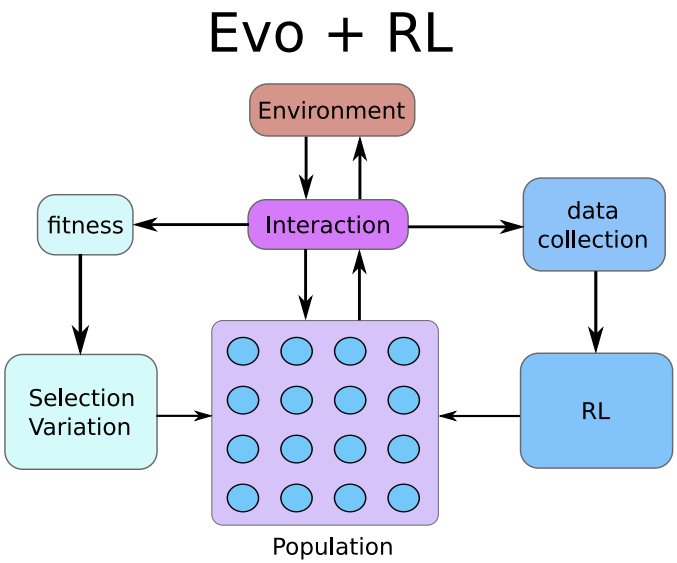
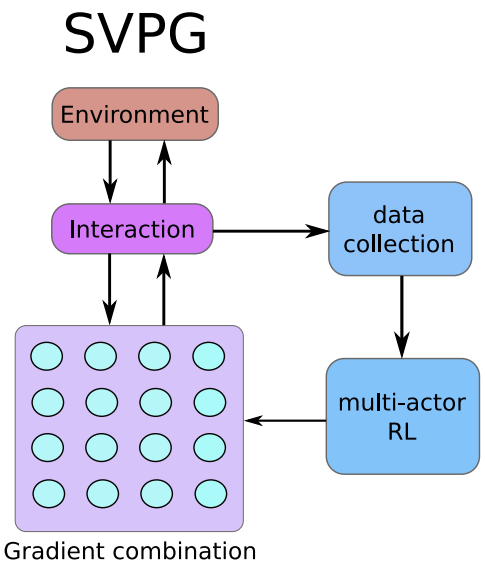
Stein Variational Policy Gradient

Olivier Sigaud

Sorbonne Université
http://people.isir.upmc.fr/sigaud

Why this paper?

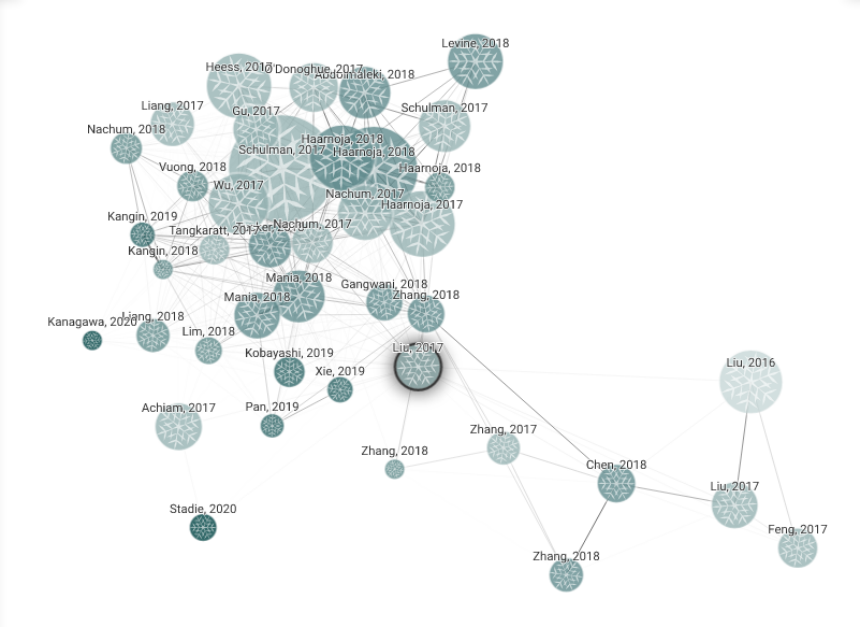




- Does not fit in the Evo + RL framework
- Instead of selection in a population of agents, coordination of a team of agents through gradient composition

