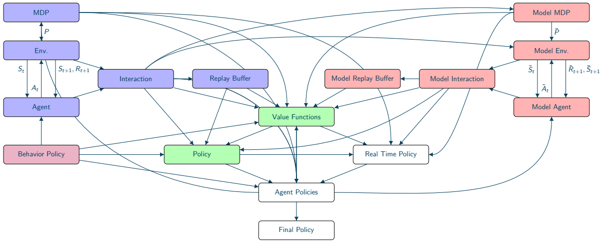
Reinforcement Learning
Reinforcement Learning: Prediction and Planning in the Tabular Setting

E. Le Pennec

M2 DS - Fall 2022

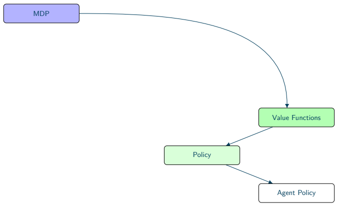
RL: What Are We Going To See?



Outline

- Operations Research and MDP.
- Reinforcement learning and interactions.
- More tabular reinforcement learning.
- Reinforcement and approximation of value functions.
- Actor/Critic: a Policy Point of View

Operations Research and MDP

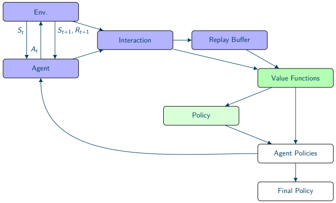


How to find the best policy knowing the MDP?

- Is there an optimal policy?
- How to estimate it numerically?

- Finite states/actions space assumption (tabular setting).
- Focus on interative methods using value functions (dynamic programming).
- Policy deduced by a statewise optimization of the value function over the actions.
- Most results for the discounted setting.

Reinforcement Learning and Interactions



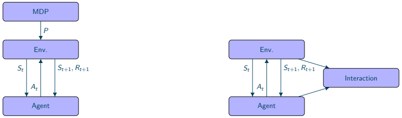
How to find the best policy not knowing the MDP?

- How to interact with the environment to learn a good policy?
- Can we use a Monte Carlo strategy outside the episodic setting?
- How to update value functions after each interaction?

- Focus on stochastic methods using tabular value functions (Q learning, SARSA. . .)
- Policy deduced by a statewise optimization of the value function over the actions.

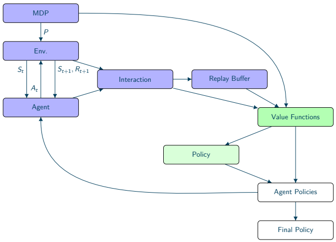
Outline

Reinforcement Learning



- What to do if one has no knowledge of the underlying MDP?
- Only information through interactions!
- Prediction? Planning?
- Focus on the discrete setting

Reinforcement Learning



Outline

Monte Carlo, i.e. Just Play!

- Most simple way to evaluate a policy.

Just Play Following Policy Π

- Play N episodes following the policy.
- During each episode, compute the (discounted) gain.
- Compute the average gain.

- What is computed?

Average Gain or Value Function

$$\mathbb{E}[G_0] \quad \text{vs} \quad v_{t,\Pi}(s) = \mathbb{E}[G_t | S_t = s]$$

Prediction as Value Function Evaluation

- Not the same goal.
- By construction,

$$\mathbb{E}[G_0] = \sum_s \mu_0(s) v_{t,\Pi}(s)$$

- Much easier to compute the average gain than the value function (even if we use a stationary policy)

- Average gain is nevertheless the most classical way to evaluate a policy (with a single number).
- Implicit episodic setting if we do not want to use approximated gain.

Average Gain Estimation

Episodic: Evaluation by MC

input: MDP environment, initial state distribution μ_0, policy Π and discount factor γ
parameter: Number of episodes N
init: $\tilde{V} = 0$,$n = 0$
repeat
    $n \leftarrow n + 1$
    $t \leftarrow 0$
    $G \leftarrow 0$
    Pick initial state S_0 following μ_0
    repeat
        Pick action A_t according to $\pi(\cdot|S_t)$
        $G \rightarrow G + \gamma^t R_{t+1}$
        $t \leftarrow t + 1$
    until *episod ends at time* T
    $\tilde{V} \leftarrow \tilde{V} + G$
until $n == N$
$\tilde{V} \leftarrow \tilde{V}/N$
output: Average gain $\tilde{V}$

Monte Carlo Prediction

- How to estimate $v_{t,\Pi}$?

