Two time scale stochastic approximation for reinforcement learning with linear function approximation

Pedro A. Santos

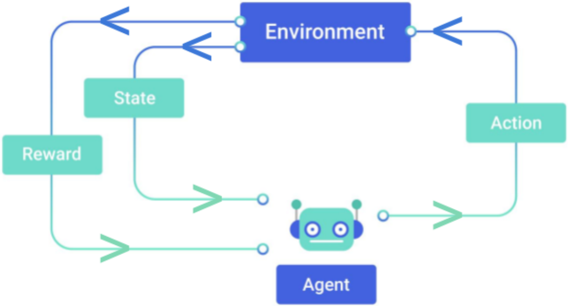
Joint work with Diogo Carvalho and Francisco Melo

MPML April 9th 2021

Contents

Contents

Reinforcement learning



$x_0, a_0, r_0, x_1, a_1, r_1, x_2, a_2, r_2, x_3 \; \ldots$

Contents

Markov Decision Processes

Definition (MDP)

A Markov decision process is a tuple $\{\mathcal{X}, \mathcal{A}, \mathcal{P}, r, \gamma\}$, where

- $\mathcal{X}$ denotes a finite set of n states;
- $\mathcal{A}$ a finite set of m actions;
- $\mathcal{P}$ is a set of $n \times n$ stochastic matrices P_a associated with each action $a \in \mathcal{A}$ with entries $[P_a]_{x,y} \in [0, 1]$ representing the probability that the state transitions from x to y given that the action a was performed;
- $R : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is a reward stochastic mapping;
- $\gamma \in [0, 1[$ is a discount factor.

The Markov Zoo

States	Actions	State Knowledge	
No	Yes		Multi-armed Bandit
Yes	No	Yes	Markov Process
Yes	No	No	Hidden MM
Yes	Yes	Yes	MDP
Yes	Yes	No	POMDP

Policy

Definition (Policy)

A policy $\pi : \mathcal{X} \to \Delta(\mathcal{A})$ is a stochastic map from states to actions.

The state value function of a given policy π:

Definition (State value function)

The state value function $V_\pi : \mathcal{X} \to \mathbb{R}$ is

$$V_\pi(x) := \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_t | x_0 = x],$$

where the expectation is taken with respect to the states $x_{t+1} \sim [P_{a_t}]_{x_t,\cdot}$ and the actions $a_t \sim \pi(x_t)$.

The Q function

Definition (State action value function)

The state action value function $Q_\pi : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is

$$Q_\pi(x, a) := \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_t | x_0 = x, a_0 = a],$$

where the expectation is taken with respect to the states $x_{t+1} \sim [P_{a_t}]_{x_t,\cdot}$ and the actions $a_t \sim \pi(x_t)$.

The Q function

,	Action 1	Action 2
State 1	0	5
State 2	0	5
State 3	0	5
State 4	20	0

The Prediction Problem

Evaluate a policy π by approximating its state value function V_π

Proposition (Fixed point equation for the state value function)

The following relation holds:

$$V_\pi(x) = \mathbb{E}[R(x,a) + \gamma V_\pi(y)], \tag{1}$$

where the expectation is taken with respect to the next state $y \sim [P_a]_{x,\cdot}$ *and the action* $a \sim \pi(x)$.

The Control Problem

Find a policy π^* that maximizes the cumulative reward $V_{(\cdot)}$

A policy π is better than another policy π' if $V_\pi(x) \geq V_{\pi'}(x)$ for all $x \in \mathcal{X}$.

Definition

An optimal policy π^* is any such that, for any policy π,

$$V_{\pi^*}(x) \geq V_\pi(x),$$

for all $x \in \mathcal{X}$.

Optimal Policy

Theorem (Existence of solution)

There exists an optimal policy π^*.

Corollary (Optimal Value Equation)

The state value function of the optimal policy π^*, which we shall denote by V^*, verifies

$$V^*(x) = \max_{a \in \mathcal{A}} \mathbb{E}[R(x,a) + \gamma V^*(y)] \tag{2}$$

for all $x \in \mathcal{X}$, where the expectation is taken with respect to the next state $y \sim P_{a_{x},\cdot}$.

