Lecture 13: Further Contemporary RL Algorithms

Oliver Wallscheid

Table of contents

Reinterpreting the stochastic policy gradient (1)

- In the following we will focus on stochastic policies $\pi(\boldsymbol{u}|\boldsymbol{x})$.
- First, we rewrite the performance metric (11.7) to obtain

$$J_\pi = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_k \right]. \tag{13.1}$$

- Using the advantage $a_\pi(\boldsymbol{x}, \boldsymbol{u}) = q_\pi(\boldsymbol{x}, \boldsymbol{u}) - v_\pi(\boldsymbol{x})$ we can calculate the performance of an updated policy $\boldsymbol{\pi} \rightarrow \tilde{\boldsymbol{\pi}}$[1]:

$$J_{\tilde{\pi}} = J_\pi + \int_{\mathcal{X}} p^{\tilde{\pi}}(\boldsymbol{x}) \int_{\mathcal{U}} \tilde{\pi}(\boldsymbol{u}|\boldsymbol{x}) a_\pi(\boldsymbol{x}, \boldsymbol{u}). \tag{13.2}$$

- While for finite MDPs, the policy improvement theorem guaranteed $J_{\tilde{\pi}} \geq J_\pi$ for each policy update, there might be some states where $\int_{\mathcal{U}} \tilde{\pi}(\boldsymbol{u}|\boldsymbol{x}) a_\pi < 0$ for continuous MDPs using function approximation.

[1] proof from: S. Kakade and J. Langford, *Approximately optimal approximate reinforcement learning*, ICML, vol. 2, pp 267-274, 2002

Reinterpreting the stochastic policy gradient (2)

- For easier calculation, we introduce a local approximation to (13.2)

$$\mathcal{L}_\pi(\tilde{\pi}) = J_\pi + \int_{\mathcal{X}} p^\pi(\boldsymbol{x}) \int_{\mathcal{U}} \tilde{\pi}(\boldsymbol{u}|\boldsymbol{x}) a_\pi(\boldsymbol{x}, \boldsymbol{u}) \tag{13.3}$$

  where $p^\pi(\boldsymbol{x})$ is used instead of $p^{\tilde{\pi}}(\boldsymbol{x})$, i.e., neglecting the state distribution change due to a policy update.
- For any parametrized and differentiable policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})$, it can be shown that

$$\begin{aligned} \mathcal{L}(\pi_{\boldsymbol{\theta}_0}) &= J(\pi_{\boldsymbol{\theta}_0}), \\ \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\pi_{\boldsymbol{\theta}_0}}(\pi_{\boldsymbol{\theta}})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} &= \nabla_{\boldsymbol{\theta}} J(\pi_{\boldsymbol{\theta}})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} \end{aligned} \tag{13.4}$$

  for any initial parameter set $\boldsymbol{\theta}_0$.
- For a sufficiently small step size, improving $\mathcal{L}_{\pi_{\boldsymbol{\theta}_0}}$ will also improve J.

However, we do not know how much the actual stochastic policy will change while moving through the parameter space. Hence, we do not have a good decision basis to choose the policy gradient step size.

Adding a trust region constraint (1)

- From the previous discussion it can be concluded that we want a metric describing how much a policy is changed in the action space when updating the policy in the parameter space.
- Against this background, we make use of the Kullback-Leibler divergence (also called relative entropy)

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) \mathrm{d}x \tag{13.5}$$

  defined for continuous distributions P and Q with their probability densities p and q.
- Example: for two multivariate Gaussian distributions of equal dimensions d, with means $\boldsymbol{\mu}_0, \boldsymbol{\mu}_1$ and with (non-singular) covariance matrix $\boldsymbol{\Sigma}_0, \boldsymbol{\Sigma}_1$ we receive

$$D_{\mathrm{KL}}\left(\mathcal{N}_0 \parallel \mathcal{N}_1\right) = \frac{1}{2} \Big(\mathrm{tr}\left(\boldsymbol{\Sigma}_1^{-1} \boldsymbol{\Sigma}_0\right) + \left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0\right)^{\mathsf{T}} \boldsymbol{\Sigma}_1^{-1} \left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0\right) - d + \ln \left(\frac{\det \boldsymbol{\Sigma}_1}{\det \boldsymbol{\Sigma}_0} \right) \Big).$$

Adding a trust region constraint (2)