Just Play Following Policy Π

- Play N episodes following the policy.
- During episode, record S_t and R_t.
- After each episode, compute recursively for each time t the gain G_t.
- Estimate $v_{t,\Pi}(s)$ by the average G_t over all trajectories such that $S_t = s$

- **May require a lot of game to have a non empty set for each state s at each time t**

Monte Carlo Prediction

- How to estimate v_Π for a stationary policy?

Just Play Following Policy Π

- Play N episodes following the policy.
- During episode, record S_t and R_t.
- After each episode, compute recursively for each time t the gain G_t.
- Estimate $v_\Pi(s)$ by the average G_t over all trajectories such that $S_t = s$, whatever t.

- The same state may be reached several time during a single episode...
- First-visit variant: Use only the first visit of s for each episode.

Monte Carlo Prediction

Episodic: Prediction by MC

input: MDP environment, initial state distribution μ_0, policy Π and discount factor γ
parameter: Number of episodes N
init: $\forall s, \tilde{V}(s), n = 0, N(s) = 0$
repeat
  $n \leftarrow n + 1$
  $t \leftarrow 0$
  Pick initial state S_0 following μ_0
  repeat
    (If First-visit) $N(S_t) \leftarrow N(S_t) + 1$
    Pick action A_t according to $\pi(\cdot|S_t)$
    Record R_{t+1}, S_{t+1}
    $t \leftarrow t + 1$
  until *episod ends at time* T
  $G_{T+1} = 0$
  $t \rightarrow T + 1$
  repeat
    $t \leftarrow t - 1$
    Compute $G_t = R_{t+1} + \gamma G_{t+1}$
    (If First-visit) $\tilde{V}(S_t) = \tilde{V}(S_t) + G_t$
  until $t = 0$
until $n == N$
for $s \in \mathcal{S}$ **do**
  $\tilde{V}(s) \leftarrow \tilde{V}(s)/N(s)$
end
output: Value function $\tilde{V}$

Monte Carlo Prediction Analysis

First-Visit Variant Analysis

- Straightforward analysis as all the used values for a given state s are independent.
- Variance of order $1/N(s)$ where $N(s)$ is the number of episod where s is visited.
- Convergence if the number of visit goes to ∞.
- Strong assumption is practice as some states may not be visited by a given policy (if we cannot play on the initial state).

- Every-visit works. . . but not necessarily better!

Outline

Monte Carlo Planning

- Can we use a MC approach to find a good policy?

A First Attempt

- Estimate $v_\pi(s)$ by MC.
- Compute $q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) v_\pi(s)$
- Enhance the current policy by setting $\pi(s) = \operatorname{argmax}_a q_\pi(s,a)$

- Inspired by the Operations Research results...
- But unusable as r and p are unknown!

Monte Carlo Planning

A Second Attempt

- Estimate $q_\pi(s, a)$ by MC.
- Enhance the current policy by setting $\pi(s) = \operatorname{argmax}_a q_\pi(s, a)$

- Requires that $N(s, a)$ the number of times that an episode contains the state s followed by action a goes to ∞.
- Impossible with a deterministic policy!

Monte Carlo Planning

Classical Exploratory Policies...

- Stochastic policies ensuring that any action can occurs at any state.
- ϵ-exploratory policy: use a determistic policy and replace it with a random action with probability ϵ.
- Gibbs policy: use a policy where $\pi(a|s) \propto e^{G(a,s)} > 0$.

A Final Attempt

- Start from an exploratory policy.
- Estimate $q_\pi(s, a)$ by MC.
- Enhance the current policy while remaining a exploratory policy.

- Last step is not straightforward...
- except for ϵ-deterministic policy for which the ϵ-exploratory policy with $\pi(s) = \operatorname{argmax}_a q_\pi(s, a)$ works.
- No convergence proof.