Optimal state value function

Letting Q^* denote Q_{π^*}, we have that

Proposition (Optimal Q equation)

The following relation holds:

$$Q^*(x, a) = \mathbb{E}[R(x, a) + \gamma \max_{a' \in \mathcal{A}} Q^*(y, a')], \tag{3}$$

where the expectation is taken with respect to the next state $y \sim [P_a]_{x,\cdot}$ *and the action* $a' \sim \pi(x)$

Finding V_π

- As a system of $n \times n$ linear equations:

$$\begin{aligned} V_\pi(x) &= \mathbb{E}[R(x,a) + \gamma V_\pi(y)] \\ &= \sum_a \pi(a|x) \sum_{y,r} p(y,r|x,a)(r + \gamma V_\pi(y)) \end{aligned}$$

$$V_\pi = TV_\pi$$

- Iterative policy evaluation:

$$V_{k+1} = TV_k$$

Approximating π^*

Given V_π,

$$\pi(x) \leftarrow \underset{a}{\operatorname{argmax}} \sum_{y,r} p(y, r|x, a)(r + \gamma V_\pi(y))$$

Policy Iteration

$$\pi_0 \rightarrow V_{\pi_0} \rightarrow \pi_1 \rightarrow V_{\pi_1} \rightarrow \pi_2 \rightarrow \ldots \approx \pi^*$$

Problems with this simplistic approach

- We need to work with the whole state space

- We need to know the model of the world

Contents

Who needs a model?

Model-free Methods

- Monte Carlo Methods
- Temporal-Difference Learning

Q-learning

Q-learning algorithm for estimating π^*

Initialize $Q(x, a)$ for all $x \in \mathcal{X}$ and $a \in \mathcal{A}$ (e.g. $Q(x, a) = 0$);
repeat for each Episode
    Choose an initial state x;
    repeat for each step of Episode
        Choose action a using policy derived from Q (e.g. ϵ-greedy);
        Execute a, observe reward r and new state x';
        $Q(x, a) \leftarrow (1 - \alpha)Q(x, a) + \alpha\big(r + \gamma \max_{a'} Q(x', a')\big)$;
        $x \leftarrow x'$
    until x *is terminal*;
until *satisfied*;

Q-learning

$$Q(x, a) \leftarrow (1 - \alpha)Q(x, a) + \alpha\big(\ r\ +\ \gamma \max_{a'} Q(x', a')\ \big)$$

Q-learning

$$Q(x,a) \leftarrow (1-\alpha)Q(x,a) + \alpha\big(\, r \,+\; \gamma \max_{a'} Q(x',a')\;\big)$$

Q-learning

$$Q(x,a) \leftarrow (1-\alpha)Q(x,a) + \alpha\big(r + \gamma \max_{a'} Q(x',a') \big)$$

Q-learning

$$Q(x,a) \leftarrow Q(x,a) + \alpha\big(\, r \,+\, \gamma \max_{a'} Q(x',a') \,- Q(x,a)\big)$$

$$Q(x,a) \leftarrow Q(x,a) + \alpha\delta$$

with $\delta = r + \gamma \max_{a'} Q(x',a') - Q(x,a)$

Q-learning with function approximation

We wish to approximate Q^* using $\mathcal{Q} = \{Q_w : w \in \mathbb{R}^k\}$

Fixed point equation for the optimal state action value function

$$Q^*(x,a) = \mathbb{E}[R(x,a) + \gamma \max_{a' \in \mathcal{A}} Q^*(y,a')].$$

Loss function:

$$L(w) = \frac{1}{2}\mathbb{E}_\mu[(Q^*(x,a) - Q_w(x,a))^2]$$

$$w \leftarrow w + \alpha \mathbb{E}_\mu[(\ Q^*(x,a)\ - Q_w(x,a))\nabla_w Q_w(x,a)]$$

$$w \leftarrow w + \alpha \mathbb{E}_\mu[(\boxed{Q^*(x,a)} - Q_w(x,a))\nabla_w Q_w(x,a)]$$

