LEARNING AND USING THE RETURN DISTRIBUTION IN OFF-POLICY POLICY OPTIMIZATION

Alex Lewandowski

BATCH OFF-POLICY POLICY OPTIMIZATION

- Batch of N trajectories of length T_n generated by behavior policy π_b

$$\{\{(s_t, a_t, \pi_b(a_t|s_t), r_{t+1})\}_{t=0}^{T_n}\}_{n=1}^{N}.$$

- How to best use this experience to learn some other target policy π_θ?

SOME BACKGROUND AND NOTATION..

SOME BACKGROUND AND NOTATION..

- Discrete set of actions : $1 \ldots A$.

SOME BACKGROUND AND NOTATION..

- Discrete set of actions : $1 \ldots A$.
- Episodic, with $\gamma = 1$.

SOME BACKGROUND AND NOTATION..

- Discrete set of actions : $1 \ldots A$.
- Episodic, with $\gamma = 1$.
- Boltzmann Policy:

$$\pi_\theta(a_t^j | s_t) = \frac{e^{q_\theta(s_t, a_t^j)}}{\sum_{i=1}^{A} e^{q_\theta(s_t, a_t^i)}}$$

BATCH OPTIMIZATION WITH EXPECTED RETURN OBJECTIVE

BATCH OPTIMIZATION WITH EXPECTED RETURN OBJECTIVE

- No exploration just maximize the expected return:

$$J(\theta) = \sum_{s} \mu(s) \left[\sum_{i=1}^{A} \pi_\theta(a^i|s) q_{\pi_\theta}(s, a^i) \right].$$

BATCH OPTIMIZATION WITH EXPECTED RETURN OBJECTIVE

- No exploration just maximize the expected return:

$$J(\theta) = \sum_s \mu(s) \left[\sum_{i=1}^{A} \pi_\theta(a^i | s) q_{\pi_\theta}(s, a^i) \right].$$

- We can approximate $q(s, a^i)$ for seen state-action pairs $q(s_t, a_t^*)$ using Monte-Carlo returns

$$q(s_t, a_t^*) \approx G_t^* = \sum_{t'=t}^{T} r_{t'}.$$

- Note: a_t^* is the action chosen at time t by the policy that generated the data.

MONTE-CARLO GRADIENT ESTIMATOR: REINFORCE

$$\nabla J(\theta) = E_{\pi_\theta}\left[G_t^* \nabla \log \pi_\theta(a_t^* | s_t)\right]$$

MONTE-CARLO GRADIENT ESTIMATOR: REINFORCE

$$\nabla J(\theta) = E_{\pi_\theta}\left[G_t^* \nabla \log \pi_\theta(a_t^* | s_t)\right]$$

- Convenient formulation but does not use all information

MONTE-CARLO GRADIENT ESTIMATOR: REINFORCE

$$\nabla J(\theta) = E_{\pi_\theta}\left[G_t^* \nabla \log \pi_\theta(a_t^* | s_t)\right]$$

- Convenient formulation but does not use all information
- High variance!

OFF-POLICY: IMPORTANCE CORRECTION

$$\nabla J(\theta) = E_{\pi_b}\left[\rho_t G_t^* \nabla \log \pi_\theta(a_t^* | s_t)\right]$$

OFF-POLICY: IMPORTANCE CORRECTION

$$\nabla J(\theta) = E_{\pi_b}\left[\rho_t G_t^* \nabla \log \pi_\theta(a_t^* | s_t)\right]$$

- Where the importance correction is denoted as

$$\rho_t = \prod_{t'=0}^{t} \frac{\pi_\theta(a_{t'}^* | s_{t'})}{\pi_b(a_{t'}^* | s_{t'})}$$

OFF-POLICY: IMPORTANCE CORRECTION

$$\nabla J(\theta) = E_{\pi_b}\left[\rho_t G_t^* \nabla \log \pi_\theta(a_t^* | s_t)\right]$$

- Where the importance correction is denoted as

$$\rho_t = \prod_{t'=0}^{t} \frac{\pi_\theta(a_{t'}^* | s_{t'})}{\pi_b(a_{t'}^* | s_{t'})}$$

- Higher variance!

MOTIVATION

MOTIVATION

- Formal motivation: Expected return can have exponentially many local minima.

MOTIVATION

- Formal motivation: Expected return can have exponentially many local minima.
- My opinion: Batch policy optimization is the simplest problem that is not well understood.

MOTIVATION

- Formal motivation: Expected return can have exponentially many local minima.
- My opinion: Batch policy optimization is the simplest problem that is not well understood.
- (Ilyas, A., et. al (2018). Are Deep Policy Gradient Algorithms Truly Policy Gradient Algorithms?)

BUT WHY A RETURN DISTRIBUTION?

- Estimating the return distribution -> Auxillary tasks
- Opens the door to many interesting objectives!

CONNECTION TO SUPERVISED LEARNING

Reinforcement learning is more general than supervised learning because:

CONNECTION TO SUPERVISED LEARNING

Reinforcement learning is more general than supervised learning because:

- Temporally extended - actions affect the next state.