Outline

Advanced Implementation of Monte Carlo Prediction

$$\tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(G_t - \tilde{v}_\pi(S_t))$$

$$M_{n-1} = \frac{G^{(1)} + \dots + G^{(n-1)}}{n-1}$$

$$\frac{G^{(1)} + \dots + G^{(n)}}{n} = \frac{(n-1)M_{n-1} + G^{(n)}}{n} = M_{n-1} + \frac{1}{n}(G^{(n)} - M_{n-1})$$

On-Line Monte Carlo

- Average for a given state can be updated each time we have the gain G_t for a state S_t.
- Just use $\alpha(N) = 1/N$ and increment $N(S_t)$.
- No need to record the values between episodes...

- We still need to wait until the end of each episode to compute G_t.
- Can we do better?

Advanced MC Prediction

Episodic: Prediction by MC

input: MDP environment, initial state distribution μ_0, policy Π and discount factor γ
parameter: Number of episodes N
init: $\forall s, \tilde{V}(s), n = 0, N(s) = 0$
repeat
  $n \leftarrow n + 1$
  $t \leftarrow 0$
  Pick initial state S_0 following μ_0
  repeat
    (If First-visit) $N(S_t) \leftarrow N(S_t) + 1$
    Pick action A_t according to $\pi(\cdot|S_t)$
    Record R_{t+1}, S_{t+1}
    $t \leftarrow t + 1$
  until *episod ends at time T*
  $G_{T+1} = 0$
  $t \rightarrow T + 1$
  repeat
    $t \leftarrow t - 1$
    Compute $G_t = R_{t+1} + \gamma G_{t+1}$
    (If First-visit) $\tilde{V}(S_t) = \tilde{V}(S_t) + \frac{1}{N(S_t)} (G_t - \tilde{V}(S_t))$
  until $t = 0$
until $n == N$
output: Value function $\tilde{V}$

- We still need to wait until the end of each episode to compute G_t.
- Can we do better?

Prediction with Temporal Differencies

$$\text{From} \quad \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(G_t - \tilde{v}_\pi(S_t))$$

$$\text{to} \quad \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))\underbrace{(R_{t+1} + \gamma\tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t))}_{\delta_t}$$

Bootstrap Strategy

- Replace G_t by an instantaneous estimate $R_{t+1} + \gamma\tilde{v}_\pi(S_{t+1})$.
- Amounts to replace $\gamma R_{t+2} + \gamma^2 R_{t+2}$ by an approximation of its expectation given S_{t+1}: $v_\pi(S_{t+1})$.
- Bootstrap as we use the current estimate $\tilde{v}_\pi(S_{t+1})$ instead of the true value.
- $\delta_t = R_{t+1} + \gamma\tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t)$ is called a temporal difference.

- No need to wait until the end of the episodes!
- Can be used in the discounted setting.

TD Prediction

Discounted: Prediction by TD

input: MDP environment, initial state distribution μ_0, policy Π and discount factor γ
parameter: Number of step T
init: $\forall s, \tilde{V}(s), n = 0, N(s) = 0,\ t' = 0$
repeat
- $t \leftarrow 0$
- Pick initial state S_0 following μ_0
- **repeat**
  - $N(S_t) \leftarrow N(S_t) + 1$
  - Pick action A_t according to $\pi(\cdot|S_t)$
  - $\tilde{V}(S_t) = \tilde{V}(S_t) + \alpha(N(S_t))\left(R_{t+1} + \gamma \tilde{V}(S_{t+1}) - \tilde{V}(S_t)\right)$
  - $t \leftarrow t + 1$ / $t' \leftarrow t' + 1$
- **until** *episod ends at time T' or $t' == T$*

until $t' == T$
output: Value function $\tilde{V}$

- **But does this work?**

Discounted

Prediction with Temporal Differencies

$$\mathbb{E}[\delta_t|S_t] \mathbb{E}[R_{t+1} + \gamma\tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t)|S_t] = (\mathcal{T}^\pi - \mathrm{Id})\,\tilde{v}_\pi(S_t)$$

TD and Bellman Operator

- TD as an approximate Policy Iteration:
  $$\mathbb{E}[\tilde{v}_\pi](S_t) \leftarrow \tilde{v}_\pi + \alpha(N(S_t))\,(\mathcal{T}^\pi - \mathrm{Id})\,\tilde{v}_\pi(S_t)$$
- Proof of convergence of this algorithm to a zero of $\mathcal{T}^\pi - \mathrm{Id}$, i.e. the fixed point of $\mathcal{T}^\pi$!
- Proof requires a mild assumption of α (satisfied by $\alpha(N) = 1/N$) and the strong assumption that $N(s)$ goes to ∞.