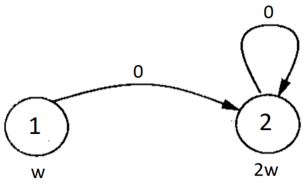
$$w \leftarrow w + \alpha \mathbb{E}_\mu[(\boxed{R(x,a) + \gamma \max_{a' \in \mathcal{A}} Q_w(y,a')} - Q_w(x,a))\nabla_w Q_w(x,a)]$$

Q-learning with function approximation

The $w \to 2w$ example (Tsitsiklis and Van Roy 1996)

Consider the state space $\mathcal{X} = \{x_1, x_2\}$, one action, all rewards 0, and the transition matrix

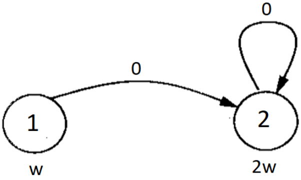
$$P = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$



$$Q^* = V = 0$$

Q-learning with function approximation

The $w \to 2w$ example (Tsitsiklis and Van Roy 1996)



$$Q^* = V = 0$$

$$\mathcal{Q} = \{w\phi,\ w \in \mathbb{R}\}$$

with $\phi : \mathcal{X} \to \mathbb{R}$ such that $\phi(x_1) = 1, \phi(x_2) = 2$

The spiral example (Tsitsiklis and Van Roy 1997)

Markov chain

$\mathcal{X} = \{s_1, s_2, s_3\}$, $\mathcal{A} = \{a\}$ and

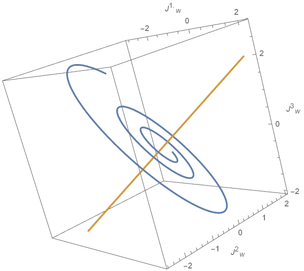
$$\mathrm{P} = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}.$$

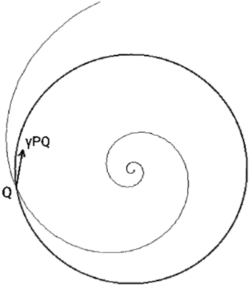
Approximation architecture

$$\frac{dQ_w}{dw} = (S + \epsilon \mathrm{I}) Q_w,$$

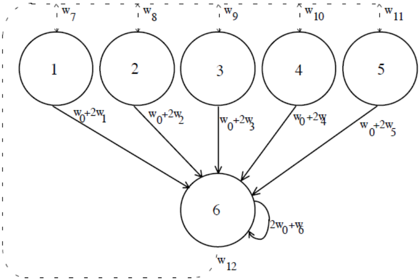
where ϵ is very small and

$$S = \begin{bmatrix} 1 & \frac{1}{2} & \frac{3}{2} \\ \frac{3}{2} & 1 & \frac{1}{2} \\ \frac{1}{2} & \frac{3}{2} & 1 \end{bmatrix}.$$





MDP Example (Baird 1995)

