Lecture 10: Policy Gradient Methods

Friday, January 21, 2022

Reinforcement Learning, Winter Term 2021/22

Joschka Boedecker, Gabriel Kalweit, Jasper Hoffmann

Neurorobotics Lab
University of Freiburg

Lecture Overview

Lecture Overview

Recap: Eligibility Traces and λ-return

Eligibility traces unify and generalize TD and Monte Carlo methods

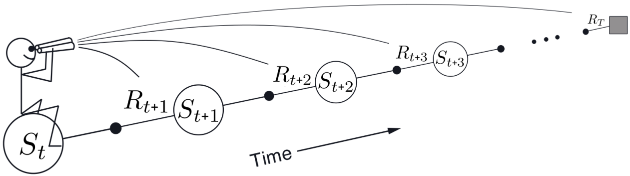
- MC methods at one end ($\lambda = 1$) and one-step TD methods at the other ($\lambda = 0$)
- almost any temporal-difference (TD) method can be combined with eligibility traces to (maybe) learn more efficiently

λ-return

- For infinite control tasks: $G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}$
- For episodic control tasks:

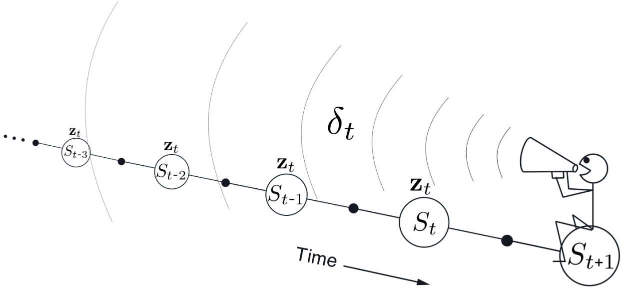
$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t-1} G_t$$

Recap: Forward View



- Update value function towards the λ-return
- Forward-view looks into the future to compute G_t^λ
- Like MC, can only be computed from complete episodes

Recap: Backward View



- look at the current TD error δ_t
- assign it backward to each prior state according to how much that state contributed to the current eligibility trace at that time

Recap: Eligibility Traces

- Eligibility Traces assign credit to components of the weight vector according to their contribution to state valuations
- They combine heuristics of *Frequency* and *Recency* (implemented by a λ-decay)
- With function approximation, the eligibility trace is a vector $\mathbf{z}_t \in \mathbb{R}$, initialized by $\mathbf{z}_{-1} = \mathbf{0}$ and incremented on each time step by:

$$\mathbf{z}_t = \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t, \mathbf{w}_t), \quad 0 \leq t \leq T,$$

where λ is called trace-decay parameter.

Lecture Overview

Policy Gradient Methods

- Up to this point, we represented a model or a value function by some parameterized function approximator and extracted the policy implicitly
- Today, we are going to talk about *Policy Gradient Methods*: methods which consider a parameterized *policy*

$$\pi(a|s, \boldsymbol{\theta}) = \mathsf{Pr}\{A_t = a|S_t = s, \boldsymbol{\theta}_t = \boldsymbol{\theta}\},$$

  with parameters $\boldsymbol{\theta}$
- Policy Gradient Methods are able to represent stochastic policies and scale naturally to very large or continuous action spaces

Policy Gradient Methods

- We update these parameters based on the gradient of some performance measure $J(\boldsymbol{\theta})$ that we want to maximize, i.e. via *gradient ascent*:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \widehat{\nabla J(\boldsymbol{\theta}_t)},$$

  where $\widehat{\nabla J(\boldsymbol{\theta}_t)} \in \mathbb{R}^d$ is a stochastic estimate whose expectation approximates the gradient of the performance measure w.r.t. $\boldsymbol{\theta}_t$

