Apprentissage par renforcement appliqué

Algorithmes fondamentaux [revision 3.2]

Brahim Chaib-draa

Brahim.Chaib-Draa@ift.ulaval.ca

Université Laval

2020-05-26

1. Concept clé en RL
2. Paysage algorithmique de l'apprentissage par renforcement
3. Modèle connu et parfait $\Longrightarrow$ pas d'aprentissage requis
4. Apprentissage par renforcement (sans modèle)
5. Pour aller plus loin

Concept clé en RL

1. **Concept clé en RL**
   - Apprentissage *sans modèle* vs apprentissage *basée sur un modèle*
   - Apprentissage vs Planification
   - Prediction vs Contrôle
   - Apprentissage EN-ligne vs HORS-ligne
2. **Paysage algorithmique de l'apprentissage par renforcement**
3. **Modèle connu et parfait $\Longrightarrow$ pas d'aprentissage requis**
4. **Apprentissage par renforcement (sans modèle)**
5. **Pour aller plus loin**

Concept clé en RL

Apprentissage *sans modèle* vs apprentissage *basée sur un modèle*

Concept HAUT niveau :

- Apprentissage **sans modèle** vs apprentissage **basée sur un modèle**
  [*Model-free* vs *model-based*]
- **Apprentissage** vs **Planification**
  [*Learning* vs *Planning*]

Concept BAS niveau :

- **Prédiction** vs **Contrôle**
  [*Prediction* vs *Control*]
- Apprentissage **EN-ligne** vs **HORS-ligne**
  [*ON-line learning* vs *OFF-line learning*]

Apprentissage sans modèle vs apprentissage basée sur un modèle

Concept HAUT niveau

Méthode sans modèle [*Model-free*]

- L'agent **ne dépend pas d'un model** pour apprendre ;
- Il apprend par essai et erreur ;

Méthode basée sur un modèle [*Model-based*]

1. Soit l'agent **apprend un modèle** de A à Z ;
2. ou l'agent **améliore un modèle** par expérience ;
3. ou l'agent connait le modèle à priori ;

Concept clé en RL

Apprentissage vs Planification

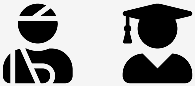
Apprentissage vs Planification

Concept HAUT niveau

Apprentissage [*Learning*]

Acquérir des habiletés ou des connaissances soit par :

1. essai et erreur ;
2. imitation ;
3. ou sous la supervision d'un professeur Ex. *curiculum learning* ;



Planification [*Planning*]

Déterminer les étapes nécessaires à l'atteinte d'un **but** et mesurer les efforts nécessaires ;

Prérequis : avoir **un modèle de l'environnement**

Ex. un calendrier, une carte, les règles de physique, les règles d'un jeu ...

Concept clé en RL

Prediction vs Contrôle

Prediction vs Contrôle

Concept BAS niveau

Prediction [*Prediction*]

Basée sur mes **connaissances actuelles**, qu'est-ce que je peux **déduire/induire** (à propos du **futur**)[1].

Implique une notion d'incertitude.

Ex. de prédiction : estimer Q^π ou V^π

Contrôle [*Control*]

Choisir une stratégie **optimale** afin d'atteindre un **but**.

Ex. de contrôle : estimer $V^\star$, $Q^\star$ ou $\pi^\star$

1. Contrairement à son utilisation dans le domaine des statistiques où le terme prédiction n'implique pas nécessairement la notion de futur.

Concept clé en RL

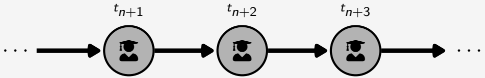
Apprentissage EN-ligne vs HORS-ligne

Apprentissage EN-ligne vs HORS-ligne

Concept BAS niveau

Apprentissage EN-ligne [*ON-line learning*]

L'apprentissage se fait **au fur et à mesure** que l'agent exécute des actions dans l'environnement.



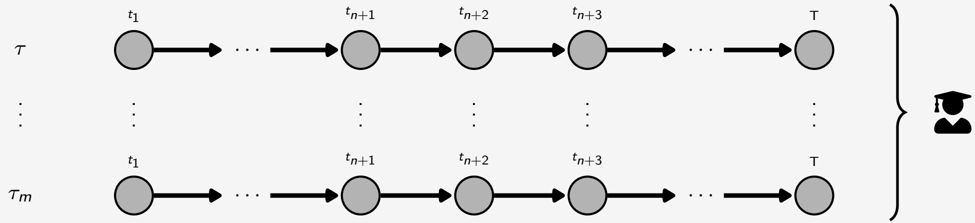
Exemple : *TD-learning*.

Apprentissage EN-ligne vs HORS-ligne

Concept BAS niveau

Apprentissage HORS-ligne [*OFF-line learning*]

La phase d'apprentissage est **retardée après la collecte d'une série de trajectoires** τ.



avec m le nombre de trajectoire collecté [$m =$ *batch size*]

Ex : *Batch RL*. ▸ wikipédia.

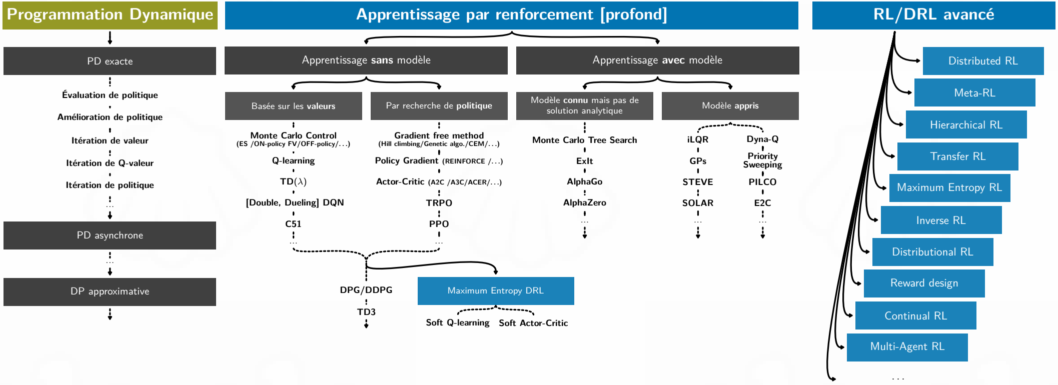
Paysage algorithmique de l’apprentissage par renforcement

Paysage algorithmique de l'apprentissage par renforcement

Taxonomie du RL/DRL

Apprentissage par renforcement [profond]

- Apprentissage **sans** modèle
  - Basée sur les **valeurs**
    - **Monte Carlo Control** (ES /ON-policy FV/OFF-policy/...)
    - **Q-learning**
    - **TD(λ)**
    - **[Double, Dueling] DQN**
    - **C51**
    - ...
  - Par recherche de **politique**
    - **Gradient free method** (Hill climbing/Genetic algo./CEM/...)
    - **Policy Gradient** (REINFORCE /...)
    - **Actor-Critic** (A2C /A3C/ACER/...)
    - **TRPO**
    - **PPO**
    - ...
  - (valeurs + politique)
    - **DPG/DDPG**
    - **TD3**
    - Maximum Entropy DRL
      - **Soft Q-learning**
      - **Soft Actor-Critic**
- Apprentissage **avec** modèle
  - Modèle **connu** mais pas de solution analytique
    - **Monte Carlo Tree Search**
    - **ExIt**
    - **AlphaGo**
    - **AlphaZero**
    - ...
  - Modèle **appris**
    - **iLQR**
    - **GPs**
    - **STEVE**
    - **SOLAR**
    - ...
    - **Dyna-Q**
    - **Priority Sweeping**
    - **PILCO**
    - **E2C**
    - ...


Programmation Dynamique
PD exacte
Évaluation de politique
Amélioration de politique
Itération de valeur
Itération de Q-valeur
Itération de politique
...
PD asynchrone
...
DP approximative
Apprentissage par renforcement [profond]
Apprentissage sans modèle
Apprentissage avec modèle
Basée sur les valeurs
Par recherche de politique
Modèle connu mais pas de solution analytique
Modèle appris
Monte Carlo Control
(ES /ON-policy FV/OFF-policy/...)
Q-learning
TD(λ)
[Double, Dueling] DQN
C51
...
Gradient free method
(Hill climbing/Genetic algo./CEM/...)
Policy Gradient (REINFORCE /...)
Actor-Critic (A2C /A3C/ACER/...)
TRPO
PPO
...
DPG/DDPG
TD3
Maximum Entropy DRL
Soft Q-learning
Soft Actor-Critic
Monte Carlo Tree Search
ExIt
AlphaGo
AlphaZero
...
iLQR
GPs
STEVE
SOLAR
...
Dyna-Q
Priority Sweeping
PILCO
E2C
...
RL/DRL avancé
Distributed RL
Meta-RL
Hierarchical RL
Transfer RL
Maximum Entropy RL
Inverse RL
Distributional RL
Reward design
Continual RL
Multi-Agent RL
...

Pourquoi autant d'algorithmes ?

Pourquoi autant d'algorithmes ?

