Reinforcement Learning: Policy gradient and TRPO

E0-270 Machine Learning

Dhiraj Shanbhag

Ashish Raghuvanshi

Waquar Azam

Motivation:

Evaluate performance of:

- Vanilla policy gradient
- Shortcomings of policy gradient.
- TRPO

- What is POLICY ?
- What is REWARD ?
- What is TRAJECTORY ?

Policy Gradient

- Motivation for Policy Gradient.
- Variations of Policy Gradient
  - REINFORCE
  - Baseline technique.
  - Actor-Critic

Policy Gradient

$$\underbrace{p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{\pi_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$\theta^\star = \arg\max_\theta E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

How to get probability now ?

$$J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right] \approx \frac{1}{N} \sum_i \sum_t r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

Updating Policy Parameters.

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t} | \mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

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

Cure for Variance : Baselines , Causality

- One among many baseline technique.

$$\nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$$

- Causality : Policy at a time t' can't affect reward at previous time t.
- Q-value : Q(s,a) = one step reward + discount * Value(s')

Actor-Critic

- Value Neural Network assists Policy neural network.
- Advantage function.

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

TRPO

- Problems with policy gradient:
  - Sample efficiency is poor in case of policy gradient.
  - Distance in parameter space is not equal to distance in policy space
    - Step size is hard to get right as a result.

