

usepackagecolor

Model Free Prediction and Control

Clément Rambour

Plan

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

Plan

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

Table of Contents

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

Échantillonnage de Monte Carlo

But

- Apprendre v_π d'après les épisodes vécus en suivant π :

$$S_1, A_1, R_1, \dots, s_k \sim \pi$$

- La fonction de valeur d'un état v_π est donnée par :

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

$$\text{avec } G_t = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{T-1} R_T$$

Échantillonnage de Monte Carlo

But

- Apprendre v_π d'après les épisodes vécus en suivant π :

$$S_1, A_1, R_1, \dots, s_k \sim \pi$$

- La fonction de valeur d'un état v_π est donnée par :

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

$$\text{avec } G_t = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{T-1} R_T$$

Monte Carlo

Façon la plus simple d'estimer la valeur $\rightarrow$ moyenne des retours

Échantillonnage de Monte Carlo

First Visit Monte-Carlo Policy Evaluation

$$\hat{v}_{\pi}(s) = \frac{S(s)}{N(s)}$$

- $N(s)$: compteur incrémenté si s est vu au moins **une fois** dans un épisode
- $S(s) \leftarrow S(s) + G_t$: retour total pour l'état s

Échantillonnage de Monte Carlo

Every Visit Monte-Carlo Policy Evaluation

$$\hat{v}_{\pi}(s) = \frac{S(s)}{N(s)}$$

- $N(s)$: compteur incrémenté **à chaque fois** que s est vu dans un épisode
- $S(s) \leftarrow S(s) + G_t$: retour total pour l'état s

Moyenne incrémentielle

$$\begin{aligned}\mu_k &= \frac{1}{k} \sum_{j=1}^k x_j \\ &= \frac{1}{k} \left(x_k + \sum_{j=1}^{k-1} x_j \right) \\ &= \frac{1}{k} (x_k + (k-1)\mu_{k-1}) \\ &= \mu_{k-1} + \frac{1}{k} (x_k - \mu_{k-1})\end{aligned}$$

Moyenne incrémentielle

$$\mu_k = \frac{1}{k} \sum_{j=1}^k x_j$$

$$= \frac{1}{k} \left(x_k + \sum_{j=1}^{k-1} x_j \right)$$

$$= \frac{1}{k} (x_k + (k-1)\mu_{k-1})$$

$$= \mu_{k-1} + \frac{1}{k} (x_k - \mu_{k-1})$$

- $V(s)$ est mis à jour après chaque épisode :

foreach S_t **do**

$N(S_t) \leftarrow N(S_t) + 1$

$V_\pi(S_t) \leftarrow V_\pi(S_t) + \frac{1}{N(S_t)} (G_t - V_\pi(S_t))$

end

Moyenne incrémentielle

$$\begin{aligned}
 \mu_k &= \frac{1}{k} \sum_{j=1}^k x_j \\
 &= \frac{1}{k} \left(x_k + \sum_{j=1}^{k-1} x_j \right) \\
 &= \frac{1}{k} (x_k + (k-1)\mu_{k-1}) \\
 &= \mu_{k-1} + \frac{1}{k} (x_k - \mu_{k-1})
 \end{aligned}$$

- $V(s)$ est mis à jour après chaque épisode :
foreach S_t **do**
 $N(S_t) \leftarrow N(S_t) + 1$
 $V_\pi(S_t) \leftarrow V_\pi(S_t) + \frac{1}{N(S_t)} (G_t - V_\pi(S_t))$
end
- En cas de non stationnarité, la moyenne peut être mise à jour *ie.* le poids des anciens épisodes $\rightarrow 0$:

$$V_\pi(S_t) \leftarrow V_\pi(S_t) + \alpha (G_t - V_\pi(S_t))$$

$$\alpha \ll 1$$

Monte Carlo prediction : pros and cons

pros

- Intuitif
- Facile à implémenter
- Sans biais

Cons

- Ne peut être mis à jour qu'après un épisode complet *ie.* ne peut pas être utilisé *online*
- Long à converger

Table of Contents

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

De Monte Carlo à TD(0)

Monte Carlo

- Mise à jour basée sur le retour G_t :

$$v_{\pi}(S_t) \leftarrow v(S_t) + \alpha(G_t - v(S_t))$$

- *Offline*

De Monte Carlo à TD(0)

Monte Carlo

- Mise à jour basée sur le retour G_t :

$$v_{\pi}(S_t) \leftarrow v(S_t) + \alpha(G_t - v(S_t))$$

- *Offline*

TD(0)

- D'après les équations de Bellman :

$$v_{\pi}(s) = \mathbb{E}[R_{t+1} + v_{\pi}(S_{t+1}) | S_t = s]$$

- Mise à jour basée sur le retour estimé $R_{t+1} + \gamma v_{\pi}(S_{t+1})$:

$$v_{\pi}(S_t) \leftarrow v_{\pi}(S_t) + \alpha(R_{t+1} + \gamma v_{\pi}(S_{t+1}) - v_{\pi}(S_t))$$