CONNECTION TO SUPERVISED LEARNING

Reinforcement learning is more general than supervised learning because:

- Temporally extended - actions affect the next state.
- **Rewards/returns are known only for actions taken.**

ESTIMATING THE RETURN DISTRIBUTION

ESTIMATING THE RETURN DISTRIBUTION

- If we had full information, then

$$p(G_t^j) = \frac{e^{G_t^j}}{\sum_{i=1}^{A} e^{G_t^i}}$$

ESTIMATING THE RETURN DISTRIBUTION

- If we had full information, then

$$p(G_t^j) = \frac{e^{G_t^j}}{\sum_{i=1}^{A} e^{G_t^i}}$$

- But we do have estimates on what could-have been: $q_\theta(s, a)$.

ESTIMATING THE RETURN DISTRIBUTION

- If we had full information, then

$$p(G_t^j) = \frac{e^{G_t^j}}{\sum_{i=1}^{A} e^{G_t^i}}$$

- But we do have estimates on what could-have been: $q_\theta(s, a)$.
- How to incorporate this with observed return?

NAIVE IMPUTATION :

$$\hat{G}_t^i = c + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - c}{\beta_t}$$

NAIVE IMPUTATION :

$$\hat{G}_t^i = c + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - c}{\beta_t}$$

- Estimate return by some constant c for all actions not taken

NAIVE IMPUTATION :

$$\hat{G}_t^i = c + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - c}{\beta_t}$$

- Estimate return by some constant c for all actions not taken
- We already do this! $c = 0$

DOUBLY ROBUST ESTIMATOR :

$$\hat{G}_t^i = q_\theta(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - q_\theta(s_t, a_t^*)}{\beta_t}$$

DOUBLY ROBUST ESTIMATOR :

$$\hat{G}_t^i = q_\theta(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - q_\theta(s_t, a_t^*)}{\beta_t}$$

- Old idea, semi-recently used for policy evaluation

DOUBLY ROBUST ESTIMATOR :

$$\hat{G}_t^i = q_\theta(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - q_\theta(s_t, a_t^*)}{\beta_t}$$

- Old idea, semi-recently used for policy evaluation
- Uses more information and leverages generalization

DOUBLY ROBUST ESTIMATOR :

$$\hat{G}_t^i = q_\theta(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{G_t^* - q_\theta(s_t, a_t^*)}{\beta_t}$$

- Old idea, semi-recently used for policy evaluation
- Uses more information and leverages generalization
- **This estimator is used when** $N = 1$

DOUBLY ROBUST ADVANTAGE ESTIMATOR :

$$\hat{G}_t^i = A(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{(G_t^* - b) - A(s_t, a_t^*)}{\beta_t}$$

DOUBLY ROBUST ADVANTAGE ESTIMATOR :

$$\hat{G}_t^i = A(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{(G_t^* - b) - A(s_t, a_t^*)}{\beta_t}$$

- New idea: include baseline b to lower variance

DOUBLY ROBUST ADVANTAGE ESTIMATOR :

$$\hat{G}_t^i = A(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{(G_t^* - b) - A(s_t, a_t^*)}{\beta_t}$$

- New idea: include baseline b to lower variance
- Where $v_\theta(s_t) = \sum_{i=1}^{A} \pi_\theta(a_t^i | s_t) q_\theta(s_t, a_t^i)$ and $A(s_t, a_t^i) = q_\theta(s_t, a_t^i) - v_\theta(s_t)$

DOUBLY ROBUST ADVANTAGE ESTIMATOR :

$$\hat{G}_t^i = A(s_t, a_t^i) + \mathbb{1}_{a_t^i = a_t^*} \frac{(G_t^* - b) - A(s_t, a_t^*)}{\beta_t}$$

- New idea: include baseline b to lower variance
- Where $v_\theta(s_t) = \sum_{i=1}^{A} \pi_\theta(a_t^i | s_t) q_\theta(s_t, a_t^i)$ and $A(s_t, a_t^i) = q_\theta(s_t, a_t^i) - v_\theta(s_t)$
- **This estimator is used to compare against baseline approaches ($N > 1$)**

OBJECTIVE FUNCTIONS FROM SUPERVISED LEARNING

FORWARD KL DIVERGENCE :

$$J(\theta) = \sum_{t=0}^{T_n} \sum_{i=1}^{A} p(\hat{G}_t^i) \log \frac{p(\hat{G}_t^i)}{\pi_\theta(s_t, a_t^i)}$$

FORWARD KL DIVERGENCE :

$$J(\theta) = \sum_{t=0}^{T_n} \sum_{i=1}^{A} p(\hat{G}_t^i) \log \frac{p(\hat{G}_t^i)}{\pi_\theta(s_t, a_t^i)}$$

- Very common classification objective, convex in θ if $p(G)$ is fixed.

FORWARD KL DIVERGENCE :

$$J(\theta) = \sum_{t=0}^{T_n} \sum_{i=1}^{A} p(\hat{G}_t^i) \log \frac{p(\hat{G}_t^i)}{\pi_\theta(s_t, a_t^i)}$$

- Very common classification objective, convex in θ if $p(G)$ is fixed.
- But $p(\hat{G})$ depends on θ!

