Cours 4
Policy Gradients

I. Introduction

Méthodes Value-Based

Toutes les méthodes vues précédemment travaillaient sur des estimations de valeurs espérées selon la politique courante π.

$$V^{\pi}(s_t) = E_{\pi}[R_t | s_t = s]$$
$$Q^{\pi}(s, a) = E_{\pi}[R_t | s_t = s, a_t = a]$$

} Émettent des avis sur les actions possibles

Selon ces valeurs, on re-définit la politique π :

$$\pi(s) = \arg\max_{a' \in \mathcal{A}(s)} Q^{\pi}(s, a)$$ (sélection greedy)

Problèmes :

- avec des algorithmes tabulaires et des grandes cartes, il y a risque d'**explosion mémoire**
- ces méthodes sont sujettes à de grosses oscillations durant l'apprentissage : l'action préférée peut changer radicalement pour une modification mineure des valeurs

  ➔ problème de convergence
- on ne peut pas appliquer ces méthodes quand le monde est **continu**

Policy Gradient

Les méthodes Policy Gradients s'intéressent directement à la politique :

$$\pi_{\theta}(a|s) = P[a|s, \theta]$$

La probabilité d'une trajectoire τ est donc :

$$\pi_{\theta}(\tau) = P(s_1) \prod_{t=1}^{|\tau|-1} \pi_{\theta}(a_t|s_t) P(s_{t+1}|s_t, a_t)$$

La fonction à optimiser, $J(\theta)$, correspond à **l'espérance des rewards** (la somme des probabilités des trajectoires x les rewards correspondants) :

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=1}^{|\tau|-1} \mathcal{R}(s_t, a_t, s_{t+1}) \right] = \sum_{\tau} \pi_\theta(\tau) \mathcal{R}(\tau)$$

L'objectif est de **trouver une politique** π_θ qui génère des **trajectoires maximisant l'espérance des rewards** :

$$\theta^* = \arg\max_\theta J(\theta) = \arg\max_\theta \sum_{\tau} \pi_\theta(\tau) \mathcal{R}(\tau)$$

⊕ **Convergence** : les mises à jour sont plus "smooth"

⊕ Amélioration de la politique souvent **plus simple** que l'apprentissage des valeurs

⊕ Les méthodes Policy Gradient peuvent travailler avec un **nombre d'actions infini**

⊕ Possible intégration de récompenses d'exploration

Optimisation

Les méthodes Policy Gradient travaillent par montées de gradient successives :

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

Log-derivative trick : $\nabla_x f(x) = f(x) \frac{\nabla_x f(x)}{f(x)} = f(x) \nabla_x \log f(x)$

En appliquant le log-derivative trick sur $\nabla J(\theta)$, on obtient :

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\mathcal{R}(\tau) \nabla_\theta \log \pi_\theta(\tau) \right]$$

⊕ Passage à des log-vraisemblances de trajectoires

⊕ Le policy gradient s'exprime donc sous la forme d'une espérance : on peut donc **échantillonner des trajectoires pour l'approximer**.

On a alors à considérer $\nabla_\theta \log \pi_\theta(\tau)$ pour chaque trajectoire τ :

$$\begin{aligned}
\nabla_\theta \log \pi_\theta(\tau) &= \nabla_\theta \left[\log \left(P(s_1) \prod_{t=1}^{|\tau|-1} \pi_\theta(a_t|s_t) P(s_{t+1}|s_t, a_t) \right) \right] \\
&= \nabla_\theta \left[\log P(s_1) + \sum_{t=1}^{|\tau|-1} \log \pi_\theta(a_t|s_t) + \log P(s_{t+1}|s_t, a_t) \right] \\
&= \sum_{t=1}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t)
\end{aligned}$$

On obtient :

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)} \left[R(\tau) \sum_{t=1}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \right]$$

II. Algorithme REINFORCE

On a alors :

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[\mathcal{R}(\tau^{(i)}) \sum_{t=1}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^i|s_t^i) \right] = \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[\left(\sum_{t=1}^{|\tau^{(i)}|-1} r(s_t^i, a_t^i) \right) \left(\sum_{t=1}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^i|s_t^i) \right) \right]$$

Correspond à une log-vraisemblance : mesure la **vraisemblance de la trajectoire étant donné la politique courante**

↳ en le multipliant avec les rewards associés à la trajectoire, on **renforce la probabilité des trajectoires associées à de fortes récompenses positives** et, au contraire, on **diminue la vraisemblance d'une politique qui produit des récompenses fortement négatives**.