- *Online*

De Monte Carlo à TD(0)

Monte Carlo

- Mise à jour basée sur le retour G_t :

$$v_{\pi}(S_t) \leftarrow v(S_t) + \alpha(G_t - v(S_t))$$

- *Offline*

TD(0)

- D'après les équations de Bellman :

$$v_{\pi}(s) = \mathbb{E}[R_{t+1} + v_{\pi}(S_{t+1}) | S_t = s]$$

$$v_{\pi}(S_t) \leftarrow v_{\pi}(S_t) + \alpha \underbrace{(R_{t+1} + \gamma v_{\pi}(S_{t+1}) - v_{\pi}(S_t))}_{\text{TD Error} = \delta_t}$$

Temporal Difference

TD(0)

Intuition : Estimation de la valeur basée sur la connaissance de l'état suivant

$$V_{\pi}(S_t) \leftarrow V_{\pi}(S_t) + \alpha(R_{t+1} + \gamma V_{\pi}(S_{t+1}) - V_{\pi}(S_t))$$

Temporal Difference

TD(0)

Intuition : Estimation de la valeur basée sur la connaissance de l'état suivant

$$V_{\pi}(S_t) \leftarrow V_{\pi}(S_t) + \alpha(R_{t+1} + \gamma V_{\pi}(S_{t+1}) - V_{\pi}(S_t))$$

Généralisation : n -step

Généralisation possible en se basant sur les n états suivants.

- D'après les équations de Bellman :

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^n v_{\pi}(S_{t+n})$$

- Mise à jour de la valeur :

$$V_{\pi}(S_t) \leftarrow V_{\pi}(S_t) + \alpha(G_t^{(n)} - V_{\pi}(S_t))$$

Temporal Difference

TD(0)

Intuition : Estimation de la valeur basée sur la connaissance de l'état suivant

$$V_{\pi}(S_t) \leftarrow V_{\pi}(S_t) + \alpha(R_{t+1} + \gamma V_{\pi}(S_{t+1}) - V_{\pi}(S_t))$$

Généralisation : n -step

Généralisation possible en se basant sur les n états suivants.

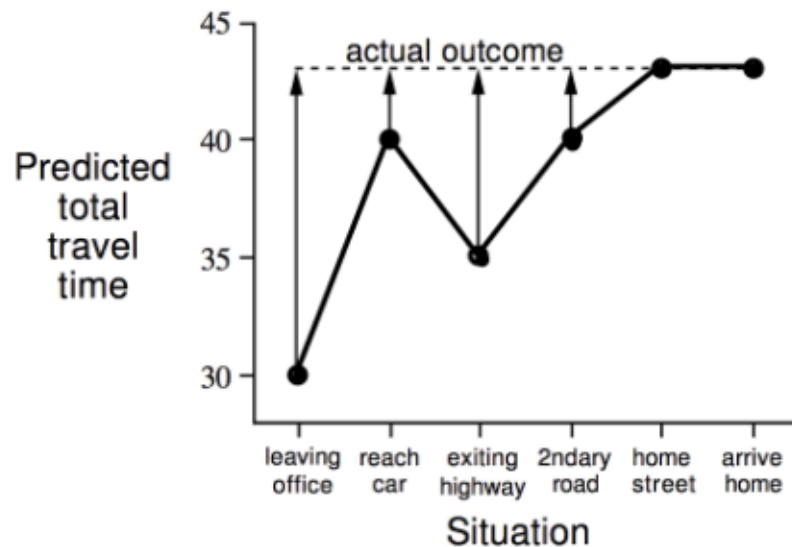
- Équivalent à Monte-Carlo pour $n \rightarrow \infty$
- Nécessite de stocker n récompenses et valeurs d'état

Exemple

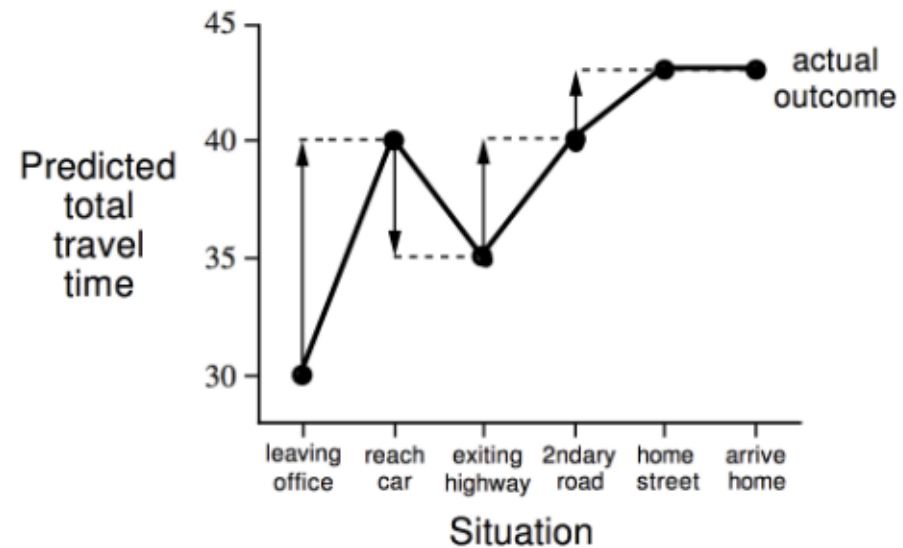
State	Elapsed Time (minutes)	Predicted Time to Go	Predicted Total Time
leaving office	0	30	30
reach car, raining	5	35	40
exit highway	20	15	35
behind truck	30	10	40
home street	40	3	43
arrive home	43	0	43

