POLITECNICO DI MILANO
School of Industrial and Information Engineering
Department of Mathematics

Master of Science Degree in Mathematical Engineering

A study of Importance Sampling Techniques for Policy Optimization

AI & R Lab
Laboratorio di Intelligenza Artificiale
e Robotica del Politecnico di Milano

Supervisor: Prof. Marcello Restelli
Co-supervisors: Dott. Alberto Maria Metelli
Dott. Matteo Papini
Prof. Jürgen Schmidhuber

Author:
Francesco Faccio, 863358

Academic Year 2017-2018

Abstract

Policy optimization is an effective reinforcement learning approach to solve continuous control tasks. Recent achievements have shown that alternating online and offline optimization is a successful choice for efficient trajectory reuse. However, deciding when to stop optimizing and collect new trajectories is non-trivial, as it requires to account for the variance of the objective function estimate. In this work, we propose a novel, model-free, policy search algorithm, POIS, applicable in both action-based and parameter-based settings. We first derive a high-confidence bound for importance sampling estimation; then we define a surrogate objective function, which is optimized offline whenever a new batch of trajectories is collected. Finally, the algorithm is tested on a selection of continuous control tasks, with both linear and deep policies, and compared with state-of-the-art policy optimization methods.

Estratto in Lingua Italiana

I metodi di ottimizzazione della politica si sono rivelati essere degli approcci molto effettivi per la soluzioni di problemi di controllo continuo nell'apprendimento per rinforzo. Alcuni recenti risultati hanno mostrato che alternare l'ottimizzazione online e offline ÃÍ una scelta di successo per riutilizzare l'informazione proveniente da una traiettoria. Tuttavia, decidere quando fermare il processo di ottimizzazione ed iniziare a raccogliere nuove traiettorie non ÃÍ banale, in quanto deve tenere conto della varianza della funzione obiettivo stimata. In questo lavoro, proponiamo un nuovo algoritmo di ricerca della politica, POIS, che puÃš essere applicato sia nel setting delle azioni che in quello di parametri. Per prima cosa, deriviamo un limite inferiore per la stima da campionamento di importanza; poi definiamo una funzione obiettivo surrogata, che viene ottimizzata offline quando un nuovo gruppo di traiettorie viene raccolto. Infine, l'algoritmo ÃÍ testato su una selezione di problemi di controllo continuo, sia usano politiche lineari, che usando Reti Neurali profonde, e confrontato con lo stato dell'arte dei metodi di ottimizzazione della politica.

Ringraziamenti

Vorrei ringraziare innanzitutto il Prof. Marcello Restelli, il Dott. Alberto Maria Metelli e il Dott. Matteo Papini, per avermi accompagnato e sostenuto in questo progetto di ricerca.
Ringrazio la mia famiglia, Bruno, Teresa e Federico, per tutto il supporto ricevuto in questi anni. Grazie ad Erin, per avermi dato la forza nei momenti in cui ne avevo piú bisogno.
Grazie infine ai soci dell'AIM e SIAM, ai coinquilini di Via Lambrate, Jacopo, Giordano e Tommaso, a tutti coloro che hanno partecipato ai σ-party.

Contents

Chapter 1

Introduction

Reinforcement Learning [48] (RL) is a Machine Learning framework for modeling the interactions between an artificial agent and the environment. In recent years, a class of RL methods, called policy search methods [8] have achieved state-of-the-art results in continuous control tasks. [21, 39, 41, 40], robotic locomotion [51, 18] and partially observable environments [29]. Policy search algorithms can be classified in two different sets, depending whether the search is performed in the space of parametrized policies (like policy gradient), or directly in the space of parameters, by exploiting global optimizers [38, 15, 46, 50] or following a proper gradient direction like in Policy Gradients with Parameter-based Exploration (PGPE) [43, 57, 44]. In this work, we focus on a very general problems of Reinforcement Learning: given a dataset of trajectories collected from an agent which interacts with the environment using a fixed policy, how can we use the information gathered in the most efficient way?
On-policy methods use a batch of trajectories for doing a single update step in the optimization process. These methods are the most widely used, and they can use stochastic policy gradient [49], like REINFORCE [58] and G(PO)MDP [4] or more advanced contrained methods such as Trust Region Policy Optimization (TRPO) [39]. However, these methods do not exploit efficiently the information contained in the trajectories, since they require a new batch for each update step. On the other hand, off-policy methods collect trajectories using a behavioral policy and then perform multiple optimization steps evaluating and improving a target policy. Off-policy learning has been adapted first in value-based RL [56, 31, 27] and has been then extended to Deterministic Policy Gradient (DPG) [45] and Deep Deterministic Policy Gradient (DDPG) [21]. Parameter-based methods have also been adapted in order to perform off-policy optimization.

