Introduction to Reinforcement Learning

Monte Carlo Methods & TD methods

Stéphane Airiau

Université Paris Dauphine

Monte Carlo method : Evaluation of an arbitrary policy π

"first visit" using Exploring Start "cheat"

$q \in \mathbb{R}^{|S|\times|A|}$, $count \in \mathbb{N}^{|S|\times|A|}$, $Acc \in \mathbb{R}^{|S|\times|A|}$
initialise $q(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise $count(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise $Acc(s,a) = 0$ for each $s \in S$ and each action $a \in A$

Repeat
    Sample uniformly an initial state s_0 and an initial action a_0
    Run a complete episode starting from s_0, executing a_0,
        and follow using policy π
    for each transition t, compute G_t
    for each pair (s,a) in the episode (where action a was executed in state s)
        for the **first iteration** t where a was executed in state s in the episode
            $Acc(s,a) \leftarrow Acc(s,a) + G_t$
            $count(s,a) \leftarrow count(s,a) + 1$
            $q(s,a) \leftarrow \frac{Acc(s,a)}{count(s,a)}$

Each sample value G_t is i.i.d $\Rightarrow \lim_{count(s,a)\to\infty} q(s,a) = q_\pi(s,a)$

the standard deviation of the error falls as $\frac{1}{\sqrt{n}}$

Learn an **optimal** policy

Use a similar idea as in policy iteration :
we alternate

- the evaluation of the current policy,
- an improvement of the current policy

⇨ hope to reach the optimal policy !

$$\pi_0 \overset{evaluate}{\rightarrow} q_{\pi_0} \overset{improve}{\rightarrow} \pi_1 \overset{evaluate}{\rightarrow} q_{\pi_1} \rightarrow \cdots \rightarrow \pi_\star \overset{evaluate}{\rightarrow} q_{\pi_\star}$$

- use an infinity of episodes to estimate $q_\pi(s,a)$ with "exploring starts" (at least a large number of episodes !)
- one step greedy improvement $\pi_{k+1}(s) = \text{argmax}_a \, q_{\pi_k}(s,a)$

⇨ same guarantees as policy iteration

can we use a more practical algorithm ?

Learn an **optimal** policy

Use same ideas as in policy iteration

- Use only a criterion of convergence with precision ϵ for estimating $q_\pi(s,a)$
  we could compute some theoretical bounds to estimate the number of episodes required
  ⇨ we can guarantee convergence, but it may not be a practical solution (too costly computationally)
- more extreme : alternate **one** step of evaluation and one step of improvement
  ⇨ we do not have a proof yet (nor a counter example) !

Learn an **optimal** policy : "Monte Carlo with Exploring Starts"

$q \in \mathbb{R}^{|S|\times|A|}$, $count \in \mathbb{N}^{|S|\times|A|}$, $Acc \in \mathbb{R}^{|S|\times|A|}$
initialise $q(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise $count(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise $Acc(s,a) = 0$ for each $s \in S$ and each action $a \in A$

Repeat
    Sample uniformly an initial state s_0 and an initial action a_0
    Run a complete episode starting from s_0, executing a_0,
        and follow using policy π
    for each transition t, compute G_t
    for each pair (s,a) in the episode (where action a was executed in state s)
        for the **first iteration** t where a was executed in state s in the episode
            $Acc(s,a) \leftarrow Acc(s,a) + G_t$
            $count(s,a) \leftarrow count(s,a) + 1$
            $q(s,a) \leftarrow \frac{Acc(s,a)}{count(s,a)}$
            For each s from the episode
                $\pi(s) \leftarrow arg\max_{a\in A} q(s,a)$

no convergence proof nor proof of a counter example!!!
(we do not care so much, exploring starts is not a realistic assumption)

Avoid the "exploring starts" trick

Thus far, we have assumed

the MDP is episodic ⇨ we wait for the end of the episode to perform an update

We can freely choose the initial state of each episode

We would like to

- perform an update **not** at the end of an episode, but **after** a transition (no need to wait for the end of the episode !)
  ⇨ we will see this in the second part of the lecture
- use the MDP's description of the initial state