Différent compromis Ex : On a peu de puissance de calcul, mais générer de nouvelles trajectoires est très rapides, donc c'est plus efficace d'utiliser un algorithme de type *Monté Carlo* qu'un par *Différence Temporelle* ;

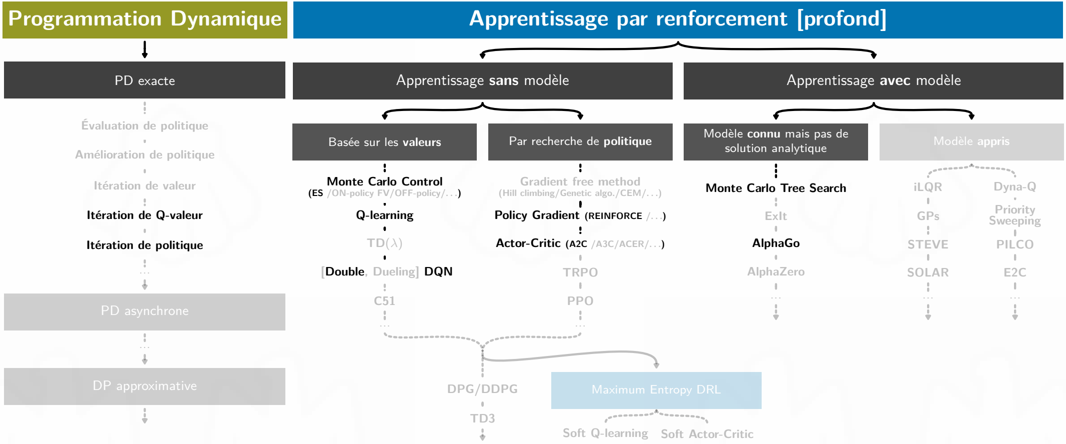
Différentes assomptions Ex : On fait l'assomption qu'on a accès à un modèle parfait, donc on peut utiliser un algorithme de *Programmation Dynamique* ;

Différente contrainte Ex : On a besoin d'un algorithme qui fonctionne sur des espaces d'actions continue, donc on ne peux pas utiliser les algorithmes de la famille *Q-learning* ;

Plus de détail au fur et à mesure

Note : Voir *Reinforcement Learning coach/Selecting an Algorithm* pour un bon exemple illustrant le niveau de complexité que représente le choix d'algorithme en RL. ▸ RL Coach / selecting an algorithm

Algorithme couvert dans le cours



Algorithme vue dans cette partie

Problème de décision séquentiel (PDS)

« Modèle connu **et** parfait »

Programmation dynamique (exacte)

Itération de Q-valeur

Itération de politique

« Modèle inconnu **ou** imparfait »

Apprentissage par renforcement

Basé sur les **valeurs**

Monte Carlo
avec *Départ Exploratoire*

Différence Temporelle
Q-learning (tabulaire)

Par recherche de **politique**

Paysage algorithmique de l'apprentissage par renforcement

Comparaison des types de méthode en RL

Terminologie

« Backup Operation »

L'opération de **mise à jour** d'un *V-valeur* (ou *Q-valeur*) par une **estimation** de son possible successeur [1]

$$V\text{-}valeur \longleftarrow f_{\text{approximation du successeur possible de } V\text{-}valeur}$$

1. Voir l'excellente explication par Andre G. Barto pour plus de détail dans REINFORCEMENT LEARNING AND DYNAMIC PROGRAMMING, section 2.2 [1]

Programmation dynamique : Estimer la valeur en utilisant l'**espérance mathématique**, le **modèle** de l'environnement et l'**estimations courantes** ;

Ex. *Backup operation* pour Itération de Q-valeur

$$Q^{\pi}_{i+1}(\mathbf{s},\mathbf{a}) \longleftarrow \mathbb{E}_{\mathbf{s}' \sim p}\left[r(\mathbf{s},\mathbf{a},\mathbf{s}') + \gamma \max_{\mathbf{a}' \in \mathcal{A}(\mathbf{s}')} Q^{\pi}(\mathbf{s}',\mathbf{a}') \right]$$

RL par méthode Monté Carlo : Estimer la valeur en utilisant la **moyenne empirique** sur des **échantillons de trajectoire τ complets** ;

Ex. *Backup operation* pour *Monte Carlo ES*

$$Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) \longleftarrow Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) + \tfrac{1}{k}\left[G_t(\tau) - Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) \right]$$

RL par Différence Temporelle : Estimer la valeur en utilisant des **échantillons de trajectoire τ partielle + l'estimation courantes** du reste de la trajectoire ;

Ex. *Backup operation* pour *Q-learning*

$$Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) \longleftarrow Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) + \alpha\left[r(\mathbf{s}_t,\mathbf{a}_t,\mathbf{s}_{t+1}) + \gamma \max_{\mathbf{a}' \in \mathcal{A}(\mathbf{s}_{t+1})} Q^{\pi}(\mathbf{s}_{t+1},\mathbf{a}') - Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) \right]$$

Programmation dynamique : Estimer la valeur en utilisant l'**espérance mathématique**, le **modèle** de l'environnement et l'**estimations courantes** ;

Ex. *Backup operation* pour Itération de Q-valeur

$$Q^{\pi}_{i+1}(\mathbf{s},\mathbf{a}) \longleftarrow \mathbb{E}_{\mathbf{s}' \sim p}\left[\underbrace{r(\mathbf{s},\mathbf{a},\mathbf{s}')}_{\textit{modèle}} + \gamma \max_{\mathbf{a}' \in \mathcal{A}(\mathbf{s}')} Q^{\pi}(\mathbf{s}',\mathbf{a}')\right]$$

RL par méthode Monté Carlo : Estimer la valeur en utilisant la **moyenne empirique** sur des **échantillons de trajectoire τ complets** ;

Ex. *Backup operation* pour *Monte Carlo ES*

$$Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) \longleftarrow Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) + \frac{1}{k}\left[\underbrace{G_t(\tau)}_{\textit{échantillon}} - Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t)\right]$$

RL par Différence Temporelle : Estimer la valeur en utilisant des **échantillons de trajectoire τ partielle** + **l'estimation courantes** du reste de la trajectoire ;

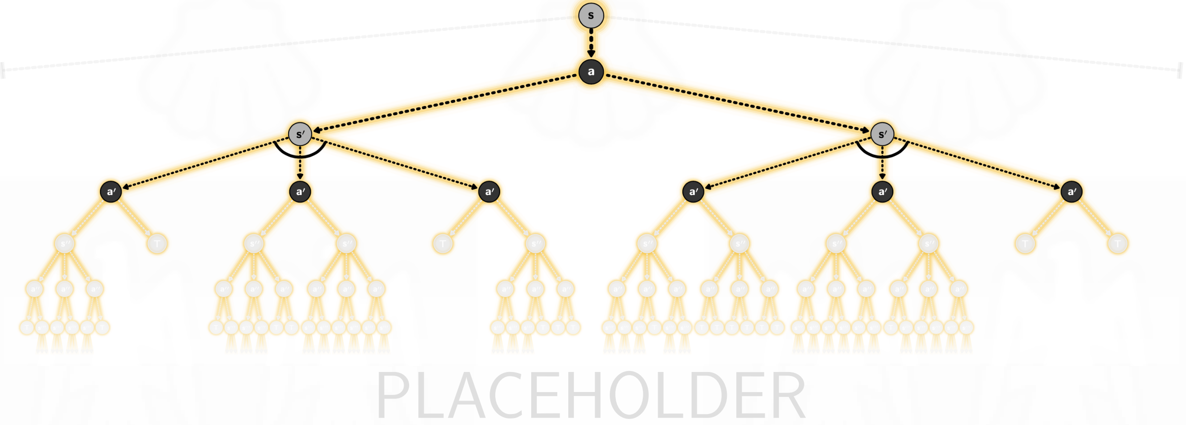
Ex. *Backup operation* pour *Q-learning*

$$Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) \longleftarrow Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t) + \alpha\left[\underbrace{r(\mathbf{s}_t,\mathbf{a}_t,\mathbf{s}_{t+1})}_{\textit{échantillon}} + \gamma \max_{\mathbf{a}' \in \mathcal{A}(\mathbf{s}_{t+1})} Q^{\pi}(\mathbf{s}_{t+1},\mathbf{a}') - Q^{\pi}(\mathbf{s}_t,\mathbf{a}_t)\right]$$

Backup Diagram de Programmation Dynamique

Ex.: Algorithme *Itération de Q-valeur*

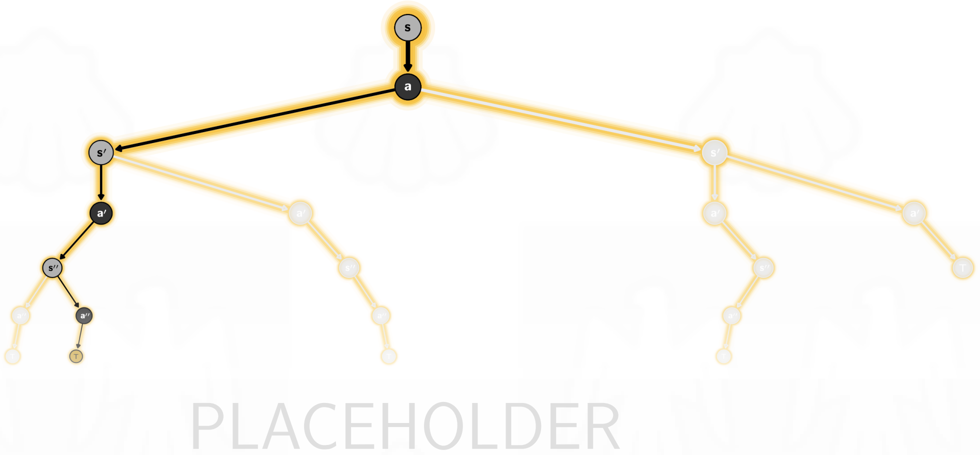
$$Q^{\pi}_{i+1}(\mathbf{s},\mathbf{a}) \longleftarrow \mathbb{E}_{\mathbf{s}'\sim p}\left[r(\mathbf{s},\mathbf{a},\mathbf{s}') + \gamma \max_{\mathbf{a}'\in\mathcal{A}(\mathbf{s}')} Q^{\pi}(\mathbf{s}',\mathbf{a}')\right]$$