Problems with policy gradient

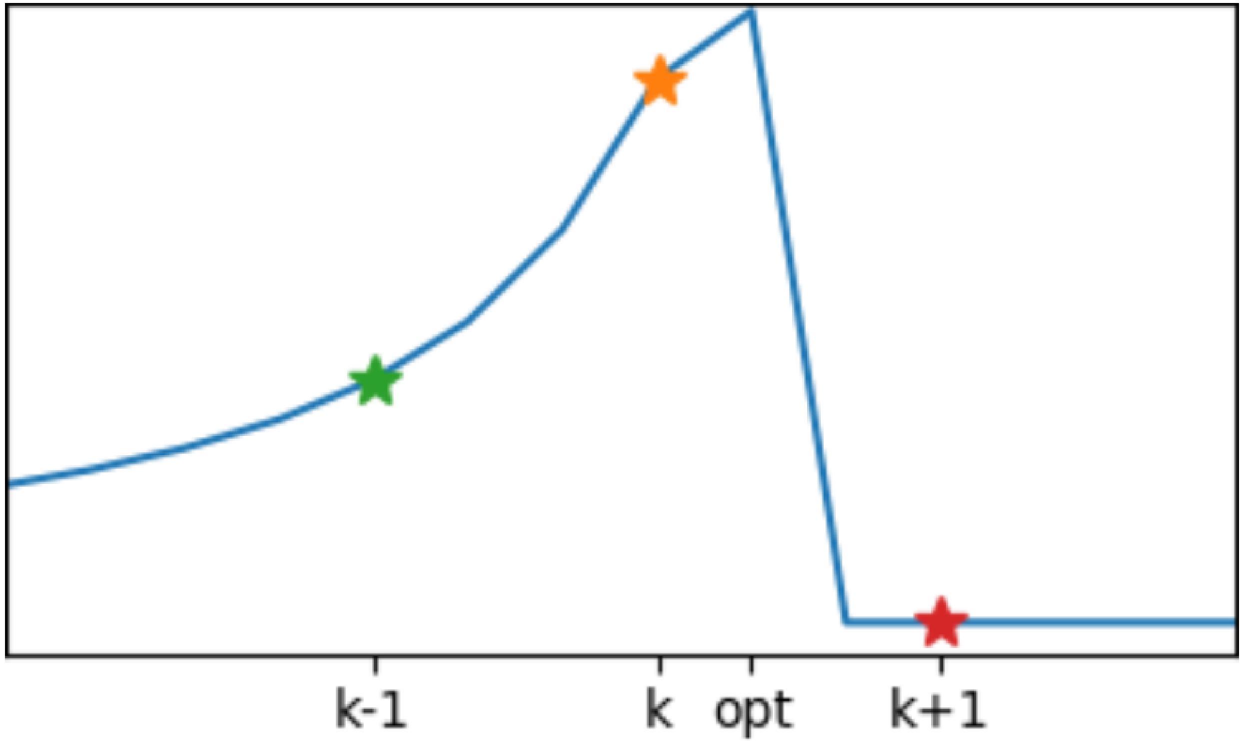


Figure: Policy parameters on x-axis and performance on y-axis. A bad step can lead to performance collapse, which may be hard to recover from.

Reference: Joshua Achiam (UC Berkeley, OpenAI)

Relative Performance of Two Policies

- In a policy optimization algorithm, we want an update step that
  - uses episodes collected from the most recent policy as efficiently as possible,
  - and takes steps that respect distance in policy space instead of distance in parameter space.
- Relative policy performance:

$$J(\pi') - J(\pi) = \mathop{\mathrm{E}}_{\tau \sim \pi'} \left[\sum_{t=0}^{\infty} \gamma^t A^{\pi}(s_t, a_t) \right]$$

Relative performance of Two Policies

$$\left|J(\pi') - \left(J(\pi) + \mathcal{L}_\pi(\pi')\right)\right| \leq C\sqrt{\mathop{\mathrm{E}}_{s\sim d^\pi}\left[D_{KL}(\pi'||\pi)[s]\right]}$$

- Gradient of this surrogate function is equal to the gradient of policy gradient.

- We are guaranteed to improve the policy using MM algorithm w.r.t the true objective.

$$\pi_{k+1} = \arg\max_{\pi'} \mathcal{L}_{\pi_k}(\pi') - C\sqrt{\mathop{\mathrm{E}}_{s\sim d^{\pi_k}}\left[D_{KL}(\pi'||\pi_k)[s]\right]}$$

TRPO Algorithm

- C provided by theory is quite high when discount factor is near 1, which makes step size very small.

- So we use KL constraint instead of KL penalty

- From the constraint, step respect distance in policy space!

  Update is parameterization-invariant.

TRPO Algorithm

Input: initial policy parameters θ_0

for $k = 0, 1, 2, \ldots$ **do**

Collect set of trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$

Estimate advantages $\hat{A}_t^{\pi_k}$ using any advantage estimation algorithm

Form sample estimates for

- policy gradient $\hat{g}_k$ (using advantage estimates)
- and KL-divergence Hessian-vector product function $f(v) = \hat{H}_k v$

Use CG with n_{cg} iterations to obtain $x_k \approx \hat{H}_k^{-1} \hat{g}_k$

Estimate proposed step $\Delta_k \approx \sqrt{\frac{2\delta}{x_k^T \hat{H}_k x_k}} x_k$

