Vega - RL

Marc Sabate-Vidales
The University of Edinburgh

Outline

Environment Dynamics

Finite MDP consits of:

- Finite sets of states $\mathcal{S}$, actions A.
- Environment dynamics. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. For $a \in A, y, y' \in \mathcal{S}$ **we are given** $p^a(y, y')$ of a discrete time Markov chain $(X_n^\alpha)_{n=0,1,\dots}$ so that

$$\mathbb{P}(X_{n+1}^\alpha = y' | X_n^\alpha = y) = p^{\alpha_n}(y, y')$$

- α is the control process. α is measurable with respect to $\sigma(X_k^\alpha, k \leq n)$. In other words, α **can't look into the future**.

Value Function

- Let $\gamma \in (0,1)$ be a fixed discount factor
- Let $f : \mathcal{S} \times A \to \mathbb{R}$ be a running reward.
- Our aim is to maximize the expected return

$$J^\alpha(x) = \mathbb{E}^x \left[\sum_{n=0}^{\infty} \gamma^n f(\alpha_n, X_n^\alpha) \right]$$

  over all controlled processes, where $\mathbb{E}^x := \mathbb{E}[\cdot | X_0^\alpha = x]$
- For all $x \in \mathcal{S}$, we define the value function and the optimal value function as

$$v^\alpha(x) = J^\alpha(x), \quad v^*(x) := \max_{\alpha \in \mathcal{A}} J^\alpha(x)$$

Dynamic Programming for controlled Markov Processes

Theorem (DPP)

Let f be bounded. Then for all $x \in S$ we have

$$v^*(x) = \max_{a \in A} \mathbb{E}^x \left[f^a(x) + \gamma v^*(X_1^a) \right]$$

Corollary

Among all admissible control processes, it is enough to consider the ones that depend only on the current state.

Policy Iteration

Start with initial guess of $\alpha^0(x_i)$ for $i = 1, \dots, |\mathcal{S}|$. Let $V^k(x_i), \alpha^k(x_i)$ be defined through the iterative procedure

1. **Evaluate** the current policy

$$V^{k+1}(x_i) = f(x_i, \alpha^k(x_i)) + \gamma \underbrace{\mathbb{E}\left[V^k(X_1^{\alpha_k}) | X_0^{\alpha_k} = x_i\right]}_{p^{\alpha^k(x_i)}(y,y') \text{ needed!}}$$

2. **Improve** the policy

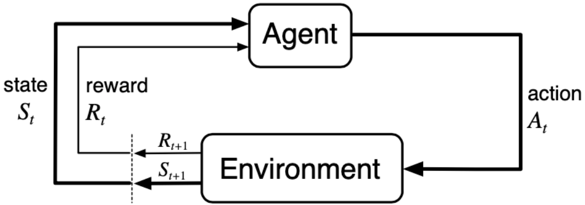
$$\alpha^{k+1}(x_i) \in \arg\max_{a \in A} f(x_i, a) + \gamma \overbrace{\mathbb{E}\left[V^{k+1}(X_1^a) | X_0^a = x_i\right]}^{p^a(y,y') \text{ needed!}}$$

Outline

1. Markov Decision Process (MDP)
2. Reinforcement Learning
    - Q-learning
    - Policy Gradient

Remark

In policy iteration, we need to know the transition probabilities $p^a(y, y')$), f and g! This is not the usual case. The alternative is to learn the policy from data, collected from interacting with the environment.



Q-learning

Definition (Q-function)

$$Q^{\alpha}(x,a) := r(x,a) + \gamma \mathbb{E}[v^{\alpha}(X_1^a)]$$

$$Q^{*}(x,a) := r(x,a) + \gamma \mathbb{E}[v^{*}(X_1^a)]$$

From DPP, we know that, $\max_a Q^*(x,a) = v^*(x)$, therefore

$$Q^*(x,a) = r(x,a) + \gamma \mathbb{E}^x[\max_{b \in A} Q^*(X_1^a, b)].$$

Re-arranging,

$$0 = r(x,a) + \gamma \mathbb{E}^x[\max_{b \in A} Q^*(X_1^a, b)] - Q^*(x,a)$$

Q-learning Algorithm - Stochastic Approximation

Stochastic approximation arises when one wants to find the root θ^* of the following expression

$$0 = C(\theta) := \mathbb{E}_{X\sim\mu}(c(X,\theta))$$

If we have access to unbiased approximations of $C(\theta)$, namely $\tilde{C}(\theta)$, then the following updates

$$\theta \leftarrow \theta - \delta_n \tilde{C}(\theta)$$

with $\delta_n \in (0,1)$ satisfying

$$\sum_n \delta_n = +\infty, \quad \sum_n \delta_n^2 < +\infty$$

will converge to θ^*

Going back to Q-learning, we want to find an unbiased approximation of

$$r(x,a) + \gamma \mathbb{E}^x[\max_{b\in A} Q^*(X_1^a, b)] - Q^*(x,a)$$

Q-learning Algorithm

Recall $\mathcal{S}, A$ are finite (they can be big). Transition probabilities, running cost and final cost are unknown, but we can observe tuples (x_n, a_n, r_n, x_{n+1}) from interacting with the environment.

1. Make initial guess, for $Q^*(x,a)$ denoted by $Q(x,a)$ for all x, a.
2. We select and perform an action a (either by following the current policy, or by doing some sort of exploration).
3. We select the state we landed in, denoting it by y. If it is not terminal, adjust

   $$Q(x,a) \leftarrow Q(x,a) + \delta_n \left(r(x,a) + \gamma \max_{b \in A} Q(y,b) - Q(x,a) \right)$$

   Note: we are doing Stochastic Approximation using $max_{b \in A} Q(y,b)$ as an unbiased approximation of $\mathbb{E}^x[\max_{b \in A} Q(X_1^a, b)]$.
4. Go back to (2)

Q-learning Algorithm - Function approximation

In practice, the state space might be very large (or continuous). It is then infeasible to sample (x_n, a, r, x_{n+1}) to explore all the space.
Alternatively, Q can be approximated with a Neural Network with parameters θ.
The optimal policy will be defined as $\alpha(x) = \max_{b \in A} Q_{\theta^*}(x, a)$ for some optimal parameters θ^*.