Avoid the "exploring starts" trick

- to be able to estimate all actions in all states, we need to **explore** : sometimes randomly try something "new"
- A contrario, when we decide to choose the best action according to our current estimate, we will say that we **exploit**

  An important dilemma in reinforcement learning is the tread-off exploration/exploitation
  - exploration may help to discover new and potentially better behaviour but it incurs also many failures (but we are learning these were not good choices)
  - if we exploit too soon, the behaviour may be sub-optimal
- Two variants when using exploration
  - use the statistics to improve the current policy ⇨"**on policy**" approach
  - use the statistics to compute "on the side" a theoretical policy (that we do not use, but that shall turn out to become the optimal policy) ⇨ "**off policy**" approach

"on-policy" Monte Carlo

- we iterate : we generate data using the current policy, we estimate the current policy using that data, and we improve the policy using that data

- use "soft policy" only (take each action with a positive probability) $\forall s \in S, a \in A : \pi(s,a) > 0$ "

➡ ideally, we will want to gradually update this policy to a deterministic (an optimal !) policy.

- For any policy (soft or not) we can always add **exploration**

ex : ϵ-greedy policy based on the Q-table

$$\underset{Q}{\epsilon\text{-greedy}}(s) = \begin{cases} \underset{a \in A}{\operatorname{argmax}} q(s,a) \text{ with probability } 1-\epsilon \\ \\ \text{uniform sample } a \in A \text{ with probability } \epsilon \end{cases}$$

Learning optimal policy : "on policy Monte Carlo"

First visit variant

$q \in \mathbb{R}^{|S|\times|A|}$, $count \in \mathbb{N}^{|S|\times|A|}$, $Acc \in \mathbb{R}^{|S|\times|A|}$
initialise $q(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise $count(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise $Acc(s,a) = 0$ for each $s \in S$ and each action $a \in A$
initialise an arbitrary soft policy π

Repeat
    Run a complete episode using π
    for each transition t, compute G_t
    for each pair (s,a) in the episode (where action a was executed in state s)
        for the **first iteration** t where a was executed in state s in the episode
            $Acc(s,a) \leftarrow Acc(s,a) + G_t$
            $count(s,a) \leftarrow count(s,a) + 1$
            $q(s,a) \leftarrow \dfrac{Acc(s,a)}{count(s,a)}$
        For each s from the episode
            $a^\star \leftarrow arg\max_{a\in A} q(s,a)$ (départage arbitraire des ex-æquos)
            Pour chaque action a

$$\pi(s,a) \leftarrow \begin{cases} 1-\epsilon+\frac{\epsilon}{|A|} \text{ if } a = a^\star \\ \frac{\epsilon}{|A|} \text{ if } a \neq a^\star \end{cases}$$

Improvement check

Let us consider policy π, q its corresponding Q-table and π' the policy derived from π using ϵ-greedy.

Note that for all π, we have $\sum_{a \in A} \left(\pi(s,a) - \frac{\epsilon}{|A|} \right) = 1 - \epsilon$

$$
\begin{aligned}
q_\pi(s,\pi'(s)) &= \sum_{a \in A} \pi'(s,a) q_\pi(s,a) \\
&= \frac{\epsilon}{|A|} \sum_{a \in A} q_\pi(s,a) + (1-\epsilon) \max_{a \in A} q_\pi(s,a) \\
&= \frac{\epsilon}{|A|} \sum_{a \in A} q_\pi(s,a) + (1-\epsilon) \sum_{a \in A} \frac{\pi(s,a) - \frac{\epsilon}{|A|}}{1-\epsilon} \max_{a \in A} q_\pi(s,a) \\
&\geqslant \frac{\epsilon}{|A|} \sum_{a \in A} q_\pi(s,a) + (1-\epsilon) \sum_{a \in A} \frac{\pi(s,a) - \frac{\epsilon}{|A|}}{1-\epsilon} q_\pi(s,a) \\
&\geqslant \frac{\epsilon}{|A|} \sum_{a \in A} q_\pi(s,a) + \sum_{a \in A} \pi(s,a) q_\pi(s,a) - \frac{\epsilon}{|A|} \sum_{a \in A} q_\pi(s,a) \\
&\geqslant \sum_{a \in A} \pi(s,a) q_\pi(s,a) = q_\pi(s,a) \text{ indeed, we improve} \quad \square
\end{aligned}
$$

