NEURAL FITTED ACTOR-CRITIC

{ MATTHIEU.ZIMMER, YANN.BONIFACE AND ALAIN.DUTECH }@LORIA.FR

PROBLEM

Reinforcement learning (RL) is a framework for solving sequential decision problems where an agent interacts with its environment and adapts its policy based on a reward signal.
We present an RL algorithm respecting two main requirements :

1. dealing with **continuous action** and state spaces
2. knowledge added by the designer to the agent should be minimal

CONTRIBUTIONS

Inspired by Fitted Q Iteration (FQI) [1] and Continuous Actor Critic Learning Automaton (CACLA) [2], we formulated a new **on-policy**, **non-linear**, **off-line**, **model-free**, **actor-critic** algorithm. Unlike FQI, it deals with continuous action and state spaces and performs better than CACLA with fewer meta-parameters.

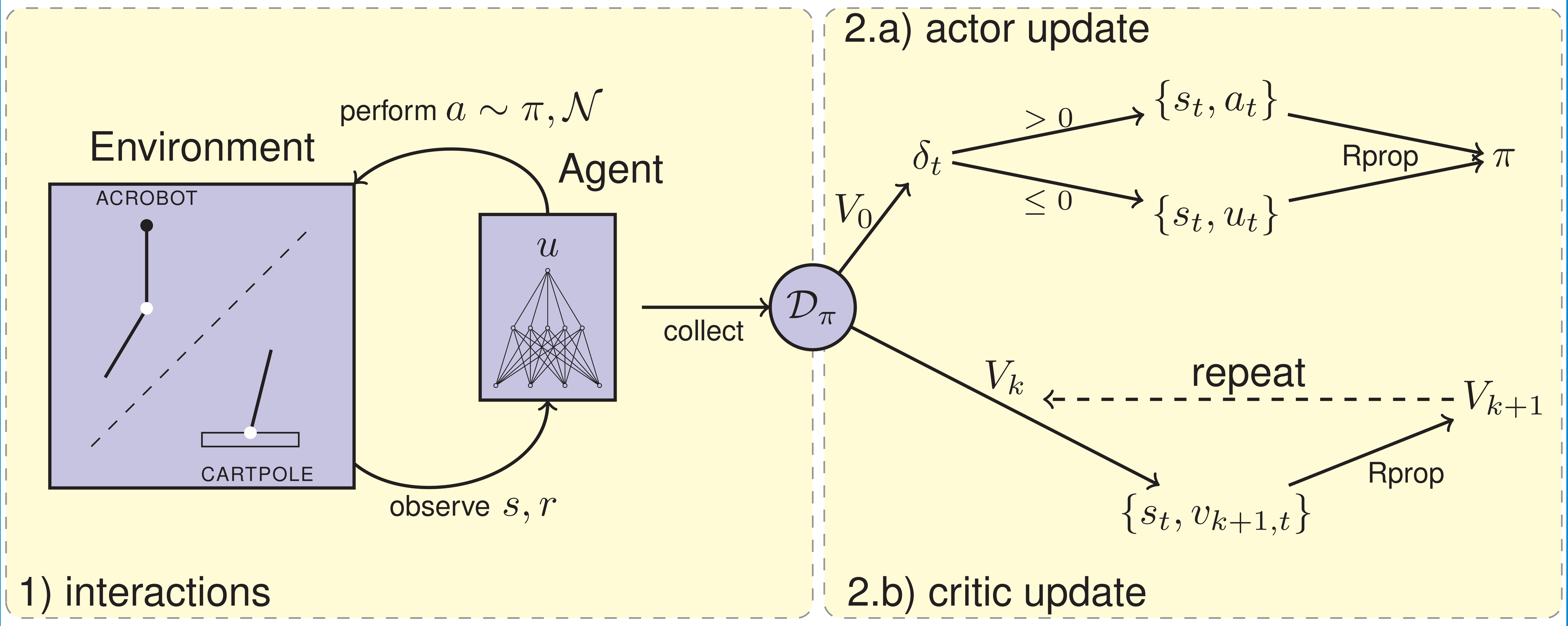
ALGORITHM

Randomly initialize critic network V
Randomly initialize actor network π
for $episode \leftarrow 0$ **to** M **do**
  Receive initial state s_0
  $t \leftarrow 0$
  while $s_t \notin S^*$ **and** $t < T_{max}$ **do**
    Select action $u_t \leftarrow \pi(s)$
    Add exploration noise $a_t = g(u_t, \mathcal{N})$
    Perform action a_t, observe r_{t+1} and s_{t+1}
    Store transition $(s_t, a_t, u_t, r_{t+1}, s_{t+1})$ in $\mathcal{D}_\pi$
