Introduction to Reinforcement Learning: Part V Q function and Q learning

Dr. Mayank S JHA

Maitre de Conférences (Associate Professor),
CRAN,
UMR 7039, CNRS,
Polytech Nancy,
Office: C 225,
Université de Lorraine
France.

Table of Contents

Table of Contents

Q-function

Given some fixed policy $\pi(x, u)$, define the Q function for that policy as:

$$\begin{aligned} Q_k^\pi(x, u) &= E_\pi \left\{ r_k + \gamma V_{k+1}^\pi \left(x' \right) \mid x_k = x, u_k = u \right\} \\ &= \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \gamma V_{k+1}^\pi \left(x' \right) \right] \end{aligned}$$

This function is equal to the expected return for taking an arbitrary action u at time k in state x and thereafter following the existing policy $\pi(x, u)$.

Q-function

Note: $V_k^{\pi}(x) = Q_k^{\pi}(x, \pi(x, u))$
Then, the backward recursion in the Q function:

$$Q_k^{\pi}(x, u) = \sum_{x'} P_{xx'}^{u} \left[R_{xx'}^{u} + \gamma Q_{k+1}^{\pi} \left(x', \pi \left(x', u' \right) \right) \right]$$

Q function

The conditional expected value:

$$
\begin{aligned}
Q_k^*(x,u) &= \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \gamma V_{k+1}^*\left(x'\right)\right] \\
&= E_\pi \left\{r_k + \gamma V_{k+1}^*\left(x'\right) \mid x_k = x, u_k = u\right\}
\end{aligned}
$$

is known as the optimal Q function.
The letter Q comes from "quality function."
The Q function is also called *action-value* function.

Bellman Optimality

In terms of the Q function, the Bellman optimality equation has the particularly simple form

$$V_k^*(x) = \min_u Q_k^*(x, u),$$

Q function

$$u_k^* = \arg\min_u Q_k^*(x, u).$$

- V-function based minimization requires knowledge of the state transition probabilities, which correspond to the system dynamics, and costs.
- Q-function based the minimization requires knowledge only of the Q function and not the system dynamics.
- Q function contains information about control actions in every state. As such, the best control in each state can be selected using by knowing only the Q function.
- the Q function can be estimated online in real time directly from date observed along the system trajectories, without knowing the system dynamics information, that is, the

Observations

- The Q function is a function of both the current state x and the action u.
- By contrast, the value function is a function of the state.
- For finite MDP, the Q function can be stored as a lookup table for each state/action pair.

Q function based analysis

The infinite-horizon Q function for a prescribed fixed policy is given by

$$Q^{\pi}(x,u) = \sum_{x'} P^{u}_{xx'}\left[R^{u}_{xx'} + \gamma V^{\pi}\left(x'\right)\right]$$

The Q function also satisfies a Bellman equation. Given a fixed policy $\pi(x,u)$,

$$V^{\pi}(x) = Q^{\pi}(x, \pi(x,u)),$$

Hence, the Q function satisfies the Bellman equation

$$Q^{\pi}(x,u) = \sum_{x'} P^{u}_{xx'}\left[R^{u}_{xx'} + \gamma Q^{\pi}\left(x', \pi\left(x',u'\right)\right)\right].$$

Bellman Optimality Q-function

Bellman optimality equation for the Q function is

$$Q^*(x,u) = \sum_{x'} P^u_{xx'} \left[R^u_{xx'} + \gamma Q^* \left(x', \pi^* \left(x', u' \right) \right) \right]$$

$$Q^*(x,u) = \sum_{x'} P^u_{xx'} \left[R^u_{xx'} + \gamma \min_{u'} Q^* \left(x', u' \right) \right].$$

Compare to :

$$V^*(x) = \min_u \sum_{x'} P^u_{xx'} \left[R^u_{xx'} + \gamma V^* \left(x' \right) \right]$$

, where the minimum operator and the expected value operator are reversed.

Policy iteration and value iteration are especially easy to implement in terms of the Q function.

Table of Contents

PI Algorithm

PI using Q function

Policy Evaluation (Value Update)

$$Q_j(x,u) = \sum_{x'} P^u_{xx'} \left[R^u_{xx'} + \gamma Q_j\left(x', \pi\left(x', u'\right)\right)\right], \text{ for all } x \in X$$

Policy Improvement

$$\pi_{j+1}(x,u) = \arg\min_u Q_j(x,u), \text{ for all } x \in X.$$

Observations

- These algorithms can be implemented online in real time, without knowing the system dynamics, by measuring data along the system trajectories.
- These algorithms are an implementation of optimal adaptive control, that is, adaptive control algorithms that converge online to optimal control solutions.

