^\pi}\left[r\left(s,\pi(s)\right)\right]$ (d^π is the discounted state-visitation distribution of π.)
- $\nabla_\theta V_D(\theta) = \sum_s d^\pi(s)\nabla_\theta\pi_\theta(s)\nabla_a Q^\pi(s,a)|_{a=\pi_\theta(s)} = \mathbb{E}_{s\sim d^\pi}\left[\nabla_\theta\pi_\theta(s)\nabla_a Q^\pi(s,a)|_{a=\pi_\theta(s)}\right]$

Plug it into an actor-critic framework

- Use $TD(0)$ to update a parametric representation of Q^π

$$\delta_t = r_{t+1} + \gamma Q_w(s_{t+1},a_{t+1}) - Q_w(s_t,a_t) \quad \text{; TD error in SARSA}$$
$$w_{t+1} = w_t + \alpha_w \delta_t \nabla_w Q_w(s_t,a_t)$$
$$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_a Q_w(s_t,a_t)\nabla_\theta\pi_\theta(s)\Big|_{a=\pi_\theta(s)} \quad \text{; Deterministic policy gradient theorem}$$

- Issue: Need to explicitly force exploration, e.g. "behavior policy" with additive $\beta(\cdot)\sim\mathcal{N}(\theta,\sigma\beta^2)$

D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," *ICML*, 2014.

Outline

1. Compatible function approximation
2. Advantages and Advantage Actor-Critic (A2C)
3. Asynchronous A2C (A3C)
4. Deep Deterministic Policy Gradient (DDPG)
5. **Soft Actor-Critic (SAC)**

Policy Iteration: Recap

Let π_0 be an arbitrary stationary policy.

while $k = 1, \dots, K$ **do**

Policy Evaluation: given π_k compute $V_k = V^{\pi_k}$

Policy Improvement: find π_{k+1} that is better than π_k

- e.g. compute the *greedy* policy:

$$\pi_{k+1}(s) \in \arg\max_{a \in \mathcal{A}} \left\{ r(s,a) + \gamma \sum_y p(y|s,a) V^{\pi_k}(y) \right\}$$

return the last policy π_K

end

- Convergence is finite and monotonic [Bertsekas, 2007] (in exact settings)

? Issues: Function approximation for $V^{\pi_k} \Longrightarrow$ Does it still converge?

Continuous Actions?

Recap: Approximate Policy Iteration with Q Functions

Recall the state-action cost-to-go function: $Q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) Q_\pi\big(s', \pi(s')\big)$

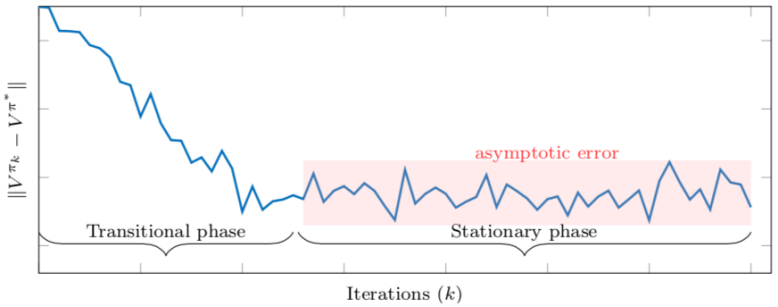
Approximate PI:

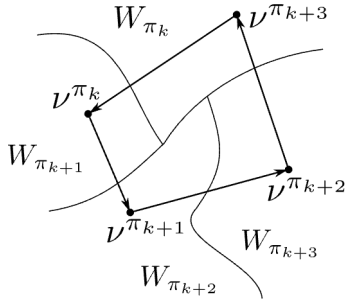
- For $k = 0, 1, 2, \ldots$
  1. Approximate the value under π_k: $Q_{\theta_k} \approx Q_{\pi_k}$
  2. Solve for an improved policy

$$\pi_{k+1}(s) \in \underset{a \in A(s)}{\operatorname{argmin}} Q_{\theta_k}(s,a) \qquad \forall s \in \mathcal{S}$$

Q_{π_k} can be approximated by either TD or Monte Carlo methods.

