RL for MDP
Chap. 4: Policy-based algorithms

Idris Kharroubi

LPSM
Sorbonne Université

DUFE

Outline

1. Policy-based methods
2. Policy gradient methods
3. Actor-critic method

In value-based methods presented above, one focuses first on the approximation and learning of the optimal value function.

Then, the optimal policy is computed from the value function learned before.

This kind of algorithms are sometimes called **critic algorithm**:

one can try the derived strategy and test its efficiency.

This value based approach, and especially the Q-learning, works well in the case where the state and actions spaces are small.

Another kind of algorithms called **Policy based algorithms** consists in computing the optimal policy at first.

For that, the policy is represented by a parametric approximation function.

More precisely, The control is replaced by stochastic randomized control π^θ depending on a parameter θ with densities $a \mapsto p_n^\theta(x, a)$ w.r.t. some fixed measure ν on the action space A.

This leads to two kind of policy based methods

- policy gradient method,
- actor method.

We present in the sequel those two methods.

Outline

1. Policy-based methods
2. Policy gradient methods
3. Actor-critic method

For a parametrized policy π^θ, we consider the reward function J define as a function of the parameter θ by

$$J(\theta) \quad = \quad \mathbb{E}_{\pi^\theta}\Big[\sum_{k=0}^{N-1} r_k(X_k, \alpha_k) + g_N(X_N)\Big] \, .$$

To maximize $J(\theta)$ over the parameter θ, the policy gradient method consists to apply a gradient ascent algorithm on J to approximate the optimal θ.

To compute the gradient of J, we introduce some notations. Define $S = (X_0, \alpha_0, \ldots, X_{N-1}, \alpha_{N-1}, X_N)$ the state/action trajectory and $R(S)$ the related total reward:

$$R(S) = \sum_{k=0}^{N-1} r_k(X_k, \alpha_k) + g_N(X_N)$$

In particular we have $J(\theta) = \mathbb{E}_{\pi^\theta}[R(S)]$.

Theorem (Gradient representation)

We have

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi^\theta}\left[R(S) \sum_{k=0}^{N-1} \nabla_\theta \log p_k^\theta(X_k, \alpha_k)\right].$$

Proof of Gradient representation THM

Denote by $\Pi^\theta(ds)$ the law of $S = (X_0, \alpha_0, \ldots, X_T)$ when $X_0 \sim \mu_0$ and $\alpha \sim \pi^\theta$. We then have

$$\begin{aligned}\Pi^\theta(ds) &= \Big(\prod_{n=0}^{N_1} p_n^\theta(x_n, a_n)\Big)\Big(\mu_0(dx_0)\prod_{n=0}^{N-1}\nu(da_n)P_n(x_n, a_n, dx_{n+1})\Big)\\ &=: \widetilde{p}^\theta(s)\widetilde{P}(ds)\,.\end{aligned}$$

We next have

$$J(\theta) = \int R(s)\Pi^\theta(ds) = \int R(s)\widetilde{p}^\theta(s)\widetilde{P}(ds)\,.$$

Proof of Gradient representation THM

This gives by the derivation under expectation theorem

$$\begin{aligned}
\nabla_\theta J(\theta) &= \int R(s)\big[\nabla_\theta \widetilde{p}^\theta(s)\big]\widetilde{P}(ds) \\
&= \int R(s)\big[\nabla_\theta \log(\widetilde{p}^\theta(s))\big]\widetilde{p}^\theta(s)\widetilde{P}(ds) \\
&= \mathbb{E}\big[R(S)\nabla_\theta \log(\widetilde{p}^\theta(S))\big] \\
&= \mathbb{E}\big[R(S)\sum_{n=0}^{N-1}\nabla_\theta \log(p_n^\theta(X_n,\alpha_n))\big].
\end{aligned}$$

□

The algorithm can be described as follows.

Algorithm 1: Reinforce.

Data: Initialize parameter θ, number of iteration M.

Result: Approximated optimal parameter θ.

for $m = 1$ **to** M **do**

Generate a batch of K trajectories $S = (X_0, \alpha_0, \dots, X_T)$, and rewards $(r_0, \dots, r_{N-1}, g)$ according to Π^θ.

Update θ according to the policy gradient

$$\theta \leftarrow \theta + \eta \Big[\frac{1}{K} \sum_{k=1}^{K} R(S^k) \sum_{n=0}^{N-1} \nabla_\theta \log(p_n^\theta(X_n^k, \alpha_n^k)) \Big] .$$

The previous algorithm needs to sample both states and actions over the whole trajectory.

This may lead to a **high variance** which impacts in an **negative** way the precision of the algorithm.