- The trust region policy optimization (TRPO) updates the policy parameters while constraining the KL divergence between the new and the old policy distribution:

$$\begin{aligned} &\max_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}), \\ \text{s.t.} \quad &\overline{D}_{\mathrm{KL}}(\boldsymbol{\theta}_k, \boldsymbol{\theta}) \leq \kappa \end{aligned} \tag{13.6}$$

with

$$\overline{D}_{\mathrm{KL}}(\boldsymbol{\theta}_k, \boldsymbol{\theta}) = \overline{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}_k}, \pi_{\boldsymbol{\theta}}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}_k}(\cdot|\boldsymbol{X}) \parallel \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{X})) \right].$$

- Hence, we want to limit the average KL divergence w.r.t. the states visited by the old policy.
- The constraint κ is a TRPO hyperparameter (typically $\kappa << 1$).
- Although (13.6) does not provide any formal convergence guarantee, we at least have a link between changes in the parameter and policy distribution space. Therefore, we can use this tool to prevent erratic policy changes.

Smooth policy updates via TRPO

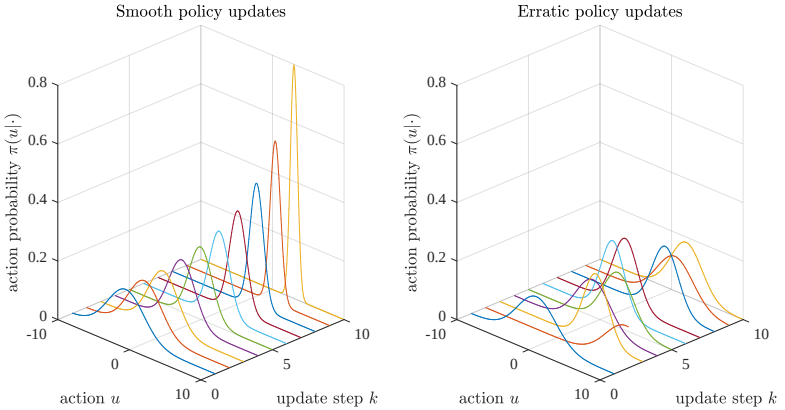

Fig. 13.1: Simplified representation of the policy evolution for a scalar action given some fixed state. Left: TRPO-style updates finding the optimal action with increasing probability. Right: Unmonitored policy distributions not converging towards an optimal policy ('policy chattering').

Sample-based objective and constraint estimation (1)

- To actually solve (13.6) we will make use of samplings from Monte Carlo rollouts.
- Expanding the objective yields

$$\max_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} J_{\pi_k} + \int_{\mathcal{X}} p^{\pi_k}(\boldsymbol{x}) \int_{\mathcal{U}} \pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x}) a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}). \tag{13.7}$$

- The first term J_{π_k} can be dropped, since it is irrelevant for the optimization result (constant).
- Using samples we can approximate $\int_{\mathcal{X}} p^{\pi_k}(\boldsymbol{x}) \approx \frac{1}{1-\gamma} \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}}[\boldsymbol{X}]$.
- Moreover, $\int_{\mathcal{U}} \pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x}) a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}) \approx \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})} a_{\pi_k}(\boldsymbol{X}, \boldsymbol{U}) \right]$ is also approximated applying importance sampling based on data from the old policy.
- Hence, the sampled objective is

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})} a_{\pi_k}(\boldsymbol{X}, \boldsymbol{U}) \right]. \tag{13.8}$$

Sample-based objective and constraint estimation (2)

- Applying the previous sample-based estimation we obtain

$$\begin{aligned} \boldsymbol{\theta}_{k+1} &= \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})} a_{\pi_k}(\boldsymbol{X}, \boldsymbol{U}) \right], \\ \text{s.t.} \quad & \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}_k}(\cdot|\boldsymbol{X}) \parallel \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{X})) \right] \leq \kappa. \end{aligned} \tag{13.9}$$

- Hence, we have a three-step procedure for each TRPO update:

1. Use Monte Carlo simulations based on the old policy to obtain data.
2. Use the data to construct (13.9).
3. Solve the constrained optimization problem to update the policy parameter vector.

Solving (13.9) is generally a nonlinear optimization problem. The original TRPO implementation uses a local objective and constraint approximation together with conjugate gradient and line search algorithms. However, many other constrained-nonlinear solvers are also applicable.