Exemple : MC vs TD

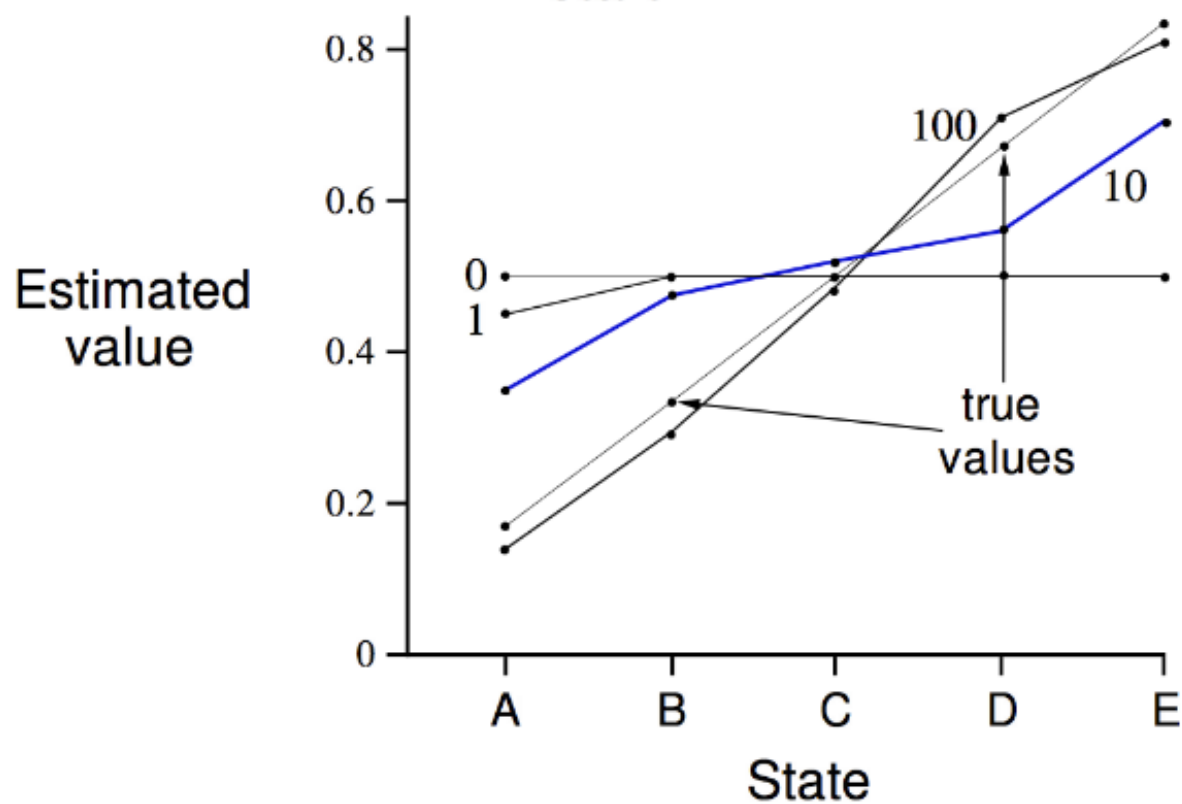
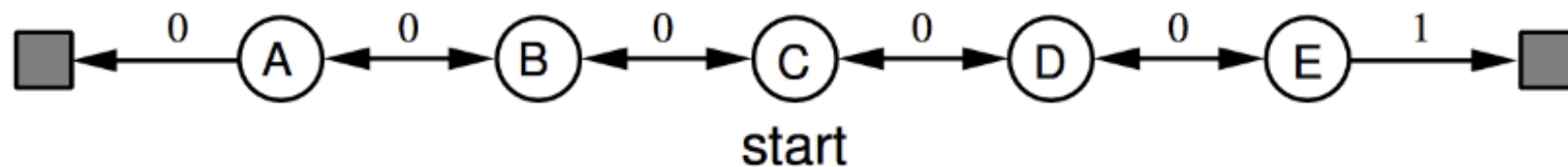
Changes recommended by
Monte Carlo methods ($\alpha=1$)