A possible approach to reduce the variance is to withdraw some constant C to the reward R. In this case, the gradient remains the same:

$$\begin{aligned}
\mathbb{E}_{\pi^\theta}\left[(R(S)-C)\sum_{k=0}^{N-1}\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right] &= \\
\mathbb{E}_{\pi^\theta}\left[R(S)\sum_{k=0}^{N-1}\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right] & \\
-C\mathbb{E}_{\pi^\theta}\left[\sum_{k=0}^{N-1}\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right] &= \\
\nabla_\theta J(\theta) - C\nabla_\theta \int \Pi^\theta(ds) &= \nabla_\theta J(\theta)\,.
\end{aligned}$$

Actually the best constant C to reduce the variance is $\mathbb{E}_{\pi_\theta}[R(S)]$ which is usually inaccessible.

However, as for Monte Carlo Methods, an approximation is sufficient.

This approach is actually related to actor critic methods.

Outline

1. Policy-based methods
2. Policy gradient methods
3. Actor-critic method

• The actor-critic method consists in Learning both the value function (the critic) and the policy (the actor) by neural networks.

• The mains steps are to update alternatively the critic and the actor networks :

- update critic by policy evaluation of the actor,
- update actor by policy gradient in direction suggested by critic.

We now present the gradient representation result base on the DP representation of the value function.

Theorem (Actor/Critic gradient representation)

We have

$$\begin{aligned}\nabla_\theta J(\theta) &= \sum_{k=0}^{N-1} \mathbb{E}_{\pi^\theta}\left[\left(r_k(X_k,\alpha_k) + V_{k+1,\pi^\theta}(X_{k+1})\right)\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right] \\ &= \sum_{k=0}^{N-1} \mathbb{E}_{\pi^\theta}\left[\left(r_k(X_k,\alpha_k) + V_{k+1,\pi^\theta}(X_{k+1}) - V_{k,\pi^\theta}(X_k)\right)\right. \\ &\qquad\qquad \left.\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right],\end{aligned}$$

where we recall that V_{π^θ} is the value function associated to the policy π^θ.

Proof of A/C gradient representation

The RI relation for the value function gives

$$\begin{aligned} V_{k,\pi^\theta}(x) &= \mathbb{E}_{\pi^\theta}\left[r_k(X_k,\alpha_k) + V_{k+1,\pi^\theta}(X_{k+1})\middle| X_k = x\right] \\ &= \int \left(r_k(x,a) + V_{k+1,\pi^\theta}(x')\right) p_k^\theta(x,a)\nu(da)P_k(x,a,dx') \ . \end{aligned}$$

By differentiating w.r.t. the parameter θ we get

$$\begin{aligned} \nabla_\theta V_{k,\pi^\theta}(x) &= \int \nabla_\theta V_{k+1,\pi^\theta}(x') p_k^\theta(x,a)\nu(da)P_k(x,a,dx') + \\ &\quad \int \left(r_k(x,a) + V_{k+1,\pi^\theta}(x')\right) \nabla_\theta p_k^\theta(x,a)\nu(da)P_k(x,a,dx') \\ &= \int \nabla_\theta V_{k+1,\pi^\theta}(x') p_k^\theta(x,a)\nu(da)P_k(x,a,dx') + \\ &\quad \int \left(r_k(x,a) + V_{k+1,\pi^\theta}(x')\right) \\ &\qquad \nabla_\theta \log(p_k^\theta(x,a)) p_k^\theta(x,a)\nu(da)P_k(x,a,dx') \ . \end{aligned}$$

Proof of A/C gradient representation

Therefore, we get

$$\begin{aligned}\nabla_\theta V_{k,\pi^\theta}(x) &= \mathbb{E}_{\pi^\theta}\left[\nabla_\theta V_{k+1,\pi^\theta}(X_{k+1})\middle|X_k = x\right] \\ &\quad + \mathbb{E}_{\pi^\theta}\left[\left(r_k(X_k,\alpha_k) + V_{k+1,\pi^\theta}(X_{k+1})\right)\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right]\end{aligned}$$

Iterating the equality over k and since $\nabla_\theta V_{N,\pi^\theta}(x) = \nabla_\theta g(x) = 0$, we get

$$\nabla_\theta J(\theta) = \sum_{k=0}^{N-1} \mathbb{E}_{\pi^\theta}\left[\left(r_k(X_k,\alpha_k) + V_{k+1,\pi^\theta}(X_{k+1})\right)\nabla_\theta \log p_k^\theta(X_k,\alpha_k)\right] .$$

The second equality comes from

$$\begin{aligned}\mathbb{E}_{\pi^\theta}\left[V_{k,\pi^\theta}(X_k)\nabla_\theta \log p_k^\theta(X_k,\alpha_k)|X_k = x\right] &= \\ V_{k,\pi^\theta}(x)\mathbb{E}_{\pi^\theta}\left[\nabla_\theta \log p_k^\theta(X_k,\alpha_k)|X_k = x\right] &= 0 .\end{aligned}$$

□

The A/C algorithm learns simultaneously via fixed-point iterations the policy (actor) by policy gradient, and the value function (critic) by performance evaluation from the DP relation.