Generalized advantage estimation

- Having data $\langle \boldsymbol{x}, \boldsymbol{u}, r, \boldsymbol{x}' \rangle$ in $\mathcal{D}$ from a Monte Carlo rollout available, an imporant problem is to estimate $a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u})$ in (13.9).
- A particular suggestion in the TRPO context is to use a generalized advantage estimator (GAE) [1] defined as

$$\hat{a}_k^{(\gamma,\lambda)} = \sum_{i=0}^{\infty} (\gamma\lambda)^i \delta_{k+i}. \tag{13.10}$$

- Here, $\delta_k = r_k + \gamma v(\boldsymbol{x}_{k+1}) - v(\boldsymbol{x}_k)$ is a single advantage sample.
- Hence, the GAE is the exponentially-weighted average of the discounted advantage samples with an additional weighting λ.
- Similar formulation compared to TD(λ) but the estimator's target is the advantage.
- The choice of $(\gamma\lambda)$ trade-offs the bias and variance of the estimator.

[1] cf. J. Schulmann et al., *High Dimensional Continuous Control Using Generalized Advantage Estimation*, https://arxiv.org/abs/1506.02438, 2015

TRPO summary

The TRPO's key facts are:

- The TRPO constrains policy distribution changes when updating the policy parameters (for stochastic policies and on-policy learning).
- The objective is to enable a monotonically improving learning process.
- Using trust regions, erratic policy updates should be prevented.

The TRPO's main hurdles are:

- Constructing the objective function and constraint requires Monte Carlo rollouts (time consuming, data inefficient).
- When the sampled optimization problem is set up, a nonlinear and constrained optimization step is required (no simple policy gradient, computational costly).

We will not provide any specific TRPO implementation suggestion at this point, since this is rather cumbersome. Instead we will move forward to a similar algorithm which is pursuing the same goal (prevent erratic policy changes) with a much simpler implementation.

Table of contents

Background and motivation

- The upcoming proximal policy optimization (PPO) algorithm tries to mimic the constrained TRPO problem

$$\begin{aligned} \boldsymbol{\theta}_{k+1} &= \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})} a_{\pi_k}(\boldsymbol{X}, \boldsymbol{U}) \right], \\ \text{s.t.} \quad & \mathbb{E}_{\pi_{\boldsymbol{\theta}_k}} \left[D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}_k}(\cdot|\boldsymbol{X}) \parallel \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{X})) \right] \leq \kappa. \end{aligned}$$

based on related unconstrained problems.

- Hence, the objective will be reformulated to incorporate mechanisms preventing excessively large variations of the policy distribution during a parameter update (leading to an updated policy with sufficient proximity to the old one).
- Moreover, PPO incorporates two variants which we will discuss:
  1. Clipping the surrogate objective,
  2. Adaptive tuning of a KL-associated penalty coefficient.

Clipped surrogate objective

- The first approach is based on the following clipped objective:

$$\mathbb{E}_{\pi_{\boldsymbol{\theta}_k}}\left[\min\left\{\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})}a_{\pi_k}(\boldsymbol{X},\boldsymbol{U}), \text{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})}, 1-\epsilon, 1+\epsilon\right)a_{\pi_k}(\boldsymbol{X},\boldsymbol{U})\right\}\right]. \tag{13.11}$$

- Above, $\epsilon < 1$ is a PPO hyperparameter serving as a regularizer.
- The first element of $\min\{\cdot\}$ is the previous TPRO objective.
- The second element of $\min\{\cdot\}$ modifies the surrogate objective by clipping the importance sampling ratio $\pi_{\boldsymbol{\theta}}/\pi_{\boldsymbol{\theta}_k}$.
- The latter should remove the incentive for moving the importance sampling ratio outside of the interval $[1-\epsilon, 1+\epsilon]$.
- The modified objective is therefore a lower bound of the unclipped TRPO objective.

Clipped surrogate objective: positive advantage

- Consider a single sample $(\boldsymbol{x}, \boldsymbol{u})$ with a positive advantage $a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u})$:

$$\max_{\boldsymbol{\theta}} \min \left\{ \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})} a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}), \operatorname{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})}, 1-\epsilon, 1+\epsilon \right) a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}) \right\}.$$

- Because the advantage is positive, the objective will increase if the action becomes more likely, i.e., if $\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})$ increases.
- If $\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x}) > (1+\epsilon)\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})$ the clipping becomes active.
- Hence, the objective reduces to

$$\max_{\boldsymbol{\theta}} \min \left\{ \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})}, 1+\epsilon \right\} a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}).$$

- Due to the $\min\{\cdot\}$ operator, the entire objective is therefore limited to $(1+\epsilon)a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u})$.
- Interpretation: the new policy does not benefit from going very away from the old policy distribution.