- MC could be interpreted in a similar way (stochastic approximation) by noticing that $\mathbb{E}[G_t - \tilde{v}_\pi(S_t)|S_t] = v_\pi(S_t) - \tilde{v}_\pi(S_t)$.
- Often use with a constant α

MC vs TD

$$\tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(G_t - \tilde{v}_\pi(S_t))$$

or $$\tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))\underbrace{(R_{t+1} + \gamma\tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t))}_{\delta_t}$$

MC vs TD

- Both are based on stochastic approximation.
- Both converges (under similar assumptions) to the correct value function.
- TD does not require to wait until the end of the episode.
- No theorical difference in the speed of convergence but often TD is better...
- Solve different approximate problems when used with a finite set of episodes:
  - MC compute the empirical gain from any state.
  - TD compute the value function of the empirical Bellman operator (the one obtained by using the empirical transition probabilities)

- If $\tilde{v}_\pi$ is kept constant during an episode

$$G_t - \tilde{v}_\pi(S_t) = \sum \gamma^{t'-t}\delta_{t'}$$

Outline

Stochastic Approximation

$$\theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k) \quad \text{with} \quad h_k(\theta) = H(\theta) + \epsilon_k + \eta_k$$
$$\implies \theta_k \to \{\theta, H(\theta) = 0\}$$

Stochastic Approximation

- Family of sequential stochastic algorithm converging to a zero of a function.
- Classical assumptions:
  - $\mathbb{E}[\epsilon_k] = 0$, $\mathbb{V}\text{ar}\,[\epsilon_k] < \sigma^2$, and $\mathbb{E}[\|\eta_k\|] \to 0$,
  - $\sum_k \alpha_k \to \infty$ and $\sum_k \alpha_k^2 < \infty$,
  - the algorithm converges if we replace h_k by H.

- Convergence toward a neighborhood if α is kept constant (as often in practice).
- Most famous example are probably Robbins-Monro and SGD.
- Proof quite technical in general.
- The convergence with H is easy to obtain for a contraction.

Stochastic Approximation and ODE

From $\theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k)$ with $h_k(\theta) = H(\theta) + \epsilon_k + \eta_k$

to $\frac{d\tilde{\theta}}{dt} = H(\tilde{\theta})$ $\left(\tilde{\theta}(t+1) = \tilde{\theta}(t) + \delta H(\tilde{\theta}_t) \right)$ Flow

ODE Approach

- General proof showing that the algorithm converges provided the ODE converges.
- Rely on the rewriting the equation
$$\frac{\theta_{k+1} - \theta_k}{\alpha_k} = h_k(\theta_k) = H(\theta_k) + \epsilon_k + \eta_k$$
- α_k can be interpreted as a time difference allowing to define a time $t_k = \sum_{k' \le k} \alpha_{k'}$.
- Equation be interpreted as the derivative at time $t \in (t_k, t_{k+1})$ of a piecewise affine function $\theta(t)$.
- This piecewise function remains close to any solution of the ODE starting from θ_k for an arbitrary amount of time provided k is large enough.

Sketch of Proof

- Difference between θ and a solution of the ODE with $\tilde{\theta}(t_k) = \theta_k$ at t_{k+l}:

$$
\begin{aligned}
\theta(t_{k+l}) - \tilde{\theta}(t_{k+l}) &= \int_{t_k}^{t_{k+l}} \left(\theta'(u) - \tilde{\theta}'(u)\right) du \\
&= \sum_{k'=k}^{k+l-1} \int_{t_{k'}}^{t_{k'+1}} \left(H(\theta(t_k)) + \epsilon_k + \eta_k - H(\tilde{\theta}(u))\right) du \\
&= \sum_{k'=k}^{k+l-1} \int_{t_{k'}}^{t_{k'+1}} \left(H(\theta(t_k)) - H(\tilde{\theta}(u))\right) du \\
&\quad + \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} + \sum_{k'=k}^{k+l-1} \alpha_{k'} \eta_{k'}
\end{aligned}
$$

- The last two term are going to be small by construction...