Score Function

- Likelihood ratios exploit the following identity:

$$\begin{aligned}
\overbrace{\nabla_{\boldsymbol{\theta}} \pi(a|s,\boldsymbol{\theta})}^{\text{We want the expectation of this}} &= \pi(a|s,\boldsymbol{\theta}) \frac{\nabla_{\boldsymbol{\theta}} \pi(a|s,\boldsymbol{\theta})}{\pi(a|s,\boldsymbol{\theta})} \\
&= \underbrace{\pi(a|s,\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log \pi(a|s,\boldsymbol{\theta})}_{\text{Easy to take the expectation because we can sample from } \pi\text{!}}
\end{aligned}$$

- $\nabla_{\boldsymbol{\theta}} \log \pi(a|s,\boldsymbol{\theta})$ is called the **score function**

Score Function: Example

Consider a Gaussian policy, where the mean is a linear combination of state features: $\pi(a|s, \boldsymbol{\theta}) \sim \mathcal{N}(s^\top \boldsymbol{\theta}, \sigma^2)$, i.e.

$$\pi(a|s, \boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2} \frac{(s^\top \boldsymbol{\theta} - a)^2}{\sigma^2})$$

Exercise (5min)

Derive the score function.

Score Function: Example

Consider a Gaussian policy, where the mean is a linear combination of state features: $\pi(a|s,\boldsymbol{\theta}) \sim \mathcal{N}(s^\top\boldsymbol{\theta}, \sigma^2)$, i.e.

$$\pi(a|s,\boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2}\frac{(s^\top\boldsymbol{\theta} - a)^2}{\sigma^2})$$

Solution

The log yields

$$\log \pi(a|s,\boldsymbol{\theta}) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(s^\top\boldsymbol{\theta} - a)^2$$

and the gradient

$$\nabla_{\boldsymbol{\theta}} \log \pi(a|s,\boldsymbol{\theta}) = -\frac{1}{2\sigma^2}(s^\top\boldsymbol{\theta} - a)2s = \frac{(a - s^\top\boldsymbol{\theta})s}{\sigma^2}.$$

Policy Gradient Theorem

Policy Objective Functions:

- For episodic problems we define performance as: $J(\boldsymbol{\theta}) = \eta(\pi_{\boldsymbol{\theta}}) = \mathbb{E}_{s_0 \sim \rho_0}[v_{\pi_{\boldsymbol{\theta}}}(s_0)]$
- For continuing problems: $J(\boldsymbol{\theta}) = \sum_s \mu(s) v_{\pi_{\boldsymbol{\theta}}}(s)$

Policy Gradient Theorem

For any differentiable policy $\pi(a|s, \boldsymbol{\theta})$ and any of the above policy objective functions, the policy gradient is:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\pi}[\nabla_{\boldsymbol{\theta}} \log \pi(a|s, \boldsymbol{\theta}) q_{\pi}(s, a)]$$

Reminder: $v_{\pi_{\boldsymbol{\theta}}} = \sum_a \pi(a|s) q_{\pi}(s, a)$

Policy Gradient Theorem

Proof (episodic case):

$$
\begin{aligned}
\nabla v_\pi(s) &= \nabla \left[\sum_a \pi(a|s) q_\pi(s,a) \right], \quad \text{for all } s \in \mathcal{S} \\
&= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla q_\pi(s,a) \right] \ \text{(product rule of calculus)} \\
&= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla \sum_{s',r} p\left(s', r | s, a\right) \left(r + v_\pi\left(s'\right)\right) \right] \\
&= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p\left(s' | s, a\right) \nabla v_\pi\left(s'\right) \right] \\
&= \sum_a \Bigg[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p\left(s' | s, a\right) \\
&\qquad \sum_{a'} [\nabla \pi(a'|s') q_\pi\left(s', a'\right) + \pi\left(a'|s'\right) \sum_{s''} p\left(s''|s', a'\right) \nabla v_\pi(s'')] \Bigg] \\
&= \sum_{x \in \mathcal{S}} \sum_{k=0}^{\infty} \Pr(s \to x, k, \pi) \sum_a \nabla \pi(a|x) q_\pi(x,a)
\end{aligned}
$$