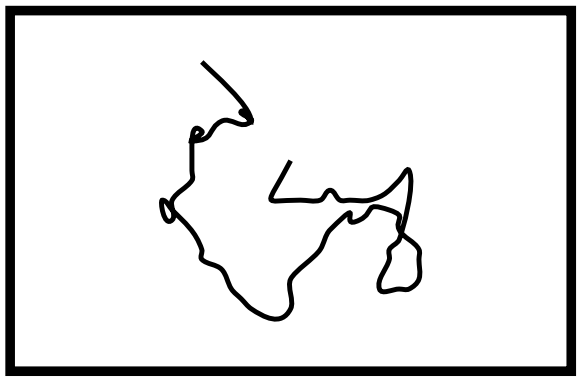
Liu, Y., Ramachandran, P., Liu, Q., and Peng, J. (2017) Stein Variational Policy Gradient. *arXiv preprint arXiv:1704.02399*, UAI

Why this paper (2)?

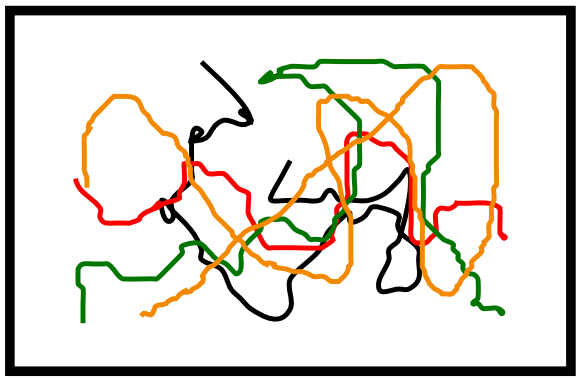


- A rather ignored important paper (UAI 2017, cited 106 times)
- Links with several more recent papers which ignore it

Distributed RL



One worker



Many workers

- Distributed RL often outperforms the single agent version
- Several workers in parallel: more i.i.d and faster exploration
- The acceleration is better than linear in the number of workers
- Independent workers might be or become very redundant
- SVPG: the gradient of a worker should depend on the distance to other workers

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. (2018) Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*

Adamski, I., Adamski, R., Grel, T., Jedrych, A., Kaczmarek, K., & Michalewski, H. (2018) Distributed deep reinforcement learning: Learn how to play atari games in 21 minutes. *arXiv preprint arXiv:1801.02852*

Overview, positioning

- Bayesian Optimization approach, with particles
  - Relationship to MCMC
- Has an entropy term
  - Relationship to natural gradient, to SAC
- The ES view, the distributed RL view
  - Relationship to monomodal evolution strategies
- Manages the distance between workers
  - Relationship to Novelty search, QD, P3S-TD3, DvD
- Opening: adding selection into SVPG instead of CEM in ERL

The distributed RL objective

- We have policy parameters $\boldsymbol{\theta}$, a distribution over these parameters $q(\boldsymbol{\theta})$
- We want to optimize the expected return of policies over $q(\boldsymbol{\theta})$
- If we have some prior $q_0(\boldsymbol{\theta})$, we want our solution to be close to $q_0(\boldsymbol{\theta})$
- We solve

$$\max_q \mathbb{E}_{q(\boldsymbol{\theta})}[J(\boldsymbol{\theta})] - \alpha D_{KL}\left(q \| q_0\right) \tag{1}$$

- If we have no prior (i.e. $q_0(\boldsymbol{\theta}) = const$, uniform prior), we get

$$\max_q \mathbb{E}_{q(\boldsymbol{\theta})}[J(\boldsymbol{\theta})] + \alpha \mathcal{H}(q) \tag{2}$$

- That is, we maximize the entropy $\mathcal{H}(q) = \mathbb{E}_{q(\boldsymbol{\theta})}[-\log q(\boldsymbol{\theta})]$.
- Relationship to SAC: SVPG maximises the entropy over policy parameters, SAC maximizes the entropy over actions given states $\mathcal{H}(\pi_{\boldsymbol{\theta}}(.|\mathbf{s}_t)) = \mathbb{E}_{\mathbf{a}_t \sim \pi_{\boldsymbol{\theta}}(.|\mathbf{s}_t)}\left[-\log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t|\mathbf{s}_t)\right]$

The SVPG approach

- Reminder (1):

$$\max_q \mathbb{E}_{q(\boldsymbol{\theta})}[J(\boldsymbol{\theta})] - \alpha D_{KL}\left(q \| q_0\right)$$

