

Comparaison de méthodes évolutionnaires et d'apprentissage par renforcement sur des benchmarks de contrôle classique

Un projet ANDROIDE encadré par Olivier Sigaud

Damien Legros et Hector Kohler

Master ANDROIDE
Université - Sorbonne Université

May 26, 2021

Objectif du projet

Mieux comprendre les différences entre la recherche directe de politiques et la recherche basée sur un gradient, à travers l'étude des performances de CEM et PG sur les environnements Pendulum et CartPoleContinuous.

Présentation des méthodes étudiées (CEM et PG)

Algorithm 1: Cross-Entropy Method

Paramètres :

θ_i : ensemble des paramètres de la solution i

J_i : fitness de la solution i

$(\theta_i, J_i)_{i=0\dots N}$: ensemble de N paires des valeurs des solutions

ρ : proportion des meilleures solutions à utiliser par itération

σ_{bruit}^2 : terme de bruit additionnel

for iteration=1,2,... **do**

Génère N paires (θ_i, J_i) selon une loi normale $\mathcal{N}(\mu, \sigma^2 + \sigma_{bruit}^2)$

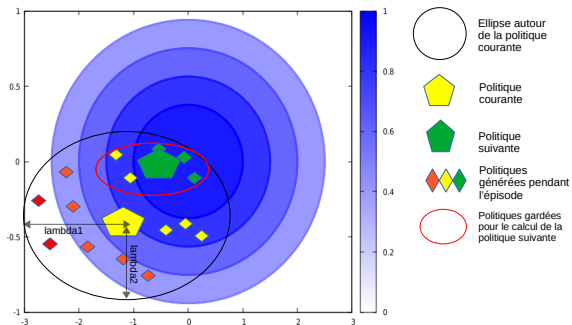
Calcule l'ensemble S_ρ des meilleures solutions de $\max(1, N \times \rho)$

$\mu \leftarrow \text{moyenne}(S_\rho)$

$\sigma^2 \leftarrow \text{écart} - \text{type}(S_\rho) + \sigma_{bruit}^2$

end for

Recherche directe : Cross-Entropy Method (CEM)



Présentation des méthodes étudiées (CEM et PG)

Algorithm 2: Policy Gradient

Initialiser les paramètres de la politique θ

for iteration=1,2,... **do**

Collecter l'ensemble des trajectoires $(s_t^{(i)}, a_t^{(i)}, r_t^{(i)})$, $i \in 1, H$ en exécutant la politique courante.

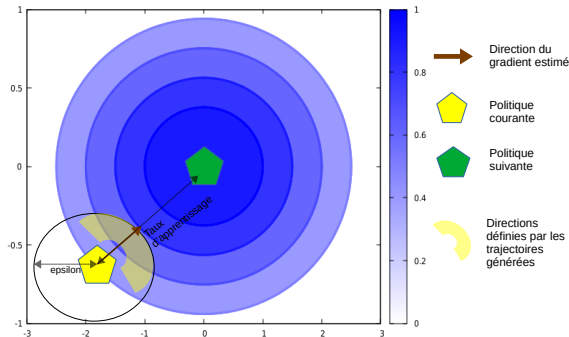
A chaque pas pour chaque trajectoire, calculer la somme des récompenses

$$R_t = \sum_{t'=t}^{T-1} r_{t'}$$

Mettre à jour la politique, à l'aide de l'estimation du gradient de la politique \hat{g} , la moyenne des termes $\nabla_{\theta} \log \pi(a_t | s_t, \theta) R_t$.

end for

Recherche basée sur un gradient : Policy Gradient (PG)



Méthode

- Modification d'une librairie existante pour la compatibilité avec CEM:
 - Ajout de fonctions de manipulation de poids compatibles avec PyTorch.
- Ajout de visualisations des déplacements pour la recherche dans l'espace des politiques :
 - Distance entre deux politiques, produit scalaire entre deux directions de recherche, normes de la covariance utilisée par CEM.
- Modification de la librairie pour la compatibilité avec l'outil Vignette (visualisation du paysage de valeur dans l'espace des politiques).