Backup Diagram de méthode Monté Carlo

Ex.: Algorithme *Monté Carlo ES*

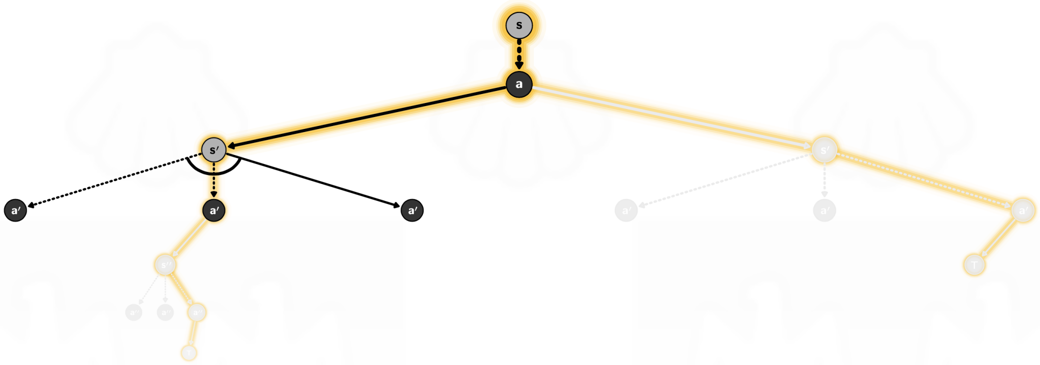
$$Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t) + \frac{1}{k}\left[G_t(\tau) - Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t)\right]$$



Backup Diagram de méthode par Différence Temporelle

Ex.: Algorithme *Q-Learning*

$$Q^{\pi}(s_t, a_t) \longleftarrow Q^{\pi}(s_t, a_t) + \alpha \left[r(s_t, a_t, s_{t+1}) + \gamma \max_{a' \in \mathcal{A}(s_{t+1})} Q^{\pi}(s_{t+1}, a') - Q^{\pi}(s_t, a_t) \right]$$



PLACEHOLDER

Modèle connu et parfait $\Longrightarrow$ pas d'aprentissage requis

1 Concept clé en RL

2 Paysage algorithmique de l'apprentissage par renforcement

3 **Modèle connu et parfait $\Longrightarrow$ pas d'aprentissage requis**
- Méthode par Programmation Dynamique exacte
- DP : Itération de Q-valeur
- DP : Itération de politique

4 Apprentissage par renforcement (sans modèle)

5 Pour aller plus loin

Modèle connu **et** parfait $\Longrightarrow$ pas d'aprentissage requis

---

Méthode par Programmation Dynamique exacte

Programmation dynamique (exacte) et processus de décision séquentielle

« Suposont qu'après avoir lu un livre sur les cubes Rubik, on sait comment résoudre tout les cubes Rubik . . . juste en réfléchissant et sans jamais en avoir pris un dans ses mains. »

Image : Cube Rubik

Caractéristique :

- Résolution par planification $\Longrightarrow$ **requiert un modèle** de l'environnement ;
- Type d'algorithme **hors-ligne** et **sans phase d'apprentissage** ;
  c.-à-d. que la solution est calculée sans que l'agent n'ait à intéragir avec l'environnement

Idées clés :

1. **Décomposer** le problème difficile **en sous-problèmes** plus simple ;
2. Trouver une **solution optimale** à chaque sous-problèmes ;
3. **Résoudre récursivement** en utilisant les solutions des sous-problèmes ;

Programmation dynamique (exacte) et processus de décision séquentielle

Pro :
- **Garantie** de trouver une solutions optimale exacte ;

Con :
- **Avoir accès au modèle** de l'environnement ;
- Fonctionne sous l'**assomption** que le modèle est **parfait** (c.a.d. sans zone grise, ni erreur) ;
- Victime de la **malédiction de la dimensionnalité** :

$\Longrightarrow$ Très gourmand en mémoire et sur le plan computationnel
Perd en efficacité pratique plus l'espaces d'état/action devient massif [1] ;

Pourquoi étudier ces méthodes dans un cours sur le RL alors ?

- Toutes les méthodes de RL **aspirent à des résultats similaire** à ceux obtenue par DP, **mais en se libérant des contraintes** du DP [2] ;
- Procure une solide **fondation** pour comprendre certains mécanismes fondamentaux de l'apprentissage par renforcement ;

1. Voir « Reinforcement Learning : An introduction », chapitre 4, section 7 « Efficiency of Dynamic Programming » par Sutton & Barto [2]
2. Quite à perde certaines propriété du DP à l'ocasion : garantie de solution optimale, production d'une solution exacte

Programmation dynamique (exacte) et processus de décision séquentielle

Pro :
- **Garantie** de trouver une solutions optimale exacte ;

Con :
- **Avoir accès au modèle** de l'environnement ;
- Fonctionne sous l'**assomption** que le modèle est **parfait** (c.a.d. sans zone grise, ni erreur) ;
- Victime de la **malédiction de la dimensionnalité** :

$\Longrightarrow$ Très gourmand en mémoire et sur le plan computationnel

Perd en efficacité pratique plus l'espaces d'état/action devient massif [1] ;

Pourquoi étudier ces méthodes dans un cours sur le RL alors ?

- Toutes les méthodes de RL **aspirent à des résultats similaire** à ceux obtenue par DP, **mais en se libérant des contraintes** du DP [2] ;
- Procure une solide **fondation** pour comprendre certains mécanismes fondamentaux de l'apprentissage par renforcement ;

1. Voir « Reinforcement Learning : An introduction », chapitre 4, section 7 « Efficiency of Dynamic Programming » par Sutton & Barto [2]
2. Quite à perde certaines propriété du DP à l'ocasion : garantie de solution optimale, production d'une solution exacte

Terminologie

« Backup Operation »

L'opération de **mise à jour** d'un *V-valeur* (ou *Q-valeur*) par une **estimation** de son possible successeur [1]

$$V\text{-}valeur \longleftarrow f_{\text{approximation du successeur possible de } V\text{-}valeur}$$

« Sweep »

Une execution du « *Backup Operation* » sur chaque *V-valeurs* (ou *Q-valeurs*) d'un espace.

1. Voir l'excellente explication par Andre G. Barto pour plus de détail dans REINFORCEMENT LEARNING AND DYNAMIC PROGRAMMING, section 2.2 [1]

Modèle connu et parfait $\Longrightarrow$ pas d'aprentissage requis

DP : Itération de Q-valeur

Itération de Q-valeur [*Q-value itération*]

« Think about every possible scenario . . . obsessively, leave no stone unturned. »

Idées clés :

1. **Estimer et stocker** la Q-valeur **de chaque couples** *état-action* de l'univers ;
2. **Répéter [infiniment]** en améliorant progressivement l'estimation jusqu'à convergence vers une solution exacte ;

$$Q_i^\pi \xrightarrow{i \to \infty} Q^\star$$

Pro :

- Intuitif et rapide à implémenter ;
- Convergence garantie [1] ;

Con :

- Mémoire requise pour la *Q-table* [2] : $2 \cdot |\mathcal{S} \times \mathcal{A}|$
- Limiter aux problèmes avec espace d'état/action de taille modeste ;
  Exemple : espace d'action discrète, facteur de branchement limité, . . .

1. Voir « Reinforcement Learning : An introduction », chapitre 4 (introduction, 4.4 et 4.6) par Sutton & Barto [2]
2. Version synchrone de l'algorithme : les *backup operation* sont effectué en parallèle

Itération de Q-valeur [*Q-value itération*]

« Think about every possible scenario . . . obsessively, leave no stone unturned. »

Idées clés :

1. **Estimer et stocker** la Q-valeur **de chaque couples** *état-action* de l'univers ;
2. **Répéter [infiniment]** en améliorant progressivement l'estimation jusqu'à convergence vers une solution exacte ;

$$Q_i^{\pi} \xrightarrow{i \to \infty} Q^{\star}$$

Pro :

- Intuitif et rapide à implémenter ;
- Convergence garantie [1] ;

Con :

- Mémoire requise pour la *Q-table* [2] : $2 \cdot |\mathcal{S} \times \mathcal{A}|$
- Limiter aux problèmes avec espace d'état/action de taille modeste ;
  Exemple : espace d'action discrète, facteur de branchement limité, . . .

