Apprentissage par renforcement

Cours 4: Policy Gradients

Sylvain Lamprier

UE RLD - Master DAC

2020

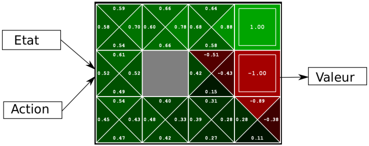
Méthodes Value-Based

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

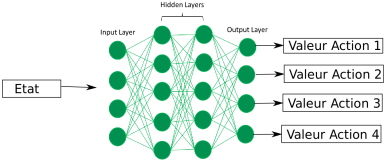
$$V^\pi(s_t) = E_\pi[R_t | s_t = s]$$

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

Tabular Q-learning



Deep Q-learning



Et une re-définition de la politique π selon ces valeurs (sélection greedy ici) :

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

Policy-Gradients

Les méthodes Policy-Gradients proposent de s'intéresser directement à la politique :

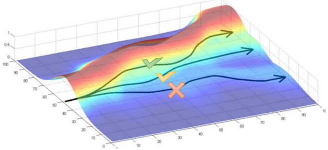
$$\pi_\theta(a|s) = P[a|s,\theta]$$

Dans ce cadre, la probabilité d'une trajectoire $\tau = (s_1, a_1, s_2, a_2, \ldots, s_{|\tau|})$ est donnée par :

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

L'objectif est de s'orienter plus probablement vers les trajectoires maximisant les récompenses :

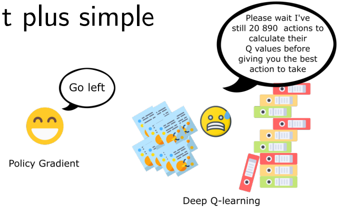
$$\begin{aligned} \theta^* &= \arg\max_\theta J(\theta) \\ &= \arg\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=1}^{|\tau|-1} \mathcal{R}(s_t, a_t, s_{t+1}) \right] \\ &= \arg\max_\theta \sum_\tau \pi_\theta(\tau) R(\tau) \end{aligned}$$



Policy-Gradients

Avantages des méthodes Policy-gradients :

- Convergence : Méthodes Value-based sujettes à de grosses oscillations durant l'apprentissage
  - L'action préférée peut changer radicalement pour une modification mineure des valeurs (même avec Bolzmann softmax selection)
  - Policy-gradients : mises à jour plus "smooth"
- Amélioration de la politique souvent plus simple que l'apprentissage des valeurs
- Policy gradients peuvent travailler avec un nombre d'actions infini



- Possible intégration de récompenses d'exploration

Policy-Gradients

Les méthodes Policy-gradients travaillent par montées de gradient successives :

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

Problème : Comment calculer le gradient de $J(\theta)$... ?

- ... qui correspond à une somme possiblement infinie sur l'ensemble des trajectoires ?
- ... dont la probabilité des longues trajectoires tend vers 0 (avec passage au log impossible directement du fait de la somme externe) ?

REINFORCE trick : Exploiter une propriété essentielle de la dérivée de la fonction log

$$\nabla_x f(x) = f(x) \frac{\nabla_x f(x)}{f(x)} = f(x) \nabla_x \log f(x)$$

Comment en tirer parti dans notre cas ?

Policy-Gradients

Log-derivative trick pour Policy gradients :

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \mathbb{E}_{\tau \sim \pi_\theta(\tau)}\left[R(\tau)\right] \\
&= \nabla_\theta \sum_\tau \pi_\theta(\tau) R(\tau) \\
&= \sum_\tau \nabla_\theta (\pi_\theta(\tau) R(\tau)) \\
&= \sum_\tau R(\tau) \nabla_\theta \pi_\theta(\tau) \\
&= \sum_\tau \pi_\theta(\tau) R(\tau) \nabla_\theta \log \pi_\theta(\tau) \\
&= \mathbb{E}_{\tau \sim \pi_\theta(\tau)}\left[R(\tau) \nabla_\theta \log \pi_\theta(\tau)\right]
\end{aligned}
$$

- Passage à des log-vraisemblances de trajectoires
- Possibilité d'échantillonner les trajectoires pour l'optimisation