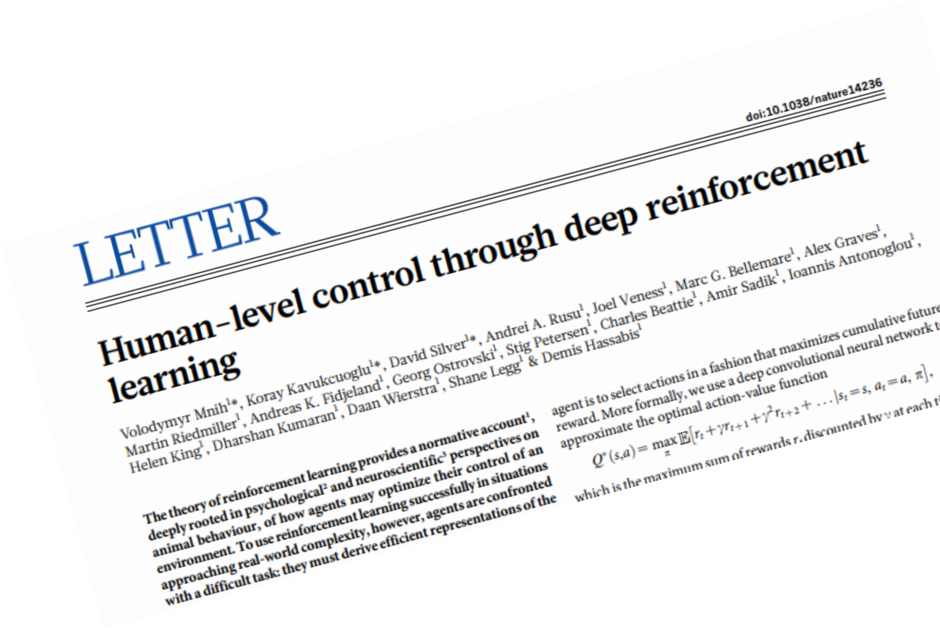


$Q_0(\text{Solid}) > Q_0(\text{Dashed})$
$Q_0(6 \to 6)$ largest

The Deadly Triad

- Function Approximation
- Bootstraping
- Off-policy training

Deepmind's breakthrough



doi:10.1038/nature14236

LETTER

Human-level control through deep reinforcement learning

Volodymyr Mnih[1*], Koray Kavukcuoglu[1*], David Silver[1*], Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]

The theory of reinforcement learning provides a normative account[1], deeply rooted in psychological[2] and neuroscientific[3] perspectives on animal behaviour, of how agents may optimize their control of an environment. To use reinforcement learning successfully in situations approaching real-world complexity, however, agents are confronted with a difficult task: they must derive efficient representations of the

agent is to select actions in a fashion that maximizes cumulative future reward. More formally, we use a deep convolutional neural network to approximate the optimal action-value function

$$Q^*(s,a) = \max_\pi \mathbb{E}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \mid s_t = s,\, a_t = a,\, \pi\right],$$

which is the maximum sum of rewards r_t discounted by γ at each ti

Deepmind's breakthrough

Two special techniques used:

- Replay Buffer
- Target Network

$$w \leftarrow w + \alpha \mathbb{E}_\mu[(\boxed{R(x,a) + \gamma \max_{a' \in \mathcal{A}} Q_w(y,a')} - Q_w(x,a))\nabla_w Q_w(x,a)]$$

$$w \leftarrow w + \alpha \mathbb{E}_\mu[(\boxed{R(x,a) + \gamma \max_{a' \in \mathcal{A}} Q_u(y,a')} - Q_w(x,a))\nabla_w Q_w(x,a)]$$

Contents

Stochastic Approximation

$$w = h(w) = \mathbb{E}_\mu[H(w,X)]$$

$$w \leftarrow w + \alpha(\ \mathbb{E}_\mu[H(w,X)] \ - w)$$

$$w \leftarrow w + \alpha(\ \mathbb{E}_\mu[H(w,X)] \ - w)$$

$$\begin{aligned}
w_{t+1} &= w_t + \alpha\big(H(w_t, x_t) - w_t\big) \\
&= w_t + \alpha\big(H(w_t, x_t) - w_t + h(w) - h(w)\big) \\
&= w_t + \alpha\big(h(w) - w + \ H(w_t, x_t) - h(w) \ \big) \\
&= w_t + \alpha\big(h(w) - w + \ H(w_t, x_t) - h(w) \ \big) \\
&= w_t + \alpha\big(h(w) - w + M_t\big)
\end{aligned}$$

Stochastic Approximation

The O.D.E. approach

The equation

$$w_{t+1} = w_t + \alpha_t\big(h(w_t) - w + M_t\big)$$

can be thought as a noisy discretization for the o.d.e.

$$\dot{w} = h(w) - w$$

$$\sum_t \alpha_t = \infty, \qquad \sum_t \alpha_t^2 < \infty$$

M_t is a Martingale difference sequence

Two time scale stochastic approximation

$$\begin{cases} v_{t+1} = v_t + \alpha_t(f(v_t, u_t) + M_{t+1}) \\ u_{t+1} = u_t + \beta_t(g(v_t, u_t) + N_{t+1}) \end{cases}, t \in \mathbb{N}$$

Assumptions:

- $\sum_t \beta_t = \infty, \sum_t \alpha^2 < \infty,\ \beta_t/\alpha_t \to 0$
- $f : \mathbb{R}^{k+d} \to \mathbb{R}^k$, $g : \mathbb{R}^{k+d} \to \mathbb{R}^d$ are locally Lipschitz
- $\sup_t(||v_t|| + ||u_t||) < \infty \quad w.p.1.$
- $M_t \in \mathbb{R}^k$ and $N_t \in \mathbb{R}^d$ are Martingale difference sequences and

$$\mathbb{E}[||M_t||^2] \leq c_M(1 + ||v_t||^2 + ||u_t||^2)$$
$$\mathbb{E}[||N_t||^2] \leq c_N(1 + ||v_t||^2 + ||u_t||^2)$$

Two time scale stochastic approximation

$$\begin{cases} v_{t+1} = v_t + \alpha_t(f(v_t, u_t) + M_{t+1}) \\ u_{t+1} = u_t + \beta_t(g(v_t, u_t) + N_{t+1}), \end{cases}, t \in \mathbb{N}$$

Assumptions (cont):

- The o.d.e.

$$\dot{v}_t = f(v_t, u)$$

  has a unique globally asymptotically stable equilibrium $\lambda(u)$, where $\lambda : \mathbb{R}^d \to \mathbb{R}^k$ is a Lipschitz-continuous function.
- The o.d.e.

$$\dot{u}_t = g(\lambda(u_t), u_t)$$

  has a unique globally asymptotically stable equilibrium u^*.

Two time scale stochastic approximation

Theorem (Borkar 2008)

If the assumptions are true, then

$$(v_t, u_t) \rightarrow^{a.s.} (\lambda(u^*), u^*).$$

Contents

Coupled Q-learning

Update rule

$$v_{t+1} = v_t + \alpha_t\big(r_t + \max_{a'\in\mathcal{A}} Q_{u_t}(y_t, a') - Q_{v_t}(x_t, a_t)\big)\nabla Q_{v_t}(x_t, a_t)$$
$$u_{t+1} = u_t + \beta_t(\nabla Q_{v_t}(x_t, a_t)Q_{v_t}(x_t, a_t) - u_t)$$

Each function $Q_w : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is such that

$$Q_w(x, a) = \phi^T(x, a)w$$

we call $\phi : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^k$ the feature vector

Then $\nabla_w Q_w(x, a) = \phi(x, a)$

Convergence result

Assumptions

1. State-action pairs are independent and identically distributed to μ;
2. $\Sigma_\mu = \mathbb{E}_\mu[\phi(x_t, a_t)\phi^T(x_t, a_t)]$ is non-singular; $||\phi(x, a)||_2 \leq 1$;
3. $\sum_{t=0}^{\infty} \beta_t = \infty$, $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$ and $\beta_t = o(\alpha_t)$.

Theorem (Carvalho, Melo and Santos, 2020)

Under assumptions 1 through 3, the sequence $\{u^{(t)}, v^{(t)}\}$ generated by coupled Q-learning converges a.s. to a single limit solution (u^*, v^*).

Error bounds

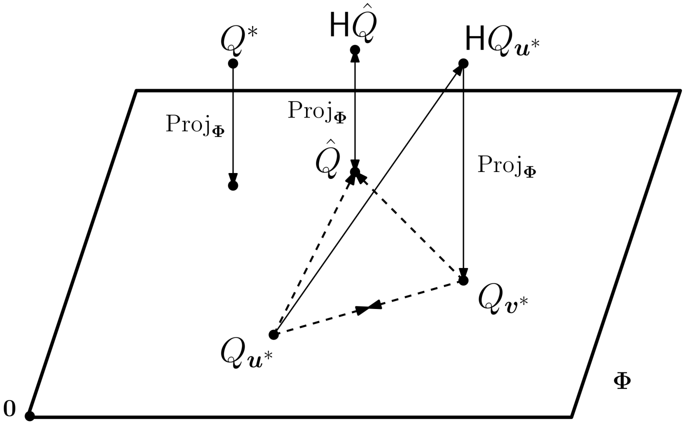
Assume $\{\phi_k\}$ are ortogonal and $\mathbb{E}_\mu[\phi_k] = \sigma$, i.e. $\Sigma_\mu = \sigma I$

Theorem

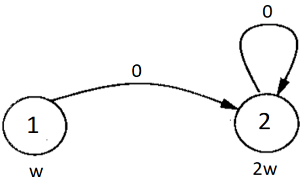
The limit solution Q_{v^*} of coupled Q-learning verifies