Sketch of Proof

- Difference between θ and a solution of the ODE with $\tilde{\theta}(t_k) = \theta_k$ at t_{k+l}:

$$\theta(t_{k+l}) - \tilde{\theta}(t_{k+l}) = \sum_{k'=k}^{k+l-1} \int_{t_{k'}}^{t_{k'+1}} \Big(H(\theta(t_k)) - H(\tilde{\theta}(u)) \Big)\, du + \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} + \sum_{k'=k}^{k+l-1} \alpha_{k'} \eta_{k'}$$

- The last two term are going to be small by construction:

$$\mathbb{E}\left[\sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} \right] = 0 \quad \text{and} \quad \mathbb{V}\mathrm{ar}\left[\sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} \right] < \sigma^2 \sum_{k'=k}^{k+l-1} \alpha_{k'}^2 \to 0$$

$$\| \sum_{k'=k}^{k+l-1} \alpha_{k'} \eta_{k'} \| \leq (t_{k+l-1} - t_k) \sup_{k' \geq k} \|\eta_{k'}\|$$

which is small if we assume that $t_{k+l-1} - t_k \leq \Delta$.

Sketch of Proof

- We can now use a Lipchitz assumption on H to obtain:

$$\begin{aligned}\left\|\int_{t_{k'}}^{t_{k'+1}}\left(H(\theta(t_{k'}))-H(\tilde{\theta}(u))\right)du\right\| &\leq L\int_{t_{k'}}^{t_{k'+1}}\|\theta(t_{k'})-\tilde{\theta}(u)\|du\\ &\leq L\alpha_{k'}\|\theta(t_{k'})-\tilde{\theta}(t_{k'})\|+L\int_{t_{k'}}^{t_{k'+1}}\|\theta(\tilde{t}_{k'})-\tilde{\theta}(u)du\|\\ &\leq L\alpha_{k'}\|\theta(t_{k'})-\tilde{\theta}(t_{k'})\|+L\|H\|_{\infty}\alpha_{k'}^{2}\end{aligned}$$

- Combinining all the result leads to

$$\begin{aligned}\|\theta(t_{k+l})-\tilde{\theta}(t_{k+l})\| &\leq L\sum_{k'=k}^{k+l-1}\alpha_{k'}\|\theta(t_{k'})-\tilde{\theta}(t_{k'})\|\\ &+L\|H\|_{\infty}\sum_{k'=k}^{k+l-1}\alpha_{k'}^{2}+\left\|\sum_{k'=k}^{k+l-1}\alpha_{k'}\epsilon_{k'}\right\|+\left\|\sum_{k'=k}^{k+l-1}\alpha_{k'}\eta_{k'}\right\|\end{aligned}$$

- A Gronwall type Lemma allows to conclude.

Sketch of Proof

- Combinining all the results leads to

$$\|\theta(t_{k+l}) - \tilde{\theta}(t_{k+l})\| \leq L \sum_{k'=k}^{k+l-1} \alpha_{k'} \|\theta(t_{k'}) - \tilde{\theta}(t_{k'})\|$$

$$+ L\|H\|_\infty \sum_{k'=k}^{k+l-1} \alpha_{k'}^2 + + \left\| \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} \right\| + \sum_{k'=k}^{k+l-1} \alpha_{k'} \|\eta_{k'}\|$$

- Using a discrete Gronwall Lemma,

$$\forall l \leq l'', z_l \leq L \sum_{l'=0}^{l-1} \alpha_{l'} z_{l'} + A \Rightarrow z_{l''} \leq A e^{L \sum_{l'=0}^{l''-1} \alpha_{l'}},$$

we obtain that if $t_{k+l} - t_k \leq \Delta$

$$\|\theta(t_{k+l}) - \tilde{\theta}(t_{k+l})\| \leq \left(\underbrace{L\|H\|_\infty \sum_{k'=k}^{\infty} \alpha_{k'}^2 + \sup_{l' \leq l} \left\| \sum_{k'=k}^{k+l'-1} \alpha_{k'} \epsilon_{k'} \right\| + L \sup_{k' \geq k} \|\eta_{k'}\|}_{\longrightarrow 0 \text{ when } k \rightarrow \infty} \right) e^{L\Delta}$$