- By taking the derivative of (1) and setting it to 0, we get

$$q(\boldsymbol{\theta}) \propto \exp(\frac{1}{\alpha} J(\boldsymbol{\theta})) q_0(\boldsymbol{\theta}). \tag{3}$$

- This can be seen as Bayesian inference where $q_0(\boldsymbol{\theta})$ is the prior, $\exp(J(\boldsymbol{\theta})/\alpha)$ is the likelihood function and $q(\boldsymbol{\theta})$ is the posterior.
- One way to solve (3) is through MCMC, but MCMC suffers from variance and slow convergence or oscillations
- SVPG rather works tries to maximize (1) following $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ and uses Stein Variational Gradient Descent (SVGD)
- Instead of considering an expectation over a continuous distribution, approximate it with a set of particles (in practice, 16 agents)

The SVGD solution (technical core)

- Get a set of "particles" $\{\boldsymbol{\theta}_i\}$
- For each particle, apply $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i + \epsilon\phi^*(\boldsymbol{\theta}_i)$ (gradient step)
- $\phi^*(\boldsymbol{\theta})$ is an optimal gradient descent direction (maximally decreases the KL divergence between the particles and the target distribution $q(\boldsymbol{\theta})$)
- I omit the derivation from the SVGD paper, but we get:

$$\phi^*(\boldsymbol{\theta}) = \mathbb{E}_{\vartheta\sim\rho}[\nabla_\vartheta \log q(\vartheta)k(\vartheta,\boldsymbol{\theta}) + \nabla_\vartheta k(\vartheta,\boldsymbol{\theta})] \tag{4}$$

  where ρ is the current distribution over $\boldsymbol{\theta}$ and $k(\vartheta,\boldsymbol{\theta})$ is a distance kernel
- Instead of the expectation, we average over particles $\{\boldsymbol{\theta}_i\}$

$$\hat{\phi}_i(\boldsymbol{\theta}_i) = \frac{1}{n}\sum_{j=1}^{n}[\nabla_{\boldsymbol{\theta}_j} \log q(\boldsymbol{\theta}_j)k(\boldsymbol{\theta}_j,\boldsymbol{\theta}_i) + \nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_j,\boldsymbol{\theta}_i)] \tag{5}$$

- Remind (3): $q(\boldsymbol{\theta}) \propto \exp(\frac{1}{\alpha}J(\boldsymbol{\theta}))q_0(\boldsymbol{\theta})$, we replace in (5) and get:

$$\hat{\phi}_i(\boldsymbol{\theta}_i) = \frac{1}{n}\sum_{j=1}^{n}[\nabla_{\boldsymbol{\theta}_j}(\frac{1}{\alpha}J(\boldsymbol{\theta}_j) + \log q_0(\boldsymbol{\theta}_j))k(\boldsymbol{\theta}_j,\boldsymbol{\theta}_i) + \nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_j,\boldsymbol{\theta}_i)]. \tag{6}$$

- Working with particles scales better than MCMC

Liu, Q. and Wang, D. (2016) Stein Variational Gradient Descent: A general purpose Bayesian inference algorithm. *arXiv preprint arXiv:1608.04471*

Closer look

$$\hat{\phi}_i(\theta_i) = \frac{1}{n}\sum_{j=1}^{n}[\underbrace{\nabla_{\theta_j}(\frac{1}{\alpha}J(\theta_j) + \log q_0(\theta_j))k(\theta_j,\theta_i)}_{\text{return maximization term}} + \underbrace{\nabla_{\theta_j}k(\theta_j,\theta_i)}_{\text{repulsive term}}] \quad (6)$$

- For the distance kernel, following SVGD, they use:

$$k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i) = \exp(\frac{-||\boldsymbol{\theta}_i - \boldsymbol{\theta}_j||_2^2}{h}) \quad (7)$$

  where $h = med^2/\log(n+1)$ and med is the median of pairwise distances between particles $\{\boldsymbol{\theta}_i\}$
- "This simple heuristic allows the bandwidth to adaptively change as the particles move, ensuring that there is always a significant number of particles that interact with each other."
- From the code, the gradient $\nabla_{\boldsymbol{\theta}_j}k(\boldsymbol{\theta}_j,\boldsymbol{\theta}_i)$ at a $\boldsymbol{\theta}_i$ is $\nabla_{\boldsymbol{\theta}_j}k(\boldsymbol{\theta}_j,\boldsymbol{\theta}_i) = \frac{2}{h}(\boldsymbol{\theta}_j - \boldsymbol{\theta}_i)\exp(\frac{-||\boldsymbol{\theta}_i-\boldsymbol{\theta}_j||_2^2}{h})$
- $\nabla_{\boldsymbol{\theta}_j}J(\boldsymbol{\theta}_j)$ is obtained by any PG algorithm (they use REINFORCE and A2C)