Structure d'une politique basée sur Chou2017 [CMS]

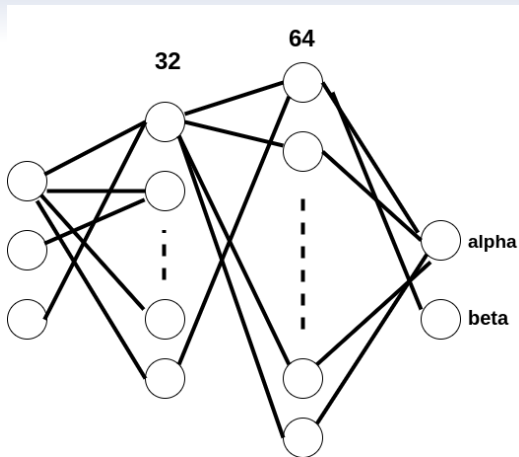


Figure: Réseau à deux couches cachées inspiré de Chou2017 [CMS]

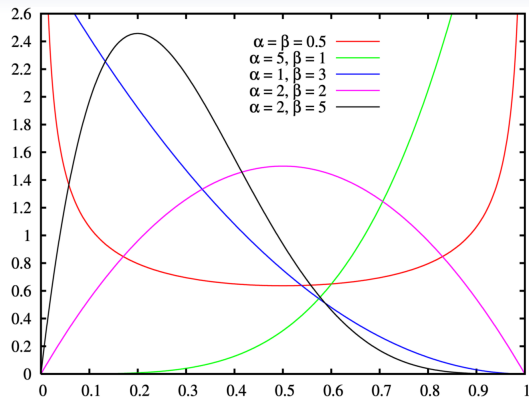


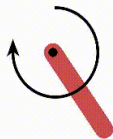
Figure: Densité de probabilité de la distribution Beta

Processus expérimental pour une comparaison fiable

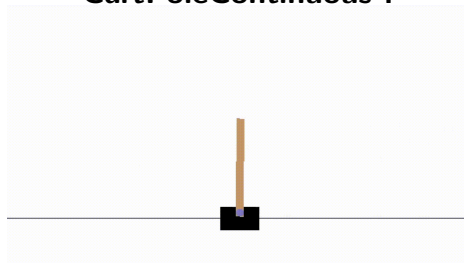
- On s'assure que la politique de départ soit la même pour CEM et PG.
- Les paramètres de CEM et PG sont fixés tout au long des expériences: taux d'apprentissage de 10^{-4} pour PG, bruit égal à la matrice identité pour CEM.
- Même budget de 50 évaluations durant l'apprentissage:
 1. CEM génère 25 politiques par itération qu'il évalue chacune 2 fois.
 2. PG génère 50 trajectoires depuis la politique courante qu'il évalue chacune 1 fois.

Benchmarks classiques: Pendulum et CartPoleContinuous

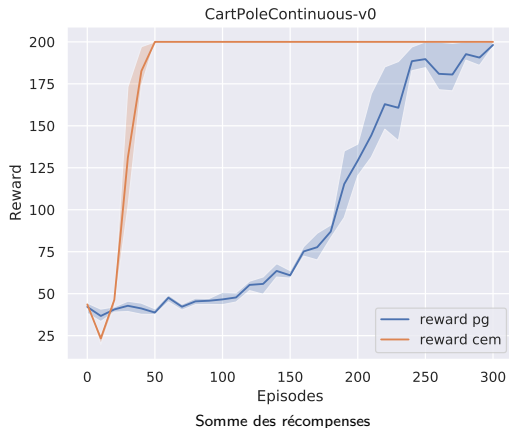
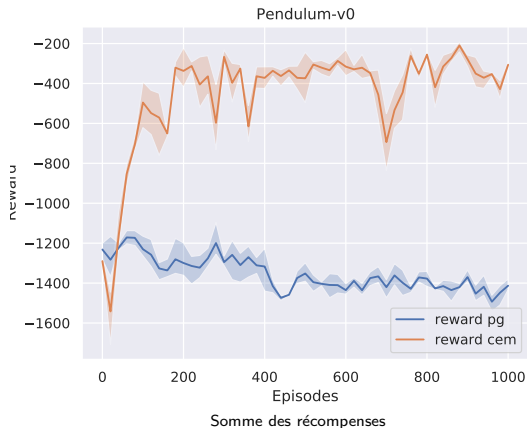
Pendulum :