Policy Gradient Theorem

Proof (episodic case):

$$
\begin{aligned}
\nabla J(\boldsymbol{\theta}) &= \nabla v_\pi\left(s_0\right) \\
&= \sum_s\left(\sum_{k=0}^{\infty} \operatorname{Pr}\left(s_0 \rightarrow s, k, \pi\right)\right) \sum_a \nabla \pi(a \mid s) q_\pi(s, a) \\
&= \sum_s \eta(s) \sum_a \nabla \pi(a \mid s) q_\pi(s, a) \\
&= \sum_{s^{\prime}} \eta\left(s^{\prime}\right) \sum_s \frac{\eta(s)}{\sum_{s^{\prime}} \eta\left(s^{\prime}\right)} \sum_a \nabla \pi(a \mid s) q_\pi(s, a) \\
&= \sum_{s^{\prime}} \eta\left(s^{\prime}\right) \sum_s \mu(s) \sum_a \nabla \pi(a \mid s) q_\pi(s, a) \\
&\propto \sum_s \mu(s) \sum_a \nabla \pi(a \mid s) q_\pi(s, a)
\end{aligned}
$$

(Q.E.D.)

Lecture Overview

REINFORCE

- REINFORCE: Monte Carlo Policy Gradient
- Builds upon Monte Carlo returns as an unbiased sample of q_π
- However, therefore REINFORCE can suffer from high variance

REINFORCE

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Variance Reduction with Baselines

- Vanilla REINFORCE provides *unbiased* estimates of the gradient $\nabla J(\theta)$, but it can suffer from high variance
- Goal: reduce variance while remaining unbiased
- Observation: we can generalize the policy gradient theorem by including an arbitrary *action-independent baseline* $b(s)$, i.e.

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &\propto \sum_s \mu(s) \sum_a (q_\pi(s,a) - b(s)) \nabla \pi(a|s) \\
&= \sum_s \mu(s) \left[\sum_a q_\pi(s,a) \nabla \pi(a|s) - b(s) \underbrace{\nabla \sum_a \pi(a|s)}_{=0} \right] \\
&= \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi(a|s)
\end{aligned}
$$

- Baselines can reduce the variance of gradient estimates significantly!

Variance Reduction with Baselines

- A constant value can be used as a baseline
- The state-value function can be used as a baseline

Question

Is the Q-function a valid baseline?

Question

Assume an approximation of the state-value function as a baseline. Is REINFORCE then biased?

REINFORCE with Baselines

Indeed, an estimate of the state value function, $\hat{v}(S_t, w)$, is a very reasonable choice for $b(s)$:

REINFORCE with Baseline (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Algorithm parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ $\quad (G_t)$
    $\delta \leftarrow G - \hat{v}(S_t,\mathbf{w})$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Lecture Overview

Actor-Critic Methods

- Methods that learn approximations to both policy and value functions are called actor-critic methods
  actor: learned policy
  critic: learned value function (usually a state-value function)

Question: Is REINFORCE-with-baseline considered as an actor-critic method?

Actor-Critic Methods

- REINFORCE-with-baseline is unbiased, but tends to learn slowly and has high variance
- To gain from advantages of TD methods we use actor-critic methods with a bootstrapping critic

One-step actor-critic methods

Replace the full return of REINFORCE with one-step return as follows:

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha \left(G_{t:t+1} - \hat{v}(S_t, \boldsymbol{w})\right) \frac{\nabla \pi(A_t|S_t, \boldsymbol{\theta}_t)}{\pi(A_t|S_t, \boldsymbol{\theta}_t)} \\
&= \boldsymbol{\theta}_t + \alpha \left(R_{t+1} + \gamma \hat{v}(S_{t+1}, \boldsymbol{w}) - \hat{v}(S_t, \boldsymbol{w})\right) \frac{\nabla \pi(A_t|S_t, \boldsymbol{\theta}_t)}{\pi(A_t|S_t, \boldsymbol{\theta}_t)} \\
&= \boldsymbol{\theta}_t + \alpha \delta_t \frac{\nabla \pi(A_t|S_t, \boldsymbol{\theta}_t)}{\pi(A_t|S_t, \boldsymbol{\theta}_t)}
\end{aligned}$$