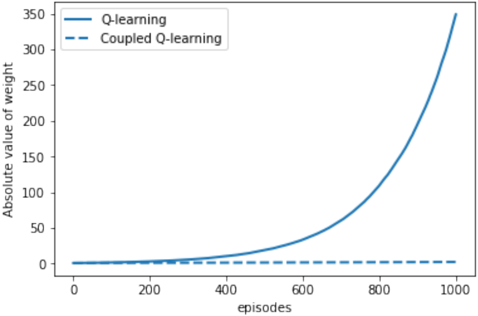
$$\|Q^* - Q_{v^*}\|_\infty \leq \frac{1}{1-\gamma}\|Q^* - Proj_\Phi\, Q^*\|_\infty + \xi_\sigma, \;\; where\ \xi_\sigma = \frac{1-\sigma}{\sigma}\frac{\gamma\rho}{(1-\gamma)^2}$$

Error bounds

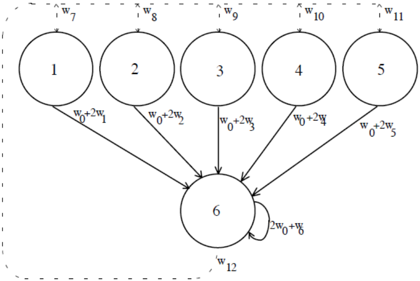


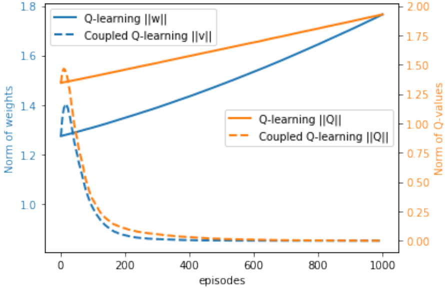
The $\theta \to 2\theta$ example



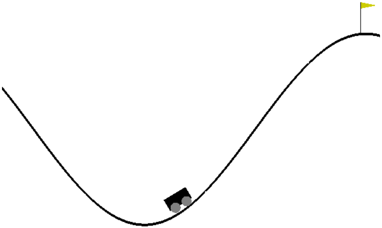


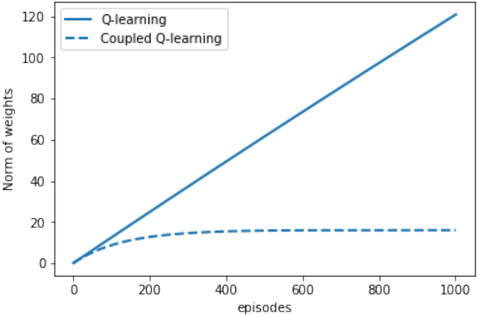
The modified star problem





The mountain car





Other applications of two-time scale stochastic approximation

Double-tap: approximate policy iteration (ongoing work)

$$w_{t+1} = w_t + \alpha_t\big(r_t + \gamma Q_{w_t}(x_{t+1}, a_{t+1}) - Q_{w_t}(x_t, a_t)\big)\nabla Q_{w_t}(x_t, a_t)$$
$$\pi_{t+1} = \pi_t + \beta_t(\Gamma Q_{w_t} - \pi_t),$$

where Γ projects a state action value function to an ϵ-soft policy and is Lipschitz-continuous.

Other applications of two-time scale stochastic approximation

Regular gradient actor critic

$$\begin{aligned} w_{t+1} &= w_t + \alpha_t \big(r_t - j_t + \gamma V_{w_t}(x_{t+1}) - V_{w_t}(x_t) \big) \nabla V_{w_t}(x_t) \\ \theta_{t+1} &= \theta_t + \beta_t \big(r_t - j_t + \gamma V_{w_t}(x_{t+1}) - V_{w_t}(x_t) \big) \nabla \pi_{\theta_t}(x_t, a_t), \end{aligned}$$

where j_t is the average reward at time t.

Contents

Conclusions

Coupled Q-learning

- Is variant of Q-learning
- The algorithm converges with linear function approximation
- This method also allows to transform outer-inner-cycle algorithms into one cycle - two-time step algorithms