Dr. Mayank S JHA, mayank-shekhar.jha@univ-lorraine.fr
Polytech Nancy, CRAN, University of Lorraine, France

Table of Contents

Q learning motivation

- Q learning reinforcement learning method results in an adaptive control algorithm that converges online to the optimal control solution for completely unknown systems.
- It solves the Bellman equation and the HJB equation online in real time by using data measured along the system trajectories, without any knowledge of the dynamics $f(x_k), g(x_k)$.

Q-learning

Q learning learns the Q function using temporal difference methods by performing an action u_k and measuring at each time stage the resulting data experience set (x_k, x_{k+1}, r_k) consisting of the current state, the next state, and the resulting stage cost. Writing the Q function Bellman equation along a sample path gives

$$Q^{\pi}(x_k, u_k) = r(x_k, u_k) + \gamma Q^{\pi}(x_{k+1}, h(x_{k+1})),$$

which defines a temporal difference error

$$e_k = -Q^{\pi}(x_k, u_k) + r(x_k, u_k) + \gamma Q^{\pi}(x_{k+1}, h(x_{k+1})).$$

Dr. Mayank S JHA, mayank-shekhar.jha@univ-lorraine.fr Polytech Nancy, CRAN, University of Lorraine, France

Q-learning

The Q function is updated using the algorithm

$$
\begin{aligned}
Q_k(x_k, u_k) = & Q_{k-1}(x_k, u_k) + \alpha_k \big[r(x_k, u_k) \\
& + \gamma \min_u Q_{k-1}(x_{k+1}, u) - Q_{k-1}(x_k, u_k) \big]
\end{aligned}
$$

It is shown the algorithm converges for a finite MDP provided that all state-action pairs are visited infinitely often and

$$
\sum_{k=1}^{\infty} \alpha_k = \infty, \quad \sum \alpha_k^2 < \infty
$$

which are standard stochastic approximation conditions. On convergence, the temporal difference error is approximately equal to zero.

The requirement that all state-action pairs are visited infinitely

Q-learning

For nonlinear systems, the Q function is parameterized as

$$Q(x, u) = W^T \phi(z),$$

for some unknown parameter vector W and basis set vector $\phi(z)$. Substituting the Q function approximation into the temporal difference error yields:

$$e_k = -W^T \phi\left(z_k\right) + r\left(x_k, u_k\right) + \gamma W^T \phi\left(z_{k+1}\right),$$

on which either policy iteration or value iteration algorithms can be based.

Q-learning

Considering the policy iteration algorithm:

$$W_{j+1}^{T}\left(\phi\left(z_k\right)-\gamma\phi\left(z_{k+1}\right)\right)=r\left(x_k,h_j\left(x_k\right)\right),$$

and the policy improvement step

$$h_{j+1}\left(x_k\right)=\arg\min_{u}\left(W_{j+1}^{T}\phi\left(x_k,u\right)\right),\text{ for all } x\in X.$$

These equations do not require knowledge of the dynamics $f(\cdot), g(\cdot)$.

Implementation

- For online implementation, batch least-squares or RLS can be used to solve for the parameter vector W_{j+1} given the regression vector $\left(\phi\left(z_k\right)-\gamma\phi\left(z_{k+1}\right)\right)$,
- The observed data at each time instant are $\left(z_k, z_{k+1}, r\left(x_k, u_k\right)\right)$ with $z_k \equiv \left[x_k^T u_k^T\right]^T$.
- The vector $z_{k+1} \equiv \left[\begin{array}{cc} x_{k+1}^T & u_{k+1}^T \end{array}\right]^T$ is computed using $u_{k+1}=h_j\left(x_{k+1}\right)$ with $h_j(\cdot)$ being the current policy.

Implementation

- Probing noise must be added to the control input to obtain persistence of excitation.
- On convergence, the action update is performed.
- This update is easily accomplished without knowing the system dynamics due to the fact that the Q function contains u_k as an argument, therefore $\partial\left(W_{j+1}^{T}\phi\left(x_k, u\right)\right)/\partial u$ can be explicitly computed.
- Due to the simple form of the action update, the actor neural network is not needed for Q learning; it can be implemented using only one neural network for Q function approximation.

References I