Fiche de cours : Fonction d'Approximation en Apprentissage par Renforcement

1 Contexte général

L'objectif d'une politique $\pi : S \to A$ est de maximiser la valeur

$$V(\pi) = \mathbb{E}_{s\sim\mu}[v^\pi(s)].$$

Dans les grands espaces d'états, les représentations tabulaires deviennent impossibles à stocker ou à estimer.

- 100 états pour une grille 10×10 ;
- 10^4 états pour une grille 100×100 ;
- $\sim 10^{20}$ états pour le Backgammon ;
- $\sim 10^{170}$ états pour le jeu de Go.

2 Problème des grands espaces

Les tables d'états deviennent inefficaces lorsque :

- la mémoire nécessaire explose ;
- la quantité de données requises pour remplir les tables devient irréaliste.

3 Fonction d'Approximation

Pour généraliser entre états similaires, on utilise une fonction paramétrée :

$$f_\theta(x),$$

où :

- x est un vecteur de caractéristiques (features) ;
- θ est un vecteur de paramètres appris.

Cela remplace les tables et permet de gérer des espaces continus.

3.1 Représentation par caractéristiques

Le vecteur $x(s)$ ou $x(s, a)$ représente l'état ou le couple état-action.
Exemples de caractéristiques :

- agrégation d'états ;
- **tile coding** (plusieurs grilles décalées) ;
- polynômes, RBF, bases de Fourier ;
- caractéristiques spécifiques à l'application (vision, robotique, etc.).

4 Politiques Paramétrées

4.1 Actions discrètes : politique Softmax

$$\pi_\theta(a \mid s) = \frac{e^{f_\theta(s,a)}}{\sum_{a'} e^{f_\theta(s,a')}}.$$

4.2 Actions continues : politique Gaussienne

$$\pi_\theta(a \mid s) = \mathcal{N}\Big(a; \mu_\theta(s), \sigma_\theta(s)^2\Big).$$

4.3 Déterministes vs stochastiques

Politiques déterministes :

- faciles à interpréter ;
- existence d'une politique optimale en MDP stationnaire.

Politiques stochastiques :

- robustes aux erreurs de modélisation ;
- intègrent naturellement l'exploration.

5 Approximation Linéaire

La forme générale est :

$$f_\theta(x) = \theta^\top x.$$

5.1 Cas importants

- **Lookup table** : $x(s)$ = indicatrice d'un état.
- **State aggregation** : regroupement d'états.
- **Tile coding** : plusieurs partitions décalées activées simultanément.

5.2 Limites

- faible pouvoir discriminant ;
- incapacité à modéliser des interactions complexes.

6 Approximation Non Linéaire : Réseaux de Neurones

6.1 Principe

Chaque neurone effectue :

$$z_i = \sigma(w_i^\top x).$$

Fonctions d'activation courantes :

- sigmoïde ;
- ReLU ;
- tanh ;
- leaky ReLU.

6.2 Universalité

Les réseaux sont des approximants universels :

- possibilité d'approcher toute fonction continue ;
- démonstration via construction de "pulses" ReLU.

6.3 Backpropagation

Algorithme en deux étapes :

1. propagation avant (forward) : calcul des activations ;
2. rétropropagation (backward) : application de la règle de dérivation en chaîne.

7 Limites des Réseaux Fully-Connected

- paramètres trop nombreux dans les entrées haute dimension (ex : image 1024×1024) ;
- inefficace pour exploiter la structure locale.

8 Convolutional Neural Networks (CNN)

8.1 Couches principales

- **convolution** : extraction locale ;
- **pooling** : sous-échantillonnage (max/average) ;
- **fully-connected** : en sortie.

8.2 Principes

- réduire progressivement la dimension des représentations ;
- n'utiliser des couches denses qu'en fin de réseau ;
- architectures nombreuses, mais la simplicité fonctionne souvent bien.

Résumé

- Les tables deviennent inutilisables dans les grands espaces.
- L'approximation de fonction permet de généraliser.
- Le choix des caractéristiques est crucial.
- Les modèles linéaires sont simples mais limités.
- Les réseaux de neurones sont puissants et universels.
- Les CNN exploitent la structure spatiale et permettent de passer à l'échelle.