1. Voir « Reinforcement Learning : An introduction », chapitre 4 (introduction, 4.4 et 4.6) par Sutton & Barto [2]
2. Version synchrone de l'algorithme : les *backup operation* sont effectué en parallèle

Algorithme | Itération de Q-valeur

Répéter avec $i = 1, 2, 3, \ldots$ **jusqu'à convergence** vers $Q^\star$:

$$Q^\pi_{i+1}(\mathbf{s}, \mathbf{a}) \longleftarrow \sum_{\mathbf{s}' \in \mathcal{S}_{t+1}} p(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \left[r(\mathbf{s}, \mathbf{a}, \mathbf{s}') + \gamma \max_{\mathbf{a}' \in \mathcal{A}(\mathbf{s}')} Q^\pi_i(\mathbf{s}', \mathbf{a}') \right] \qquad \forall\ \mathbf{s} \in \mathcal{S},\ \mathbf{a} \in \mathcal{A}(\mathbf{s})$$

En pratique | Itération de Q-valeur

On **arrête prématurément** l'algorithme au lieu d'itérer à l'infinie en implémentant un moyen d'évaluer l'estimation et en spécifiant la marge d'erreur ε visé.

Conséquence :

- L'algorithme **converge approximativement** : $Q^\pi \xrightarrow{\approx} Q^\star$
- et **retourne** $\pi \approx \pi^\star$ au lieu de la valeur exacte

Pseudocode | Itération de Q-valeur [1] (Version synchrone)

Init: **foreach** $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$ **do** `Q-table`$[s, a] \leftarrow 0$
Init: threshold `hyperparam`$(\epsilon) > 0$ « Critère d'arrêt »

repeat « sweep i »
    $\Delta \leftarrow 0$
    $\texttt{Q-table}_{old} \leftarrow \texttt{Q-table}$
    foreach $s \in \mathcal{S}$ **and** $a \in \mathcal{A}(s)$ **do**
        $\texttt{Q-table}[s, a] \leftarrow \sum_{s' \in \mathcal{S}} p(s'|s, a) \left[r(s, a, s') + \gamma \max_{a' \in \mathcal{A}(s')} \texttt{Q-table}_{old}[s', a'] \right]$ « Backup operation »
        $\Delta \leftarrow \max\left(\Delta, \left| \texttt{Q-table}_{old}[s, a] - \texttt{Q-table}[s, a] \right|\right)$
until $\Delta < \epsilon$ « $Q^{\pi} \xrightarrow{\approx} Q^{\star}$ »

Output: $\pi^\star(s) \approx \arg\max_{a \in \mathcal{A}(s)} \texttt{Q-table}[s, a]$ « Produit une politique déterministe »

1. Pseudocode inspiré de « Reinforcement Learning : An introduction », chapitre 4.4, page 83 « Value Iteration » par Sutton & Barto [2]

Modèle connu **et** parfait $\Longrightarrow$ pas d'aprentissage requis

DP : Itération de politique

Itération de politique [*Policy iteration*]

« Think freely, see where it leads you . . . explore until you've seen it all. »

Idées clés :

1. À chaque *« sweep »*, **considérer une seule action par état** (fixer la politique) ;
2. **Estimer et stocker** la V-valeur **de chaque** *état* de l'univers ;
3. **Choisir une meilleur** politique en évaluant le **futur immédiat** *(one-step look-ahead)* ;
4. **Répéter [infiniment]** en améliorant progressivement l'estimation jusqu'à convergence vers une solution exacte ;

$$V_k^{\pi} \xrightarrow{k \to \infty} V_k^{\star} \quad \text{et} \quad \pi_k \xrightarrow{k \to \infty} \pi^{\star}$$

Pro :

- Intuitif et rapide à implémenter ;
- Convergence garantie [1] et plus rapide [2] que *Itération de Q-valeur* ;

Con :

- Mémoire requise pour la *V-table* et la *π-table* [3] : $3 \cdot |\mathcal{S}|$
- Limiter aux problèmes avec espace d'état/action de taille modeste ;

1. Voir « Reinforcement Learning : An introduction », section 4.0, 4.3 et 4.6 par Sutton & Barto [2]
2. Sous certaines condition. Voir preuve de convergence
3. Version synchrone de l'algorithme : les *backup operation* sont effectué en parallèle

Itération de politique [*Policy iteration*]

« Think freely, see where it leads you . . . explore until you've seen it all. »

Idées clés :

1. À chaque *« sweep »*, **considérer une seule action par état** (fixer la politique) ;
2. **Estimer et stocker** la V-valeur **de chaque** *état* de l'univers ;
3. **Choisir une meilleur** politique en évaluant le **futur immédiat** *(one-step look-ahead)* ;
4. **Répéter [infiniment]** en améliorant progressivement l'estimation jusqu'à convergence vers une solution exacte ;

$$V_k^{\pi} \xrightarrow{k\to\infty} V_k^{\star} \quad \text{et} \quad \pi_k \xrightarrow{k\to\infty} \pi^{\star}$$

Pro :

- Intuitif et rapide à implémenter ;
- Convergence garantie [1] et plus rapide [2] que *Itération de Q-valeur* ;

Con :

- Mémoire requise pour la *V-table* et la *π-table* [3] : $3 \cdot |\mathcal{S}|$
- Limiter aux problèmes avec espace d'état/action de taille modeste ;

1. Voir « Reinforcement Learning : An introduction », section 4.0, 4.3 et 4.6 par Sutton & Barto [2]
2. Sous certaines condition. Voir preuve de convergence
3. Version synchrone de l'algorithme : les *backup operation* sont effectué en parallèle

Algorithme | Itération de politique

$\forall\ \mathbf{s} \in \mathcal{S}$, fixer arbitrairement la politique courante π_k tel que $\pi_0(\mathbf{s}) = \mathbf{a} \in \mathcal{A}(\mathbf{s})$

Répéter (étape 1 et 2) avec $k = 1, 2, 3, \ldots$ **jusqu'à convergence** vers $V^\star$ et $\pi^\star$:

1. **Évaluer la politique courante**

   Répéter avec $i = 1, 2, 3, \ldots$ **jusqu'à convergence** vers V^π :

$$V^\pi_{i+1}(\mathbf{s}) \longleftarrow \sum_{\mathbf{s}' \in \mathcal{S}_{t+1}} p(\mathbf{s}'|\mathbf{s}, \pi_k(\mathbf{s})) \left[r(\mathbf{s}, \pi_k(\mathbf{s}), \mathbf{s}') + \gamma\, V^\pi_i(\mathbf{s}') \right] \qquad \forall\ \mathbf{s} \in \mathcal{S}$$

2. **Améliorer la politique courante**

$$\pi_{k+1}(\mathbf{s}) \longleftarrow \underset{\mathbf{a} \in \mathcal{A}(\mathbf{s})}{\arg\max} \sum_{\mathbf{s}' \in \mathcal{S}_{t+1}} p(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \left[r(\mathbf{s}, \mathbf{a}, \mathbf{s}') + \gamma\, V^\pi(\mathbf{s}') \right] \qquad \forall\ \mathbf{s} \in \mathcal{S},\ \mathbf{a} \in \mathcal{A}(\mathbf{s})$$

En pratique | Itération de politique

On **arrête prématurément** l'algorithme au lieu d'itérer à l'infinie comme dans *Itération de Q-valeur*.

Conséquence :

- L'algorithme **converge approximativement** : $V^\pi \xrightarrow{\approx} V^\star$ et $\pi \xrightarrow{\approx} \pi^\star$

Pseudocode | Itération de politique [1] (Version synchrone)

Init: **foreach** $s \in \mathcal{S}$ **do** $\texttt{V-table}[s] \leftarrow 0$ **and** $\pi\texttt{-table}[s] \leftarrow a \in \mathcal{A}(s)$ arbitrarily

Init: threshold $\texttt{hyperparam}(\epsilon) > 0$ « Critère d'arrêt »

repeat

1 % .... Étape d'évaluation de politique ....

repeat « sweep i »

$\Delta \leftarrow 0$

$\texttt{V-table}_{old} \leftarrow \texttt{V-table}$

foreach $s \in \mathcal{S}$ **do**

$\texttt{V-table}[s] \leftarrow \sum_{s' \in \mathcal{S}} p\left(s' | s, \ \pi\texttt{-table}[s]\right) \left[r(s,a,s') + \gamma \ \texttt{V-table}_{old}[s'] \right]$ « Backup operation »

$\Delta \leftarrow \max\left(\Delta, \left| \texttt{V-table}_{old}[s] - \texttt{V-table}[s] \right|\right)$

until $\Delta < \epsilon$

2 % .... Étape d'amélioration de politique ....