Same story as fitted Q-iteration. No longer guaranteed to converge.





Soft policy iteration [Haarnoja, 2018]

- **Soft policy evaluation**:

We define the bellman backup operator for any $Q : \mathcal{S} \times \mathcal{A} \to \Re$:

$$\mathcal{T}^{\pi} Q(s_t, a_t) \triangleq r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p}[V(s_{t+1})]$$

where we have the soft state value function:

entropy regularization

$$V(s_t) = \mathbb{E}_{a_t \sim \pi}[Q(s_t, a_t) - \alpha \log \pi(a_t | s_t)]$$

- Under standard assumptions: Q^k will converge to the soft Q-value of π as $k \to \infty$

Soft policy iteration [Haarnoja, 2018]

- **Soft policy improvement**:

  For each state, we do the following update:

$$\pi_{\text{new}} = \arg\min_{\pi' \in \Pi} D_{\text{KL}} \left(\pi'(\cdot|s_t) \| \frac{1}{Z^{\pi_{\text{old}}}(s_t)} \cdot \exp(\frac{1}{\alpha} \cdot Q^{\pi_{\text{old}}}(s_t, \cdot) \right).$$

  Then we have

$$Q^{\pi_{\text{new}}}(s_t, a_t) \geq Q^{\pi_{\text{old}}}(s_t, a_t) \quad \forall (s_t, a_t) \in \mathcal{S} \times \mathcal{A}, |\mathcal{A}| < \infty.$$

Soft policy iteration:

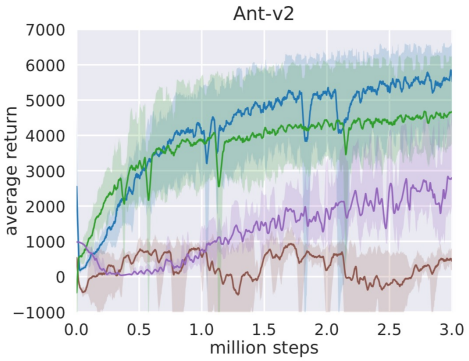
- Under standard assumptions: The sequence Q^{π_i} is monotonically increasing and bounded. So, it converges to some π^*.

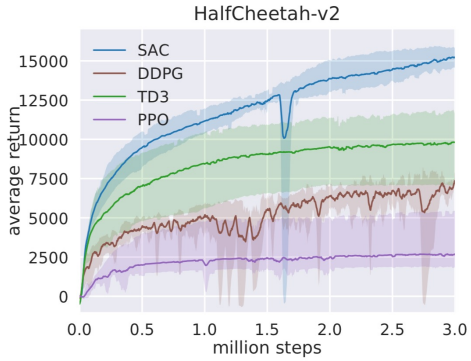
Soft actor-critic (SAC) [Haarnoja, 2018]

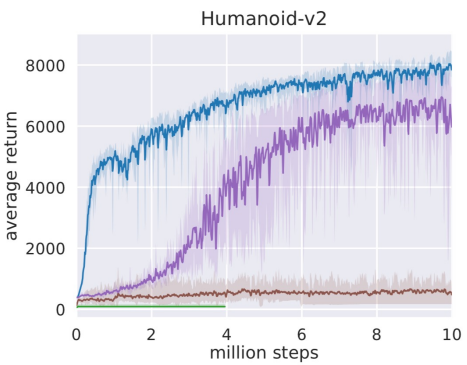
Soft policy iteration + function approximation