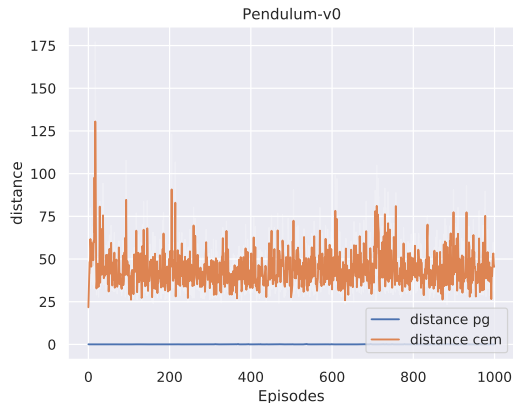
CartPoleContinuous :



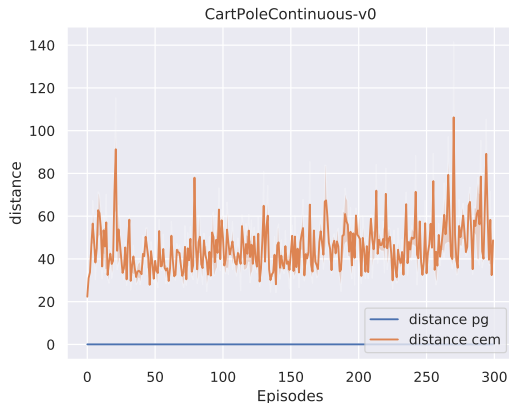
Résultat: CEM performe mieux que PG sur les benchmarks étudiés



Résultat: L'exploration de l'espace des politiques est différente avec CEM et PG sur Pendulum et CartPoleContinuous



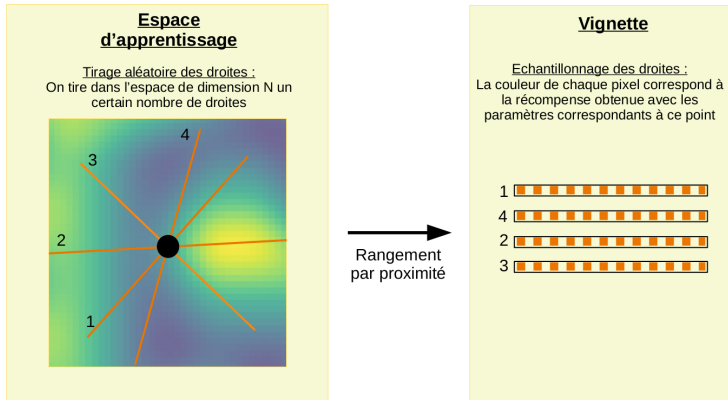
Distances entre politiques apprises successivement



Distances entre politiques apprises successivement

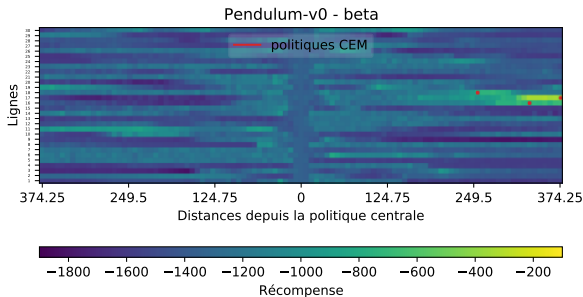
Interlude Vignette

Comment visualiser l'espace des politiques pour Pendulum ou CartPoleContinuous, espace en grandes dimensions, en 2D ou 3D?