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

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

- Somme de gradients de log-probabilités
- Plus de problème d'arrondis à 0

On a alors :

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

REINFORCE travaille par échantillonage de Monte-Carlo (Rollouts) :

REINFORCE algorithm:

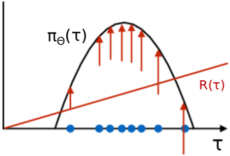
1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run the policy)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i) \right) \left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i) \right)$
3. $\theta \leftarrow \theta + \alpha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

Algorithme REINFORCE

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[R(\tau^{(i)}) \nabla_\theta \log \pi_\theta(\tau^{(i)}) \right]$$

Intuition :

- Renforcement de la probabilité des trajectoires associées à des fortes récompenses

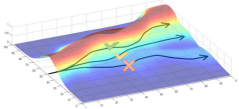


- Parallèle avec log-vraisemblance d'un ensemble de M trajectoires $\mathcal{T}$

$$\nabla_\theta L(\mathcal{T}; \theta) = \frac{1}{M} \sum_{\tau^{(i)} \in \mathcal{T}} \nabla_\theta \log \pi_\theta(\tau^{(i)})$$

Algorithme REINFORCE

Malheureusement l'algorithme REINFORCE souffre d'une très forte variance



- Pour un même couple état-action et une même politique, les sommes de rewards retournées peuvent être très différentes

$\Rightarrow$ Convergence très lente

Reduction de la variance

- Exploitation de la structure temporelle
- Introduction d'une Baseline
- Facteur de Discount

Réduction de la variance : Causalité

Causalité : Les décisions à t n'affectent en rien les récompenses obtenues à t', avec t'<t

$$\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\tau\sim\pi_\theta}\left[R(\tau)\nabla_\theta \log \pi_\theta(\tau)\right] \\
&= \sum_\tau \pi_\theta(\tau)\left[(\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t))(\sum_{t=0}^{|\tau|} r_t)\right] \\
&= \sum_\tau \pi_\theta(\tau)\left[\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\sum_{t'=t}^{|\tau|} r_{t'}\right] + \nabla_\theta C(\theta) \\
&= \sum_\tau \pi_\theta(\tau)\left[\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\sum_{t'=t}^{|\tau|} r_{t'}\right]
\end{aligned}$$

avec r_t le reward obtenu selon $\mathcal{R}(s_t, a_t, s_{t+1})$ dans τ, car :

$$\nabla_\theta C(\theta) = \sum_\tau \pi_\theta(\tau)\left[\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\sum_{t'=0}^{t-1} r_{t'}\right] = 0 \qquad (1)$$

Preuve ?

Réduction de la variance : Baseline

Introduction d'une baseline $b(s_t)$ pour réduire la variance

- Idée : retirer à $R_t(\tau)$ la moyenne des récompenses cumulées observées à partir de s_t
- Intuition : stabiliser le processus en ne conservant que l'avantage tiré de l'action choisie

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)} \left[\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t)(R_t(\tau) - b(s_t)) \right]$$

avec $b(s_t) = \frac{1}{N} \sum_\tau R_t(\tau)$ et $R_t(\tau) = \sum_{t'=t}^{|\tau|-1} r'_t$.
On a le droit de faire çà car $\forall t \in \{0..T-1\}$, $b(s_t)$ ne dépend pas de $\pi_\theta(a_t|s_t)$:

$$\mathbb{E}_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(a_t|s_t) b(s_t)] = \sum_{\tau_{0..t-1}} \pi_\theta(\tau_{0..t-1}) \sum_{s_t \in \mathcal{S}} P(s_t|s_{t-1}, a_{t-1}) b(s_t) \nabla_\theta \sum_{a_t \in \mathcal{A}(s_t)} \pi(a_t|s_t) = 0$$

On garde alors une estimation non-biaisée.

Réduction de la variance : Baseline

Baseline optimale ?