Clipped surrogate objective: negative advantage

- Consider a single sample $(\boldsymbol{x}, \boldsymbol{u})$ with a negative advantage $a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u})$:

$$\max_{\boldsymbol{\theta}} \min \left\{ \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})} a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}), \operatorname{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})}, 1-\epsilon, 1+\epsilon \right) a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}) \right\}.$$

- Because the advantage is negative, the objective will increase if the action becomes less likely, i.e., if $\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})$ decreases.
- If $\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x}) < (1-\epsilon)\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})$ the clipping becomes active.
- Hence, the objective reduces to

$$\max_{\boldsymbol{\theta}} \max \left\{ \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{u}|\boldsymbol{x})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{u}|\boldsymbol{x})}, 1-\epsilon \right\} a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u}).$$

- Due to the $\max\{\cdot\}$ operator, the entire objective is limited to $(1-\epsilon)a_{\pi_k}(\boldsymbol{x}, \boldsymbol{u})$.

Adaptive KL penalty

- The second PPO variant makes use of the following KL-penalized objective

$$\mathbb{E}_{\pi_{\boldsymbol{\theta}_k}}\left[\frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{U}|\boldsymbol{X})}{\pi_{\boldsymbol{\theta}_k}(\boldsymbol{U}|\boldsymbol{X})} a_{\pi_k}(\boldsymbol{X},\boldsymbol{U}) - \beta D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}_k}(\cdot|\boldsymbol{X}) \parallel \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{X}))\right]. \tag{13.12}$$

- Transfers the KL-based constraint into a penalty for large policy distribution changes.
- The parameter β weights the penalty against the policy improvement.
- The original PPO implementation suggests an adaptive tuning of β w.r.t. the sampled average KL divergence $\overline{D}_{\mathrm{KL}}(\boldsymbol{\theta}_k, \boldsymbol{\theta})$ estimated from previous experience

$$\begin{aligned} \overline{D}_{\mathrm{KL}}(\boldsymbol{\theta}_k, \boldsymbol{\theta}) < \overline{D}^*_{\mathrm{KL}}: &\quad \beta \leftarrow \beta/2, \\ \overline{D}_{\mathrm{KL}}(\boldsymbol{\theta}_k, \boldsymbol{\theta}) > \overline{D}^*_{\mathrm{KL}}: &\quad \beta \leftarrow \beta \cdot 2. \end{aligned} \tag{13.13}$$

with some target value of the KL divergence $\overline{D}^*_{\mathrm{KL}}$ (additional hyperparameter).

Algo. implementation: PPO

input: diff. stochastic policy fct. $\pi(\boldsymbol{u}|\boldsymbol{x},\boldsymbol{\theta})$ and value fct. $\hat{v}(\boldsymbol{x},\boldsymbol{w})$
parameter: step sizes $\{\alpha_w, \alpha_\theta\} \in \{\mathbb{R}|0 < \alpha\}$
init: weights $\boldsymbol{w} \in \mathbb{R}^\zeta$ and $\boldsymbol{\theta} \in \mathbb{R}^d$ arbitrarily, memory $\boldsymbol{\mathcal{D}}$
for $j = 1, 2, \ldots,$ *(sub-)episodes* **do**
  initialize $\boldsymbol{x}_0$ (if new episode);
  collect a set of tuples $\langle \boldsymbol{x}_k, \boldsymbol{u}_k, r_{k+1}, \boldsymbol{x}_{k+1} \rangle$ by a rollout using $\pi(\boldsymbol{u}|\boldsymbol{x},\boldsymbol{\theta}_j)$;
  store them in $\boldsymbol{\mathcal{D}}$;
  estimate the advantage $\hat{a}_{\pi_j}(\boldsymbol{x},\boldsymbol{u})$ based on $\hat{v}(\boldsymbol{x},\boldsymbol{w}_j)$ and $\boldsymbol{\mathcal{D}}$ (e.g., GAE);
  $\boldsymbol{\theta}_{j+1} \leftarrow$ policy gradient update based on the PPO variant (13.11) or (13.12);
  $\boldsymbol{w}_{j+1} \leftarrow$ minimizing the mean-squared TD errors using $\boldsymbol{\mathcal{D}}$ (critic);
  delete entries in $\boldsymbol{\mathcal{D}}$ (due to on-policy learning);