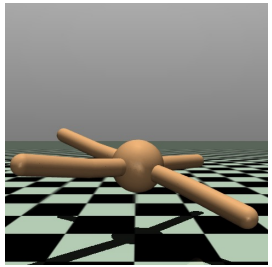
1. **[Soft policy evaluation]**
   Train the action-value function Q_θ, minimizing:
   $$\arg\min_\theta \mathbb{E}_{(s,a)\in H}\left[\frac{1}{2}\left(Q_\theta(s_t,a_t)-\left(r(s_t,a_t)+\gamma\mathbb{E}\left[V_{\bar{\psi}}(s')\right]\right)\right)^2\right]$$
   ! Fix the target network (e.g. DQN) → increase stability / break dependences
2. Train the (soft) value function V_ψ, minimizing:
   $$J_V(\psi)=\mathbb{E}_{s_t\sim D}\left[\frac{1}{2}\left(V_\psi(s_t)-\mathbb{E}_{a_t\sim\pi_\phi}\left[Q_\theta(s_t,a_t)-\log\pi_\phi(a_t|s_t)\right]\right)^2\right]$$
   entropy regularization
3. **[Soft policy improvement]**
   Fit the new (stochastic) policy π_ϕ:
   $$\arg\min_\phi \mathbb{E}_{s\in H}\left[D_{KL}\left(\pi_\phi\,||\,\frac{\exp[\eta Q_\theta]}{Z}\right)[s]\right]$$
   replace max with softmax

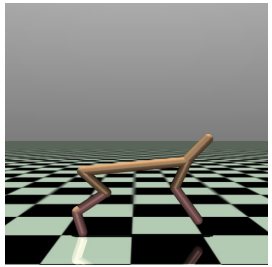
Soft actor-critic (SAC) [Haarnoja, 2018]

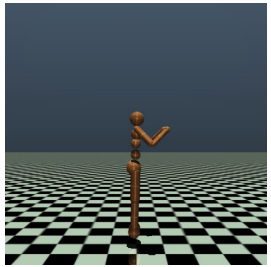










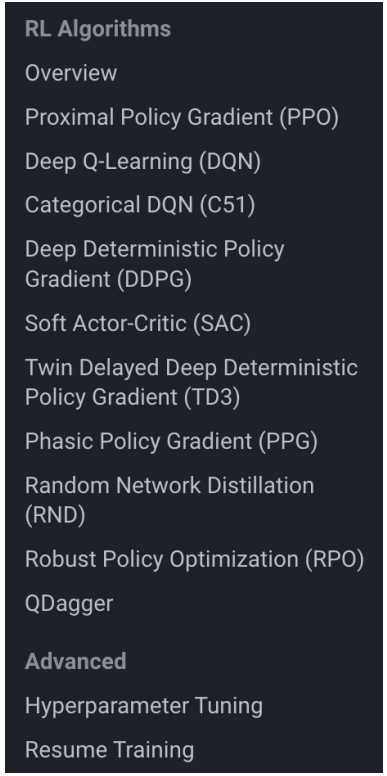


Documentation: https://gymnasium.farama.org/environments/mujoco/

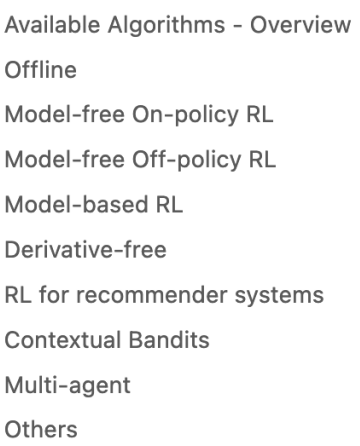
Implementations of RL algorithms

- For learning and research prototyping:
  - CleanRL: https://docs.cleanrl.dev/
    - A Deep Reinforcement Learning library that provides high-quality single-file implementation with research-friendly features
  - Mastering RL online textbook + code: https://uq.pressbooks.pub/mastering-reinforcement-learning/
    - Implementations of classic algorithms & concepts
- For research:
  - stable-baselines3: https://stable-baselines3.readthedocs.io/en/master/
    - Popular library
- For scaling up:
  - RLlib: https://docs.ray.io/en/latest/rllib/index.html
    - Industry-grade reinforcement learning
    - Built on distributed execution engine Ray

CleanRL



RLlib



References

1. Matteo Pirotta. FAIR. Reinforcement Learning. 2019, Lecture 5.

2. Matteo Pirotta. Reinforcement Learning Summer School, 2019. Policy Search: Actor-Critic Methods.