- Pour toute trajectoire τ et tout instant $t \in \{0, |\tau| - 1\}$, la meilleure baseline est celle qui minimise la variance de $\nabla_\theta \log \pi_\theta^t(\tau)(R_t(\tau) - b(s_t))$, avec $\pi_\theta^t(\tau) = \pi_\theta(\tau_{t..|\tau|}|\tau_{0..t-1})$

$$\frac{dVAR\left[\nabla_\theta \log \pi_\theta^t(\tau)(R_t(\tau) - b(s_t))\right]}{db} = \frac{d\mathbb{E}\left[(\nabla_\theta \log \pi_\theta^t(\tau)(R_t(\tau) - b(s_t)))^2\right]}{db}$$

$$\text{car : } \frac{d\mathbb{E}\left[(\nabla_\theta \log \pi_\theta^t(\tau)(R_t(\tau) - b(s_t)))\right]^2}{db} = 0 \text{ (estimateur sans biais)}$$

$$\begin{aligned}\frac{d\mathbb{E}\left[(\nabla_\theta \log \pi_\theta^t(\tau)(R_t(\tau) - b(s_t)))^2\right]}{db} &= 2b(s_t)\mathbb{E}\left[(\nabla_\theta \log \pi_\theta^t(\tau))^2\right] \\ &- 2\mathbb{E}\left[(\nabla_\theta \log \pi_\theta^t(\tau))^2 R_t(\tau)\right]\end{aligned}$$

$$\Rightarrow b(s_t) = \frac{\mathbb{E}\left[(\nabla_\theta \log \pi_\theta^t(\tau))^2 R_t(\tau)\right]}{\mathbb{E}[(\nabla_\theta \log \pi_\theta^t(\tau))^2]}$$ (=moyenne empirique pondérée par la magnitude des gradients)

- En pratique on utilise le plus souvent la moyenne empirique classique

Réduction de la variance : Discount

Jusqu'alors on a considéré $R(\tau) = \sum_{t=0}^{|\tau|-1} r_t$

Mais on peut aussi intégrer un facteur de discount comme dans les méthodes Value-based : $R(\tau) = \sum_{t=0}^{|\tau|-1} \gamma^t r_t$

On a alors :

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[\sum_{t=0}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^{(i)}|s_t^{(i)}) \gamma^t \left(\sum_{t'=t}^{|\tau^{(i)}|-1} \gamma^{t'-t} r_{t'} - b(s_t^{(i)}) \right) \right]$$

avec $b(s_t) = \frac{1}{M} \sum_{\tau^{(i)}} \sum_{t'=t}^{|\tau^{(i)}|-1} \gamma^{t'-t} r_{t'}$

Certaines approches retirent le facteur γ^t :

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{\tau^{(i)} \sim \pi_\theta} \left[\sum_{t=0}^{|\tau^{(i)}|-1} \nabla_\theta \log \pi_\theta(a_t^{(i)}|s_t^{(i)}) \left(\sum_{t'=t}^{|\tau^{(i)}|-1} \gamma^{t'-t} r_{t'} - b(s_t^{(i)}) \right) \right]$$

- c'est juste un scale qui ne change pas les rapports entre actions selon un état (du moins dans la version tabulaire)

Algorithme Vanilla REINFORCE

Algorithm 1 "Vanilla" policy gradient algorithm

Initialize policy parameter θ, baseline b
for iteration$=1, 2, \ldots$ **do**
  Collect a set of trajectories by executing the current policy
  At each timestep in each trajectory, compute
   the *return* $R_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}$, and
   the *advantage estimate* $\hat{A}_t = R_t - b(s_t)$.
  Re-fit the baseline, by minimizing $\|b(s_t) - R_t\|^2$,
   summed over all trajectories and timesteps.
  Update the policy, using a policy gradient estimate $\hat{g}$,
   which is a sum of terms $\nabla_\theta \log \pi(a_t \mid s_t, \theta) \hat{A}_t$
end for

Version deep :

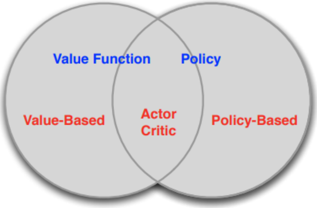
- $b(s_t) = V_\phi(s_t)$, avec V_ϕ un réseau de neurones
- Descente de gradient plutôt que minimisation à chaque itération :

$$\phi \leftarrow \phi + \epsilon \sum_{\tau^{(i)}} \sum_{t=0}^{|\tau^{(i)}|-1} (R_t^{(i)} - V_\phi(s_t^{(i)})) \nabla_\phi V_\phi(s_t^{(i)})$$