The off-policy optimization can be performed by alternating online steps, in which a behavioral policy collects new data, and offline optimization steps, in which a target policy is optimized with these new data. This idea has been used in Proximal Policy Optimization (PPO) [41], which has achieved new state-of-the-art performance in continuous control tasks. Optimizing a target policy with samples collected from a behavioral policy can be performed using the Importance Sampling (IS) [30, 16] technique, which weight each sample by the likelihood of having been generated from the target distribution. Unfortunately, the IS estimator, although unbiased, can have a very high variance when the two distribution are dissimilar.

In this work, we propose a novel, model-free, actor-only, policy optimization algorithm, named *Policy Optimization via Importance Sampling* (POIS) that alternates online and offline optimization to efficiently exploit the information contained in a batch of trajectories.

The main contributions of this paper are theoretical, algorithmic and experimental.

The structure of the remaining part of this document is as follows: Chapter 2 provides an introduction on the Reinforcement Learning framework; Chapter 3 briefly reviews the state-of-the-art methods in Policy search; Chapter 4.1 contains an analysis of the Importance sampling estimator and its relation with the dissimilarity of the policies; Chapter 5 contains the derivation of the novel algorithm, POIS, and the details of its implementation; Chapter 6 discusses the results of a comparison between POIS and the current state-of-the-art Policy search methods in a selected group of benchmark problems; Chapter 7 summarizes the work and suggests future direction of research. The implementation of POIS can be found at `https://github.com/T3p/pois`. POIS was accepted at the 32nd Conference on Neural Information Processing Systems (NIPS 2018) and selected for an oral presentation [24].

Chapter 2

Reinforcement Learning and Markov Decision Process

In many Machine Learning problems, an agent has to learn to solve a task by interacting with the environment. Markov Decision Process (MDP) [34] can be used to model this interaction.

2.1 Markov Decision Process

2.1.1 Definition

A discrete-time MDP can be formally defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma, \mu_0)$ in which:

- $\mathcal{S} \subseteq \mathcal{R}^n$ is the set of possible states of the agent $s \in \mathcal{S}$
- $\mathcal{A} \subseteq \mathcal{R}^n$ is the set of possible actions the agent can perform: $a \in \mathcal{A}$
- $P(\cdot|s, a)$ is a stochastic markovian transition matrix, which assigns for each pair (s_t, a_t) the probability of transitioning to state s_{t+1}
- $R(s, a) \in [-R_{\max}, R_{\max}]$ is the reward function, which assigns a scalar reward to the agent for performing action a in state s
- $\gamma \in [0, 1]$ is a discount factor, measuring the actual value of future reward
- μ_0 is a probability distribution over the initial state of the agent

At each discrete time-step t, the agent observes its state s_t and interacts with the environment performing action a_t. The environment receives the

action of the agent and emits a scalar reward $r_t = R(s_t, a_t)$. The state of the agent evolves following the dynamics of the transition matrix $\mathcal{P}$. The goal of the agent is to choose actions such that the cumulative discounted reward is maximized, i.e., $v = \sum_{i=0}^{H-1} \gamma^t r_t$ is maximized, where H is the horizon length. We define the return v_t as the total discounted reward from time-step t:

$$v_t = r_{t+1} + \gamma r_{t+2} + \cdots = \sum_{k=0}^{H-t-1} \gamma^k r_{t+k+1}.$$