$\pi\texttt{-IsStable} \leftarrow$ ***true***

foreach $s \in \mathcal{S}$ **do**

$a_{old} \leftarrow \pi\texttt{-table}[s]$

$\pi\texttt{-table}[s] \leftarrow \arg\max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[r(s,a,s') + \gamma \texttt{V-table}[s'] \right]$ « Backup operation »

if $a_{old} \neq \pi\texttt{-table}[s]$ **then** $\pi\texttt{-IsStable} \leftarrow$ ***false***

until $\pi\texttt{-IsStable}$ « $V^{\pi} \xrightarrow{\approx} V^{\star}$ *et* $\pi \xrightarrow{\approx} \pi^{\star}$ »

Output: $\pi^{\star}(s) \approx \pi\texttt{-table}[s]$ « Produit une politique déterministe »

1. Pseudocode inspiré de « Reinforcement Learning : An introduction », chapitre 4.3, page 80 « Policy Iteration » par Sutton & Barto [2]

Comparaison

Pseudocode | Itération de Q-valeur

Init: **foreach** $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$ **do** `Q-table[s,a]` $\leftarrow 0$
Init: threshold `hyperparam(`ϵ`)` > 0

repeat
- $\Delta \leftarrow 0$
- `Q-table`$_{old}$ $\leftarrow$ `Q-table`
- **foreach** $s \in \mathcal{S}$ **and** $a \in \mathcal{A}(s)$ **do**
  - `Q-table[s,a]` $\leftarrow \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[r(s,a,s') + \gamma \max_{a' \in \mathcal{A}(s')} \text{Q-table}_{old}[s',a'] \right]$
  - $\Delta \leftarrow \max\left(\Delta, \left| \text{Q-table}_{old}[s,a] - \text{Q-table}[s,a] \right|\right)$

until $\Delta < \epsilon$

Output: $\pi^*(s) \approx \arg\max_{a \in \mathcal{A}(s)} \text{Q-table}[s,a]$

Pseudocode | Itération de politique

Init: **foreach** $s \in \mathcal{S}$ **do** `V-table[s]` $\leftarrow 0$ **and** π`-table[s]` $\leftarrow a \in \mathcal{A}(s)$ arbitrarily
Init: threshold `hyperparam(`ϵ`)` > 0

repeat

1 % .... Étape d'évaluation de politique ....
- **repeat**
  - $\Delta \leftarrow 0$
  - `V-table`$_{old}$ $\leftarrow$ `V-table`
  - **foreach** $s \in \mathcal{S}$ **do**
    - `V-table[s]` $\leftarrow \sum_{s' \in \mathcal{S}} p(s'|s, \pi\text{-table}[s]) \left[r(s,a,s') + \gamma \, \text{V-table}_{old}[s'] \right]$
    - $\Delta \leftarrow \max\left(\Delta, \left| \text{V-table}_{old}[s] - \text{V-table}[s] \right|\right)$
- **until** $\Delta < \epsilon$

2 % .... Étape d'amélioration de politique ....
- π`-IsStable` $\leftarrow$ ***true***
- **foreach** $s \in \mathcal{S}$ **do**
  - $a_{old} \leftarrow \pi$`-table[s]`
  - π`-table[s]` $\leftarrow \arg\max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[r(s,a,s') + \gamma \text{V-table}[s'] \right]$
  - **if** $a_{old} \neq \pi$`-table[s]` **then** π`-IsStable` $\leftarrow$ ***false***

until π`-IsStable`

Output: $\pi^*(s) \approx \pi$`-table[s]`

Comparaison

>_ Pseudocode | Itération de Q-valeur

Init: **foreach** $s \in \mathcal{S}, a \in \mathcal{A}(s)$ **do** $\texttt{Q-table}[s,a] \leftarrow 0$
Init: threshold $\texttt{hyperparam}(\epsilon) > 0$

repeat
  $\Delta \leftarrow 0$
  $\texttt{Q-table}_{old} \leftarrow \texttt{Q-table}$
  foreach $s \in \mathcal{S}$ **and** $a \in \mathcal{A}(s)$ **do**
    $\texttt{Q-table}[s,a] \leftarrow \sum_{s' \in \mathcal{S}} p(s'|s, a) \left[r(s,a,s') + \gamma \max_{a' \in \mathcal{A}(s')} \texttt{Q-table}_{old}[s',a'] \right]$
    $\Delta \leftarrow \max\left(\Delta, \left| \texttt{Q-table}_{old}[s,a] - \texttt{Q-table}[s,a] \right| \right)$
until $\Delta < \epsilon$

Output: $\pi^\star(s) \approx \arg\max_{a \in \mathcal{A}(s)} \texttt{Q-table}[s,a]$

>_ Pseudocode | Itération de politique

Init: **foreach** $s \in \mathcal{S}$ **do** $\texttt{V-table}[s] \leftarrow 0$ **and** $\pi\texttt{-table}[s] \leftarrow a \in \mathcal{A}(s)$ arbitrarily
Init: threshold $\texttt{hyperparam}(\epsilon) > 0$

repeat
1 % .... Étape d'évaluation de politique ....
  repeat
    $\Delta \leftarrow 0$
    $\texttt{V-table}_{old} \leftarrow \texttt{V-table}$
    foreach $s \in \mathcal{S}$ **do**
      $\texttt{V-table}[s] \leftarrow \sum_{s' \in \mathcal{S}} p(s'|s, \pi\texttt{-table}[s]) \left[r(s,a,s') + \gamma\, \texttt{V-table}_{old}[s'] \right]$
      $\Delta \leftarrow \max\left(\Delta, \left| \texttt{V-table}_{old}[s] - \texttt{V-table}[s] \right| \right)$
  until $\Delta < \epsilon$
2 % .... Étape d'amélioration de politique ....
  $\pi\texttt{-IsStable} \leftarrow$ *true*
  foreach $s \in \mathcal{S}$ **do**
    $a_{old} \leftarrow \pi\texttt{-table}[s]$
    $\pi\texttt{-table}[s] \leftarrow \arg\max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[r(s,a,s') + \gamma \texttt{V-table}[s'] \right]$
    if $a_{old} \neq \pi\texttt{-table}[s]$ **then** $\pi\texttt{-IsStable} \leftarrow$ *false*
until $\pi\texttt{-IsStable}$

Output: $\pi^\star(s) \approx \pi\texttt{-table}[s]$

Itération de valeur/politique

Exemple interactif

Évaluation de politique dans l'environnement GridWorld

▸ Exemple interactif

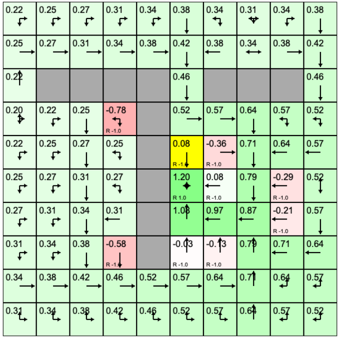


Image : REINFORCEjs, Stanford

- REINFORCEjs ▸ Projet ▸ GitHub

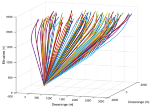
Apprentissage par renforcement (sans modèle)

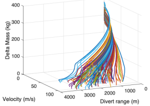
Apprentissage par renforcement (sans modèle)

*« Supposons qu'on souhaite concevoir une **fusée autonome** avec pour objectif d'attérir sur la planète Glises 581c[1] puis re-décoller, **sur quelle base** peut-on dévelloper les systèmes de **navigation, de guidage et de contrôle** ?*

*D'un point de vue appliqué, est-ce que la **garantie d'optimalité** et la **propriété d'exactitude** sont des **condition nécessaire ?** »*

Complément : Voir Deep Reinforcement Learning for Six Degree-of-Freedom Planetary Powered Descent and Landing, partie I. Introduction, pour une comparaison des méthodes de RL par rapport aux méthode traditionel par Contrôle Optimal dans le domaine de l'aérospatial. (Plus spécifiquement, dans le cadre du problème d'attérissage autonome à une coordoné précise sur Mars)


6-DOF Trajectories



6-DOF Fuel-Mass Along the Trajectories

1. « Gliese 581c . . . is classified as a super-Earth . . . there are no measurements of its radius. Furthermore, the radial velocity method used to detect it only puts a lower limit on the planet's mass, which means theoretical models of planetary radius and structure can only be of limited use . . . », wikipedia

Apprentissage par renforcement (sans modèle)

⚠ Problème : On n'a **pas de modèle** de l'environnement (Il est inconnu ou imparfait) $\Longrightarrow$ On ne peux pas prédire de comportement dans cette environnement ;

Concrètement : Pas de modèle $\Longrightarrow$ $p(\mathbf{s}'|\mathbf{s},\mathbf{a})$ et $r(\mathbf{s},\mathbf{a},\mathbf{s}')$ **inconnu** $\Longrightarrow$ **Impossible** de calculer une solution optimale **exacte** à priori ;

Solution : **Interagir** avec l'environnement (réèlement ou en simulateur) afin d'**apprendre** les composantes nécessaire pour trouver une solution optimale :

- $V^\pi(\mathbf{s})$, $Q^\pi(\mathbf{s},\mathbf{a})$, $\pi(\mathbf{a}|\mathbf{s})$ pour le RL **sans** model ;
- $p(\mathbf{s}'|\mathbf{s},\mathbf{a})$ et $V^\pi(\mathbf{s})$, $Q^\pi(\mathbf{s},\mathbf{a})$, $\pi(\mathbf{a}|\mathbf{s})$ pour le RL **avec** model (appris) ;

Apprentissage par renforcement (sans modèle)

RL par méthode Monte-Carlo [MC]

Méthodes Monté Carlo et processus de décision séquentielle

« Try many many time, learn from mistakes . . . once in a while. »

Caractéristiques :

- **Intéragit** avec l'environnement pour recuillir de l'information ;
- Collecte des **échantillons** de trajectoire τ **complètes** ;
- Phase d'apprentissage **hors-ligne** ;
- Apprentissage **sans-modèle** ;
- Pas de « Bootstrap » contrairement aux méthodes de *DP* ;
- N'exploite pas la *propriété de Markov* ;

Idées clés : Utiliser la **moyenne empirique** de plusieurs **trajectoires τ complètes** pour approximer l'espérance mathématique des *équations de Bellman* ;

Pourquoi ça marche ?