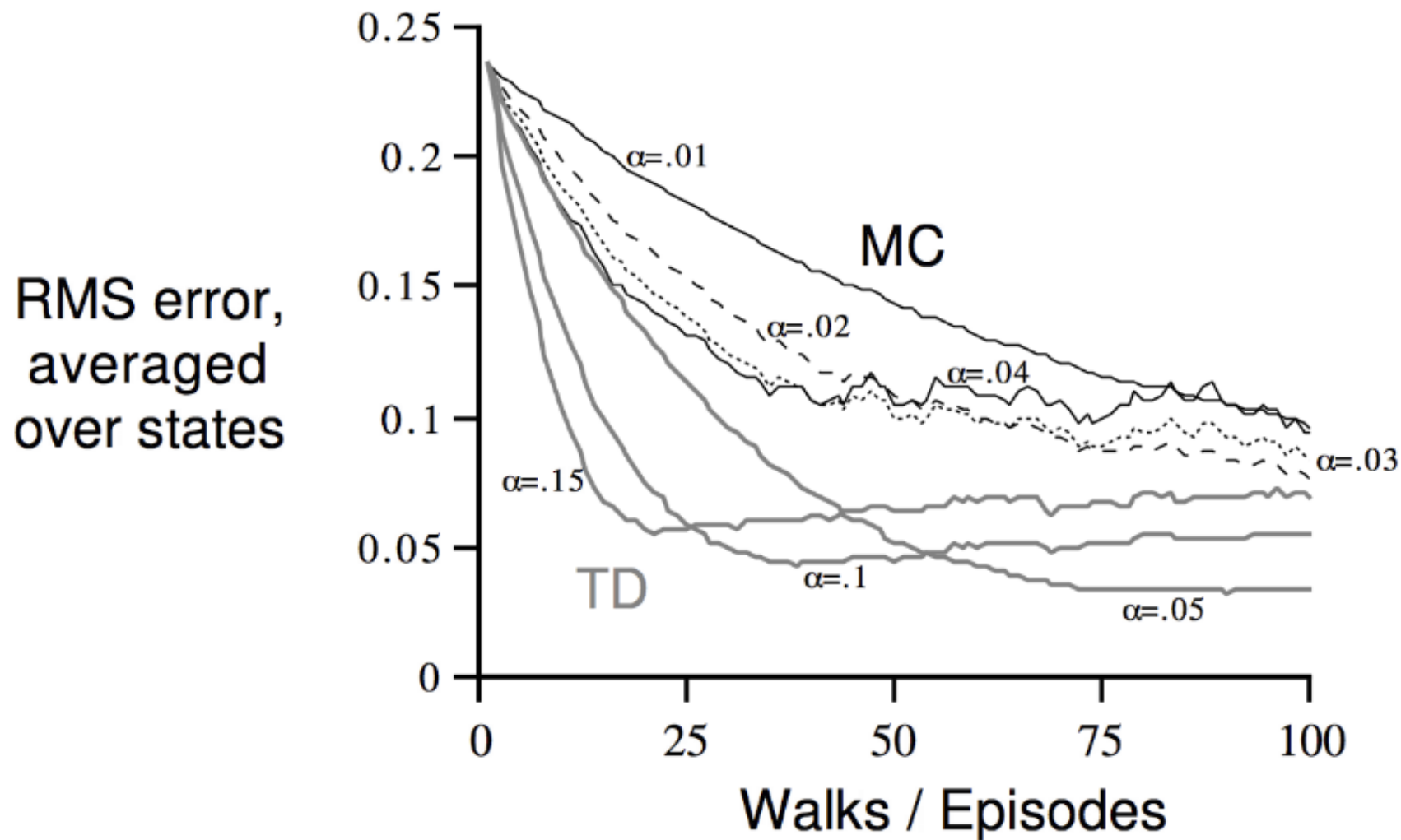
Changes recommended
by TD methods ($\alpha=1$)