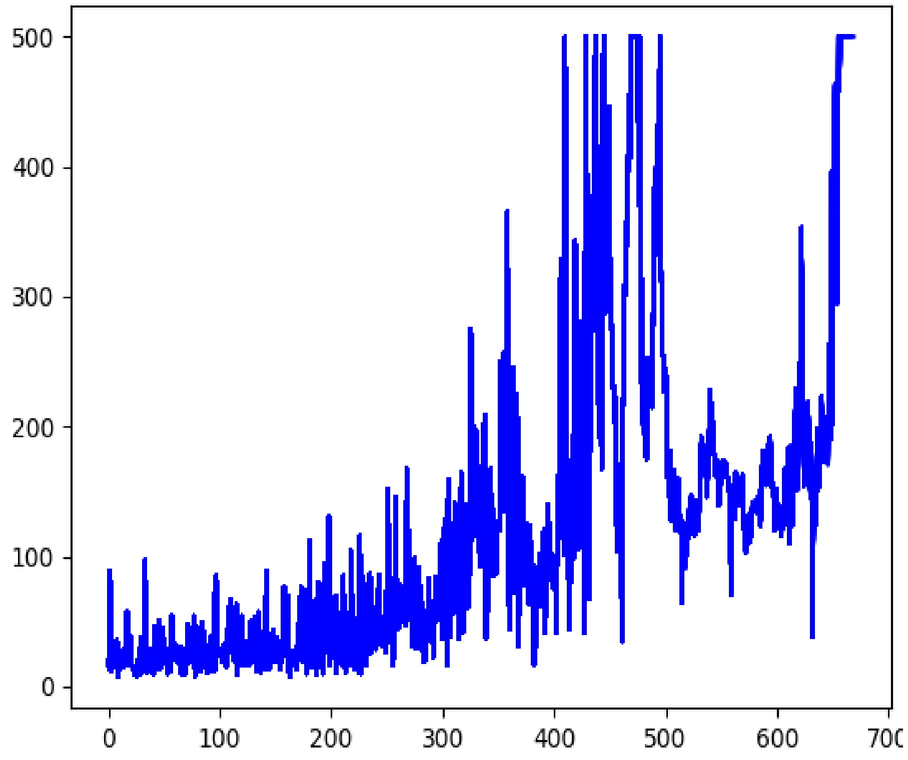
Perform backtracking line search with exponential decay to obtain final update