Convergence

We need to prove convergence to the optimal ϵ-soft policy

We want to show that when we do not find an improvement, we have reached an optimal soft policy (i.e. no other policy can dominate the current policy)

Let us consider an environment ϵ-$\mathcal{E}$ derived from the environment $\mathcal{E}$: when executing action a in ϵ-$\mathcal{E}$, the outcome is to execute a uniform action with probability ϵ in $\mathcal{E}$ and to execute action a with probability $1-\epsilon$ in $\mathcal{E}$.

Let $\widetilde{v}^\star$ and $\widetilde{q}^\star$ be the optimal value functions in ϵ-$\mathcal{E}$.

A policy π is optimal in $\mathcal{E}$ among all soft policies iff $v_\pi = \widetilde{v^\star}$

$\widetilde{v^\star}$ must be the unique solution to

$$\begin{aligned}
\widetilde{v^\star}(s) &= (1-\epsilon)\max_a \widetilde{q^\star}(s,a) + \frac{\epsilon}{|A|}\sum_{a\in A}\widetilde{q^\star}(s,a) \\
&= (1-\epsilon)\max_a \sum_{s',r} p(s',r\mid s,a)\left[r+\gamma\widetilde{v^\star}(s')\right] \\
&\quad + \frac{\epsilon}{|A|}\sum_{a\in A}\sum_{s',r} p(s',r\mid s,a)\left[r+\gamma\widetilde{v^\star}(s')\right]
\end{aligned}$$

When no improvement is found, π must satisfy

$$\begin{aligned}
v_\pi(s) &= (1-\epsilon)\max_a q_\pi(s,a) + \frac{\epsilon}{|A|}\sum_{a\in A} q_\pi(s,a) \\
&= (1-\epsilon)\max_a \sum_{s',r} p(s',r\mid s,a)\left[r+\gamma v_\pi(s')\right] \\
&\quad + \frac{\epsilon}{|A|}\sum_{a\in A}\sum_{s',r} p(s',r\mid s,a)\left[r+\gamma v_\pi(s')\right]
\end{aligned}$$

As $\widetilde{v^\star}$ is the *unique* solution, we must have $\widetilde{v^\star} = v_\pi$

Where we are at

- We can find the optimal ϵ-soft policy
- No need to use the "exploring starts" trick (which is not realistic)

⇨ near optimal policy, with fixed exploration

this is the on-policy approach.

off policy approach

Now we consider the off policy approach.
We use two policies

- one to generate data (our exploration strategy)
- the other is our estimate of the optimal policy
  (that we keep on improving)

off policy approach

Now we consider the off policy approach.
We use two policies

- one to generate data (our exploration strategy)
- the other is our estimate of the optimal policy
  (that we keep on improving)

- Let us assume we follow policy π' (or we use data generated from π')

off policy approach

Now we consider the off policy approach.
We use two policies

- one to generate data (our exploration strategy)
- the other is our estimate of the optimal policy
  (that we keep on improving)

- Let us assume we follow policy π' (or we use data generated from π')
- For any given π, can we estimate v_π using that data.

off policy approach

Now we consider the off policy approach.
We use two policies

- one to generate data (our exploration strategy)
- the other is our estimate of the optimal policy
  (that we keep on improving)

- Let us assume we follow policy π' (or we use data generated from π')
- For any given π, can we estimate v_π using that data.
- we need at least to ensure that $\pi(s,a) > 0 \Rightarrow \pi'(s,a) > 0$

off policy approach

Now we consider the off policy approach.
We use two policies

- one to generate data (our exploration strategy)
- the other is our estimate of the optimal policy (that we keep on improving)