Algo. 13.1: Proximal policy optimization (output: parameter vectors $\boldsymbol{\theta}^*$ for $\pi^*(\boldsymbol{u}|\boldsymbol{x},\boldsymbol{\theta}^*)$) and $\boldsymbol{w}^*$ for $\hat{v}^*(\boldsymbol{x},\boldsymbol{w}^*)$)

Some PPO remarks

- Clipping the surrogate objective (13.11) was reported to achieve higher performances than the KL penalty (13.12).[1]
- Like TRPO, PPO is an on-policy algorithm. Hence, the memory $\mathcal{D}$ is not a rolling replay buffer (cf. off-policy algorithms like DQN, DDPG or TD3) but a rollout buffer using one fixed policy.
- These rollouts are likely to result in an increased sample demand either using a simulator or a real experiment.

Although PPO is derived from a TRPO background pursuing monotonically increasing policy performance, its realization is based on multiple heuristics and approximations. Hence, there is no guarantee on achieving this goal and the specific performance of the PPO algorithm must be evaluated empirically given a certain application.

[1] cf. original PPO paper results by J. Schulman et al., *Proximal Policy Optimization Algorithms*, https://arxiv.org/abs/1707.06347, 2017

Exemplary performance comparison

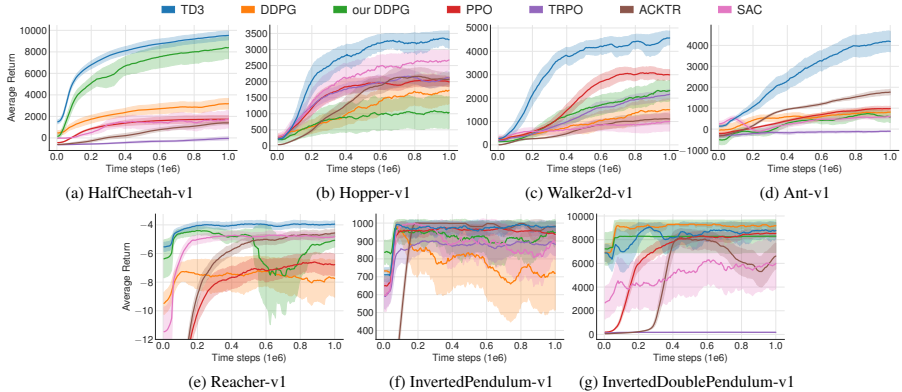

Fig. 13.2: Learning curves for OpenAI Gym continuous control tasks. The shaded region represents half a standard deviation of the average evaluation over ten trials (source: S. Fujimoto et al., *Addressing Function Approximation Error in Actor-Critic Methods*, 2018).

Algorithmic outlook: other contemporary model-free algorithms (1)

The selection of algorithms appears endless:

- DQN variants such as
  - (Prioritized) dueling DQN
  - Noisy DQN
  - Distributional DQN
- Rainbow (combining multiple DQN extensions)
- Soft actor-critic (SAC)
- Actor critic using Kronecker-factored trust region (ACKTR)
- Asynchronous advantage actor-critic (A3C)
- ....

Remarks:

- You have already learned the basic building blocks in order to make yourself familiar with any value-/policy-based model-free RL approach.
- Use this knowledge!
- Focus on primary scientific literature for self-studying and not on unreliable sources!

Algorithmic outlook: other contemporary model-free algorithms (2)

Algorithm collections with tutorial-style documentation:

- Intel Reinforcement Learning Coach
- OpenAI Spinning Up

Algorithm collections with decent application-oriented documentation:

- RLlib (Ray)
- Stable Baselines3
- Acme
- Garage
- Google Dopamine
- Tensorforce
- TF-Agents
- ...

Summary: what you've learned today

- Trust region policy optimization (TRPO) pursues monotonically increasing policy performance by limiting policy distribution changes.
- This results in a nonlinear constrained optimization problem adding computational complexity (no simple policy gradients).
- Proximal policy optimization (PPO) converts the TRPO idea into an unconstrained optimization problem by a modified objective. Likewise, the PPO's objective is to prevent erratic policy distribution changes.