In addition to randomized policies π^θ, we also parametrize the value function, e.g. by a Neural Network $\mathcal{V}_n^\phi(x)$ of parameter ϕ.

The parameters (θ, ϕ) are updated as follows.

- Update θ from the gradient representation :

$$\theta_{m+1} = \theta_m + \eta \mathbb{E}_{\pi^{\theta_m}} \Big[\sum_{k=0}^{N-1} \big(r_k(X_k, \alpha_k) + \mathcal{V}_{k+1}^{\phi_m}(X_{k+1}) - \mathcal{V}_k^{\phi_m}(X_k) \big) \nabla_\theta \log p_k^{\theta_m}(X_k, \alpha_k) \Big] ,$$

- update ϕ by minimizing the square regression error from the DP relation :

$$\phi_{m+1} \in \arg\min_{\phi} \mathbb{E}_{\pi^{\theta_m}} \Big[\Big| \sum_{k=0}^{N-1} r_k(X_k, \alpha_k) + \mathcal{V}_{k+1}^{\phi_m}(X_{k+1}) - \mathcal{V}_k^{\phi}(X_k) \Big|^2 \Big] .$$

Algorithm 2: Actor/Critic (offline).

Data: Number of episodes M, mini-batch size K, learning rates η^G , η^V for policy and value function estimation; Parametrized family π^θ and V^ϕ

for $m = 1$ **to** M **do**

  for $k = 1$ **to** K **do**

    Generate initial state $X_0^{(k)}$.

    for $n = 1$ **to** N **do**

      Generate $\alpha_n^{(k)} \sim p_n(X_n^{(k)}, a)\nu(da)$; $X_{n+1}^{(k)} \sim P_n(X_n^{(k)}, \alpha_n^{(k)}, dx)$

    Compute

$$\Gamma_\theta^{(k)} = \sum_{n=0}^{N-1} \left(r_n(X_n^{(k)}, \alpha_n^{(k)}) + \mathcal{V}_{n+1}^\phi(X_{n+1}^{(k)}) - \mathcal{V}_n^\phi(X_k^{(k)})\right) \nabla_\theta \log p_k^\theta(X_n^{(k)}, \alpha_n^{(k)})$$

$$\Delta_\phi^{(k)} = \sum_{n=0}^{N-1} \left(r_n(X_n^{(k)}, \alpha_n^{(k)}) + \mathcal{V}_{n+1}^\phi(X_{n+1}^{(k)}) - \mathcal{V}_n^\phi(X_k^{(k)})\right) \nabla_\theta \mathcal{V}_n^\phi(X_k^{(k)}))$$

  Actor update: $\theta \leftarrow \theta + \eta^G \frac{1}{K} \sum_{k=1}^K \Gamma_\theta^{(k)}$.

  Critic update: $\phi \leftarrow \phi + \eta^V \frac{1}{K} \sum_{k=1}^K \Delta_\theta^{(k)}$.

Result: π^θ, $\mathcal{V}^\phi$.

Algorithm 3: Actor/Critic (online).

Data: Number of episodes M, mini-batch size K, learning rates η^G , η^V for policy and value function estimation ; Parametrized families π^θ and V^ϕ;

for $m = 1$ **to** M **do**

  for $k = 1$ **to** K **do**

    Generate initial state $X_0^{(k)}$.

    for $n = 1$ **to** N **do**

      Generate $\alpha_n^{(k)} \sim p_n(X_n^{(k)}, a)\nu(da)$; $X_{n+1}^{(k)} \sim P_n(X_n^{(k)}, \alpha_n^{(k)}, dx)$

      Actor update

$$\theta \leftarrow \theta + \eta^G \big(r_n(X_n^{(k)}, \alpha_n^{(k)}) + \mathcal{V}_{n+1}^{\phi}(X_{n+1}^{(k)}) - \mathcal{V}_n^{\phi}(X_k^{(k)})\big) \nabla_\theta \log p_k^\theta(X_n^{(k)}, \alpha_n^{(k)}) \, .$$

      Critic update

$$\phi \leftarrow \phi + \eta^V \big(r_n(X_n^{(k)}, \alpha_n^{(k)}) + \mathcal{V}_{n+1}^{\phi}(X_{n+1}^{(k)}) - \mathcal{V}_n^{\phi}(X_k^{(k)})\big) \nabla_\theta \mathcal{V}_n^{\phi}(X_k^{(k)}))$$

Result: π^θ, $\mathcal{V}^\phi$.