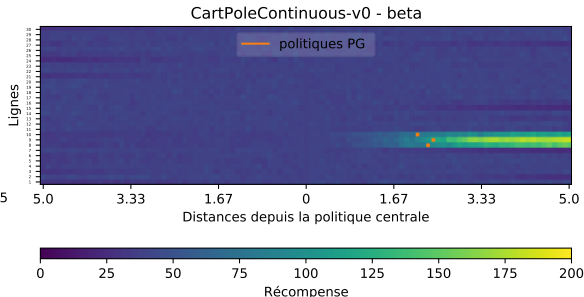


Vue de l'esprit de l'espace d'apprentissage (bruit Perlin)

Résultat: Les pentes améliorantes dans le paysage des récompenses sont loin de l'initialisation sur Pendulum, et proches sur CartPoleContinuous



Visualisation du paysage des récompenses autour de l'initialisation et des 3 premières politiques obtenues avec CEM dans une pente



Visualisation du paysage des récompenses autour de l'initialisation et des 3 premières politiques obtenues avec PG dans une pente

Liste des résultats tirés des expériences

Pendulum :

1. Bonne performance de CEM et mauvaise performance de PG.
2. Grandes distances d'une politique à l'autre pour CEM et petites pour PG.
3. Paysage plat autour de l'initialisation et pentes améliorantes vers les optima loin de l'initialisation.

CartPoleContinuous :

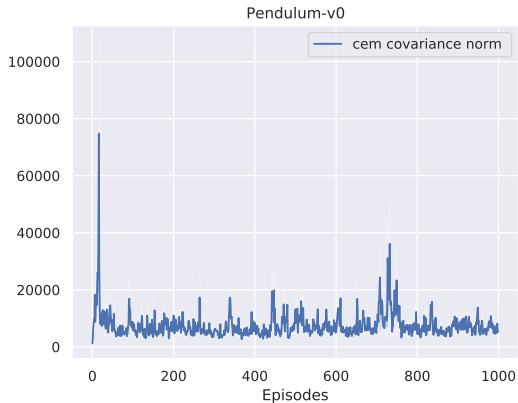
1. Les deux méthodes ont de bonnes performances.
2. Grandes distances d'une politique à l'autre pour CEM et petites pour PG.
3. Paysage plat autour de l'initialisation et pentes améliorantes vers les optima proches de l'initialisation.

Pourquoi CEM performe t-il mieux que PG sur Pendulum ?

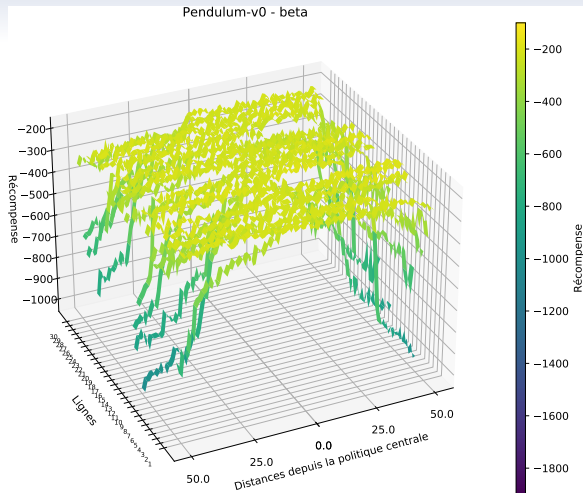
- CEM peut effectuer de grands déplacements et s'éloigner rapidement de la zone plate autour de l'initialisation contrairement à PG.
- Cela est nécessaire pour des espaces de recherche dans lesquels les optima sont loin de l'initialisation comme sur Pendulum.
- Une zone de plat dans un paysage à faibles récompenses, comme autour de l'initialisation Pendulum, représente un ensemble de politiques peu performantes. Comme PG estime la direction dans laquelle explorer de manière locale à partir de la politique courante, la direction estimée ne sera jamais améliorante.
- Alors que sur CartPoleContinuous, il existe des pentes améliorantes (des bonnes politiques) proches de l'initialisation. PG peut donc estimer une direction améliorante et explorer vers celle-ci.

PG a donc du mal à quitter l'espace plat d'initialisation de Pendulum ce qui explique ses faibles performances → *Problème d'exploration de PG.*

Autre résultat: diversité des politiques optimales



Norme de la covariance utilisée pour tirer des politiques avec CEM



Visualisation du paysage des récompenses autour d'un optimum local obtenu avec CEM.