- Let us assume we follow policy π' (or we use data generated from π')
- For any given π, can we estimate v_π using that data.
- we need at least to ensure that $\pi(s,a) > 0 \Rightarrow \pi'(s,a) > 0$
- use the idea of "importance sampling".
  - we have received the rewards $r_{t+1}, r_{t+2}, \ldots, r_T$ using π'
  - Of course, $G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots$ is a sample for $v_\pi(s_t)$ but not for $v_{\pi'}(s_t)$
  - ? Can we weight r_t in our discounted sum to estimate $v_\pi(s_t)$

Let us assume we are in a state S_t, the probability of observing the execution trace $S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2}, \ldots, S_T$ following policy π is

$$\mathbb{P}(A_t, S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2}, \ldots, S_T \mid S_t, \pi)$$

$$\begin{aligned} &= \pi(A_t \mid S_t) T^{A_t}_{S_t, S_{t+1}} \pi(A_{t+1} \mid S_{t+1}) \ldots T^{A_{T-1}}_{S_{T-1}, S_T} \\ &= \prod_{k=t}^{T-1} \pi(A_k \mid S_k) T^{A_k}_{S_k, S_{k+1}} \end{aligned}$$

Let us assume we are in a state S_t, the probability of observing the execution trace $S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2}, \ldots, S_T$ following policy π is

$$\mathbb{P}(A_t, S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2}, \ldots, S_T \mid S_t, \pi)$$

$$\begin{aligned} &= \pi(A_t \mid S_t) T^{A_t}_{S_t, S_{t+1}} \pi(A_{t+1} \mid S_{t+1}) \ldots T^{A_{T-1}}_{S_{T-1}, S_T} \\ &= \prod_{k=t}^{T-1} \pi(A_k \mid S_k) T^{A_k}_{S_k, S_{k+1}} \end{aligned}$$

the relative probability of that execution trace (i.e. the importance sampling ratio) is

$$\rho_{t:T-1} = \frac{\prod_{k=t}^{T-1} \pi(A_k \mid S_k) T^{A_k}_{S_k, S_{k+1}}}{\prod_{k=t}^{T-1} \pi'(A_k \mid S_k) T^{A_k}_{S_k, S_{k+1}}} = \frac{\prod_{k=t}^{T-1} \pi(A_k \mid S_k)}{\prod_{k=t}^{T-1} \pi'(A_k \mid S_k)}$$

note that it is independent from the transition model of the MDP!

Evaluate a policy while following a different one

$$v_{\pi'}(s) = \mathbb{E}\left[G_t \mid S_t = s\right]$$

Following policy π', $v_{\pi'}(s)$ is the average of the returns G_t.

To estimate $v_\pi(s)$, we can use

$$v_\pi(s) = \mathbb{E}\left[\rho_{t:T-1} G_t \mid S_t = s\right]$$

"Ordinary" importance sampling

Notations :

- We label the iterations, independently of the episode it belongs to
  ex : two episodes,the first ends after 3 iterations, the second after 4, but we label the rewards with a unique integer in $\{1,\dots,7\}$
- $\mathcal{T}(s)$
  - "every visit" : $\mathcal{T}(s)$ is the set of all iterations where the current state is s.
  - "first visit" : $\mathcal{T}(s)$ is the set of iterations where it is the first time in the episode where s is visited.
- $T(t)$ is the final iteration of the episode that include iteration t.

To compute an estimate $\tilde{v}$ of v_π, we can compute the average

$$\tilde{v}(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{|\mathcal{T}(s)|}$$

Weighted importance sampling

To compute an estimate $\tilde{v}$ of v_π, we can also use a weighted average

$$\tilde{v}(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1}}$$

Importance sampling : ordinary or weighted

- "first visit" variant
  - ordinary version : unbiased but variance may be large (even infinite)
  - weighted version : biased, if rewards are bounded, the bias converges to 0
  - in practice, the weighted version is preferred due to (much) lower variance.
- "every visit", in both versions, it is biased, but it converges to 0 with the number of samples
  ⇨ much simpler to implement !