Sketch of Proof

Asynchronous Update

$$\text{From} \quad \theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k) \quad \text{with} \quad h_k(\theta) = H(\theta) + \epsilon_k + \eta_k$$
$$\text{to} \quad \forall i, \theta_{k+1}(i) = \theta_k(i) + \alpha_k(i) h_k(\theta_k)(i)$$

Asynchronous Update

- Componentwise action on θ.
- Not necessarily the same stepsize $\alpha_k(i)$ for all components.
- $\alpha_k(i) = 0$ is permitted!
- Previous results hold provided for every component i, $\sum_k \alpha_k(i) \to \infty$ and $\sum_k \alpha_k^2(i) < \infty$,

- Exact setting of TD approximation!

Outline

Planning with Temporal Differencies

A State Value Function Attempt

- $V_\star$ is the fixed point of $\mathcal{T}^\star$.
- Approximate it as the zero of $\mathcal{T}^\star - \mathrm{Id}$.
- By construction

$$\mathcal{T}^\star v(S_t) = \max_a \mathbb{E}[R_{T+1} + \gamma v(S_{t+1}) | S_t, a]$$

- Not an expectation!

A State-Action Value Function Attempt

- $q_\star$ is the fixed point of $\mathcal{T}^\star$.
- Approximate it as the zero of $\mathcal{T}^\star - \mathrm{Id}$.
- By construction

$$\mathcal{T}^\star q(S_t, A_t) = \mathbb{E}\left[R_{t+1} + \gamma \max_a q(S_{t+1}, a) \middle| S_t, A_t\right]$$

- An expectation!

Discounted: Planning by Q-Learning

input: MDP environment, initial state distribution μ_0, policy Π and discount factor γ
parameter: Number of step T
init: $\forall s, a, \tilde{Q}(s,a), N(s,a) = 0$, n=0, $t' = 0$
repeat
- $t \leftarrow 0$
- Pick initial state S_0 following μ_0
- **repeat**
  - $N(S_t) \leftarrow N(S_t) + 1$
  - Pick action A_t according to $\pi(\cdot|S_t)$
  - $\tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t))\left(R_{t+1} + \gamma \max_a \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t)\right)$
  - $t \leftarrow t + 1$
  - $t' \leftarrow t' + 1$
- **until** *episod ends at time* T' *or* $t' == T$

until $t' == T$
output: Deterministic policy $\tilde{\pi}(s) = \text{argmax}_a \tilde{Q}(s,a)$

Planning with Q Learning

$$\tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \left(\underbrace{R_{t+1} + \gamma \max_a \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t)}_{\delta_t} \right)$$

Q-Learning

- Update is independent of the policy Π.
- Convergence of the Q-value function provided the policy is such that $N(s, a)$ tends to ∞ for any state and any action.
- Implies a convergence of the policy.
- Relies on temporal difference.

- Most classical (tabular) planning algorithm!

Outline

Planning with Policy Improvement

from $$\tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \left(\underbrace{R_{t+1} + \gamma \max_a \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t)}_{\delta_t} \right)$$

to $$\tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \left(\underbrace{R_{t+1} + \gamma \tilde{Q}(S_{t+1}, A_{t+1}) - \tilde{Q}(S_t, A_t)}_{\delta_t} \right)$$

$$\Pi(S_t) = \operatorname*{argmax}_a \tilde{Q}(S_t, a) (\text{plus exploration})$$

Policy Improvement

- More emphasis on the policy with a link between the policy used to play and the optimized policy.
- Almost equivalent to use the current policy in the Q-Learning algorithm.