1. Initialise network's parameters θ.
2. Sample tuples $(x_n, a_n, r_n, x_{n+1})_{n=1,\dots,M}$ from the environment, using some exploration-exploitation heuristics.
3. Find θ^* that minimise the L_2-error

$$J(\theta) = \frac{1}{2} \mathbb{E}_{x,a\sim\mu} \left(Q_\theta(x, a) - (r(x, a) + \gamma \mathbb{E}^x \max_{b \in A} Q_{\bar{\theta}}(X, b)) \right)^2$$

where μ is the empirical measure of the visited action-states, using gradient ascent. We use the following approximation of the gradient

$$\nabla_\theta J = \mathbb{E}_{x,a\sim\mu} \left(Q_\theta(x, a) - (r(x, a) + \gamma \mathbb{E}^x \max_{b \in A} Q_{\bar{\theta}}(X, b)) \right) \nabla_\theta Q_\theta(x, a)$$

Soft Policies

From DPP it follows that the optimal policy is a deterministic function of the state. In practice, since the environment and the running cost/reward function are unkown, we will use **soft policies**,

$$\pi : \mathcal{S} \to \mathcal{P}(A)$$

where $\mathcal{P}(A)$ is the space of probability meaures on A.
I will abuse the notation, and I will indistinctively use $\pi(\cdot|x)$ for the distribution, the probability mass function (or the density) of $\pi(x)$.

Remark (Relationship between the value function and the Q-function)

$$v^\pi(x) = \mathbb{E}_{A\sim\pi(\cdot|x)} Q(x, A)$$

Policy Gradient for Soft Policies I

Consider a soft (random) policy with probability mass function $\pi_\theta(\cdot|x)$ parametrised by some parameters θ. Let ρ be some initial state distribution.
Instead of finding the optimal policy through the Q-function, we directly maximise the expected return for all $x \in S$.

$$J^{\pi_\theta}(\theta) = \mathbb{E}_{A_n \sim \pi(\cdot|X_n)} \left[\sum_{n=0}^{\infty} \gamma^n r(A_n, X_n^\alpha) \middle| X_0 \sim \rho \right]$$

Assume we know an expression for $\nabla_\theta J^{\pi_\theta}$ (next slide). Then $\arg\max_\theta J^{\pi_\theta}(\theta)$ is found using gradient ascent using a learning rate τ

$$\theta \leftarrow \theta + \tau \cdot \nabla_\theta J^{\pi_\theta}$$

Policy Gradient for Soft Policies II

We need to find an expression for $\nabla_\theta J^{\pi_\theta}$. This is given by The Policy Gradient Thm, Section 13.2 in [Sutton and Barto, 2018]

Theorem (Policy Gradient Theorem)

$$
\begin{aligned}
\nabla_\theta J^{\pi_\theta}(\theta) &\propto \sum_{x \in \mathcal{S}} \mu(x) \sum_{a \in A} \nabla_\theta \pi_\theta(a|x) Q_{\pi_\theta}(x, a) \\
&\propto \mathbb{E}_{X_n \sim \mu}\left[\mathbb{E}_{A_n \sim \pi_\theta(\cdot|X_n)} \nabla_\theta \log(\pi_\theta(A_n|X_n)) Q_{\pi_\theta}(X_n, A_n)\right]
\end{aligned}
$$

where μ is the visitation measure.

We need to approximate Q_{π_θ}!

Policy Gradient for Deterministic Policies

If we have a deterministic policy with continuous actions $\alpha_\alpha : \mathcal{S} \to A$, then the Deterministic Policy Gradient for Reinforcement Learning with continuous actions is given by Theorem 1 in [Silver et al., 2014]

Theorem

$$\nabla_\theta J^{\alpha_\theta}(\theta) = \mathbb{E}_{X_n \sim \mu}\left[\nabla_\theta \alpha_\theta(x) \nabla_a Q_{\alpha_\theta}(X_n, \alpha_\theta(s))\right]$$

We need to approximate Q_{α_θ}

Actor-Critic type Algorithms

Policy Gradient theorems include the Q-function. In practice, one can either

- approximate it using Monte Carlo (i.e. by simulating several games starting from (x, a) and approximate it with the average). This is expensive and might have a high variance.
- Using a function approximation $Q_\psi(x, a)$ with parameters ψ. This motivates **actor-critic** algorithms:
  1. **Policy evaluation**: approximate the Q-function (the critic) using for example the Bellman equation.

  $$\psi^* = \arg\max_\psi \frac{1}{2}\,\mathbb{E}_{x,a\sim\mu}\left(Q_\psi(x,a) - (r(x,a) + \gamma\mathbb{E}^x v_{\bar{\psi}}(X))\right)^2$$

     where we recall that $v_{\bar{\psi}}(X) = \mathbb{E}_{a\sim\pi_\theta(\cdot|X)}[Q_\psi(X, a)]$
  2. **Policy improvement** improve the policy (the actor) with gradient ascent using the Policy Gradient theorems.

References I

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014).
Deterministic policy gradient algorithms.
In *International conference on machine learning*, pages 387–395. PMLR.

Sutton, R. S. and Barto, A. G. (2018).
Reinforcement learning: An introduction.
MIT press.