Autre résultat: diversité des politiques optimales

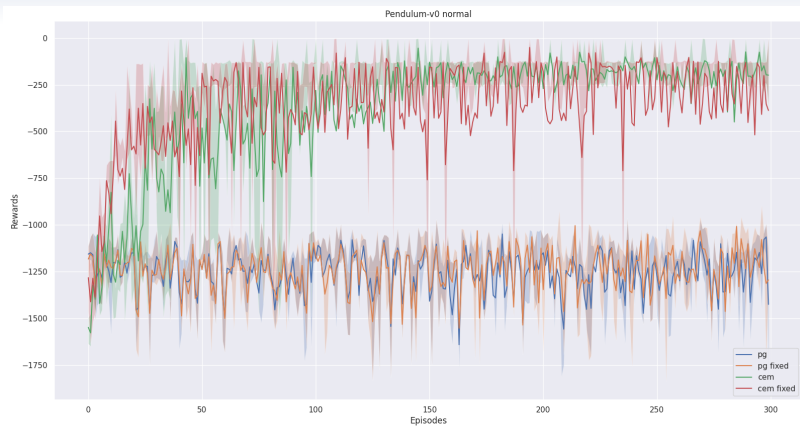


Figure: Performances de PG et CEM avec couches fixées sur Pendulum

Conclusion et travaux futurs

Conclusion





- CEM : Effectue de grands déplacements pour quitter les vastes zones plates de l'initialisation sur Pendulum.
- PG : Effectue de petits déplacements et n'arrive pas à quitter les vastes zones plates de l'initialisation sur Pendulum.
- Si PG performe mieux sur CartPoleContinuous c'est grâce à la présence de pentes améliorantes proches de l'initialisation.
- Il existe une grande diversité de politiques optimales qui se traduit par de vastes plateaux de bonnes récompenses dans l'espace des politiques sur Pendulum.

Travaux futurs






- Etude de différentes initialisations.
- Etude de différentes structures de réseaux de neurones (par exemple, augmenter le nombre de neurones par couche).
- Etude des différents hyper-paramètres.

Remerciements

References I

-  Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba, *OpenAI gym*.
-  Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter, *Back to basics: Benchmarking canonical evolution strategies for playing atari*.
-  Po-Wei Chou, Daniel Maturana, and Sebastian Scherer, *Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution*, International Conference on Machine Learning, PMLR, ISSN: 2640-3498, pp. 834–843.
-  Yan Duan, Xi Chen, Rein Houthoofd, John Schulman, and Pieter Abbeel, *Benchmarking deep reinforcement learning for continuous control*, International Conference on Machine Learning, PMLR, ISSN: 1938-7228, pp. 1329–1338.

References II

-  Marc Peter Deisenroth, *A survey on policy search for robotics*, no. 1, 1–142.
-  Scott Gigante, Adam S. Charles, Smita Krishnaswamy, and Gal Mishne, *Visualizing the PHATE of neural networks*.
-  Shie Mannor, Reuven Y. Rubinstein, and Yohai Gat, *The cross entropy method for fast policy search*.
-  Paolo Pagliuca, Nicola Milano, and Stefano Nolfi, *Efficacy of modern neuro-evolutionary strategies for continuous control optimization*.
-  Jan Peters and Stefan Schaal, *Reinforcement learning by reward-weighted regression for operational space control*, Proceedings of the 24th international conference on Machine learning, ICML '07, Association for Computing Machinery, pp. 745–750.

References III

-  Richard S. Sutton and Andrew G. Barto, *Reinforcement learning, second edition: An introduction*, MIT Press, Google-Books-ID: uWV0DwAAQBAJ.
-  Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever, *Evolution strategies as a scalable alternative to reinforcement learning*.
-  István Szita and András Lörincz, *Learning tetris using the noisy cross-entropy method*, no. 12, 2936–2941.
-  Olivier Sigaud and Freek Stulp, *Policy search in continuous action domains: an overview*.
-  Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba, *Benchmarking model-based reinforcement learning*.

References IV



Nir Ben Zrihem, Tom Zahavy, and Shie Mannor, *Visualizing dynamics: from t -SNE to SEMI-MDPs*.