Exemple : MC vs TD



Exemple : MC vs TD



Exemple : MC vs TD

Two states A, B ; no discounting; 8 episodes of experience

$A, 0, B, 0$

$B, 1$

$B, 1$

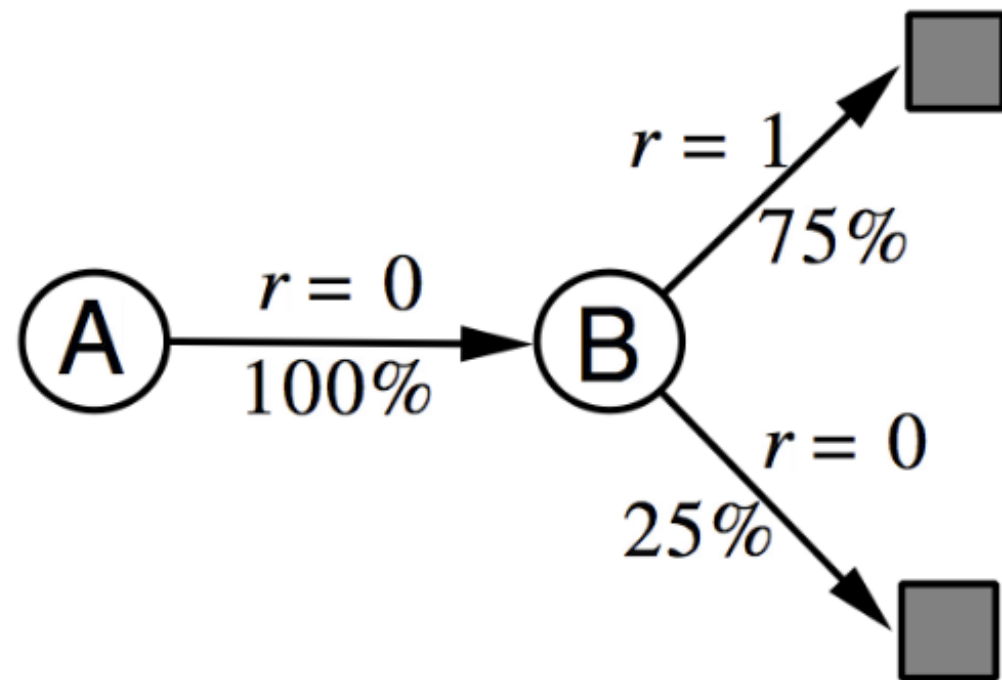
$B, 1$

$B, 1$

$B, 1$

$B, 1$

$B, 0$



What is $V(A), V(B)$?

MC vs TD

- batch-MC converge vers la solution correspondant aux moindres carrés
- Batch TD(0) converge vers le maximum de vraisemblance d'après la dynamique du modèle conditionné par la politique

Table of Contents

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

TD(λ)

TD(λ)

- Utilisation de toutes les approximations du retour :

$$G_t^\lambda = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

- (*forward-view*) TD(λ) :

$$V_\pi(S_t) \rightarrow V_\pi(s_t) + \alpha(G_t^\lambda - V_\pi(S_t))$$

- Nécessite des épisodes complets (comme MC)
- *offline*

TD(λ)

TD(λ)

- Utilisation de toutes les approximations du retour :

$$G_t^\lambda = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

- (*forward-view*) TD(λ) :

$$V_\pi(S_t) \rightarrow V_\pi(s_t) + \alpha(G_t^\lambda - V_\pi(S_t))$$

- Nécessite des épisodes complets (comme MC)
- *offline*
- Peut être approché *online* (*backward-view*)

Plan

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

Table of Contents

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

Model Free Control

- Précédemment (programmation dynamique), l'amélioration *greedy* de la politique se fait selon la fonction de valeur d'état :

$$\pi_{k+1}(s) = \operatorname{argmax}_{a \in \mathcal{A}} q_{\pi}(s, a)$$

- → on choisit les action qui amèneront aux meilleurs états
- Nécessité de stocker la valeur de chaque état $v_{\pi}(s)$

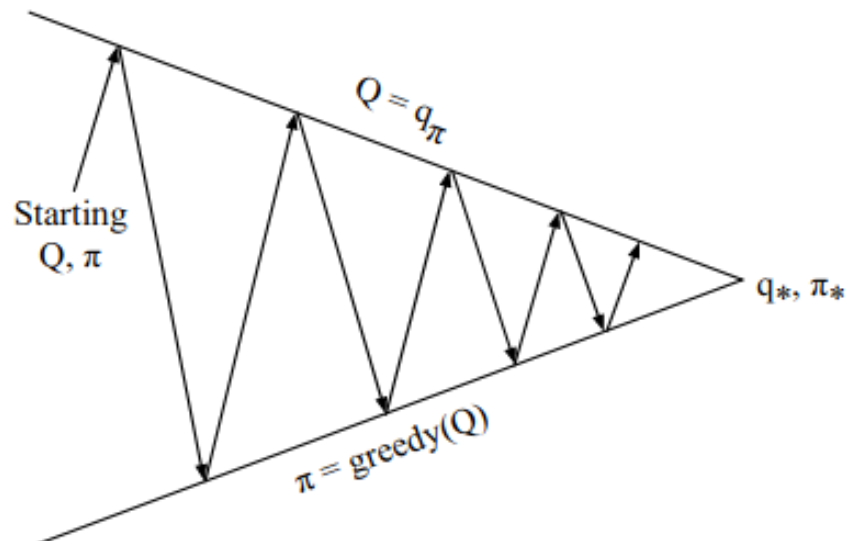
Model Free Control

- Précédemment (programmation dynamique), l'amélioration *greedy* de la politique se fait selon la fonction de valeur d'état :

$$\pi_{k+1}(s) = \operatorname{argmax}_{a \in \mathcal{A}} q_{\pi}(s, a)$$

- → on choisit les action qui amèneront aux meilleurs états
- Nécessité de stocker la valeur de chaque état $v_{\pi}(s)$
- Sans connaître la dynamique du modèle, on ne sait pas quels sont les états atteignables
- → Nécessité de stocker la valeur de chaque couple (action-état) $q_{\pi}(s, a)$