Precup, Sutton, Dasgupta (2001) Off policy temporal-difference learning with function approximation, *Proceedings of ICML 2001*

Conclusions for Monte Carlo method

In domains in which we can simulate/run many episodes

- we can evaluate any given policy
- we can find a near optimal policy

⇨ all this without any knowledge of the transition matrix T or the reward vector R

- in many applications, it is easy to simulate episodes.
- if needed, we can target some particular state and ignore the others.
- no "bootstrap", but this is coming next!

what we did not consider is incremental algorithms for efficient computation

Temporal-difference Learning

Combines ideas from
dynamic programming (DP)
with
Monte Carlo methods (MC)

"Temporal-difference" methods

- they learn from experience (as in MC), no knowledge of transition model or reward model
- they do not use any model (as in MC)
- they learn from incomplete episodes (as in DP)
- they use other estimates to update one estimate (i.e. they bootstrap) as in DP

"Temporal-difference" for the value function v

For a given fixed policy π, we have

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma v_\pi(s_{t+1}) \mid s_t = s] \end{aligned}$$

- Monte Carlo "every visit"/"first visit"

"Temporal-difference" for the value function v

For a given fixed policy π, we have

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma v_\pi(s_{t+1}) \mid s_t = s] \end{aligned}$$

- Monte Carlo "every visit"/"first visit"
  - estimate using the actual return G_t obtained during one entire episode.

$$v(s_t) \leftarrow v(s_t) + \alpha(G_t - v(s_t))$$

"Temporal-difference" for the value function v

For a given fixed policy π, we have

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma v_\pi(s_{t+1}) \mid s_t = s] \end{aligned}$$

- Monte Carlo "every visit"/"first visit"
  - estimate using the actual return G_t obtained during one entire episode.

$$v(s_t) \leftarrow v(s_t) + \alpha(G_t - v(s_t))$$

  - the value G_t can only be obtained at the **end** of an episode

"Temporal-difference" for the value function v

For a given fixed policy π, we have

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s] \\ &= \mathbb{E}_\pi[r_{t+1} + \gamma v_\pi(s_{t+1}) \mid s_t = s] \end{aligned}$$

- Monte Carlo "every visit"/"first visit"
  - estimate using the actual return G_t obtained during one entire episode.

$$v(s_t) \leftarrow v(s_t) + \alpha(G_t - v(s_t))$$

  - the value G_t can only be obtained at the **end** of an episode
- Temporal difference $\text{TD}(0)$

$$v(s_t) \leftarrow v(s_t) + \alpha\,[r_{t+1} + \gamma v(s_{t+1}) - v(s_t)]$$

update using as sample $r_{t+1} + \gamma v(s_{t+1})$

"Temporal-difference" for the value function v

$$v(s_t) \leftarrow v(s_t) + \alpha\,[r_{t+1} + \gamma v(s_{t+1}) - v(s_t)]$$

Temporal-difference : we compute the error between the estimate of $v(s_t)$ and the estimate $r + \gamma v(s_{t+1})$.

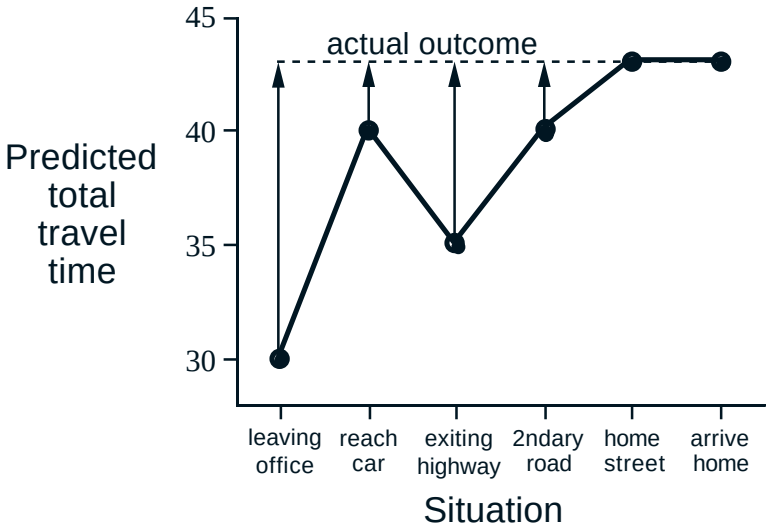
we observe that we use two different iterations in that update $t+1$ and t, hence the name.