Méthodes Monté Carlo et processus de décision séquentielle

« Try many many time, learn from mistakes . . . once in a while. »

Caractéristiques :

- **Intéragit** avec l'environnement pour recuillir de l'information ;
- Collecte des **échantillons** de trajectoire τ **complètes** ;
- Phase d'apprentissage **hors-ligne** ;
- Apprentissage **sans-modèle** ;
- Pas de « Bootstrap » contrairement aux méthodes de *DP* ;
- N'exploite pas la *propriété de Markov* ;

Idées clés : Utiliser la **moyenne empirique** de plusieurs **trajectoires τ complètes** pour approximer l'espérance mathématique des *équations de Bellman* ;

Pourquoi ça marche ?

La loi des grands nombres

Estimé de Monté Carlo (formellement)

Soit

- $\mathcal{X}$ un espace d'évènements possible,
- x une réalisation de la variable aléatoire X,
- f_{X} la distribution probabilité **continue** suivie par la v.a. X,
- $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$ des échantillons tirés de f_{X}
- $g(\cdot)$ une fonction mesurable,

alors pour N suffisament grand

$$\mathbb{E}\left[\, g(\mathrm{X}) \,\right] = \int_{x \in \mathcal{X}} g(x)\, f_{\mathrm{X}}(x)\, dx \approx \frac{1}{N} \sum_{n=1}^{N} g(\, x^{(n)} \,) = \widehat{g}_N$$

avec $\widehat{g}_N$ la moyenne empirique de l'espérance mathématique $\mathbb{E}\left[\, g(\mathrm{X}) \,\right]$

Estimé de Monté Carlo (formellement)

Soit

- $\mathcal{X}$ un espace d'évènements possible,
- x une réalisation de la variable aléatoire X,
- $\mathbb{P}_{\mathrm{X}}$ la distribution probabilité **discrète** suivie par la v.a. X,
- $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$ des échantillons tirés de $\mathbb{P}_{\mathrm{X}}$
- $g(\cdot)$ une fonction mesurable,

alors pour N suffisament grand

$$\mathbb{E}\left[\, g(\mathrm{X}) \,\right] \quad = \quad \sum_{x \in \mathcal{X}} g(x)\, \mathbb{P}(\, \mathrm{X} = x \,) \quad \approx \quad \frac{1}{N} \sum_{n=1}^{N} g(\, x^{(n)} \,) \quad = \quad \widehat{g}_N$$

avec $\widehat{g}_N$ la moyenne empirique de l'espérance mathématique $\mathbb{E}\left[\, g(\mathrm{X}) \,\right]$

Estimé de Monté Carlo (considération pratique)

$$\widehat{g}_N = \frac{1}{N}\sum_{n=1}^{N} g(\,x^{(n)}\,)$$

Dans le contexte du RL :

$$Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow \frac{1}{k}\sum_{j=1}^{k} G_t^{(j)}(\tau)$$

est un « backup operation » qui **est répéter à chaque couple *état-action* rencontrer** dans une trajectoire échantillonné.

Estimé de Monté Carlo (considération pratique)

$$\widehat{g}_N = \frac{1}{N}\sum_{n=1}^{N} g(\, x^{(n)} \,)$$

Dans le contexte du RL :

$$Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow \frac{1}{k}\sum_{j=1}^{k} G_t^{(j)}(\tau)$$

est un « backup operation » qui **est répéter à chaque couple *état-action* rencontrer** dans une trajectoire échantillonné.

Estimé de Monté Carlo (considération pratique)

$$\widehat{g}_N = \frac{1}{N}\sum_{n=1}^{N} g(x^{(n)})$$

Dans le contexte du RL :

$$Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow \frac{1}{k}\sum_{j=1}^{k} G_t^{(j)}(\tau)$$

est un « backup operation » qui **est répéter à chaque couple *état-action* rencontrer** dans une trajectoire échantillonné.

Problème : Plus le nombre d'échantillon d'épisode k devient grand, plus ce calcule devient couteux en mémoire et en temps de calcule. **C'est ineficace** ⚠

Estimé de Monté Carlo (considération pratique)

$$\widehat{g}_N = \frac{1}{N}\sum_{n=1}^{N} g(x^{(n)})$$

Dans le contexte du RL :

$$Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow \frac{1}{k}\sum_{j=1}^{k} G_t^{(j)}(\tau)$$

est un « backup operation » qui **est répéter à chaque couple *état-action* rencontrer** dans une trajectoire échantillonné.

Problème : Plus le nombre d'échantillon d'épisode k devient grand, plus ce calcule devient couteux en mémoire et en temps de calcule. **C'est ineficace ⚠**

Solution : Implémentation incrémental du calcule de la moyenne

Estimé de Monté Carlo (considération pratique)

Implémentation incrémental du calcule de la moyenne :

$$
\begin{aligned}
Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) &= \frac{1}{k}\sum_{j=1}^{k} G_t^{(j)}(\tau) \\
&= \frac{1}{k}\left(G_t^{(k)}(\tau) + \sum_{j=1}^{k-1} G_t^{(j)}(\tau) \right) \\
&= \frac{1}{k}\left(G_t^{(k)}(\tau) + (k-1)\frac{1}{k-1}\sum_{j=1}^{k-1} G_t^{(j)}(\tau) \right) \\
&= \frac{1}{k}\left(G_t^{(k)}(\tau) + (k-1) Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) \right) \\
&= \frac{1}{k}\left(G_t^{(k)}(\tau) + k\, Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) - Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) \right) \\
&= Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) + \frac{1}{k}\left[G_t^{(k)}(\tau) - Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) \right]
\end{aligned}
$$

Estimé de Monté Carlo (considération pratique)

Implémentation incrémental du calcule de la moyenne :

$$
\begin{aligned}
Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) &= \frac{1}{k} \sum_{j=1}^{k} G_t^{(j)}(\tau) \\
&= \frac{1}{k} \left(G_t^{(k)}(\tau) + \sum_{j=1}^{k-1} G_t^{(j)}(\tau) \right) \\
&= \frac{1}{k} \left(G_t^{(k)}(\tau) + (k-1) \frac{1}{k-1} \sum_{j=1}^{k-1} G_t^{(j)}(\tau) \right) \\
&= \frac{1}{k} \left(G_t^{(k)}(\tau) + (k-1) Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) \right) \\
&= \frac{1}{k} \left(G_t^{(k)}(\tau) + k\, Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) - Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) \right) \\
&= Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) + \frac{1}{k} \left[G_t^{(k)}(\tau) - Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) \right]
\end{aligned}
$$

Estimé de Monté Carlo (considération pratique)

« Update rule », forme générale [1]

$$\text{Estimate}^{\text{New}} \longleftarrow \text{Estimate}^{\text{Old}} + \text{StepSize}\Big[\underbrace{\text{Target} - \text{Estimate}^{\text{Old}}}_{\text{estimate « error »}}\Big]$$

En pratique | Méthode Monté Carlo

Implémentation incrémental du calcule de la moyenne

$$Q^{\pi}_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t) + \frac{1}{k}\Big[G_t^{(k)}(\tau) - Q^{\pi}_{k}(\mathbf{s}_t, \mathbf{a}_t)\Big]$$

1. Voir explication dans « Reinforcement Learning : An introduction », chapitre 2, page 31 « Incremental Implementation » par Sutton & Barto [2]

Méthodes Monté Carlo et processus de décision séquentielle

Pro :

- Pas besoin de modèle ;
- Phase d'**apprentissage** exécuté par **« batch »** $\Longrightarrow$ est un avantage lorsque les ressources computationnel sont inférieur à la capacité à produire des nouvelles trajectoires ;
- **Aucun biais**[1] ;
- Les estimés de chaque état sont indépendant l'un de l'autre $\Longrightarrow$ il est possible de restreindre l'échantillonage à un sous-ensemble d'état d'intéret ;

Con :

- **Dépend de la capacité à générer beaucoup de trajectoires**
  $\Longrightarrow$ est un enjeux si la production de trajectoire est longue, couteuse ou risqué ;
- Fonctionne seulement sur les **environnements de type épisodique** ;
  (chaque trajectoire doit terminer)
- **Beaucoup de variance**[2] ;

1. car le retour $G_t(\tau) = \sum_{t'=t}^{T} \gamma^{t'-t} r(\mathrm{s}_{t'}, \mathrm{a}_{t'}, \mathrm{s}_{t'+1})$ est un estimateur non biaisé de $V^{\pi}(\mathrm{s}_t)$
2. car le retour $G_t(\tau)$ dépend d'une grande quantité de phénomène aléatoire (probabilité de sélection d'action, probabilité de transition, probabilité de récompense)

Méthodes Monté Carlo (Enjeux pratique)

Convergence vers $Q^\star$ et $\pi^\star$ garantie sous l'**assomption** que :

1. le processus d'*évaluation de politique* se fait sur une infinité d'épisode ;
2. chaque couple état-action a été visité un nombre infini de fois dans la limite d'une infinité d'épisode ;

En pratique | Méthode de Monté Carlo

Enjeux $\longrightarrow$ contourner ses deux assomptions d'une façon qui permet de...