In order to define the behavior of the agent, we need to introduce the policy, which is a map between the set of the states $\mathcal{S}$ and a probability distribution over the action set $\mathcal{A}$. More formally, we define $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ which specifies the probability of performing action a in state s. A policy is deterministic when for each state s there exists an action a such that $\pi(a|s) = 1$. When the agent interacts with the environment for H time steps, the sequence of states and actions collected defines a trajectory, i.e., $\tau = (s_{\tau,0}, a_{\tau,0}, \ldots, s_{\tau,H-1}, a_{\tau,H-1}, s_{\tau,H})$. Given a policy π, we can compute the probability of a trajectory τ as:

$$p(\tau) = \mu_0(s_0) \prod_{t=0}^{H-1} \pi(a_t|s_t) P(s_{t+1}|s_t, a_t).$$

We can express the problem of maximizing the cumulative discounted reward in terms of the policy. The objective function is an expectation over π:

$$J^\pi = \mathbb{E}_\pi[v].$$

2.1.2 Value Functions

Given a policy π, we want to be able to evaluate its utility in each state. The state-value function, defined as the expected return for being in a state s and following policy π, is used to measure the value of the policy in each state and it is employed by many Reinforcement Learning algorithms. More formally, it is defined as:

$$V^\pi(s) = \mathbb{E}_\pi[v_t|s_t = s].$$

The value function can also be defined in a recursive way, using Bellman's Expectation Equation:

$$V^\pi(s) = \int_{\mathcal{A}} \pi(a|s) \left(R(s,a) + \gamma \int_{\mathcal{S}} P(s^{'}|s,a) V^\pi(s^{'}) \, \mathrm{d}s^{'} \right) \mathrm{d}a.$$

By integrating over the state space $\mathcal{S}$, we can express the maximization of the expected cumulative reward in terms of the value function:

$$J^{\pi} = \int_{\mathcal{S}} \mu_0(s) V^{\pi}(s)\, \mathrm{d}s.$$

Another way to measure how good is a policy is by defining the action-value function Q^{π}, which is defined as the expected return for performing action a in state s, under the policy π:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}[v_t | s_t = s, a_t = a],$$

which can also be expressed recursively, using Bellman's Expectation Equation:

$$Q^{\pi}(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} P(s^{'}|s,a) \int_{\mathcal{A}} \pi(a^{'}|s^{'}) Q^{\pi}(s^{'},a^{'})\, \mathrm{d}a^{'}\, \mathrm{d}s^{'}.$$

2.2 Dynamic Programming

For solving an MDP, we need to find the optimal policy. One naive approach would be to enumerate all possible policies, to evaluate their value function and then to return the best policy found. However, the number of deterministic markovian policies is exponential ($|\mathcal{A}|^{|\mathcal{S}|}$) when action and states are discrete and impractical when they assume continuous values.
Dynamic Programming techniques offer a general solution for complex problems which can be divided in smaller, easier subproblems. In particular, for solving an MDP, it iterates among two main steps: Prediction and Control.

2.2.1 Policy Iteration

In **Policy Iteration** algorithm, these two steps correspond to Policy Evaluation and Policy Improvement.

- **Policy Evaluation**: In Policy Evaluation, the goal is, given a policy π, to compute its action-value function Q^{π} using Bellman Equation.
- **Policy Improvement**: Given the action-value function Q^{π} for policy π, compute a better policy $\pi^{'}(s) \in \arg\max_{a \in \mathcal{A}} Q^{\pi}(s,a)$.

Iterating the procedure we obtain a deterministic policy. This method is guaranteed to find the optimal policy. However, in most Reinforcement Learning problem, the action-value function cannot be computed exactly and

needs to be estimated from sample. This can lead to an high variance estimate and to slow down or prevent the algorithm to converge. One possible solution is to use a method which does not require to know the dynamics of the MDS and which directly models the policy, constraining it in a parametric subspace. We obtain a class of methods called model-free, policy search methods.