The SVPG algorithm

Algorithm 1 Stein Variational Policy Gradient

Input: Learning rate ϵ, kernel $k(x, x')$, temperature, initial policy particles $\{\theta_i\}$.
for iteration $t = 0, 1, .., T$ **do**
  for particle $i = 0, 1, .., n$ **do**
    Compute $\nabla_{\theta_i} J(\theta_i)$ using an RL algorithm
  end for
  for particle $i = 0, 1, .., n$ **do**

$$\Delta\theta_i \leftarrow \frac{1}{n}\sum_{j=1}^{n}[\nabla_{\theta_j}\left(\frac{1}{\alpha}J(\theta_j) + \log q_0(\theta_j)\right)k(\theta_j, \theta_i) + \nabla_{\theta_j} k(\theta_j, \theta_i)]$$

$$\theta_i \leftarrow \theta_i + \epsilon\Delta\theta_i$$

  end for
end for

- The temperature is α

Overview of the main loop

1. For K iterations (say $K = 1000$)
2. Collect samples during many time steps (e.g. 50.000) for each individual
3. Compute the REINFORCE gradient:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{H} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)}) [\sum_{k=t}^{H} \gamma^k r(\mathbf{s}_k^{(i)}, \mathbf{a}_k^{(i)})] \qquad (8)$$

4. Then compute

$$\hat{\phi}_i(\theta_i) = \frac{1}{n} \sum_{j=1}^{n} [\underbrace{\nabla_{\theta_j} (\frac{1}{\alpha} J(\theta_j) + \log q_0(\theta_j)) k(\theta_j, \theta_i)}_{\text{return maximization term}} + \underbrace{\nabla_{\theta_j} k(\theta_j, \theta_i)}_{\text{repulsive term}}] \qquad (6)$$

5. Apply $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i + \epsilon \phi^*(\boldsymbol{\theta}_i)$ (gradient step)

In distributed RL, omit step 4 and apply $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i + \epsilon \nabla_{\boldsymbol{\theta}_i} J(\boldsymbol{\theta}_i)$

Relationship to Monomodal Evolution Strategies

- Note: they mention the relationship in the background
- In Evolution Strategies, a set of particles used to collectively perform approximate gradient ascent
- Each particle is a random variation of the previous solution
- In CEM and CMA-ES, the particles are sampled from a Gaussian
- In SVPG:
  - Each particle (blindly) follows its own gradient (no random variation)
  - No constraint to comply to a Gaussian distribution (a key feature of SVGD)
  - No post hoc evaluation and selection of the bests
- Idea: build a variation-selection version of SVPG, with one CEM per particle?

Relationship to TRPO

- In TRPO, one uses the Natural Gradient to enforce a "trust region"
- Apply some gradient transformation (using the inverse Fisher Information matrix)
- Used to keep a low distance to the previous policy in the policy output space
- In SVPG, keep a high distance to current other policies

Schulman, J., Levine, S., Moritz, P., Jordan, M. I., & Abbeel, P. (2015) Trust Region Policy Optimization. *CoRR, abs/1502.05477*

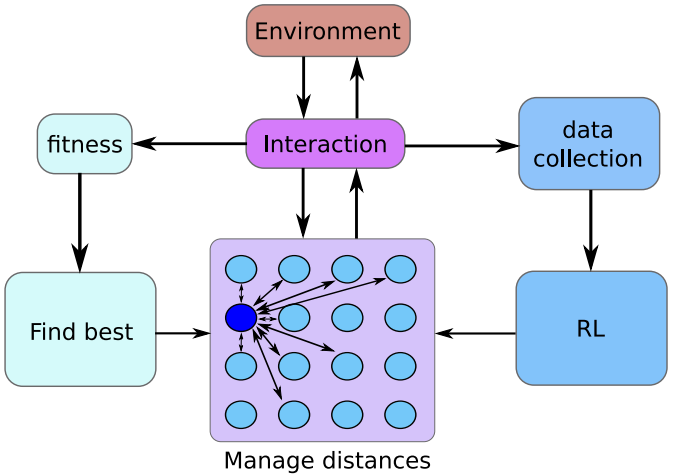
Relationship to Novelty Search, Quality Diversity

$$\hat{\phi}_i(\theta_i) = \frac{1}{n}\sum_{j=1}^{n}[\underbrace{\nabla_{\theta_j}(\frac{1}{\alpha}J(\theta_j) + \log q_0(\theta_j))k(\theta_j,\theta_i)}_{\text{return maximization term}} + \underbrace{\nabla_{\theta_j}k(\theta_j,\theta_i)}_{\text{repulsive term}}] \quad (6)$$