Generalized policy iteration



- Même principe qu'en programmation dynamique
- Alternance :
 - Estimation de q_π
 - Amélioration *greedy* de la politique

GLIE

Condition **suffisante** pour la convergence vers q_{π^*}

Définition

Un processus d'amélioration de politique est dit *Greedy in the Limit with Infinite Exploration (GLIE)* si

- Tous les états-action sont explorés infiniment :

$$\lim_{k \rightarrow \infty} N_k(s, a) = \infty$$

- La politique converge vers une politique *greedy* :

$$\lim_{k \rightarrow \infty} \pi_k(a|s) = \mathbb{1}_{\operatorname{argmax}_{a \in \mathcal{A}} q_k(s, a)}$$

GLIE

Condition **suffisante** pour la convergence vers q_{π^*}

Définition

Un processus d'amélioration de politique est dit *Greedy in the Limit with Infinite Exploration (GLIE)* si

- Tous les états-action sont explorés infiniment :

$$\lim_{k \rightarrow \infty} N_k(s, a) = \infty$$

- La politique converge vers une politique *greedy* :

$$\lim_{k \rightarrow \infty} \pi_k(a|s) = \mathbb{1}_{\operatorname{argmax}_{a \in \mathcal{A}} q_k(s, a)}$$

Par exemple : un apprentissage ϵ -*greedy* est GLIE si $\epsilon_k = \frac{1}{k}$

ϵ -greedy policy iteration

Intuition : explorer continuellement tout en convergent vers la politique optimale

- Toutes les actions ont une probabilité non nulle
- L'action qui maximise $q(s, a)$ est prise avec la probabilité ~~ϵ~~ $1 - \epsilon$
- Les autres actions peuvent être choisies avec une probabilité ϵ

$$\pi_{\text{greedy}}(a|s) = \begin{cases} \frac{\epsilon}{m} + 1 - \epsilon & \text{si } a^* = \operatorname{argmax}_{a \in \mathcal{A}} q(s, a) \\ \frac{\epsilon}{m} & \text{sinon} \end{cases}$$

avec $m = |\mathcal{A}|$

GLIE Monte Carlo

- Échantillonner k épisodes basés sur $\pi : \{S_1, A_1, R_2, \dots, S_T\}$
- Pour chaque couple (s_t, a_t) :

$$N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \frac{1}{N(s_t, a_t)} (G_t - Q(s_t, a_t))$$

- Amélioration ϵ -greedy de π :

$$\epsilon \leftarrow \frac{1}{k}$$

$$\pi \leftarrow \epsilon - \text{greedy}(Q)$$

GLIE Monte Carlo

- Échantillonner k épisodes basés sur $\pi : \{S_1, A_1, R_2, \dots, S_T\}$
- Pour chaque couple (s_t, a_t) :

$$N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \frac{1}{N(s_t, a_t)} (G_t - Q(s_t, a_t))$$

- Amélioration ϵ -greedy de π :

$$\epsilon \leftarrow \frac{1}{k}$$

$$\pi \leftarrow \epsilon - \text{greedy}(Q)$$

GLIE MC converge vers la fonction action-valeur optimale

Table of Contents

1 Introduction

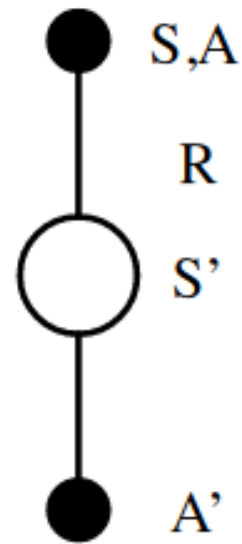
2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

SARSA



$$Q(S, A) \leftarrow Q(S, A) + \alpha (R + \gamma Q(S', A') - Q(S, A))$$

SARSA

Initialize $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\text{terminal-state}, \cdot) = 0$

Repeat (for each episode):

 Initialize S

 Choose A from S using policy derived from Q (e.g., ϵ -greedy)

 Repeat (for each step of episode):

 Take action A , observe R, S'

 Choose A' from S' using policy derived from Q (e.g., ϵ -greedy)

$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$

$S \leftarrow S'; A \leftarrow A';$

 until S is terminal

SARSA

```

Initialize  $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$ , arbitrarily, and  $Q(\text{terminal-state}, \cdot) = 0$ 
Repeat (for each episode):
  Initialize  $S$ 
  Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)
  Repeat (for each step of episode):
    Take action  $A$ , observe  $R, S'$ 
    Choose  $A'$  from  $S'$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)
     $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$ 
     $S \leftarrow S'; A \leftarrow A';$ 
  until  $S$  is terminal

```

SARSA converge vers la politique optimale ($Q(s, a) \rightarrow q^*(s, a)$) si,

- La séquence des politiques est GLIE
- Le *learning rate* α_t vérifie les conditions suivantes :

$$\sum_{t=1}^{\infty} \alpha_t = \infty$$

$$\sum_{t=1}^{\infty} \alpha_t^2 < \infty$$

Extensions

SARSA similaire à l'estimation avec TD(0) :