Chapter 3

Policy Search

In Policy Search methods [8] we consider a parametrized policy π_θ belonging to a parametric policy space $\Pi_\Theta = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p\}$. The goal is to find the parameters of the policy such that the expected cumulative reward is maximized. In a high level, we can divide these methods in two categories: action-based and parameted-based. In the following sections the two approaches are described.

3.1 Action-based Methods

Action-based methods, known also al Policy Gradient methods, optimize the objective function by following the direction of the gradient w.r.t. the parameters θ, which is estimated using a batch of trajectories generated by the policy. The expected return can be espressed as an expectation over the trajectory space:

$$J(\boldsymbol{\theta}) = \int_{\mathcal{T}} p(\tau|\boldsymbol{\theta}) R(\tau)\, \mathrm{d}\tau,$$

where $R(\tau) = \sum_{t=0}^{H-1} \gamma^t R(s_{\tau,t}, a_{\tau,t})$ is the trajectory return. The goal is to find the parameter θ^* such that $J(\boldsymbol{\theta})$ is maximized. Assuming that the policy π_θ is stochastic and differentiable, we can update the parameters using gradient ascent:

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

where:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathcal{T}} p(\tau|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\tau|\boldsymbol{\theta}) R(\tau)\, \mathrm{d}\tau.$$

3.2 Parameter-based Methods

In Parameter-based methods, like in Policy Gradients with Parameter-based Exploration (PGPE) [43, 57, 44], the agent is endowed with an hyperpolicy ν belonging to a parametric hyperpolicy space $\mathcal{N}_{\mathcal{P}} = \{\nu_{\boldsymbol{\rho}} : \boldsymbol{\rho} \in \mathcal{P} \subseteq \mathbb{R}^r\}$. At the beginning of each episode, a vector of policy parameters $\boldsymbol{\theta}$ is sampled from μ. The policy π_θ, in the parameter-based setting, is a deterministic controller and does not need to be differentiable. The expected return can be espressed as an expectation over the trajectory space and over the policy parameter space:

$$J(\boldsymbol{\rho}) = \int_{\Theta}\int_{\mathcal{T}} \nu_{\boldsymbol{\rho}}(\boldsymbol{\theta}) p(\tau|\boldsymbol{\theta}) R(\tau)\, \mathrm{d}\tau\, \mathrm{d}\boldsymbol{\theta},$$

where $p(\tau|\boldsymbol{\theta}) = \mu_0(s_0)\prod_{t=0}^{H-1}\pi_{\boldsymbol{\theta}}(a_t|s_t)P(s_{t+1}|s_t,a_t)$ is the trajectory density function. The goal is to find the hyperparameter ρ^* such that $J(\boldsymbol{\rho})$ is maximized. This can be done, without the assumption of the differentiability of the policy π_θ using gradient ascent:

$$\boldsymbol{\rho}' = \boldsymbol{\rho} + \alpha \nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}),$$

where $\alpha > 0$ and

$$\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}) = \int_{\Theta}\int_{\mathcal{T}} \nu_{\boldsymbol{\rho}}(\boldsymbol{\theta}) p(\tau|\boldsymbol{\theta}) \nabla_{\boldsymbol{\rho}} \log \nu_{\boldsymbol{\rho}}(\boldsymbol{\theta}) R(\tau)\, \mathrm{d}\tau\, \mathrm{d}\boldsymbol{\theta}$$

Sampling the parameters once per episode can reduce a lot the variance of the gradient estimate.

3.3 State-of-the-art algorithms in Policy search

Policy search methods [8] have become widely used because of their ability to solve Reinforcement Learning problems with continuous control tasks [21, 39, 41, 40], robotic locomotion [51, 18] and partially observable environments [29]. We can divide the state-of-the-art Policy search methods in three categories, depending on how the optimization process is carried on in order to find the optimal policy. The first, and most common used, category, is the one of Policy Gradient methods, in which the cumulative expected return is optimized following the gradient direction. The second category also follows the direction of the gradient, but subject to a constraints that limits updates leading to policies too far from the current policy. This constraint is typically expressed in terms of the dissimilarity between the two policies. The third category, which includes our algorithm, POIS, optimizes a surrogate objective function: a function similar w.r.t. the objective function of the problem, but much easier to optimize.