BACKWARD KL DIVERGENCE :

$$J(\theta) = \sum_{t=0}^{T_n} \sum_{i=1}^{A} \pi_\theta(s_t, a_t^i) \log \frac{\pi_\theta(s_t, a_t^i)}{p(\hat{G}_t^i)}$$

BACKWARD KL DIVERGENCE :

$$J(\theta) = \sum_{t=0}^{T_n} \sum_{i=1}^{A} \pi_\theta(s_t, a_t^i) \log \frac{\pi_\theta(s_t, a_t^i)}{p(\hat{G}_t^i)}$$

- Similar to expected return, sometimes referred to as entropy-regularized expected return.

BACKWARD KL DIVERGENCE :

$$J(\theta) = \sum_{t=0}^{T_n} \sum_{i=1}^{A} \pi_\theta(s_t, a_t^i) \log \frac{\pi_\theta(s_t, a_t^i)}{p(\hat{G}_t^i)}$$

- Similar to expected return, sometimes referred to as entropy-regularized expected return.
- **This objective is the focus of the talk!**

SOME ANALYTICAL RESULTS

BACKWARD KL DIVERGENCE INCORPORATES AN ENTROPY REGULARIZER

$$\sum_{i=1}^{A} \pi_\theta(s, a^i) \log \frac{\pi_\theta(s, a^i)}{p(\hat{G}^i)} = \underbrace{\sum_{i=1}^{A} \pi_\theta(s, a^i) \log \pi_\theta(s, a^i)}_{-\text{Entropy}} - \sum_{i=1}^{A} \pi_\theta(s, a^i) \log p(\hat{G}^i)$$

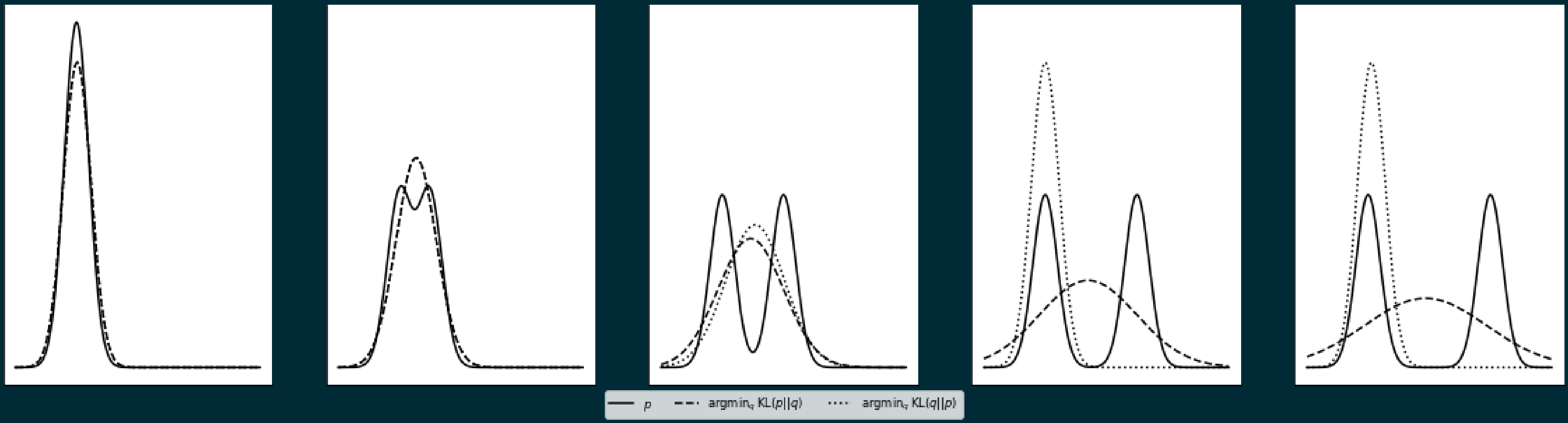
BACKWARD KL DIVERGENCE INCORPORATES AN ENTROPY REGULARIZER

$$\sum_{i=1}^{A} \pi_\theta(s, a^i) \log \frac{\pi_\theta(s, a^i)}{p(\hat{G}^i)} = \underbrace{\sum_{i=1}^{A} \pi_\theta(s, a^i) \log \pi_\theta(s, a^i)}_{-\text{Entropy}} - \sum_{i=1}^{A} \pi_\theta(s, a^i) \log p(\hat{G}^i)$$