$$\theta_{k+1} = \theta_k + \alpha^j \Delta_k$$

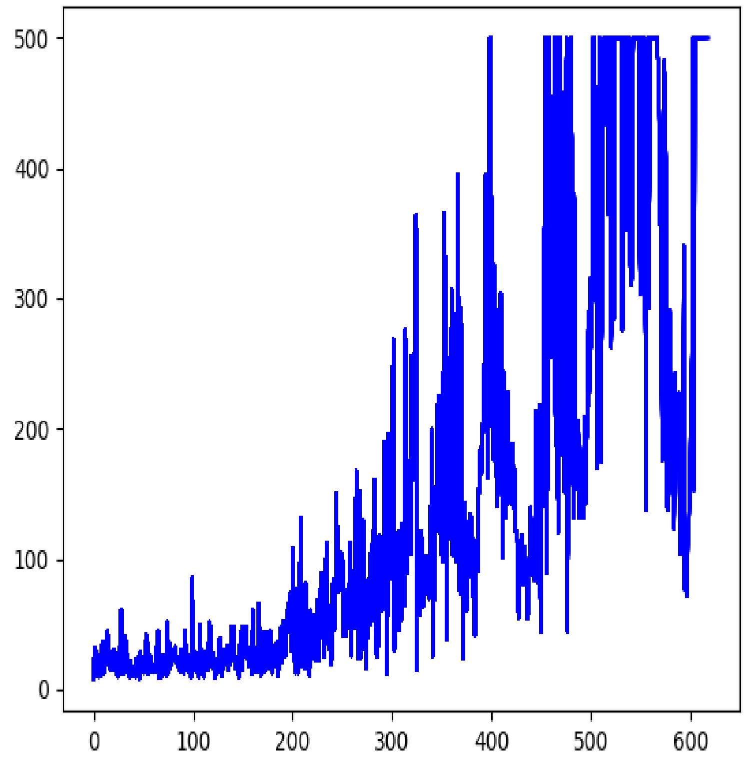
end for

Experiments: cartpole-v1

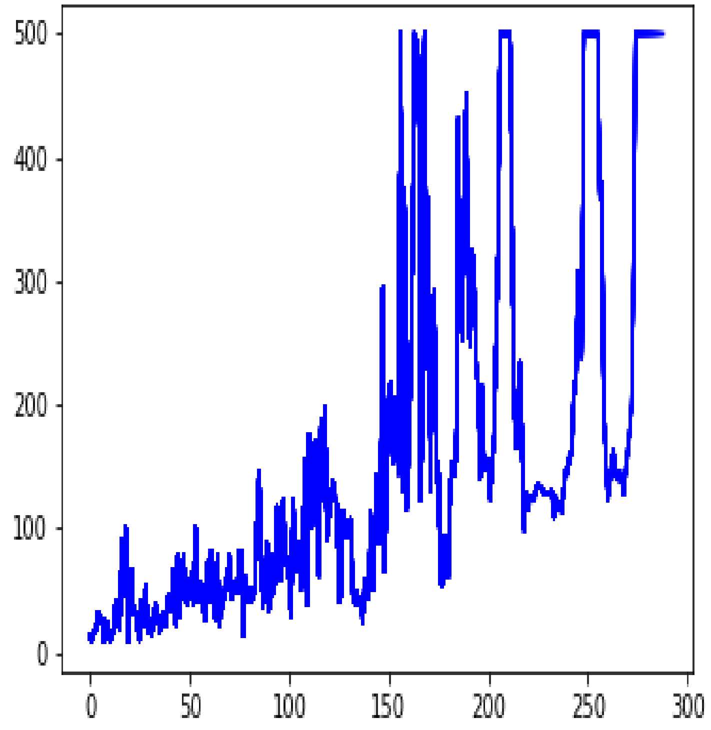
PG



Baseline



AC



X-axis : number of episodes

y: Reward gained.