3.4 Policy Gradient

In this section, we see some Policy search algorithms using policy gradient methods.

3.4.1 Policy Gradient Theorem

We can first note that when we perform a parameter update:

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

we have to compute the quantity $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. This can be done analytically, thanks to the Policy Gradient Theorem [49]:

Theorem 3.4.1. *For any Markov Decision Process, the following holds:*

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathcal{S}} \mu_0(s) \int_{\mathcal{A}} \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) Q^{\pi_{\boldsymbol{\theta}}}(s,a) \, \mathrm{d}a \, \mathrm{d}s.$$

3.4.2 REINFORCE/GPOMDP

Unfortunately, in many applications we need to estimate the term $Q^{\pi_{\boldsymbol{\theta}}}(s,a)$ with sample. One straightforward way of doing it is to use the return:

$$\hat{Q}^{\pi_{\boldsymbol{\theta}}}(s,a) \simeq \sum_{k=0}^{H-1} \gamma^k R(s_k, a_k)$$

Moreover, we can rewrite the Policy Gradient Theorem using the likelihood ratio technique [58]:

$$\nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) = \pi_{\boldsymbol{\theta}}(a|s) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s),$$

and obtain:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathcal{S}} \mu_0(s) \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a|s) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) Q^{\pi_{\boldsymbol{\theta}}}(s,a) \, \mathrm{d}a \, \mathrm{d}s.$$

Combining this expression with the approximation of the action-value function, we obtain the REINFORCE estimator [58]:

$$\hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \left\langle \left(\sum_{k=0}^{H-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_k|s_k) \right) \left(\sum_{k=0}^{H-1} \gamma^k R(s_k, a_k) \right) \right\rangle_N$$

where $\langle . \rangle_N$ denotes the sample mean over N trajectories. One problem of REINFORCE is that, in the estimator, the value of (s_k, a_k) depends on the

past reward. Making this term independent on past reward, we obtain the G(PO)MDP gradient estimator [4]:

$$\hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \left\langle \sum_{t=0}^{H-1} \left(\sum_{k=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_k|s_k) \right) \left(\gamma^t R(s_t, a_t) \right) \right\rangle_N$$

3.4.3 Natural Gradient

When searching in the space of parameters in order to find the optimal policy, the optimization process strongly depends on the choice of the parametrization used for the policy. Natural gradient methods [1] allow to search directly in the space of the policy and are invariant w.r.t. the parametrization used. The update formula for the natural gradient becomes:

$$\widetilde{\nabla}_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \mathcal{F}^{-1}(\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}),$$

where $\mathcal{F}$ is the Fisher Information Matrix[36, 2]:

$$\mathcal{F}(\boldsymbol{\theta}) = \int_{\mathcal{S}} \mu_0(s) \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a|s) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)^T \, \mathrm{d}a \, \mathrm{d}s$$

3.5 Constraint Methods

The methods belonging to this category, still follow the direction of the gradient, but with a constraint on the step size depending on how different the current policy and the policy at the update are.

3.5.1 TRPO

The most used algorithm of this category, which achieved state-of-the-art results in many continuous control tasks, is Trust Region Policy Optimization (TRPO) [39], whose update formula is:
$\max \widehat{\mathbb{E}}_{t\sim\boldsymbol{\theta}} \left[w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(a_t|s_t) \widehat{A}(s_t, a_t) \right]$
s.t. $\widehat{\mathbb{E}}_{t\sim\boldsymbol{\theta}} \left[D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}'}(\cdot|s_t) \| \pi_{\boldsymbol{\theta}}(\cdot|s_t)) \right] \leq \delta$
where $\widehat{A}(s_t, a_t)$ is an estimate of the advantage function $Q^{\pi_{\boldsymbol{\theta}}}(s, a) - V^{\pi_{\boldsymbol{\theta}}}(s)$, which measures the advantage of taking action a in state s, and $D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}'}(\cdot|s_t) \| \pi_{\boldsymbol{\theta}}(\cdot|s_t))$ is the Kullback-Liebler divergence between the two distributions.