Actor-Critic

Les méthodes Actor-Critic sont à la jonction des méthodes

- Policy-Based (Actor) : apprennent à prendre des décisions
- Value-Based (Critic) : émettent des avis sur les possibles décisions



Actor $= \pi$
Critic = récompenses estimées - baseline

Actor-Critic

Méthodes Policy Gradient (sans baseline) :

$$\nabla_\theta J(\theta) = \sum_\tau \pi_\theta(\tau) \left[\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t}^{|\tau|} \gamma^{t'-t} r_{t'} \right]$$

Méthodes Actor-Critic (sans baseline) :

$$\nabla_\theta J(\theta) = \sum_\tau \pi_\theta(\tau) \left[\sum_{t=0}^{|\tau|-1} \nabla_\theta \log \pi_\theta(a_t|s_t) Q^\pi(s_t, a_t) \right]$$

Ces deux définitions du gradient sont équivalentes (Preuve)

Forme générale des Policy Gradients [Sch+15] :

Policy gradient methods maximize the expected total reward by repeatedly estimating the gradient $g := \nabla_\theta \mathbb{E}\left[\sum_{t=0}^{\infty} r_t\right]$. There are several different related expressions for the policy gradient, which have the form

$$g = \mathbb{E}\left[\sum_{t=0}^{\infty} \Psi_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right], \tag{1}$$

where Ψ_t may be one of the following:

1. $\sum_{t=0}^{\infty} r_t$: total reward of the trajectory.
2. $\sum_{t'=t}^{\infty} r_{t'}$: reward following action a_t.
3. $\sum_{t'=t}^{\infty} r_{t'} - b(s_t)$: baselined version of previous formula.
4. $Q^\pi(s_t, a_t)$: state-action value function.
5. $A^\pi(s_t, a_t)$: advantage function.
6. $r_t + V^\pi(s_{t+1}) - V^\pi(s_t)$: TD residual.

The latter formulas use the definitions

$$V^\pi(s_t) := \mathbb{E}_{\substack{s_{t+1:\infty},\\ a_{t:\infty}}}\left[\sum_{l=0}^{\infty} r_{t+l}\right] \qquad Q^\pi(s_t, a_t) := \mathbb{E}_{\substack{s_{t+1:\infty},\\ a_{t+1:\infty}}}\left[\sum_{l=0}^{\infty} r_{t+l}\right] \tag{2}$$

$$A^\pi(s_t, a_t) := Q^\pi(s_t, a_t) - V^\pi(s_t), \quad \text{(Advantage function).} \tag{3}$$

Actor-Critic

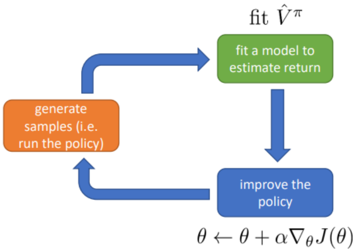
Fonction d'avantage : $A(s,a) = Q(s,a) - V(s)$

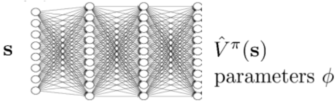
batch actor-critic algorithm:

1. sample $\{\mathbf{s}_i, \mathbf{a}_i\}$ from $\pi_\theta(\mathbf{a}|\mathbf{s})$ (run it on the robot)
2. fit $\hat{V}^\pi_\phi(\mathbf{s})$ to sampled reward sums
3. evaluate $\hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i) = r(\mathbf{s}_i, \mathbf{a}_i) + \hat{V}^\pi_\phi(\mathbf{s}'_i) - \hat{V}^\pi_\phi(\mathbf{s}_i)$
4. $\nabla_\theta J(\theta) \approx \sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}_i) \hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i)$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

$$y_{i,t} \approx r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) + \hat{V}^\pi_\phi(\mathbf{s}_{i,t+1})$$

$$\mathcal{L}(\phi) = \frac{1}{2} \sum_i \left\| \hat{V}^\pi_\phi(\mathbf{s}_i) - y_i \right\|^2$$





Actor-Critic

batch actor-critic algorithm:

1. sample $\{\mathbf{s}_i, \mathbf{a}_i\}$ from $\pi_\theta(\mathbf{a}|\mathbf{s})$ (run it on the robot)
2. fit $\hat{V}^\pi_\phi(\mathbf{s})$ to sampled reward sums
3. evaluate $\hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i) = r(\mathbf{s}_i, \mathbf{a}_i) + \gamma \hat{V}^\pi_\phi(\mathbf{s}'_i) - \hat{V}^\pi_\phi(\mathbf{s}_i)$
4. $\nabla_\theta J(\theta) \approx \sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}_i) \hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i)$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

online actor-critic algorithm:

1. take action $\mathbf{a} \sim \pi_\theta(\mathbf{a}|\mathbf{s})$, get $(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)$
2. update $\hat{V}^\pi_\phi$ using target $r + \gamma \hat{V}^\pi_\phi(\mathbf{s}')$
3. evaluate $\hat{A}^\pi(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \hat{V}^\pi_\phi(\mathbf{s}') - \hat{V}^\pi_\phi(\mathbf{s})$
4. $\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) \hat{A}^\pi(\mathbf{s}, \mathbf{a})$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

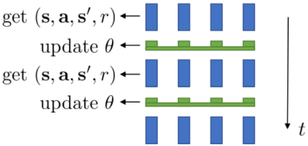
Online Actor-Critic

online actor-critic algorithm:

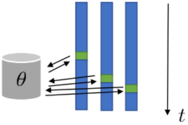
1. take action $\mathbf{a} \sim \pi_\theta(\mathbf{a}|\mathbf{s})$, get $(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)$
2. update $\hat{V}^\pi_\phi$ using target $r + \gamma \hat{V}^\pi_\phi(\mathbf{s}')$
3. evaluate $\hat{A}^\pi(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \hat{V}^\pi_\phi(\mathbf{s}') - \hat{V}^\pi_\phi(\mathbf{s})$
4. $\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) \hat{A}^\pi(\mathbf{s}, \mathbf{a})$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

works best with a batch (e.g., parallel workers)

synchronized parallel actor-critic



asynchronous parallel actor-critic



Asynchronous Advantage Actor-Critic (A3C)

Algorithm S3 Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

// Assume global shared parameter vectors θ and θ_v and global shared counter $T = 0$
// Assume thread-specific parameter vectors θ' and θ'_v
Initialize thread step counter $t \leftarrow 1$
repeat
  Reset gradients: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.
  Synchronize thread-specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$
  $t_{start} = t$
  Get state s_t
  repeat
    Perform a_t according to policy $\pi(a_t|s_t;\theta')$
    Receive reward r_t and new state s_{t+1}
    $t \leftarrow t + 1$
    $T \leftarrow T + 1$
  until terminal s_t **or** $t - t_{start} == t_{max}$
  $R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{for non-terminal } s_t \text{// Bootstrap from last state} \end{cases}$
  for $i \in \{t-1, \ldots, t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$
    Accumulate gradients wrt θ': $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i;\theta')(R - V(s_i;\theta'_v))$
    Accumulate gradients wrt θ'_v: $d\theta_v \leftarrow d\theta_v + \partial\left(R - V(s_i;\theta'_v)\right)^2 / \partial\theta'_v$
  end for
  Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
until $T > T_{max}$

(source : [Mni+16])

Actor-Critic

Actor-critic: $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \left(r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) + \gamma \hat{V}_\phi^\pi(\mathbf{s}_{i,t+1}) - \hat{V}_\phi^\pi(\mathbf{s}_{i,t})\right)$

+ lower variance (due to critic)

- not unbiased (if the critic is not perfect)

Policy gradient: $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \left(\left(\sum_{t'=t}^{T} \gamma^{t'-t} r(\mathbf{s}_{i,t'}, \mathbf{a}_{i,t'})\right) - b\right)$

+ no bias

- higher variance (because single-sample estimate)

can we use $\hat{V}_\phi^\pi$ and still keep the estimator unbiased?