REINFORCE travaille par **échantillonnage de Monte-Carlo** (rollouts) pour mettre à jour les paramètres de la politique :

REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run the policy)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)\right)\left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i)\right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

⊖ REINFORCE dépend d'une suite de décisions prises

→ **très forte variance** (log probabilités + rewards cumulés)

→ Noisy gradient

→ apprentissage **instable**

→ distribution de probabilités déviant vers une direction non-optimale

⊖ Trajectoires avec somme des rewards cumulés nulle

→ apprentissage des "bonnes" et "mauvaises" actions **impossible**

→ **Convergence très lente**

Réduction de la variance : Causalité

Les futures actions n'affectent en rien les décisions prises dans le passé. Les actions prises au présent impactent seulement le futur → on ne regarde que les récompenses futures.

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[\left(\sum_{t'=t}^{|\tau^{(i)}|-1} r(s_{t'}^i, a_{t'}^i) \right) \left(\sum_{t=1}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^i|s_t^i) \right) \right]$$

Cumul des rewards ultérieurs à l'action a_t.

Réduction de la variance : Baseline

Au lieu d'utiliser $R_t(\tau)$, on lui soustrait une baseline $b(s_t)$, la moyenne des récompenses cumulées observées à partir de s_t :

$$b(s_t) = \frac{1}{M} \sum_{\tau} \mathcal{R}_t(\tau)$$

Intuition : stabiliser le processus en ne conservant que l'avantage tiré de l'action choisie.

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \sum_{t=1}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^i | s_t^i) \left(\mathcal{R}_t(\tau) - b(s_t) \right)$$

- on veut savoir si une action donne une meilleure récompense que la moyenne
- si les rewards sont toujours positifs, alors on renforce la probabilité d'une trajectoire même si le reward associé est bien inférieur que les autres.

↳ **Recalibrer les rewards par rapport à l'action moyenne**.

- reward total plus faible → gradient plus faible → mise à jour moins drastique et plus stable

Réduction de la variance : Discount

Intégrer un facteur de discount pour retirer de l'importance aux rewards trop lointains :

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[\left(\sum_{t=1}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^i | s_t^i) \right) \left(\sum_{t'=t}^{|\tau^{(i)}|-1} \gamma^{t'-t} r(s_{t'}^i, a_{t'}^i) \right) \right]$$

Vanilla REINFORCE

Algorithm 1 Vanilla Policy Gradient Algorithm

1: Input: initial policy parameters θ_0, initial value function parameters ϕ_0
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
4: Compute rewards-to-go $\hat{R}_t$.
5: Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.
6: Estimate policy gradient as

$$\hat{g}_k = \frac{1}{|\mathcal{D}_k|} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}_t.$$

7: Compute policy update, either using standard gradient ascent,

$$\theta_{k+1} = \theta_k + \alpha_k \hat{g}_k,$$

or via another gradient ascent algorithm like Adam.
8: Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_\phi \frac{1}{|\mathcal{D}_k| T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.
9: **end for**

VPG est on-policy : il explore en échantillonnant des actions selon la politique courante.

$$\mathop{\mathrm{arg\,max}}\limits_{\theta}$$

$$\nabla_{\theta} J(\theta) \approx \dfrac{1}{M} \displaystyle\sum_{\tau^{(i)} \sim \pi_{\theta}} \left[\mathcal{R}(\tau^{(i)}) \displaystyle\sum_{t=1}^{|\tau^{(i)}|-1} \nabla_{\theta} \log \pi_{\theta} (a_{t}^{i}| s_{t}^{i})\right] = \dfrac{1}{M} \displaystyle\sum_{\tau^{(i)} \sim \pi_{\theta}} \left[\left(\displaystyle\sum_{t=1}^{|\tau^{(i)}|-1} r(s_t^{i}, a_t^{i}) \right) \left(\displaystyle\sum_{t=1}^{|\tau^{(i)}|-1} \nabla_{\theta} \log \pi_{\theta} (a_{t}^{i}| s_{t}^{i}) \right) \right]$$

$$\nabla_{\theta} J(\theta) \approx \dfrac{1}{M} \displaystyle\sum_{\tau^{(i)} \sim \pi_{\theta}} \displaystyle\sum_{t=1}^{|\tau^{(i)}|-1} \nabla_{\theta} \log \pi_{\theta} (a_{t}^{i}| s_{t}^{i}) \left(Q(s_t^{i}, a_t^{i}) - V(s_t^{i})\right)$$