3.6 Surrogate methods

This class of methods tries to maximize a surrogate objective function, which is a function similar w.r.t. the objective function of the problem, but much easier to optimize.

3.6.1 PPO

The most widely used surrogate policy gradient method is Proximal Policy Optimization (PPO) [41], which optimizes the following surrogate objective function:

$$\max \widehat{\mathbb{E}}_{t\sim\boldsymbol{\theta}}\Big[\min\Big\{w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(a_t|s_t)\widehat{A}(s_t,a_t),\quad \text{clip}\left(w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(a_t|s_t),1-\epsilon,1+\epsilon\right)\widehat{A}(s_t,a_t)\Big\}\Big].$$

Chapter 4

Importance Sampling

4.1 The Importance Sampling estimator

Importance sampling is a technique that can be used for studying a distribution, which we call target, using samples collected from another distribution, called behavioral. In particular, we want to study the problem of estimating the expected value of a deterministic bounded function f ($\|f\|_\infty < +\infty$) of a random variable x, under a target distribution P, having samples collected samples from another distribution Q. If we assume that P and Q admit p and q as Lebesgue probability density functions, we can use the importance sampling estimator (IS) [6, 30], which weights the collected samples by the likelihood of being generated by the target distribution. This correction relies on the *importance weights* (or Radon-Nikodym derivative or likelihood ratio) $w_{P/Q}(x) = p(x)/q(x)$:

$$\widehat{\mu}_{P/Q} = \frac{1}{N}\sum_{i=1}^{N}\frac{p(x_i)}{q(x_i)}f(x_i) = \frac{1}{N}\sum_{i=1}^{N} w_{P/Q}(x_i)f(x_i), \tag{4.1}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ is sampled from Q and we assume $q(x) > 0$ whenever $f(x)p(x) \neq 0$. The IS estimator is unbiased ($\mathbb{E}_{\mathbf{x}\sim Q}[\widehat{\mu}_{P/Q}] = \mathbb{E}_{x\sim P}[f(x)]$) but it can have a large variance if the two distributions differ a lot. Even in the particular case in which the two distributions are Gaussian, the IS estimator can have infinite variance.

We can measure how two distributions are dissimilar using the Rényi divergence, an information-theoretic index. [37, 54]

4.1.1 Rényi divergence

Consider two probability measures P and Q on a measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ (P is absolutely continuous w.r.t. Q) and Q is σ-finite. Let P and Q admit p and q as Lebesgue probability density functions (p.d.f.), respectively. The α-Rényi divergence is defined as:

$$D_\alpha(P\|Q) = \frac{1}{\alpha - 1} \log \int_{\mathcal{X}} \left(\frac{\mathrm{d}P}{\mathrm{d}Q}\right)^\alpha \mathrm{d}Q = \frac{1}{\alpha - 1} \log \int_{\mathcal{X}} q(x) \left(\frac{p(x)}{q(x)}\right)^\alpha \mathrm{d}x, \quad (4.2)$$

where $\mathrm{d}P/\,\mathrm{d}Q$ is the Radon-Nikodym derivative of P w.r.t. Q and $\alpha \in [0, \infty]$.
Some particular cases are the following:

- for $\alpha = 1$ we have that $D_1(P\|Q) = D_{\mathrm{KL}}(P\|Q)$
- for $\alpha = \infty$ we have $D_\infty(P\|Q) = \log \operatorname{ess\,sup}_{\mathcal{X}} \mathrm{d}P/\,\mathrm{d}Q$.

Using the notation of [7], we define the α-Rényi divergence as $d_\alpha(P\|Q) = \exp\left(D_\alpha(P\|Q)\right)$. Whenever it will be clear within the context, $D_\alpha(P\|Q)$ will be replaced by $D_\alpha(p\|q)$.