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \left(\left(\sum_{t'=t}^{T} \gamma^{t'-t} r(\mathbf{s}_{i,t'}, \mathbf{a}_{i,t'})\right) - \hat{V}_\phi^\pi(\mathbf{s}_{i,t})\right)$$

+ no bias

+ lower variance (baseline is closer to rewards)

Actor-Critic

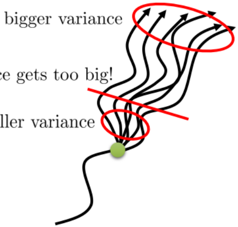
$$\hat{A}^{\pi}_{\mathrm{C}}(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \hat{V}^{\pi}_{\phi}(\mathbf{s}_{t+1}) - \hat{V}^{\pi}_{\phi}(\mathbf{s}_t)$$

\+ lower variance
\- higher bias if value is wrong (it always is)

$$\hat{A}^{\pi}_{\mathrm{MC}}(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{\infty} \gamma^{t'-t} r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) - \hat{V}^{\pi}_{\phi}(\mathbf{s}_t)$$

\+ no bias
\- higher variance (because single-sample estimate)

Can we combine these two, to control bias/variance tradeoff?



$$\hat{A}^{\pi}_{n}(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{t+n} \gamma^{t'-t} r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) - \hat{V}^{\pi}_{\phi}(\mathbf{s}_t) + \gamma^{n} \hat{V}^{\pi}_{\phi}(\mathbf{s}_{t+n})$$

choosing $n > 1$ often works better!

Generalized Actor-Critic [Sch+15]

$$
\begin{aligned}
\hat{A}_t^{(1)} &:= \delta_t^V &&= -V(s_t) + r_t + \gamma V(s_{t+1}) \\
\hat{A}_t^{(2)} &:= \delta_t^V + \gamma \delta_{t+1}^V &&= -V(s_t) + r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) \\
\hat{A}_t^{(3)} &:= \delta_t^V + \gamma \delta_{t+1}^V + \gamma^2 \delta_{t+2}^V &&= -V(s_t) + r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 V(s_{t+3})
\end{aligned}
$$

$$
\hat{A}_t^{(k)} := \sum_{l=0}^{k-1} \gamma^l \delta_{t+l}^V = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k})
$$

Comment choisir k ?

$$
\begin{aligned}
\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} &:= (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda \hat{A}_t^{(2)} + \lambda^2 \hat{A}_t^{(3)} + \ldots\right) \\
&= (1-\lambda)\left(\delta_t^V + \lambda\left(\delta_t^V + \gamma\delta_{t+1}^V\right) + \lambda^2\left(\delta_t^V + \gamma\delta_{t+1}^V + \gamma^2\delta_{t+2}^V\right) + \ldots\right) \\
&= (1-\lambda)\left(\delta_t^V\left(1 + \lambda + \lambda^2 + \ldots\right) + \gamma\delta_{t+1}^V\left(\lambda + \lambda^2 + \lambda^3 + \ldots\right)\right. \\
&\qquad \left. + \gamma^2\delta_{t+2}^V\left(\lambda^2 + \lambda^3 + \lambda^4 + \ldots\right) + \ldots\right) \\
&= (1-\lambda)\left(\delta_t^V\left(\frac{1}{1-\lambda}\right) + \gamma\delta_{t+1}^V\left(\frac{\lambda}{1-\lambda}\right) + \gamma^2\delta_{t+2}^V\left(\frac{\lambda^2}{1-\lambda}\right) + \ldots\right) \\
&= \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V
\end{aligned}
$$

Generalized Actor-Critic [Sch+15]

$$\begin{aligned}
\hat{A}_t^{(1)} &:= \delta_t^V &&= -V(s_t) + r_t + \gamma V(s_{t+1}) \\
\hat{A}_t^{(2)} &:= \delta_t^V + \gamma\delta_{t+1}^V &&= -V(s_t) + r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) \\
\hat{A}_t^{(3)} &:= \delta_t^V + \gamma\delta_{t+1}^V + \gamma^2\delta_{t+2}^V &&= -V(s_t) + r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 V(s_{t+3})
\end{aligned}$$

$$\hat{A}_t^{(k)} := \sum_{l=0}^{k-1} \gamma^l \delta_{t+l}^V = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k})$$

Comment choisir k ?