SARSA

Discounted: Planning by SARSA

input: MDP environment, initial state distribution μ_0, policy Π and discount factor γ
parameter: Number of step T
init: $\forall s, a, \tilde{Q}(s,a), N(s,a) = 0$, n=0, $t' = 0$
repeat
  $t \leftarrow 0$ Pick initial state S_0 following μ_0
  repeat
    $N(S_t) \leftarrow N(S_t) + 1$
    Pick action A_t according to $\pi(\cdot|S_t)$
    $\tilde{Q}(S_{t-1}, A_{t-1}) = \tilde{Q}(S_{t-1}, A_{t-1}) + \alpha(N(S_{t-1}, A_{t-1}))\left(R_t + \gamma\tilde{Q}(S_t, A_t) - \tilde{Q}(S_{t-1}, A_{t-1})\right)$
    $\Pi(S_{t-1}) = \text{argmax}\, \tilde{Q}(S_{t-1}, A_{t-1})$ (plus exploration)
    $t \leftarrow t + 1$
    $t' \leftarrow t' + 1$
  until *episod ends at time* T' *or* $t' == T$
until $t' == T$
output: Deterministic policy $\tilde{\pi}(s) = \text{argmax}_a \tilde{Q}(s,a)$

- Does this work?

SARSA and exploration

$$\Pi(S_t) = \operatorname*{argmax}_a \tilde{Q}(S_t, a)(\text{plus exploration})$$

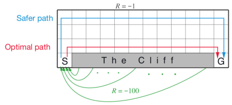
SARSA and Exploration

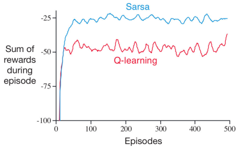
- No hope of convergence if we do not explore all possible actions (and states).
- Impossible if the policy used is deterministic.
- Exploration is required!
- Most classical choice: ϵ-greedy policy with a decaying ϵ.

- Convergence proof is harder than for Q-Learning.
- Relies on the similarity in the limit (when ϵ goes to 0) with the Q-Learning algorithm.

Outline

Q-Learning vs SARSA





How different are they?

- In Q-learning, the exploratory policy used is decoupled from the optimized policy.
- This exploratory policy may yield low rewards on average.
- In SARSA, the two policies are linked with the hope on having higher rewards during the learning step.
- Subtle different behavior even if we modify the exploratory policy in Q-Learning.

Exploration vs Exploitation

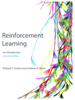
Exploration vs Exploitation

- Exploration: explore new policies to be able to discover the best ones.
- Exploitation: use good policies to obtain a good return.
- Exploration is a requirement.

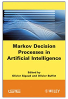
- No tradeoff if we optimize only the final result!
- Tradeoff between the two if we consider that the returns during training matters!
- Q-learning use the first approach and SARSA try to tackle the second.
- Tradeoff if we study a regret:
$$\sum_t \mathbb{E}_{\Pi_\star}[R_t] - \mathbb{E}_{\Pi_t}[R_t]$$
which forces us to be good as fast as possible.
- No natural definition in the discounted setting.

Outline

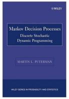
References



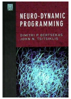
R. Sutton and A. Barto.
Reinforcement Learning, an Introduction (2nd ed.)
MIT Press, 2018



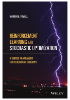
O. Sigaud and O. Buffet.
Markov Decision Processes in Artificial Intelligence.
Wiley, 2010



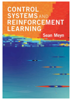
M. Puterman.
Markov Decision Processes. Discrete Stochastic Dynamic Programming.
Wiley, 2005



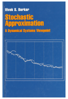
D. Bertsekas and J. Tsitsiklis.
Neuro-Dynamic Programming.
Athena Scientific, 1996



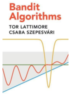
W. Powell.
Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions.
Wiley, 2022



S. Meyn.
Control Systems and Reinforcement Learning.
Cambridge University Press, 2022



V. Borkar.
Stochastic Approximation: A Dynamical Systems Viewpoint.
Springer, 2008



T. Lattimore and Cs. Szepesvári.
Bandit Algorithms.
Cambridge University Press, 2020

Licence and Contributors

Contributors

- Main contributor: E. Le Pennec
- Contributors: S. Boucheron, A. Dieuleveut, A.K. Fermin, S. Gadat, S. Gaiffas, A. Guilloux, Ch. Keribin, E. Matzner, M. Sangnier, E. Scornet.