- The repulsive term tries to maximize distance between policies
- A gradient-based approach to the search for diversity
- The return maximization term corresponds to maximizing quality
- A form of gradient-based QD? Link to QDRL
- A linear combination of both is less rich than QD approaches

Pierrot, T., Macé, V., Cideron, G., Beguir, K., Cully, A., Sigaud, O., and Perrin, N. Diversity policy gradient for sample efficient quality-diversity optimization. In *Submitted to GECCO*, 2022

Relationship to P3S-TD3



- P3S-TD3 is an intermediate between PBT and SVPG (ignores SVPG)
- Also maintains a distance between policies to prevent collapse (part not covered by the derivation)
- The mechanism is somewhat ad hoc
- By contrast, in SVPG, the repulsive part is integrated directly in the gradient update derivation

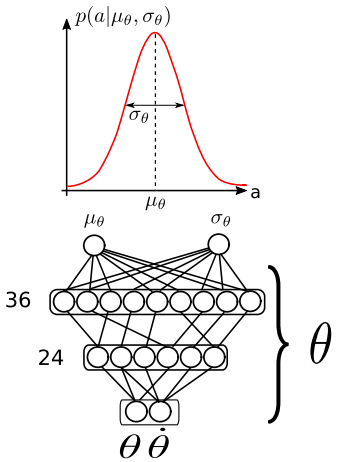
$$\hat{\phi}_i(\theta_i) = \frac{1}{n}\sum_{j=1}^{n}[\underbrace{\nabla_{\theta_j}(\frac{1}{\alpha}J(\theta_j) + \log q_0(\theta_j))k(\theta_j,\theta_i)}_{\text{return maximization term}} + \underbrace{\nabla_{\theta_j}k(\theta_j,\theta_i)}_{\text{repulsive term}}] \quad (6)$$

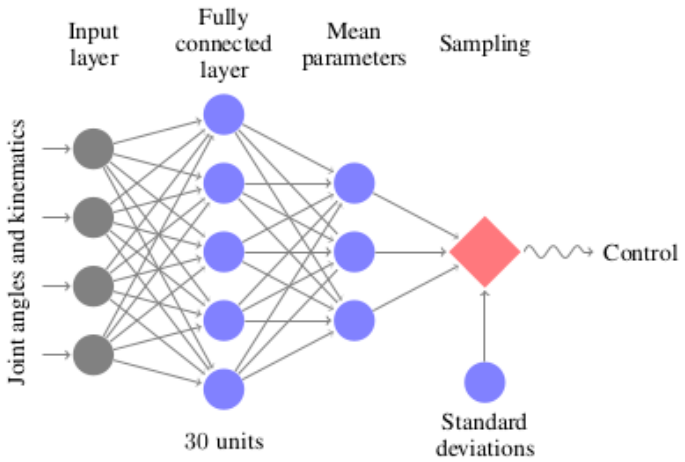
Jung, W., Park, G., and Sung, Y. (2020) Population-guided parallel policy search for reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net URL https://openreview.net/forum?id=rJeINp4KwH

Experimental setup

- rllab versions: CartPole, CartPole Swing-Up (pole initially down?), DoublePendulum, MountainCar
- All with max time = 500 steps (score of 5000 for CartPole?)
- Popsize = 16 (not studied)
- 10.000 training time steps to get a gradient (that's small)
- CartPole Swing-Up, DoublePendulum: 1000 iterations → 10 million samples!
- MountainCar and Cartpole: 50 iterations → 500K samples
- Temperature $\alpha = 10$ (best results, see below)
- 5 seeds