Example “coming home”

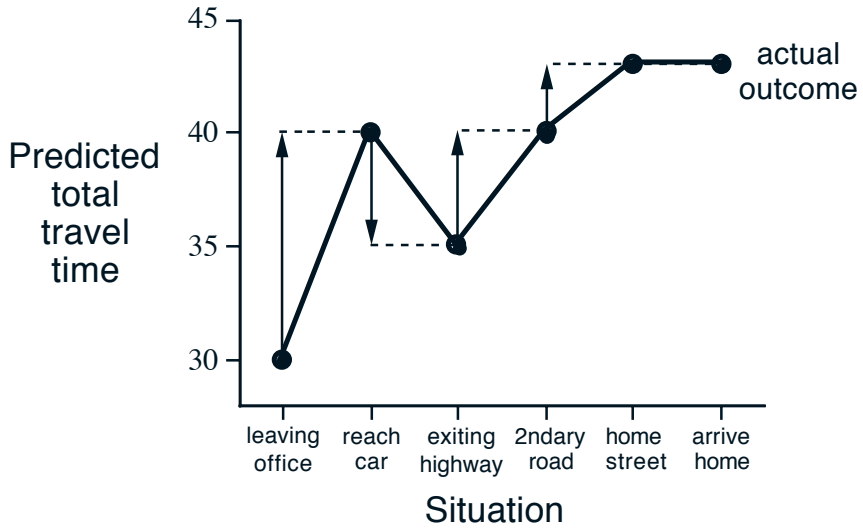
Etat	elapsed time	estimated remaining time	total predicted time
live the office	0	30	30
reached the car, it’s raining	5	35	40
leaving the highway	20	15	35
slow truck	30	10	40
in the neighbourhood	40	3	43
reached home	43	0	43

Example "coming home"