- On estime la fonction de valeur d'état-action $q(s, a)$ par mise à jour de la **TD Target** :

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

- **contrôle** : on cherche à suivre une politique optimale $\rightarrow$
 $\pi_t = \epsilon\text{-greedy}(Q(s_t, a_t))$

Extensions

SARSA similaire à l'estimation avec TD(0) :

- On estime la fonction de valeur d'état-action $q(s, a)$ par mise à jour de la **TD Target** :

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

- **contrôle** : on cherche à suivre une politique optimale $\rightarrow$
 $\pi_t = \epsilon\text{-greedy}(Q(s_t, a_t))$

L'estimation peut être étendue aux extension de TD déjà rencontrée

Table of Contents

1 Introduction

2 Model Free Prediction

- Monte Carlo sampling
- TD0 Learning
- TD(λ) Learning

3 *Model Free Control*

- Introduction
- On policy et SARSA
- Off policy et Q-learning

Off-Policy learning

Intuition : Apprendre d'une autre politique

- Évaluer une politique cible $\pi(s|a) : v_{\pi}(s), q_{\pi}(s, a)$
- Suivre une politique différente (comportement) : $\mu(a|s)$

$$\{S_1, A_1, R_2, \dots, S_T\} \sim \mu$$

- Nombreux cas d'utilisation :
 - Apprendre d'humains
 - Ré-utiliser l'expérience passée
 - Apprendre une politique **optimale** tout en suivant une politique **exploratoire**

Importance sampling

- Espérance de $f(X)$ avec $X \sim P$:

$$\mathbb{E}_P[f(X)] = \sum P(X)f(X)$$

- Espérance de $f(X)$ par rapport à une autre distribution Q :

$$\begin{aligned}
 E_P[f(X)] &= \sum_{X \sim P} P(X)f(X) \\
 &= \sum_{X \sim Q} Q(X) \frac{P(X)}{Q(X)} f(X) \\
 &= \mathbb{E}_{X \sim Q} \left[\underbrace{\frac{P(X)}{Q(X)}}_{\text{facteur de pondération (FI)}} f(X) \right]
 \end{aligned} \tag{1}$$

Off-Policy Monte-Carlo

- Utiliser les retour générer par μ (le comportement) pour évaluer π
- Pondération du retour par le facteur de correction sur tout l'épisode :

$$G_t^{\pi/\mu} = \frac{\pi(A_t|S_t)\pi(A_{t+1}|S_{t+1})\dots\pi(A_T|S_T)}{\mu(A_t|S_t)\mu(A_{t+1}|S_{t+1})\dots\mu(A_T|S_T)} \quad G_t$$

- La valeur est mise à jour grâce au retour corrigé :

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^{\pi/\mu} - V(S_t))$$

- Pour que la correction soit définie : $\mu \neq 0$ si $\pi \neq 0$
- Peut entraîner des problème de stabilité

Off-Policy TD

- Utiliser les retour génér~~é~~^{er} par μ (le comportement) pour évaluer π
- Pondération de la TD target facteur de correction sur pour une étape lors de la mise à jour :

$$V(S_t) \leftarrow V(S_t) + \alpha \left(\frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} (R_{t+1} + \gamma V(S_{t+1})) - V(S_t) \right)$$

Q-learning

- Off policy TD learning pour l'estimation de la fonction de valeur (état, action)
- Importance sampling **plus** nécessaire
- $A_{t+1} \sim \mu(.|S_{t+1})$
- On considère une action alternative $A' \sim \pi(.|S_{t+1})$
- La TD target est distribuée selon μ mais on a choisi une action A' donnée par π :

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left((R_{t+1} + \gamma Q(S_{t+1}, A')) - Q(S_t, A_t) \right)$$


 update $Q(S_t, A_t)$ towards value of alternative action $A' \sim \pi$

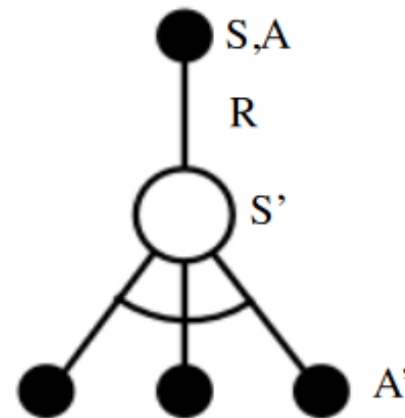
Q-learning

- Choix de μ et π ?
- μ gère l'exploration
- π cherche l'optimalité
- Simple schéma off-policy à partir d'une même politique :

$$\pi(S|A) = \underset{a'}{\text{greedy}}(Q) = \underset{a'}{\text{argmax}} Q(S, A)$$