    $t \leftarrow t+1$
  end
  for $(s_t, a_t, u_t, r_{t+1}, s_{t+1}) \in \mathcal{D}_\pi$ **do**
    $$\delta_t \leftarrow \begin{cases} r_{t+1} - V(s_t), & \text{if } s_{t+1} \in S^* \\ r_{t+1} + \gamma V(s_{t+1}) - V(s_t), & \text{otherwise} \end{cases}$$
    $$y_t \leftarrow \begin{cases} a_t, & \text{if } \delta_t > 0 \\ u_t, & \text{otherwise} \end{cases}$$
    Store (s_t, y_t) into $\mathcal{D}_{actor}$
  end
  Update actor minimazing the MSE with Rprop:
  $$\pi \leftarrow \underset{\pi \in \mathcal{F}_a}{\text{argmin}} \sum_{(s_t, y_t) \in \mathcal{D}_{actor}} \big(\pi(s_t) - y_t\big)^2$$
  $V_0 \leftarrow V$
  for $k \leftarrow 1$ **to** K **do**
    for $(s_t, r_{t+1}, s_{t+1}) \in \mathcal{D}_\pi$ **do**
      $$v_{k,t} \leftarrow \begin{cases} r_{t+1}, & \text{if } s_{t+1} \in S^* \\ r_{t+1} + \gamma V_{k-1}(s_{t+1}), & \text{otherwise} \end{cases}$$
      Store $(s_t, v_{k,t})$ in $\mathcal{D}_{critic,k}$
    end
    Update critic minimazing MSE with Rprop:
    $$V_k \leftarrow \underset{V \in \mathcal{F}_c}{\text{argmin}} \sum_{(s_t, y_t) \in \mathcal{D}_{critic,k}} \big(V(s_t) - v_{k,t}\big)^2$$
  end
  $V \leftarrow V_K$
  Clear $\mathcal{D}_\pi$, $\mathcal{D}_{actor}$ and $\mathcal{D}_{critic,*}$
end