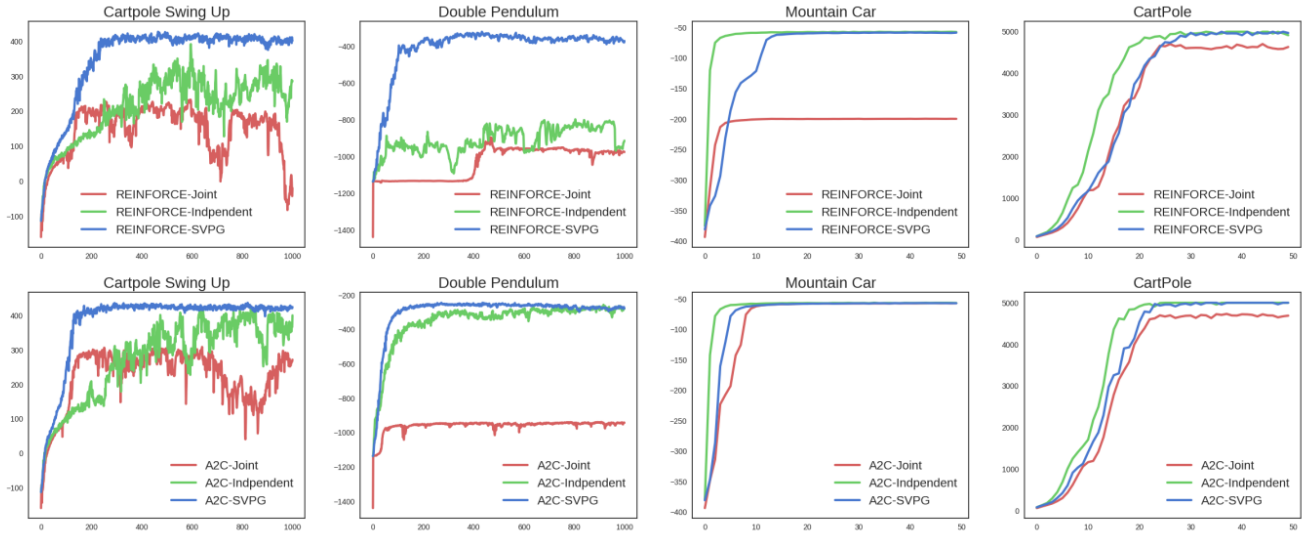
Policy architecture





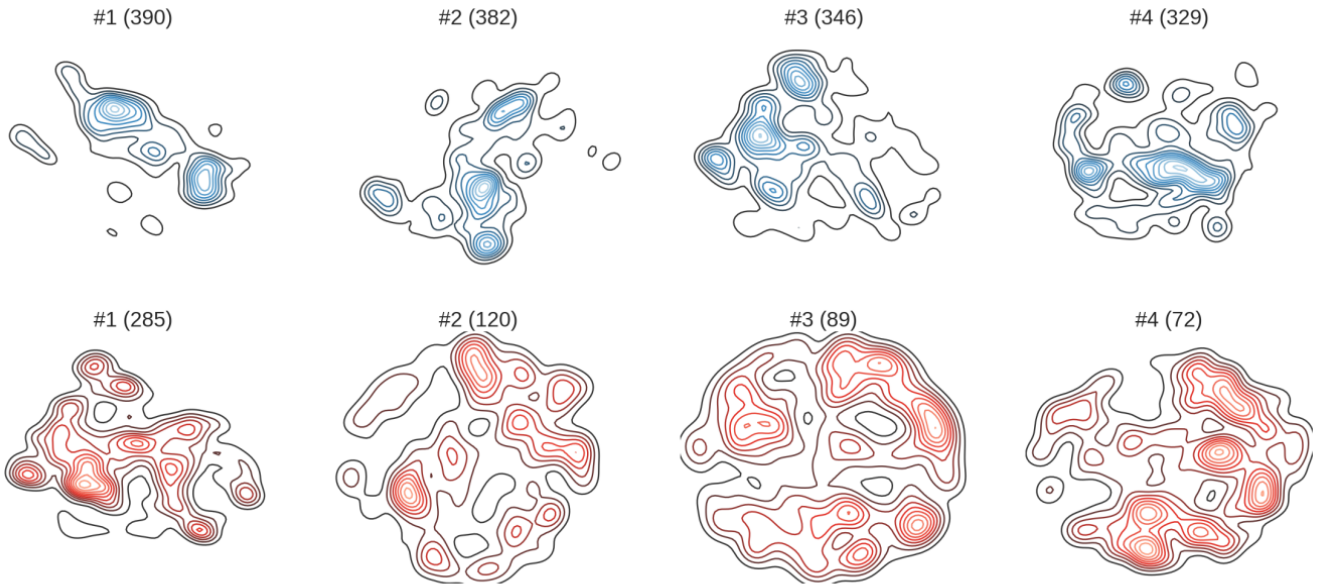
- 3 layers NN: 100 - 50 -25 (unusual), tanh activation, diagonal covariance output layer (Gaussian)
- The log standard deviation is parameterized by a vector, as suggested in (Duan et al., 2016, Schulman et al., 2015a).
- It means std parameters are not function of the state, but adapted with GD.
- In code, `nn.std = nn.Parameter(torch.randn(action_size, 1))`