Actor-Critic Methods

One-step Actor–Critic (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g., to $\mathbf{0}$)
Loop forever (for each episode):
    Initialize S (first state of episode)
    $I \leftarrow 1$
    Loop while S is not terminal (for each time step):
        $A \sim \pi(\cdot|S, \boldsymbol{\theta})$
        Take action A, observe S', R
        $\delta \leftarrow R + \gamma \hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S,\mathbf{w})$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} I \delta \nabla \ln \pi(A|S, \boldsymbol{\theta})$
        $I \leftarrow \gamma I$
        $S \leftarrow S'$

Actor-Critic Methods

Actor–Critic with Eligibility Traces (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s,\boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Parameters: trace-decay rates $\lambda^{\boldsymbol{\theta}} \in [0,1]$, $\lambda^{\mathbf{w}} \in [0,1]$; step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g., to $\mathbf{0}$)
Loop forever (for each episode):
  Initialize S (first state of episode)
  $\mathbf{z}^{\boldsymbol{\theta}} \leftarrow \mathbf{0}$ (d'-component eligibility trace vector)
  $\mathbf{z}^{\mathbf{w}} \leftarrow \mathbf{0}$ (d-component eligibility trace vector)
  $I \leftarrow 1$
  Loop while S is not terminal (for each time step):
    $A \sim \pi(\cdot|S,\boldsymbol{\theta})$
    Take action A, observe S', R
    $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
    $\mathbf{z}^{\mathbf{w}} \leftarrow \gamma\lambda^{\mathbf{w}}\mathbf{z}^{\mathbf{w}} + \nabla\hat{v}(S,\mathbf{w})$
    $\mathbf{z}^{\boldsymbol{\theta}} \leftarrow \gamma\lambda^{\boldsymbol{\theta}}\mathbf{z}^{\boldsymbol{\theta}} + I\nabla\ln\pi(A|S,\boldsymbol{\theta})$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}}\delta\mathbf{z}^{\mathbf{w}}$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}}\delta\mathbf{z}^{\boldsymbol{\theta}}$
    $I \leftarrow \gamma I$
    $S \leftarrow S'$

Lecture Overview

Proximal Policy Optimization

- We collect data with $\pi_{\boldsymbol{\theta}_{\text{old}}}$
- And we want to optimize some objective to get a new policy $\pi_{\boldsymbol{\theta}}$
- We can write $\eta(\pi_{\boldsymbol{\theta}})$ in terms of $\pi_{\boldsymbol{\theta}_{\text{old}}}$:

$$\eta(\pi_{\boldsymbol{\theta}}) = \eta(\pi_{\boldsymbol{\theta}_{\text{old}}}) + \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[\sum_{t=0}^{\infty} \gamma^t \mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s_t, a_t)]$$

where the **advantage function** is defined as

$$\begin{aligned} \mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a) &= \mathbb{E}_{\pi_{\boldsymbol{\theta}}, s_{t+1} \sim p}[q_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a) - v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s)] \\ &= \mathbb{E}_{\pi_{\boldsymbol{\theta}}, s_{t+1} \sim p}[r(s,a) + \gamma v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s') - v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s)] \end{aligned}$$

- Advantage: how much better or worse is every action than average?