1. relaxer le prérequis d'avoir à générer une infinité de trajectoire ;
2. adresser le **dilème exploration/exploitation** (assurer l'exploration suffisante de l'espace) ;

Apprentissage par renforcement (sans modèle)

Algorithme Monté Carlo avec départ exploratoire

Algorithme Monté Carlo avec départ exploratoire [*Monte Carlo ES*]

Idées clés :

1. Version *Monté Carlo* de l'algorithme *Itération de politique* ;
2. **Alterne épisode par épisode** entre *Évaluation et Amélioration de Politique* ;
3. À chaque épisode, considérer **exclusivement de la première visite** fait à un *état-action* pour les calcules (*Fisrt-visit MC*)[2] ;
4. **Choisir aléatoirement le point de départ** de chaque trajectoire échantillonné (départ exploratoire) ;

$$Q_k^{\pi} \xrightarrow{k \to \infty} Q_k^{\star} \quad \text{et} \quad \pi_k \xrightarrow{k \to \infty} \pi^{\star}$$

Pro :

- Adresse le *dilème d'exploration/exploitation* en utilisant l'idée de **départ exploratoire** ;
- Contourne le prérequis d'infinité d'épisode en acceptant le fait que chaque étape d'*évaluation de politique* provoque un **plus petit déplacement** $Q_k \to Q_k^{\pi}$;

Con :

- La preuve formelle de convergence de *Monte Carlo ES* est un problème ouvert.

2. Il existe également une version *Every-Visit MC* avec des propriété théorique légèrement différente

Algorithme | Monté Carlo avec *départ exploratoire*

$\forall\ \mathbf{s} \in \mathcal{S}$, fixer arbitrairement la politique courante π_k tel que $\pi_0(\mathbf{s}) = \mathbf{a} \in \mathcal{A}(\mathbf{s})$

Répéter (pour chaque trajectoire) avec $k = 1, 2, 3, \ldots$ **jusqu'à convergence** vers $Q^\star$ et $\pi^\star$:

- **Choisir** (le point de départ) $\mathbf{s}_0 \in \mathcal{S}$ et $\mathbf{a}_0 \in \mathcal{A}(\mathbf{s}_0)$ **aléatoirement** ;
- **Générer une trajectoire** en suivant $\pi_k(\cdot)$ jusqu'à terminaison ;

$$\tau = \mathbf{s}_0, \mathbf{a}_0, r_1, \mathbf{s}_1, \mathbf{a}_1, r_2, \ldots, \mathbf{s}_t, \mathbf{a}_t, r_{t+1}, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1}, r_T$$

Répéter (étape 1 à 3) pour chaque t de la trajectoire :

Seulement si le couple $(\mathbf{s}_t, \mathbf{a}_t)$ est une première visite (dans cette trajectoire) :

1. $\mathcal{G}(\mathbf{s}, \mathbf{a}) \leftarrow \mathcal{G}(\mathbf{s}, \mathbf{a}) \cup \{ G_t^{(k)} \}$
2. Évaluer la politique courante

$$Q^\pi_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow \frac{1}{|\mathcal{G}(\mathbf{s}, \mathbf{a})|} \sum_{G_t \in \mathcal{G}(\mathbf{s}, \mathbf{a})} G_t^{(\cdot)}$$

3. Améliorer la politique courante

$$\pi_{k+1}(\mathbf{s}_t) \longleftarrow \operatorname*{arg\,max}_{\mathbf{a} \in \mathcal{A}(\mathbf{s}_t)} Q^\pi_{k+1}(\mathbf{s}_t, \mathbf{a})$$

>_ Pseudocode | Monté Carlo avec *départ exploratoire* et *moyenne incrémentale*[1]

Init: foreach $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$ **do** `Q-table[s,a]` $\leftarrow 0$ **and** `K-table[s,a]` $\leftarrow 0$

Init: foreach $s \in \mathcal{S}$ **do** π`-table[s]` $\leftarrow a \in \mathcal{A}(s)$ arbitrarily

Init: threshold `hyperparam`$(\epsilon) > 0$, $\Delta \leftarrow 0$, π`-IsStable` $\leftarrow$ ***true*** « Critère d'arrêt »

repeat

$(s_0, a_0) \leftarrow$ `random.select(`$s \in \mathcal{S}$`)`, `random.select(`$a \in \mathcal{A}(s)$`)`

$\mathcal{T} \leftarrow$ **Generate trajectory** following π`-table[s]` from (s_0, a_0) to terminal state

$G \leftarrow 0$

foreach $t \in \mathcal{T} \mid t = T-1, T-2, \ldots, 2, 1, 0$ **do**

$G \leftarrow \gamma G + r_{t+1}$

if (s_t, a_t) *is* **not in** $\mathcal{T}[0, t-1]$ **then** it's first-visit

`Q-table`$_{old}$ $\leftarrow$ `Q-table` , $a_{old} \leftarrow \pi$`-table[`s_t`]` « pour la vérification de stabilité »

1 % .... Étape d'évaluation de politique ....

`K-table[`s_t, a_t`]` $\leftarrow$ `K-table[`s_t, a_t`]` $+1$

`Q-table[`s_t, a_t`]` $\longleftarrow$ `Q-table`$_{old}$`[`s_t, a_t`]` $+ 1/$`len(K-table[`s_t, a_t`])` $\Big[\, G_t - \gamma$ `Q-table`$_{old}$`[`s_t, a_t`]` $\Big]$ « Backup operation »

2 % .... Étape d'amélioration de politique ....

π`-table[`s_t`]` $\longleftarrow \arg\max_{a \in \mathcal{A}(s_t)}$ `Q-table[`s_t, a`]` « Backup operation »

3 % .... Étape de vérification de stabilité ....

$\Delta \leftarrow \max\Big(\Delta, \big|$ `Q-table`$_{old}$`[`s_t, a_t`]` $-$ `Q-table[`s_t, a_t`]` $\big|\Big)$

if $a_{old} \neq \pi$`-table[s]` **then** π`-IsStable` $\leftarrow$ ***false***

until $\Delta < \epsilon$ **and** π`-IsStable` for a arbitrary number of trajectories « $Q^\pi \xrightarrow{\approx} Q^\star$ *et* $\pi \xrightarrow{\approx} \pi^\star$ »

Output: $\pi^\star(s) \approx \pi$`-table[s]` « Produit une politique déterministe »

1. Pseudocode inspiré de « Reinforcement Learning : An introduction », chapitre 5.3, page 99 « Monte Carlo Control » par Sutton & Barto [2]

Apprentissage par renforcement (sans modèle)

RL par Différence temporelle [TD]

Méthodes par Différence Temporelle

« Try something & learn from mistake has you go. »

Caractéristiques :

- **Intéragit** avec l'environnement pour recuillir de l'information ;
- Collecte des **échantillons** de trajectoire τ **partiel** ;
- « Bootstrap » comme avec les méthodes de *DP* ;
- Phase d'apprentissage **en-ligne** ;
- Apprentissage **sans-modèle** ;
- Exploite la *propriété de Markov* ;

Idées clés : Apprendre **au fur et à mesure**, en se basant en partie sur des **échantillons** d'interaction avec l'environnement et en partie sur l'**estimation courante** (« Bootstrap ») ;

Méthodes par Différence Temporelle (considération pratique)

« Update rule », forme générale (rappel)

$$\text{Estimate}^{\text{New}} \longleftarrow \text{Estimate}^{\text{Old}} + \text{StepSize}\Big[\underbrace{\text{Target} - \text{Estimate}^{\text{Old}}}_{\text{estimate « error »}}\Big]$$

En pratique | Backup operation des méthodes par Différence Temporelle

Idées clés : Donner plus de poids aux expériences récentes ;

Comment ? Calculer la moyene pondéré des récompenses passés ; (aka : Exponential moving average)

Exemple (Algorithme SARSA) :

$$Q_{k+1}(s_t, a_t) \longleftarrow Q_k(s_t, a_t) + \alpha\Big[\underbrace{r(s_t, a_t, s_{t+1}) + Q_k(s_{t+1}, a_{t+1})}_{\text{TD target}} - Q_k(s_t, a_t)\Big]$$

avec le « learning rate » $\alpha \in]0, 1]$, l'échantillon $r(s_t, a_t, s_{t+1})$ et Q_k un estimer du vrai Q_k^π.

Méthodes par Différence Temporelle (considération pratique)

« Update rule », forme générale (rappel)

$$\text{Estimate}^{\text{New}} \longleftarrow \text{Estimate}^{\text{Old}} + \text{StepSize}\Big[\underbrace{\text{Target} - \text{Estimate}^{\text{Old}}}_{\text{estimate « error »}}\Big]$$

>_ En pratique | Backup operation des méthodes par Différence Temporelle

Idées clés : Donner plus de poids aux expériences récentes ;

Comment ? Calculer la moyene pondéré des récompenses passés ; (aka : Exponential moving average)

Exemple (Algorithme SARSA) :

$$Q_{k+1}(s_t, a_t) \longleftarrow Q_k(s_t, a_t) + \alpha\Big[\underbrace{r(s_t, a_t, s_{t+1}) + Q_k(s_{t+1}, a_{t+1})}_{\text{TD target}} - Q_k(s_t, a_t)\Big]$$

avec le « learning rate » $\alpha \in]0, 1]$, l'échantillon $r(s_t, a_t, s_{t+1})$ et Q_k un estimer du vrai Q_k^π.