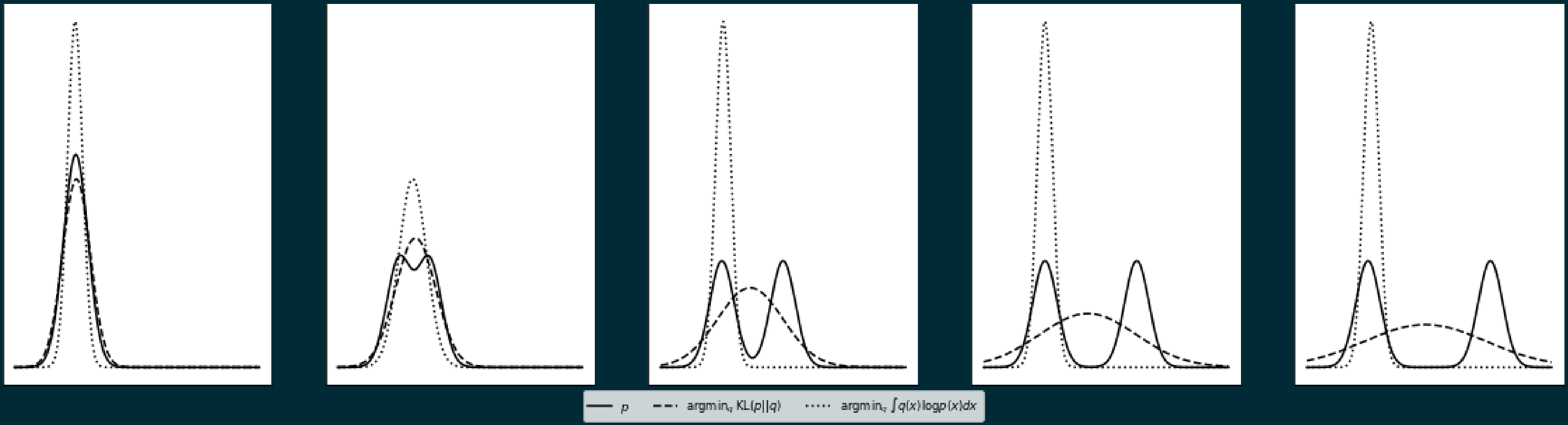
- What if we set the entropy term to zero? Only the cross-entropy term would remain:

$$\sum_{i=1}^{A} -\pi_\theta(s, a^i) \log p(\hat{G}^i) = -[\pi_\theta(a^* | s) \frac{G^* - q_\theta(s, a^*)}{\beta} + \sum_{i=1}^{A} \pi_\theta(s, a^i) q_\theta(s, a_i) - \hat{Z}_\theta]$$

FORWARD VS BACKWARD KL WITH ENTROPY TERM



FORWARD VS BACKWARD KL WITHOUT ENTROPY TERM



VARIANCE ANALYSIS

VARIANCE ANALYSIS

- Not going to hammer you with more equations..

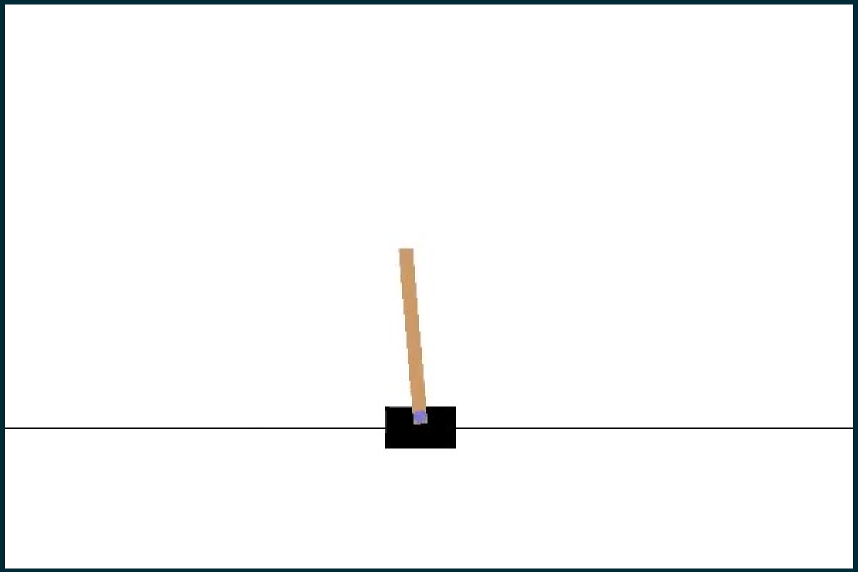
VARIANCE ANALYSIS

- Not going to hammer you with more equations..
- if $p(G)$ is fixed, the analytic gradient variance is the same except:
    - Importance corrected expected return has a $(G^*)^2$ term
    - Backward KL has a $(G^* - q(a_t^*|s_t))^2$ term.