Comparison to independent and joint workers



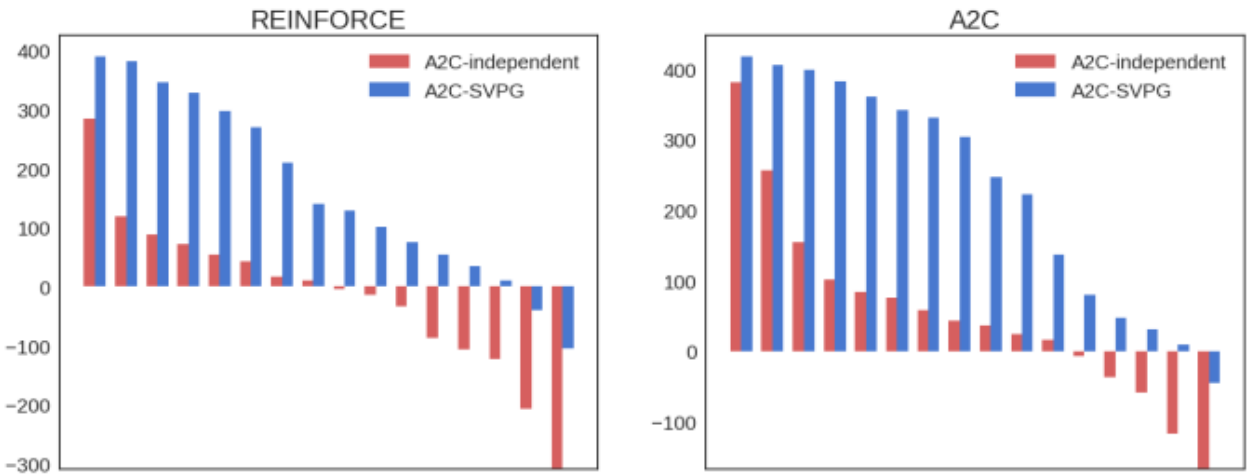
- Joint worker: a single agent with 160.000 time steps to get a gradient (same training budget)
- Independent workers seem to work better in several benchmarks

Visitation of states by the best policies



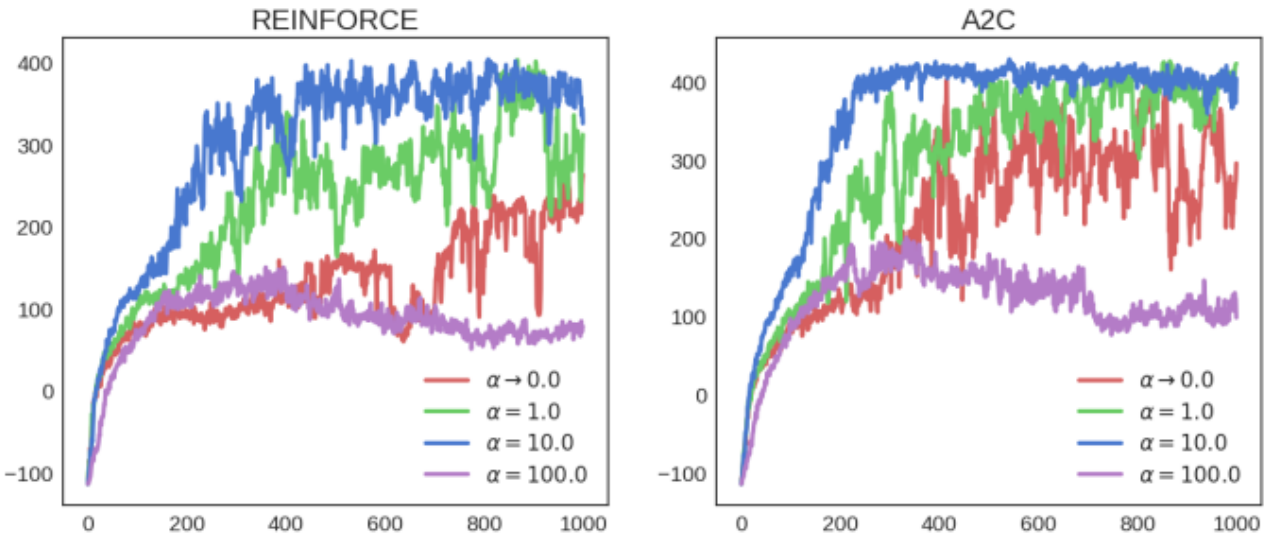
- Good SVPG policies (blue) seem to cover less states than with independent workers (red)
- A pro in RL, would be a con in a sparse reward context???