Méthodes par Différence Temporelle (considération pratique)

« Update rule », forme générale (rappel)

$$\text{Estimate}^{\text{New}} \longleftarrow \text{Estimate}^{\text{Old}} + \text{StepSize}\Big[\underbrace{\text{Target} - \text{Estimate}^{\text{Old}}}_{\text{estimate « error »}}\Big]$$

En pratique | Backup operation des méthodes par Différence Temporelle

Idées clés : Donner plus de poids aux expériences récentes ;

Comment ? Calculer la moyene pondéré des récompenses passés ; (aka : Exponential moving average)

Exemple (Algorithme SARSA) :

$$Q_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow Q_k(\mathbf{s}_t, \mathbf{a}_t) + \alpha\Big[\underbrace{r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) + Q_k(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})}_{\text{TD target}} - Q_k(\mathbf{s}_t, \mathbf{a}_t)\Big]$$

avec le « learning rate » $\alpha \in]0, 1]$, l'échantillon $r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ et Q_k un estimer du vrai Q_k^π.

Méthodes par Différence Temporelle (considération pratique)

« Update rule », forme générale (rappel)

$$\text{Estimate}^{\text{New}} \longleftarrow \text{Estimate}^{\text{Old}} + \text{StepSize}\Big[\underbrace{\text{Target} - \text{Estimate}^{\text{Old}}}_{\text{estimate « error »}}\Big]$$

>_ En pratique | Backup operation des méthodes par Différence Temporelle

Idées clés : Donner plus de poids aux expériences récentes ;

Comment ? Calculer la moyene pondéré des récompenses passés ; (aka : Exponential moving average)

Exemple (Algorithme SARSA) :

$$Q_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \longleftarrow Q_k(\mathbf{s}_t, \mathbf{a}_t) + \alpha\Big[\underbrace{r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) + Q_k(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})}_{\text{TD target}} - Q_k(\mathbf{s}_t, \mathbf{a}_t)\Big]$$

avec le « learning rate » $\alpha \in]0, 1]$, l'échantillon $r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ et Q_k un estimer du vrai Q_k^{π}.

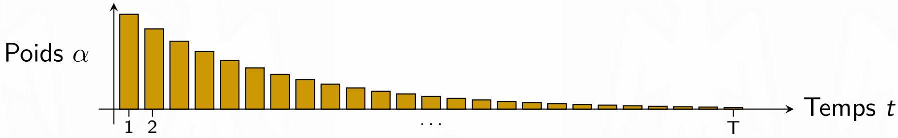
Méthodes par Différence Temporelle (considération pratique)

Learning rate $\alpha \in]0, 1]$

$$Q_{k+1}(\mathbf{s}_t, \mathbf{a}_t) \quad \longleftarrow \quad Q_k(\mathbf{s}_t, \mathbf{a}_t) + \alpha \Big[r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) + Q_k(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) - Q_k(\mathbf{s}_t, \mathbf{a}_t) \Big]$$

Exemple avec $\alpha = 0.97$: « L'agent tiens compte des expériences passée, comme si ça venait d'arrivé »

Exemple avec $\alpha = 0.5$: « L'agent est plus préoccupé par les expériences récentes »



Méthodes par Différence Temporelle

Pro :

- Pas besoin de modèle ;
- Peu apprendre **au fur et à mesure** $\Longrightarrow$ est un avantage lorsque les trajectoires sont longues à exécuter ;
- Peu apprendre **de trajectoire incomplète** $\Longrightarrow$ est un avantage lorsque on n'a pas de garantie que les trajectoires termine toujours ;
- **Peu de variance**[1] ;

Con :

- **Légèrement biaisé**[2] ;
- Les valeurs prisent lors de l'initialisation peuvent avoir une incidence sur le résultat de l'apprentissage ;

1. car le « TD-target » $r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) + \gamma Q(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})$ dépend de phénomène aléatoire sur un seul « step » (sélection d'action, transition, récompense)
2. car le « TD-target » $r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) + \gamma Q(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})$ est un estimé biaisé du vrai $Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t)$

Q-Learning (tabulaire) [*Tabular Q-Learning*]

« MOTO »

Idées clés :

1. C'est une approximation stochastique de l'algorithme *Itération de Q-valeur* ;

$$Q_i^\pi \xrightarrow{i \to \infty} Q^\star$$

Pro :

- Fusce condimentum mi et urna aliquam, non euismod quam fermentum. ;

Con :

- Proin egestas urna vulputate lorem pharetra, in posuere ipsum posuere ;

☐ ToDo: MOTO

☐ ToDo: Idées clées

☐ ToDo: Pro and Con

Algorithme | Q-Learning (tabulaire)

Répéter avec $i = 1, 2, 3, \dots$ **jusqu'à convergence** vers $Q^\star$:

$$Q^{\pi}_{i+1}(\mathbf{s}, \mathbf{a}) \longleftarrow \text{Fusce condimentum mi et urna aliquam, non euismod quam fermentum.}$$

En pratique | Q-Learning (tabulaire)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam in egestas magna, vel molestie mauris.

Conséquence :

- Sed vehicula felis at felis posuere efficitur quis sit amet arcu.

☐ ToDo: math algorithm

☐ ToDo: applied consideration

>_ Pseudocode | Q-Learning (tabulaire)[1]

Init: $\texttt{Q-table}[s, a] \leftarrow 0 \quad \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$

Init: threshold $\texttt{hyperparam}(\epsilon) > 0$ « Critère d'arrêt »

repeat « sweep i »

- $\Delta \leftarrow 0$
- **foreach** $s \in \mathcal{S}$ **and** $a \in \mathcal{A}(s)$ **do**
  - $Q_{old} \leftarrow \texttt{Q-table}[s, a]$
  - $\texttt{Q-table}[s, a] \leftarrow \sum_{s' \in \mathcal{S}} p(s'|s, a) \left[r(s, a, s') + \gamma \max_{a' \in \mathcal{A}(s')} \texttt{Q-table}[s', a'] \right]$ « Backup operation »
  - $\Delta \leftarrow \max \left(\Delta, \left| Q_{old} - \texttt{Q-table}[s, a] \right| \right)$

until $\Delta < \epsilon$ « $Q^{\pi} \xrightarrow{\approx} Q^{\star}$ »

Output: $\pi^{\star}(s) \approx \arg\max_{a \in \mathcal{A}(s)} \texttt{Q-table}[s, a]$ « Produit une politique déterministe »

☐ ToDo: PSEUDOCODE ☐ ToDo: RÉFÉRENCE

1. Pseudocode inspiré de « PAPERTITLE », section TODO, page TODO par TODO [**TODO**]

Prochaine partie

Prochaine partie

Deep Reinforcement Learning (DRL)

Apprentissage **sans** modèle

Apprentissage **avec** modèle

Basé sur les **valeurs**

Par recherche de **politique**

Modèle **connu** mais pas de solution analytique

Modèle **appris**
(Pas vue en cours)

DQN

Double DQN

Policy Gradient

Actor-Critic

Monte Carlo Tree Search

AlphaGo

Pour aller plus loin

Pour aller plus loin

Complément théorique

Pour aller plus loin

Ressource théorique

Programmation dynamique

- « Dynamic Programming and Optimal Control », Vol. 1 et 2, 4e Edition ▸ Publication
  par Dimitri P. Bertsekas

  Abstract : *. . . oriented towards modeling, conceptualization, finite-horizon problems, . . . mathematical analysis and computation, treats infinite horizon problems extensively, and provides an up-to-date account of approximate large-scale dynamic programming and reinforcement learning . . . ;*

- « Reinforcement Learning : An introduction », chapitre 4 [2] ▸ Publication
  par Sutton & Barto

  Algorithme : *Évaluation de politique, Amélioration de politique, Itération de V-valeur, Itération de politique et Itération de politique généralisé (GPI) ;*

Différence temporel [*Temporal Difference (TD)*]

- « The Paths Perspective on Value Learning », Distill, 2019 [3] ▸ www
  par Greydanus & Olah

  Algorithme : *Sarsa, Expected Sarsa, Q-learning* et *Double Q-learning ;*
  Abstract : *A closer look at how Temporal Difference learning merges paths of experience for greater statistical efficiency ;*

Pour aller plus loin

Framework et outil d'implémentation

MDP en pratique :

- Markov Decision Process (MDP) Toolbox for Python ▸ Doc

Pour aller plus loin

Références

References I

1. Barto, A. G. Reinforcement Learning and Dynamic Programming. *IFAC Proceedings Volumes* **28,** 407-412. ISSN : 14746670 (1995).
2. Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction.* 2e éd. (éd. MIT Press) ISBN : 978-0262039246. `http://incompleteideas.net/book/RLbook2018.pdf` (Cambridge, MA, 2018).
3. Greydanus, S. & Olah, C. The Paths Perspective on Value Learning. *Distill* **4.** `https://distill.pub/2019/paths-perspective-on-value-learning` (2019).