Proximal Policy Optimization

Proof:

$$
\begin{aligned}
&\mathbb{E}_{\pi_{\boldsymbol{\theta}}}[\sum_{t=0}^{\infty} \gamma^t \mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s_t, a_t)] \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}, s_{t+1} \sim p}[\sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + \gamma v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s_{t+1}) - v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s_t))] \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}, s_{t+1} \sim p}[-v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s_0) + \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)] \\
&= \mathbb{E}_{s_0 \sim p_0}[-v_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s_0)] + \mathbb{E}_{\pi_{\boldsymbol{\theta}}, s_{t+1} \sim p}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)] \\
&= -\eta(\pi_{\boldsymbol{\theta}_{\text{old}}}) + \eta(\pi_{\boldsymbol{\theta}})
\end{aligned}
$$

Proximal Policy Optimization

- In PPO, we *ignore* the change in state distribution and optimize a **surrogate objective**:

$$\begin{aligned} J_{\text{old}}(\theta) &= \mathbb{E}_{s\sim\pi_{\boldsymbol{\theta}_{\text{old}}}, a\sim\pi_{\boldsymbol{\theta}}}[\mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a)] \\ &= \mathbb{E}_{(s,a)\sim\pi_{\boldsymbol{\theta}_{\text{old}}}}\left[\frac{\pi_{\boldsymbol{\theta}}}{\pi_{\boldsymbol{\theta}_{\text{old}}}}\mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a)\right] \end{aligned}$$

- Improvement Theory: $\eta(\pi_{\boldsymbol{\theta}}) \geq J_{\text{old}}(\theta) - c \cdot \max_s \text{KL}[\pi_{\boldsymbol{\theta}_{\text{old}}}||\pi_{\boldsymbol{\theta}}]$
- If we keep the KL-divergence between our old and new policies small, optimizing the surrogate is close to optmizing $\eta(\pi_{\boldsymbol{\theta}})$!

Proximal Policy Optimization

- Clipped Surrogate Objective:

$$\mathbb{E}_{(s,a)\sim\pi_{\boldsymbol{\theta}_{\text{old}}}}\left[\min\left(\frac{\pi_{\boldsymbol{\theta}}}{\pi_{\boldsymbol{\theta}_{\text{old}}}}\mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a), \text{clip}\left(\frac{\pi_{\boldsymbol{\theta}}}{\pi_{\boldsymbol{\theta}_{\text{old}}}}, 1-\epsilon, 1+\epsilon\right)\mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a)\right)\right]$$

- Adaptive Penalty Surrogate Objective:

$$\mathbb{E}_{(s,a)\sim\pi_{\boldsymbol{\theta}_{\text{old}}}}\left[\frac{\pi_{\boldsymbol{\theta}}}{\pi_{\boldsymbol{\theta}_{\text{old}}}}\mathcal{A}_{\pi_{\boldsymbol{\theta}_{\text{old}}}}(s,a) - \beta\text{KL}[\pi_{\boldsymbol{\theta}_{\text{old}}}||\pi_{\boldsymbol{\theta}}]\right]$$

Algorithm 1 PPO

for *iteration* $i = 1, 2, \ldots$ **do**
    Run policy for T timesteps of N trajectories
    Estimate advantage function at all timesteps
    Do SGD on one of the above objectives for some number of epochs
    In case of the Adaptive Penalty Surrogate: Increase β if KL-divergence too high, otherwise decrease β

Lecture Overview

Exam

- There will be oral exams, the dates are March 23-25
- The first six minutes will be about the project, talk and discussion
- The project is designed as an exercise sheet for the last three weeks of the lecture (January 31, February 07, February 14)
- Final grade: $\frac{1}{3}$ project, $\frac{2}{3}$ questions about the rest of the lecture

Project

- You can choose to implement and apply any reinforcement learning algorithm (from the lecture or beyond) to solve this problem
- The evaluation should at least include learning curves (i.e. the return over time) of your chosen approach and settings – you can additionally think of your own metric and evaluate that as well
- It is important that your evaluation builds the basis for discussion and scientifically analyzes which are the important aspects and characteristics of your approach – your talk has to highlight your findings in a convincing manner
- You can prepare two slides, one with your approach and one with results (prepare as many backup slides for the discussion as you want)

Lecture Overview

Summary by Learning Goals

Having heard this lecture, you can now...

- explain the Policy Gradient Theorem and derive score functions for a given policy.
- explain Actor-Critic Methods.
- apply Policy Gradient algorithms, such as REINFORCE and PPO.