$$\hat{A}_t^{GAE(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V$$

- Similaire à TD(λ)
- λ est un hyper-paramètre à régler
- Décroissance exponentielle du poids des δ^V

$\Rightarrow$ $\lambda = 1$: $\hat{A}_t^{GAE(\gamma,\lambda)} = \hat{A}_t^{\infty}$ (Monte-Carlo)

$\Rightarrow$ $\lambda = 0$: $\hat{A}_t^{GAE(\gamma,\lambda)} = \hat{A}_t^{1}$ (TD(0))

Traces d'éligibilité

$$\theta \leftarrow \theta + \alpha \frac{1}{M} \sum_{\tau^{(i)}} \sum_{t=0}^{|\tau^{(i)}|-1} \hat{A}_t^{GAE(\gamma,\lambda)} \nabla_\theta \pi_\theta(a_t|s_t)$$

Comme pour $TD(\lambda)$, on peut définir des traces d'éligibilité pour faire les mises à jour de θ au fur et à mesure du processus :

$$e_0 \leftarrow 0$$

$$e_t \leftarrow \lambda\gamma e_{t-1} + \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Dont on peut se servir pour pondérer le passé et faire des mises à jour à chaque étape de la trajectoire :

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

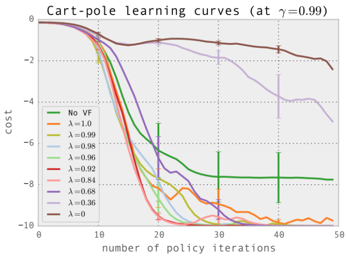
$$\theta \leftarrow \theta + \alpha \delta_t e_t$$

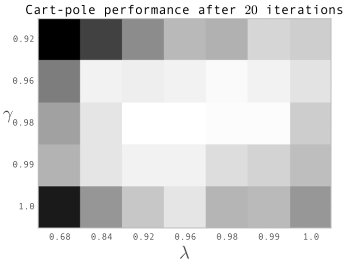
Possible de faire la même chose pour ϕ :

$$\phi \leftarrow \phi + \alpha \frac{1}{M} \sum_{\tau^{(i)}} \sum_{t=0}^{|\tau^{(i)}|-1} \nabla_\phi (\hat{A}_t^{GAE(\gamma,\lambda)})^2$$

Generalized Actor-Critic

Performances pour différents λ sur Cartpole :





- NoVF correspond à un MonteCarlo avec une baseline "moyenne glissante" ne dépendant pas de l'état courant
- Pour les autres $\hat{A}_t^{GAE(\gamma,\lambda)}$ avec Value Function apprise selon TD(0)

$\Rightarrow$ Variance augmente lorsque λ augmente

$\Rightarrow$ Biais augmente lorsque λ diminue

Actor Critic avec critique approximée

Un certain nombre d'approches proposent d'utiliser une approximation de la critique plutôt que des estimateurs de Monte-Carlo ou Temporal Difference.

Plutôt que de considérer le gradient :

$$\nabla_\theta J(\theta) = \sum_\tau \pi_\theta(\tau) \left[\sum_{t=0}^{|\tau|-1} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) Q^\pi(s_t, a_t) \right]$$

Cela revient à utiliser le gradient :

$$\hat{g} = \sum_\tau \pi_\theta(\tau) \left[\sum_{t=0}^{|\tau|-1} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) Q^\pi_\phi(s_t, a_t) \right]$$

- Découplage complet de l'acteur et de la critique

$\Rightarrow$ Objectifs :

  - Utilisation de la structure de l'espace d'états
  - Réduction de la variance

Fonctions Compatibles

Soit le gradient : $\hat{g} = \sum_{\tau} \pi_\theta(\tau) \left[\sum_{t=0}^{|\tau|-1} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) f_\phi(s_t, a_t) \right]$, avec f_ϕ une fonction $\mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ de paramètres ϕ.
Le seul moyen de rendre ce gradient non biaisé (i.e., $\hat{g} = \nabla_\theta J(\theta)$) est d'utiliser une fonction f_ϕ compatible. Deux conditions à cela [Sut+00] :