EXPERIMENTS

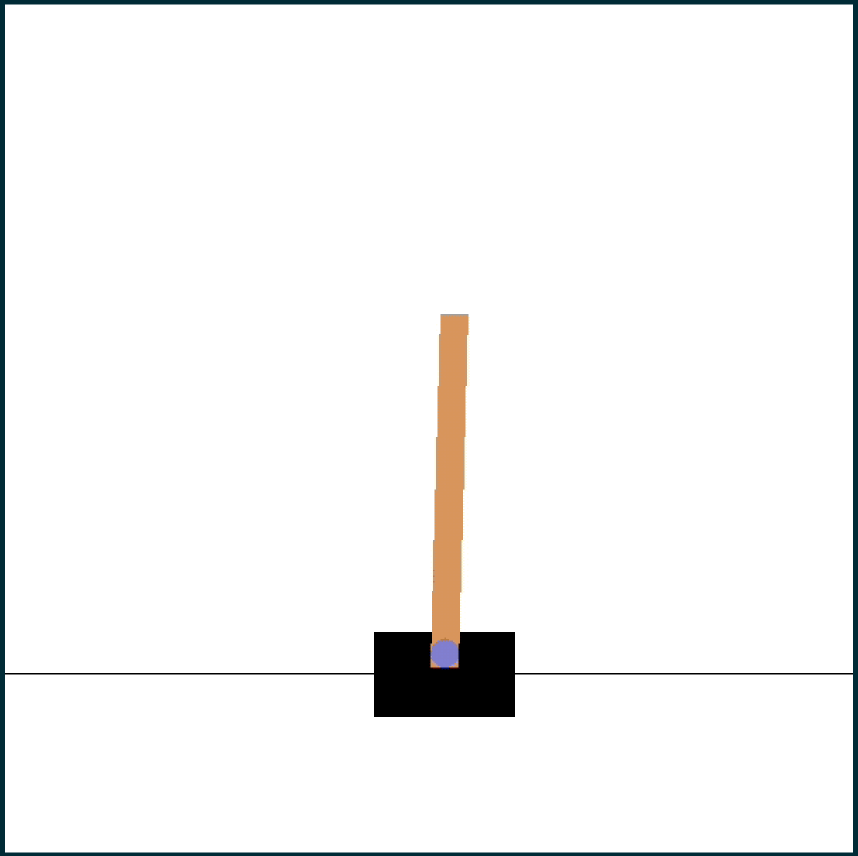
CARTPOLE

- 2 actions: move left or right
- State: (cart-position, cart-velocity, pole-angle, pole-velocity)
- Receives reward of 1 for every time step it stays upright and within a range.
- Episode terminates at $t = 200$



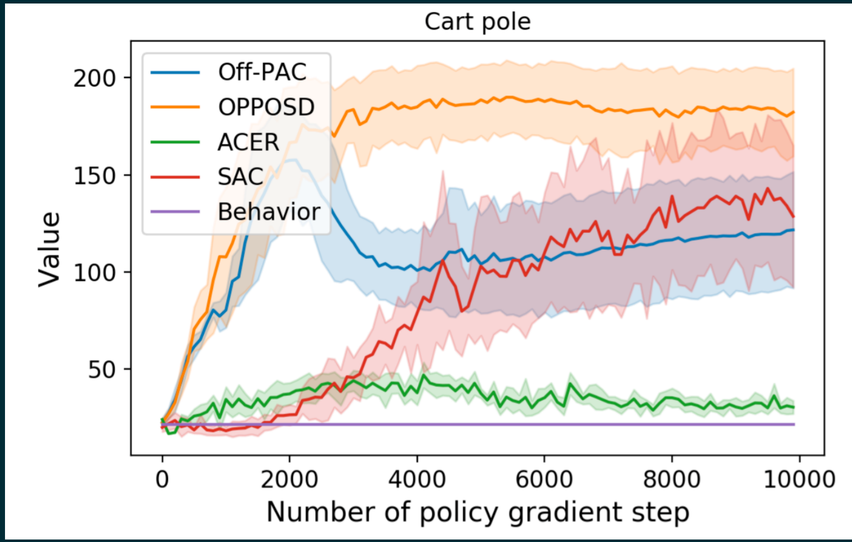
'WAY OFF-POLICY' CARTPOLE

- Uniformly random behavior policy
- Only 50 trajectories (average length of 22)