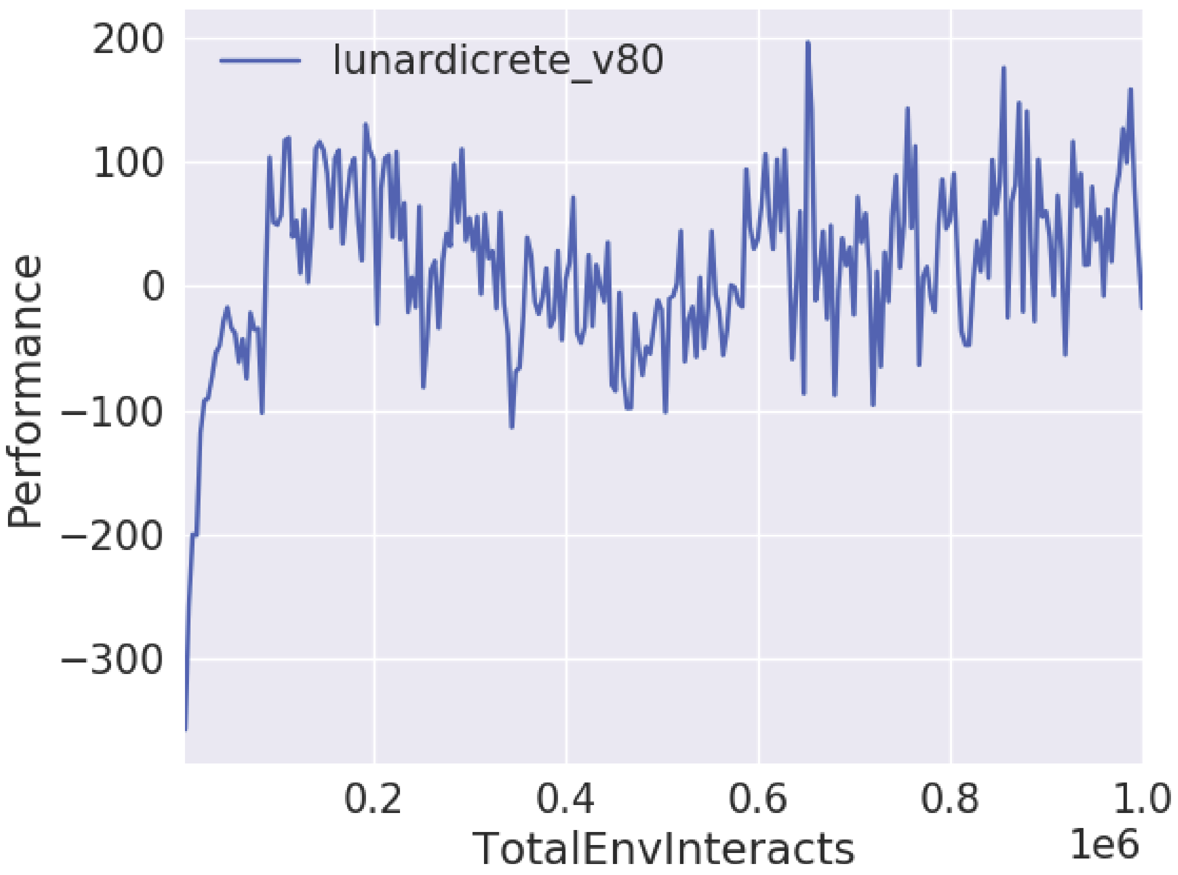
Max-reward: 500

discount_factor = 0.99

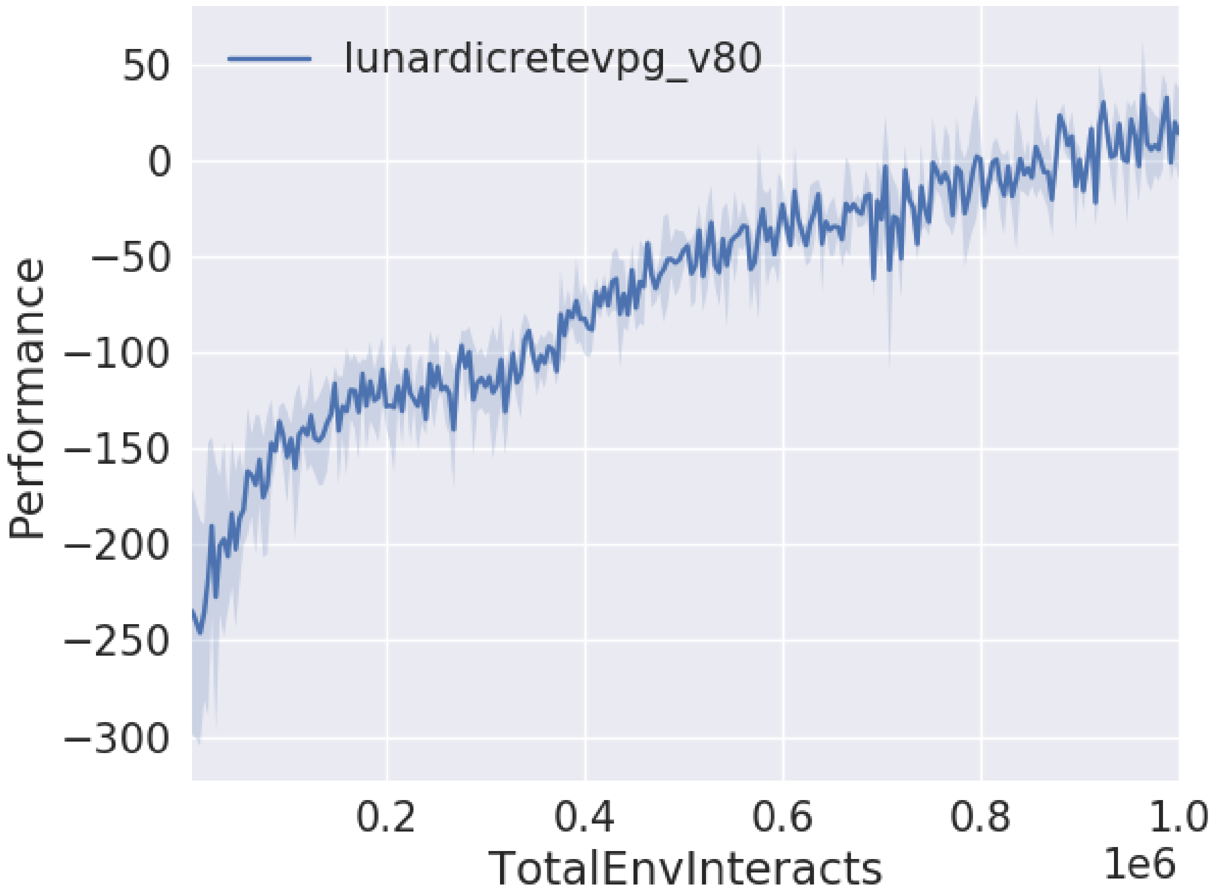
learning_rate = 0.001

Experiments

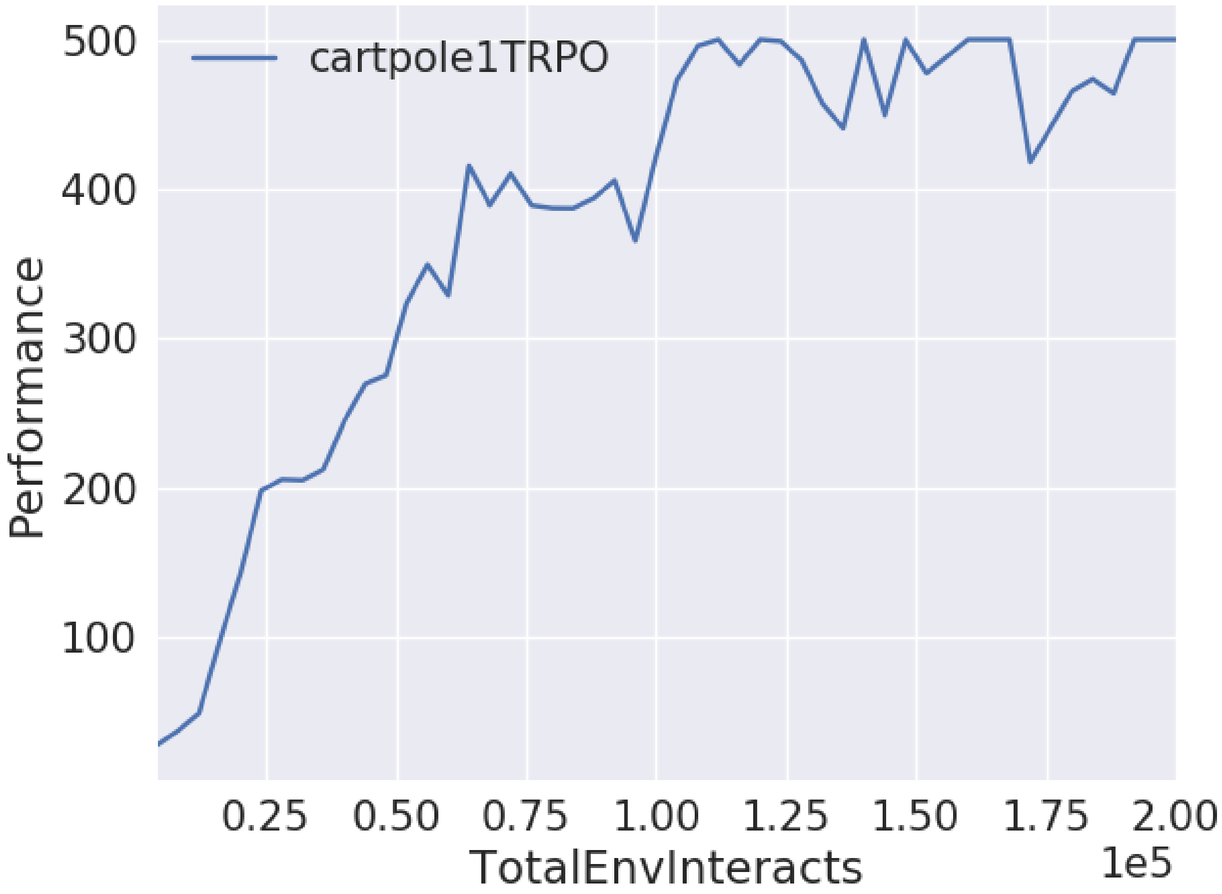
TRPO lunar lander



VPG lunar lander