Monte Carlo update
$\gamma = \alpha = 1$



TD(0) update
$\gamma = \alpha = 1$

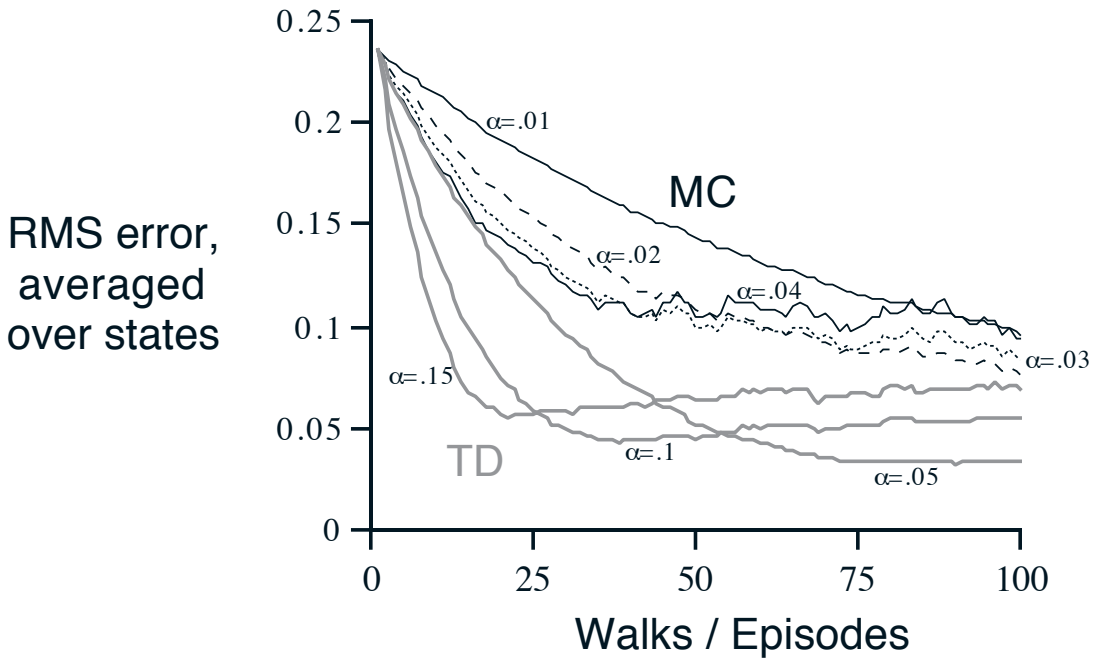


Comparison between TD and MC

- No theoretical results showing TD dominates MC methods, or the other way around.

⇨ in practice, TD methods tend to be faster on stochastic problems

same probability to move left or right.



Intuitive example

Let us consider an MDP with two states A and B. We observe the following 8 episodes :

$$\begin{array}{ll} B,0 & B,1 \\ B,1 & B,1 \\ B,1 & B,1 \\ B,1 & A,0,B,0 \end{array}$$

What is your estimate of $v(A)$ and $v(B)$?

Learn optimal policy : TD "on policy"

- we learn the Q-table
- The update rule for TD(0) is similar as for the case of v :

$$q(s_t,a_t) \leftarrow q(s_t,a_t) + \alpha\,[r_{t+1} + \gamma q(s_{t+1},a_{t+1}) - q(s_t,a_t)]$$

State–action–reward–state–action (SARSA)

Initialise $q(s,a) \in \mathbb{R}$ arbitrarily (for instance $q(s,a) = 0$)
Repeat for each episode
    start from one initial state s
    choose action $a \in A$ for state s using a policy derived from q (ex : ϵ-greedy)
    Repeat for each step of the episode
        Execute action a, observe $r \in \mathbb{R}$ and next state $s' \in S$
        if s' is final
            $q(s,a) \leftarrow q(s,a) + \alpha[r - q(s,a)]$
        else
            choose action $a' \in A$ for state s' using a policy derived from q
            $q(s,a) \leftarrow q(s,a) + \alpha[r + \gamma q(s',a') - q(s,a)]$
        $s \leftarrow s'$
        $a \leftarrow a'$
    until s is terminal

Learn optimal policy : TD "on policy"

Theorem

SARSA converges to the optimal value function Q under the following conditions :

- Conditions about exploration :
  - each `(state,action)` pairs are sufficiently explored $\lim_{k\to\infty} n_k(s,a) = \infty$
  - the policy converges to a greedy policy $\lim_{k\to\infty} \pi_k(a|s) = 1$ pour $a = arg\max_{a'\in A} q(s,a')$
- $\sum_{t=1}^{\infty} \alpha_t = \infty$
- $\sum_{t=1}^{\infty} \alpha_t^2 < \infty$

ϵ-greedy satisfies the conditions when ϵ is decreasing (ex $e_k = \frac{1}{k}$)

Learn optimal policy : TD "off policy"

Q-learning (Watkins 1989)

$$q(s_t,a_t) \leftarrow q(s_t,a_t) + \alpha \left[r_{t+1} + \gamma \max_{a \in A} q(s_{t+1},a) - q(s_t,a_t) \right]$$

Update Q-table independently from the current policy

```
1  Initialise q(s,a) ∈ ℝ arbitrarily (for instance q(s,a) = 0)
2  Repeat for each episode
3      start from one initial state s
4      choose action a ∈ A for state s using a policy derived from q (ex : ε-greedy)
5      Repeat for each step of the episode
6          Execute action a, observe r ∈ ℝ and next state s' ∈ S
7          if s' is final
8              q(s,a) ← q(s,a) + α[r − q(s,a)]
9          else
10             choose action a' ∈ A for state s' using a policy derived from q
11             q(s,a) ← q(s,a) + α[r + γ max_{a''∈A} q(s',a'') − q(s,a)]
12         s ← s'
13         a ← a'
14     until s is terminal
```

Learn an optimal policy : TD "off policy"

under the same conditions as SARSA, Q-learning converges.