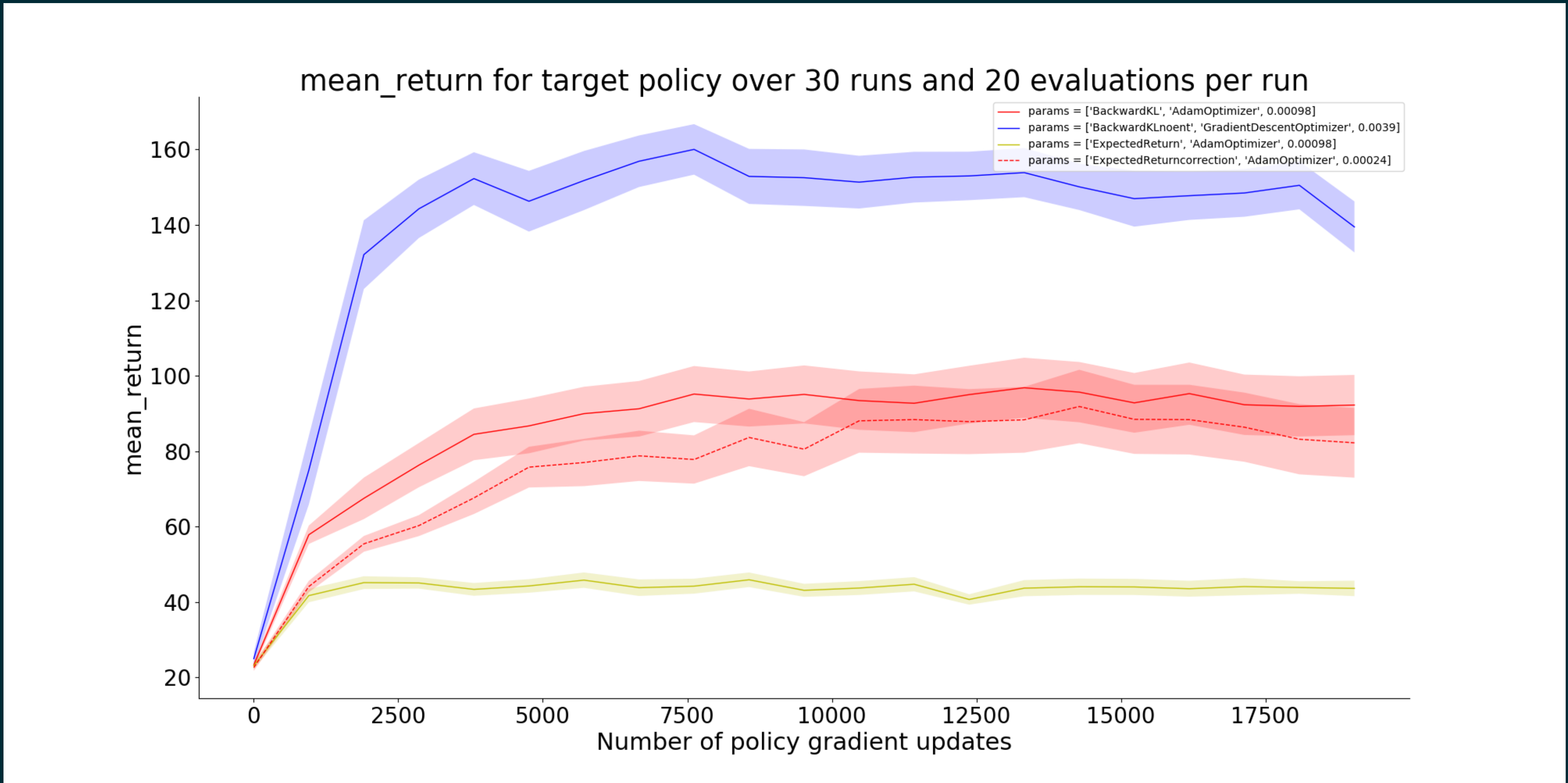
SOME PREVIOUS WORK ON THIS PROBLEM

> *"[..] is a very challenging data set for off-policy policy optimization methods to learn from as this policy does not attain the desired upright configuration for any prolonged period of time."*