- Pour tout s et a : $\nabla_\phi f_\phi(s, a) = \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)}$
- $\sum_{\tau} \pi_\theta(\tau) \left[\sum_{t=0}^{|\tau|-1} \gamma^t (Q^\pi(s_t, a_t) - f_\phi(s_t, a_t) - v_w(s_t)) \nabla_\phi f_\phi(s_t, a_t) \right] = 0$

Preuve

avec $v_w(s)$ une fonction quelconque $\mathcal{S} \rightarrow \mathbb{R}$ de paramètres w.
Soit π la fonction softmax : $\pi_\theta(a|s) = \frac{e^{h_\theta(s,a)}}{\sum_{a' \in \mathcal{A}(s)} e^{h_\theta(s,a')}}$, avec $h_\theta : S \times A \rightarrow \mathbb{R}$

Selon la 1ière condition, on a :

$f_\phi(s, a) = \left[\nabla_\theta h_\theta(s, a) - \sum_{a' \in \mathcal{A}(s)} \pi_\theta(a'|s) \nabla_\theta h_\theta(s, a') \right]^T \phi$

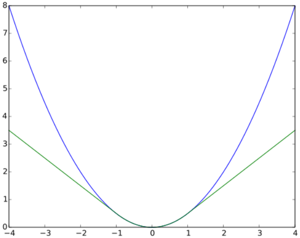
Notons que : $\sum_{a \in \mathcal{A}(s)} \pi(a|s) f_\phi(s, a) = 0$. On peut alors voir f_w comme une fonction d'avantage : $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$.

⇒ Pour l'estimation de w, la fonction v_w qui mininime la variance de l'estimation de f_ϕ est [Bha+09] : $V^\pi(s)$

⇒ Estimation de $V^\pi(s)$ selon $v_w(s)$ et de $Q^\pi(s, a)$ selon $f_\phi(s, a) + v_w(s)$ par temporal difference

Prévenir l'explosion des gradients

Huber Loss plutôt que Quadratic Loss pour l'apprentissage de V :



$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \le \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$$

Exploration : Entropie

Fréquemment, la politique converge trop vite vers des situations sous-optimales

- $\pi(s_t)$ tend vers une politique déterministe rapidement

$\Rightarrow$ Plus d'exploration, boucles infinies possibles

Possibilité de rajouter un coût d'entropie permettant de maintenir l'exploration tant qu'il reste de l'incertitude :

$$\Delta\theta = \alpha \sum_{t=0}^{T} [\nabla_\theta \log \pi_\theta(\mathbf{a}_t|s_t)(R_t - b_t(s_t)) + \beta \nabla_\theta H_\theta(s_t)]$$

$$H_\theta(s_t) := - \sum_{\mathbf{a} \in \mathcal{A}} \pi_\theta(\mathbf{a}|s_t) \log \pi_\theta(\mathbf{a}|s_t)$$

Sources

- Sergey Levine (UC Berkeley, Spring 2017)
- Daniel Takeshi : https://danieltakeshi.github.io/2017/03/28/going-deeper-into-reinforcement-learning-fundamentals-o
- Jonathan Hui : https://medium.com/@jonathan_hui/rl-deep-reinforcement-learning-series-833319a95530
- Lilian Weng : https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html
- Felix Yu : https ://flyyufelix.github.io/2017/10/12/dqn-vs-pg.html
- Joshua Achiam : http://rail.eecs.berkeley.edu/deeprlcourse-fa17/f17docs/lecture_13_advanced_pg.pdf

References I

[Bha+09] Shalabh Bhatnagar et al. « Natural actor–critic algorithms ». In : *Automatica* 45.11 (2009), p. 2471-2482.

[Gro+12] Ivo Grondman et al. « A survey of actor-critic reinforcement learning : Standard and natural policy gradients ». In : *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.6 (2012), p. 1291-1307.

[Mni+16] Volodymyr Mnih et al. « Asynchronous methods for deep reinforcement learning ». In : *International conference on machine learning*. 2016, p. 1928-1937.

[Sch+15] John Schulman et al. « High-dimensional continuous control using generalized advantage estimation ». In : *arXiv preprint arXiv :1506.02438* (2015).

[Sut+00] Richard S Sutton et al. « Policy gradient methods for reinforcement learning with function approximation ». In : *Advances in neural information processing systems*. 2000, p. 1057-1063.