Experiments

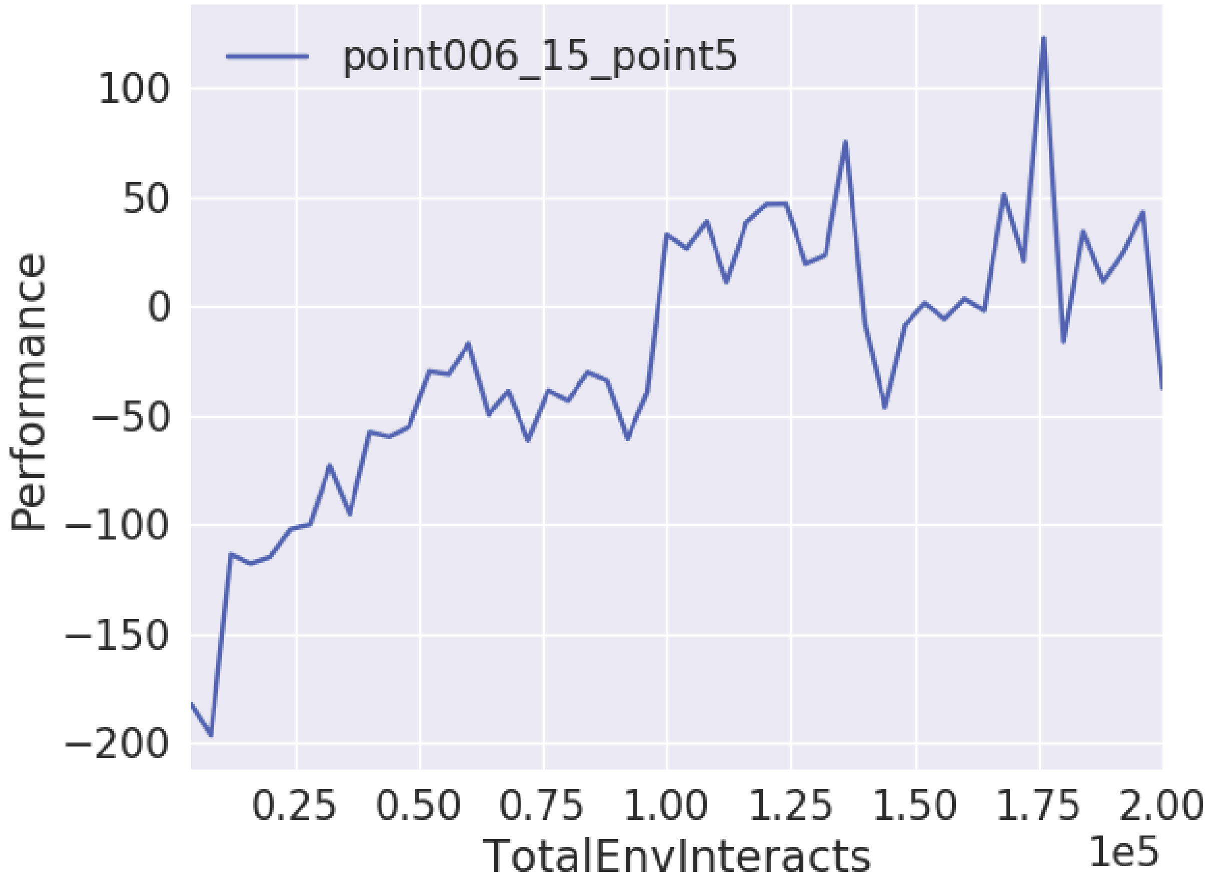


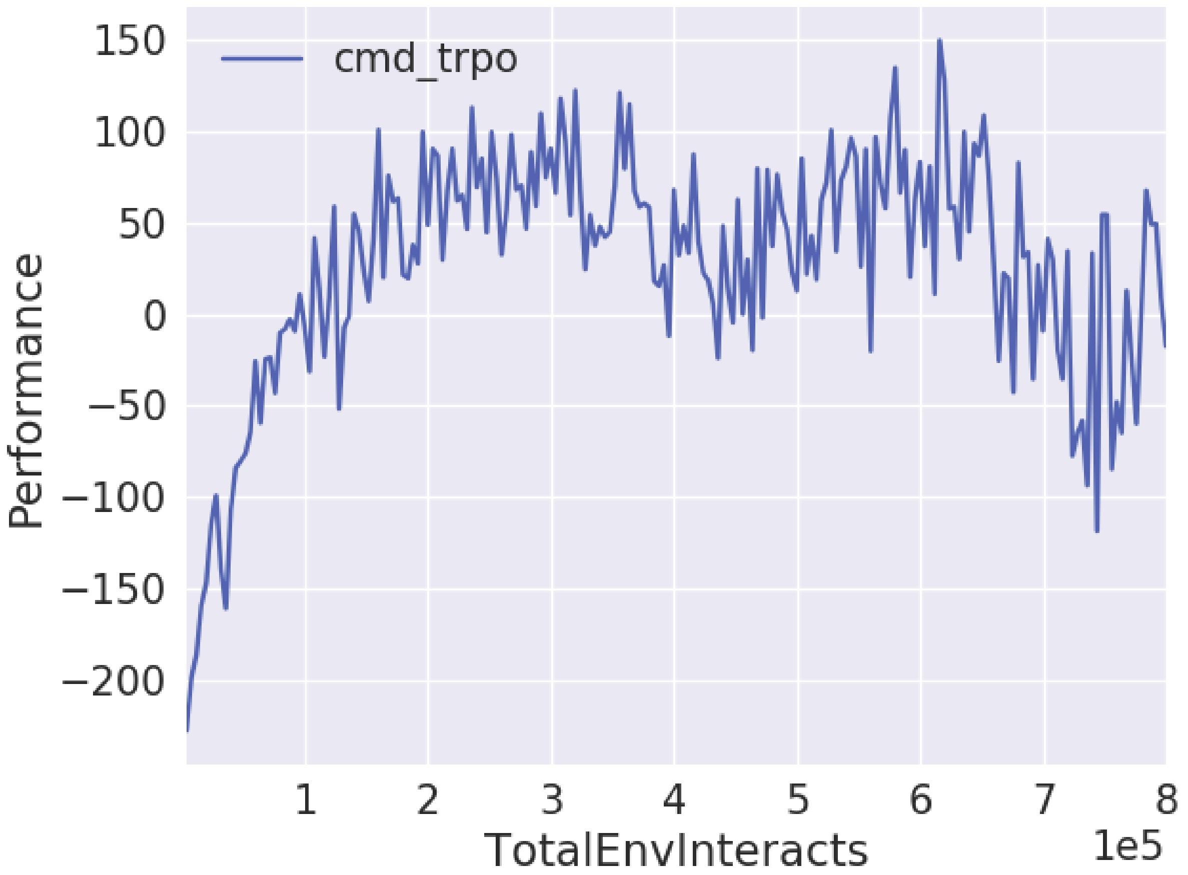
references

- J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. “Trust region policy optimization”. In: CoRR, abs/1502.05477 (2015).
- Richard S. Sutton, David McAllester, Satinder Singh, Yishay Mansour. "Policy Gradient Methods for Reinforcement Learning with Function Approximation "
- Vijay R. Konda John N. Tsitsiklis ." Actor Critic Algorithms".
- S. Kakade and J. Langford. “Approximately optimal approximate reinforcement learning”. In: ICML. Vol. 2. 2002.
- http://rail.eecs.berkeley.edu/deeprlcourse-fa17
- https://spinningup.openai.com/en/latest/algorithms

THANK YOU

Experiments





Experiments