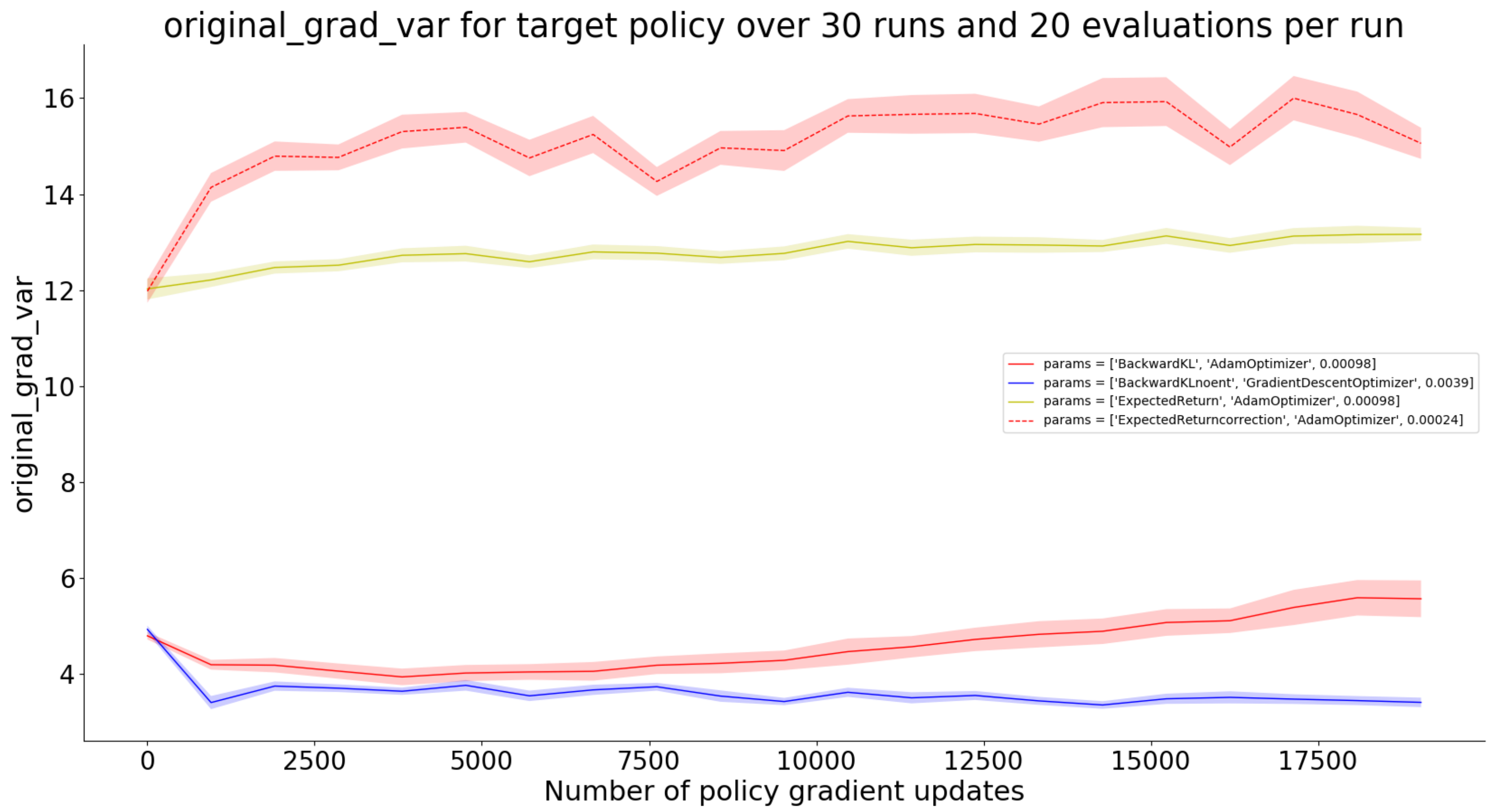


- (Liu, Y. et al. (2019). Off-policy policy gradient with state distribution correction)

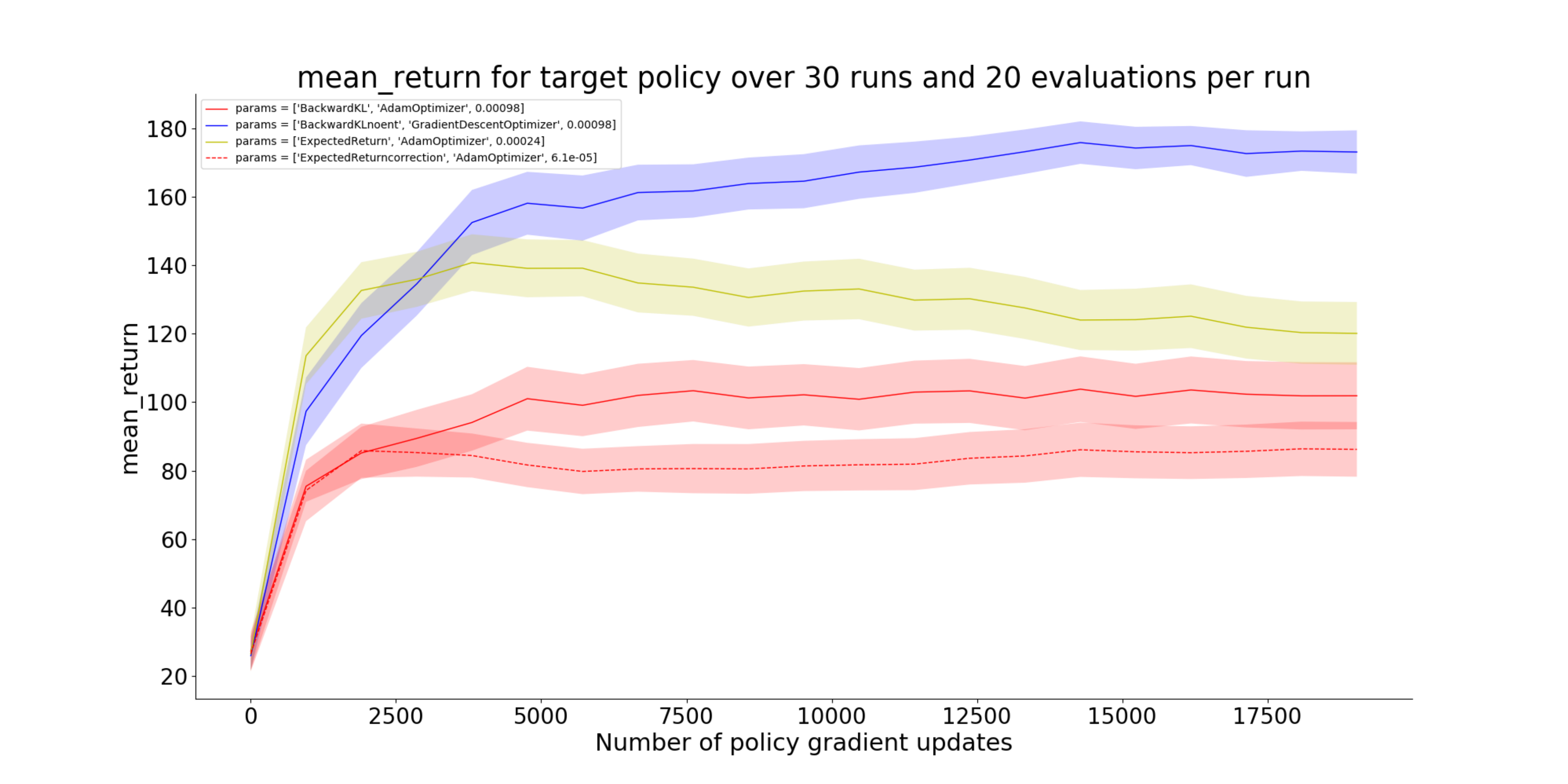
1 TRAJECTORY PER BATCH WITH NO BASELINE :