$$\mu(S|A) = \epsilon - \text{greedy}(Q)$$

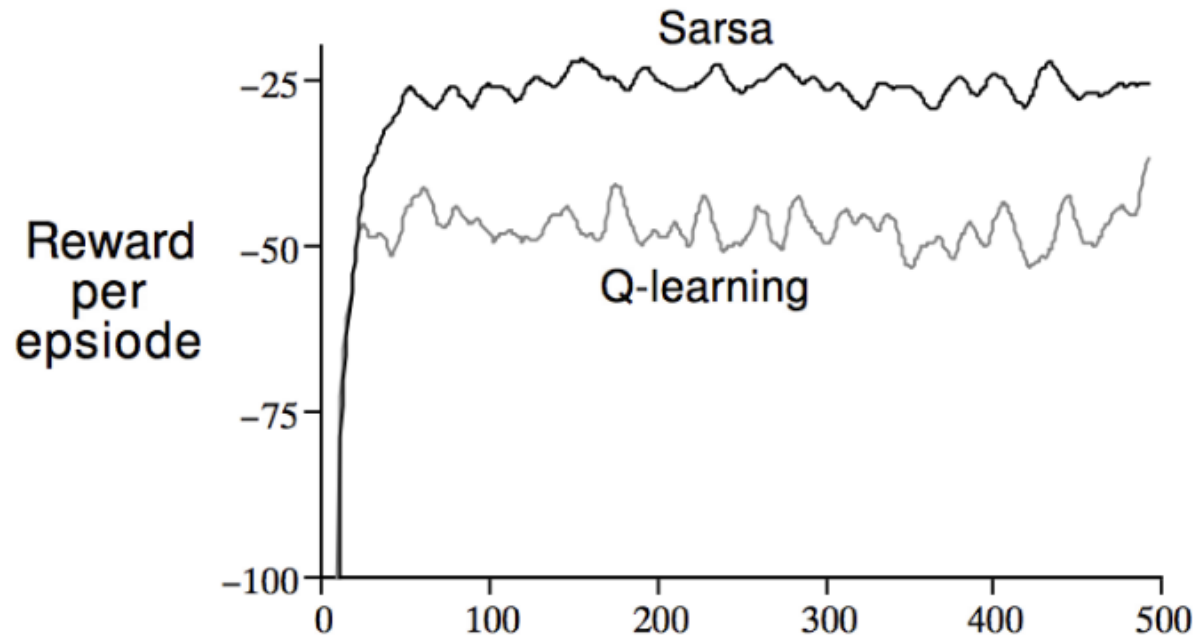
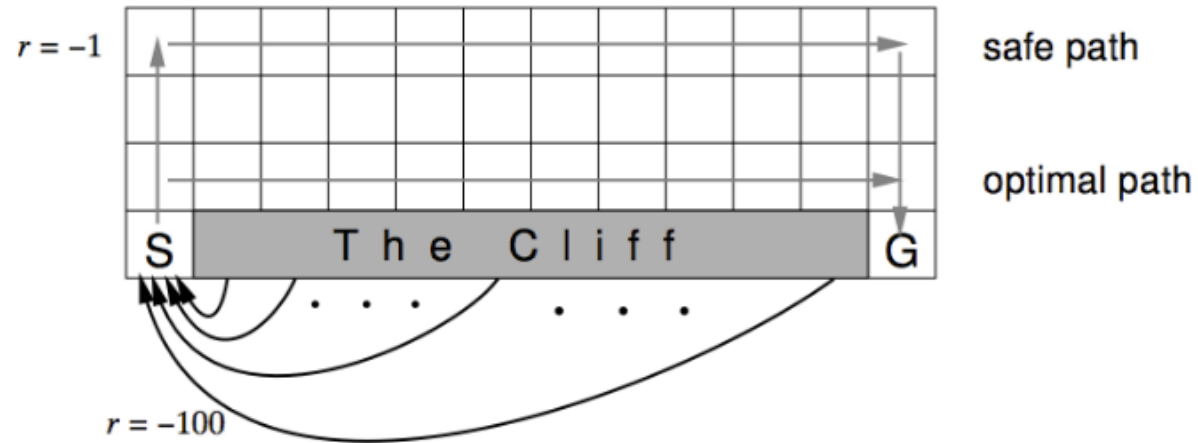
Q-learning



$$Q(S, A) \leftarrow Q(S, A) + \alpha \left(R + \gamma \max_{a'} Q(S', a') - Q(S, A) \right)$$

Q-learning

Initialize $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\text{terminal-state}, \cdot) = 0$
Repeat (for each episode):
 Initialize S
 Repeat (for each step of episode):
 Choose A from S using policy derived from Q (e.g., ϵ -greedy)
 Take action A , observe R, S'
 $Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$
 $S \leftarrow S'$;
 until S is terminal



Bibliographie

- Reinforcement Learning : an introduction, second edition, Richard S. Sutton and Andrew G. Barto
- Reinforcement Learning courses, David Silver, DeepMind (<https://www.davidsilver.uk/>)