Scores of separate policies



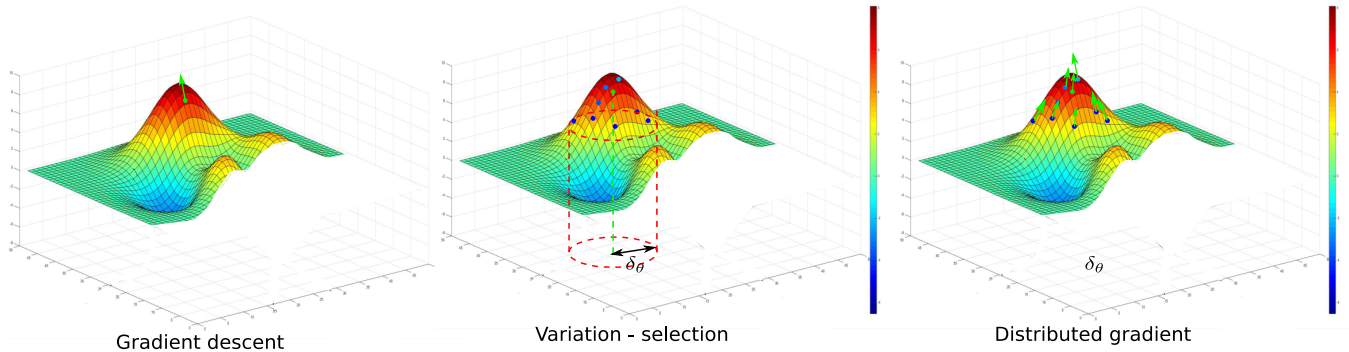
- Given the scores, that's on CartPole Swing-Up (not said)
- All policies seem to work better

Temperature tuning



- Given the scores, that's on CartPole Swing-Up (not said)
- Optuna could find something better than $\alpha = 10$?

Discussion: gradient descent vs variation-selection



Gradient descent | Variation - selection | Distributed gradient

- Distributed RL and SVPG are not the best of both worlds
- Each gradient step is still the guess of a direction
- There is no a posteriori selection as in variation-selection

Limitations, Future work

- Build a new implementation based on SaLinA or SB3
- TODO: redo with gym, more standard architectures, ...
- No study of the population size $\rightarrow$ do it
- Try with deterministic gradient, TD3
- ...

Any question?

Send mail to: `Olivier.Sigaud@upmc.fr`

Adamski, I., Adamski, R., Grel, T., Jedrych, A., Kaczmarek, K., and Michalewski, H.

Distributed deep reinforcement learning: Learn how to play atari games in 21 minutes.
arXiv preprint arXiv:1801.02852, 2018.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K.

IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures.
In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1406–1415. PMLR, 2018.
URL http://proceedings.mlr.press/v80/espeholt18a.html.

Jung, W., Park, G., and Sung, Y.

Population-guided parallel policy search for reinforcement learning.
In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
URL https://openreview.net/forum?id=rJeINp4KwH.

Liu, Q. and Wang, D.

Stein variational gradient descent: A general purpose Bayesian inference algorithm.
arXiv preprint arXiv:1608.04471, 2016.

Liu, Y., Ramachandran, P., Liu, Q., and Peng, J.

Stein variational policy gradient.
arXiv preprint arXiv:1704.02399, 2017.

Pierrot, T., Macé, V., Cideron, G., Beguir, K., Cully, A., Sigaud, O., and Perrin, N.

Diversity policy gradient for sample efficient quality-diversity optimization.
In *Submitted to GECCO*, 2022.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P.

Trust region policy optimization.
In Bach, F. R. and Blei, D. M. (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1889–1897. JMLR.org, 2015.

URL http://proceedings.mlr.press/v37/schulman15.html.