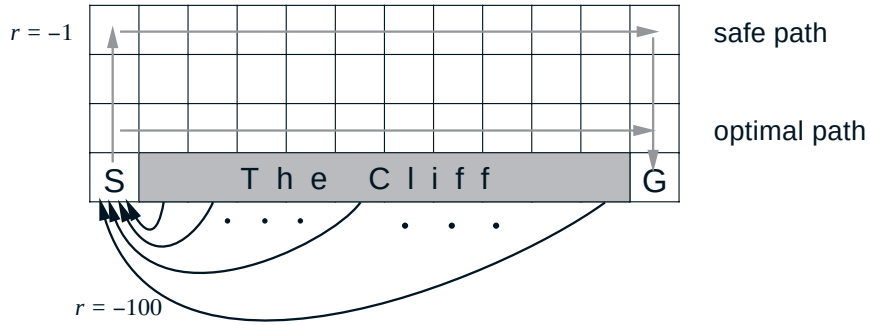
Theorem

Q-learning converges to the optimal value function Q under the following conditions :

- Conditions about exploration :
  - each `(state,action)` pairs are sufficiently explored $\lim_{k\to\infty} n_k(s,a) = \infty$
  - the policy converges to a greedy policy $\lim_{k\to\infty} \pi_k(a|s) = 1$ pour $a = arg\max_{a' \in A} q(s,a')$
- $\sum_{t=1}^{\infty} \alpha_t = \infty$
- $\sum_{t=1}^{\infty} \alpha_t^2 < \infty$

Comparaison SARSA/Q-learning

Both algorithms converges to the same optimal value function. Their path to learning may be different. If exploration is constant, they may learn different strategies



Other exploration schemes

- soft max : choose an action a with probability

$$\frac{e^{\frac{q_t(s,a)}{\tau}}}{\sum_{a' \in A} e^{\frac{q_t(s,a')}{\tau}}}$$

  - $\tau > 0$ is the temperature
  - high temperature ⇨ uniform probability
  - low temperature ⇨ greedy behaviour

- Optimistic initialisation : always behave greedily, but initialise the value optimistically. (Even-Dar & Mansour, NIPS 1994)
  ⇨ forces exploration to consider promising actions.

Central dilemma : explore or exploit

- our agent is generating its own data.
- exploration : obtains as much information as possible
  need to spend enough time exploring, but not too much, otherwise, we never learn !
- exploitation : optimises the current knowledge
  don't exploit too soon otherwise the agent obtains suboptimal performances
- what actions will lead to a better knowledge so as to obtain the best possible rewards in the long run

⇨ intelligence trait

Stratégie d'exploration

- ϵ-greedy
  - easy to implement and used a lot (kind of default solution)
  - convergence requires an **exponential** number of samples (in the MDP size).
- Boltzmann
  - in practice, I find it harder to set up (planning the decrease of temperature)
  - similar bad result for complexity.

Summary

Algorithms to

- evaluate a given policy
- find an optimal policy

- if the MDP is known ⇨ policy/value iteration
- Episodic domains only ⇨ Monte Carlo methods
- Any MDP (but with discrete actions and not too many states) ⇨ SARSA, Q-learning.

⇨ all these methods are **value**-based methods (we estimate the Q-table and we try to improve).

- other **value**-based methods : double Q-learning, n-step SARSA, td(λ), expected-SARSA, ...

- some algorithms work on building a model of the MDP to solve it. (E^3, R_{max}) : they may require less samples (polynomial sampling complexity), but more computations (no free lunch !)