METHOD

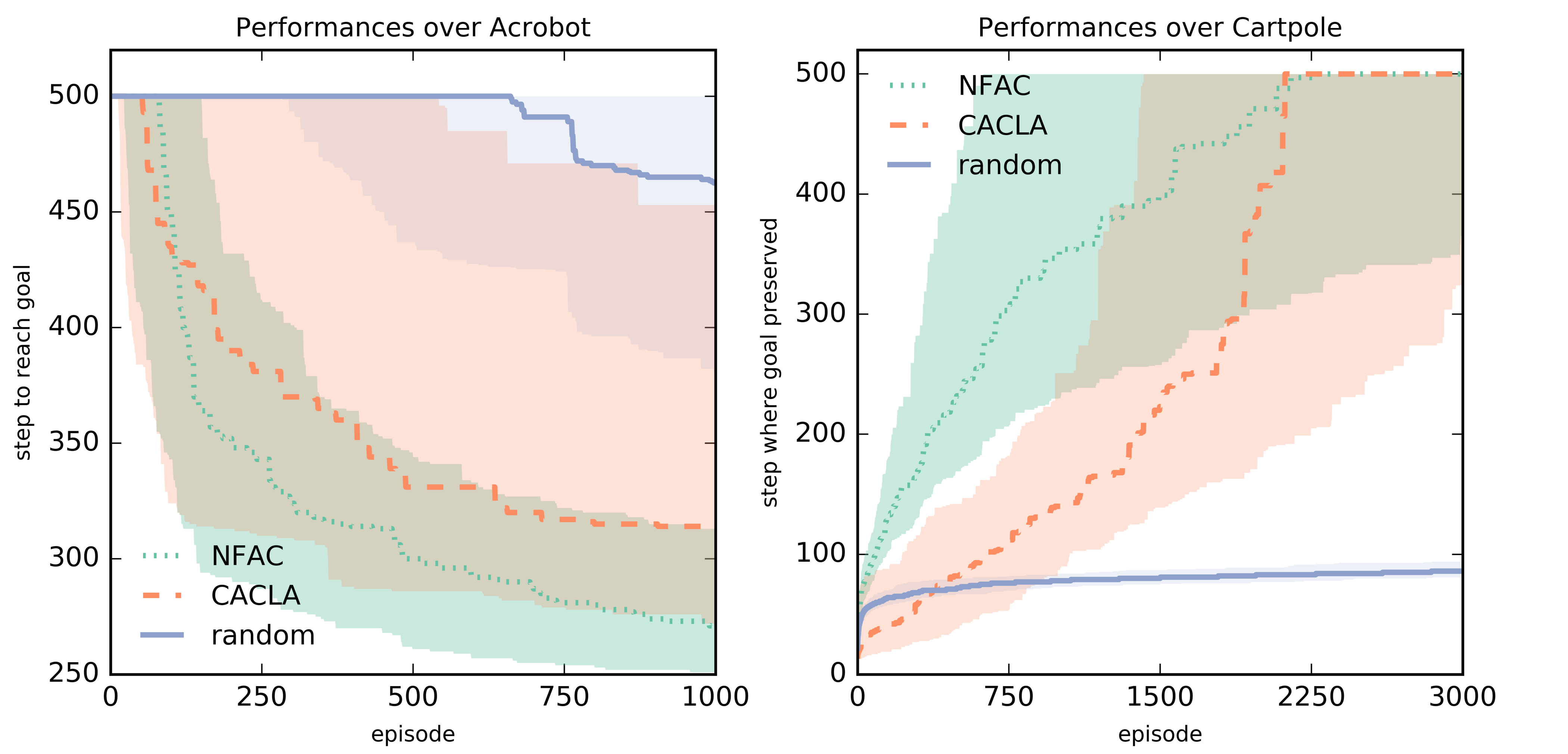
Actor-critic is a solution to handle **continuous actions spaces** in Markov Decision Process (S, A, R, T, γ). The critic learns the value-function V_π that describes the expected return following π after the state s. The goal of the actor is to decide which action to take in which state, by finding the best policy according to V :

$$\begin{cases} V_\pi(s) & = \mathbb{E}_\pi\Big[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\Big] \\ \pi_V(s_t) & = \underset{a \in A}{\text{argmax}}\ T(s_{t+1}|s_t, a)\big[R(s_t, a, s_{t+1}) + \gamma V_\pi(s_{t+1})\big] \end{cases}$$



The major difference with CACLA is the replay of previous transitions of the current episode, which increases the accuracy of the critic but also the computations needed by the agent. Instead of moving toward the direction of each good action encountered, we evaluate the efficiency of all taken actions during the episode through δ and learn a new policy with supervised learning. Compared to FQI, the argmax operator is replaced by the actor choice : V is here an **on-policy** evaluation of the current actor.

RESULTS



Median and quartile of the best registered performance in Acrobot (lower better) and Cartpole (higher better) environment during RL learning.

FUTURES DIRECTIONS

In order to increase **data efficiency**, $\mathcal{D}_\pi$ should not be cleared and the sample generated from the previous episodes should be used to improve the actor and the critic updates.

For the critic, it can be done through the importance sampling term $\frac{\pi(a_t|s_t)}{\pi_0(a_t|s_t)}$, making the actor-critic **off-policy**. The main challenge is how to use the previous samples to update the policy parameters.

REFERENCES

[1] Martin Riedmiller. Neural fitted Q iteration - First experiences with a data efficient neural Reinforcement Learning method. In *Lecture Notes in Computer Science*, volume 3720 LNAI, pages 317–328, 2005.

[2] Hado Van Hasselt and Marco A. Wiering. Reinforcement learning in continuous action spaces. In *Proceedings of the IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 272–279, 2007.

SOURCE CODE

The source code and data are available at :
`drl.gforge.inria.fr`