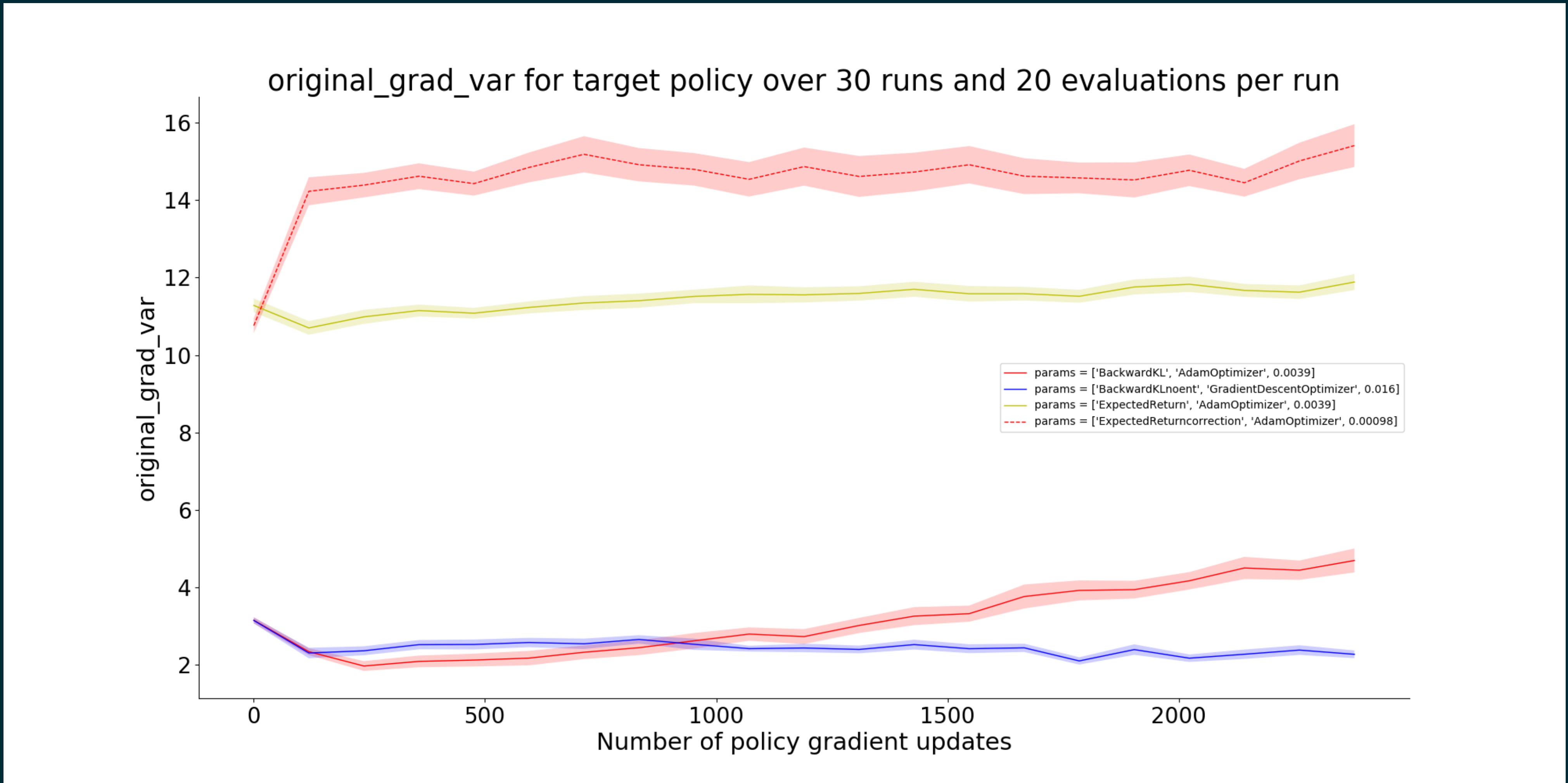
1 TRAJECTORY PER BATCH WITH NO BASELINE VARIANCE:



8 TRAJECTORY PER BATCH WITH TIME-DEPENDENT BASELINE :



8 TRAJECTORY PER BATCH WITH TIME-DEPENDENT BASELINE VARIANCE:



CONCLUSIONS

- ~~Deep~~ Reinforcement learning doesn't work **on hard problems** ~~yet~~

ADDENDUM

- Of course, that's what makes them hard.

SUMMARY

SUMMARY

- The return distribution
    - **Different from distributional RL**
    - How to extend to continuous actions?
    - Is it linked with choice in objective?

SUMMARY

- The return distribution
    - **Different from distributional RL**
    - How to extend to continuous actions?
    - Is it linked with choice in objective?
- New objectives with the return distribution
    - Many choices: unclear which objective is optimal.
    - Strong evidence that expected return is not always best.

SUMMARY

- The return distribution
  - **Different from distributional RL**
  - How to extend to continuous actions?
  - Is it linked with choice in objective?
- New objectives with the return distribution
  - Many choices: unclear which objective is optimal.
  - Strong evidence that expected return is not always best.
- Future work
  - Have been avoiding bootstrapping: does this alleviate or worsen instabilities?

THANK YOU.

QUESTIONS?