There is a tight relation between the dissimilarity of the two distributions and the moments of the importance weights. Indeed:

$$\mathop{\mathbb{E}}_{x\sim Q}\left[w_{P/Q}(x)^\alpha\right] = d_\alpha(P\|Q).$$

Moreover,

$$\mathop{\mathbb{V}\mathrm{ar}}_{x\sim Q}\left[w_{P/Q}(x)\right] = d_2(P\|Q) - 1,$$

and for [7]

$$\operatorname*{ess\,sup}_{x\sim Q} w_{P/Q}(x) = d_\infty(P\|Q).$$

4.2 Analysis of the IS estimator

In this section, the importance weights are analyzed under the assumption that both the behavioral and target distributions are Gaussian. We can note that for multivariate Gaussian distribution, there exists an analytic expression for the Rényi divergence [5]. Let $P \sim \mathcal{N}(\boldsymbol{\mu}_P, \boldsymbol{\Sigma}_P)$ and $Q \sim \mathcal{N}(\boldsymbol{\mu}_Q, \boldsymbol{\Sigma}_Q)$ and $\alpha \in [0, \infty]$:

$$D_\alpha(P\|Q) = \frac{\alpha}{2}(\boldsymbol{\mu}_P - \boldsymbol{\mu}_Q)^T \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{\mu}_P - \boldsymbol{\mu}_Q) - \frac{1}{2(\alpha - 1)} \log \frac{\det(\boldsymbol{\Sigma}_\alpha)}{\det(\boldsymbol{\Sigma}_P)^{1-\alpha} \det(\boldsymbol{\Sigma}_Q)^\alpha}, \quad (4.3)$$

where $\boldsymbol{\Sigma}_\alpha = \alpha\boldsymbol{\Sigma}_Q + (1 - \alpha)\boldsymbol{\Sigma}_P$ under the assumption that $\boldsymbol{\Sigma}_\alpha$ is positive-definite.

In the particular case of univariate Gaussian distributions, the expression of the importance weights and their probability density function f_w can be computed in closed-form. Let us consider $Q \sim \mathcal{N}(\mu_Q, \sigma_Q^2)$ as the behavioral distribution and $P \sim \mathcal{N}(\mu_P, \sigma_P^2)$ as the target distribution. Under the assumption that $\sigma_Q^2, \sigma_P^2 > 0$ we can analyze the importance weights in the following two cases: unequal variances and equal variances. The importance weight $w_{P/Q}(x)$ will be indicated with $w(x)$ for brevity.

4.2.1 Unequal variances

We start considering the case in which the variance of the behavioral and target policies differ: $\sigma_Q^2 \neq \sigma_P^2$. In this case, the importance weights assume the following expression:

$$w(x) = \frac{\sigma_Q}{\sigma_P} \exp\left(\frac{1}{2}\frac{(\mu_P - \mu_Q)^2}{\sigma_Q^2 - \sigma_P^2}\right) \exp\left(-\frac{1}{2}\frac{\sigma_Q^2 - \sigma_P^2}{\sigma_Q^2 \sigma_P^2}\left(x - \frac{\sigma_Q^2 \mu_P - \sigma_P^2 \mu_Q}{\sigma_Q^2 - \sigma_P^2}\right)^2\right), \quad (4.4)$$

for $x \sim Q$. We can study when the above equation is bounded based on the sign of the exponential: if $\sigma_Q^2 - \sigma_P^2 > 0$, then $w(x)$ is upper bounded by $A = \frac{\sigma_Q}{\sigma_P} \exp\left(\frac{1}{2}\frac{(\mu_P - \mu_Q)^2}{\sigma_Q^2 - \sigma_P^2}\right)$, while if $\sigma_Q^2 - \sigma_P^2 < 0$, then $w(x)$ is unbounded and it admits A as minimum value. We then compute the probability density function of $w(x)$:

Proposition 4.2.1. *Let $Q \sim \mathcal{N}(\mu_Q, \sigma_Q^2)$ be the behavioral distribution and $P \sim \mathcal{N}(\mu_P, \sigma_P^2)$ be the target distribution, with $\sigma_Q^2 \neq \sigma_P^2$. The probability density function of $w(x) = p(x)/q(x)$ is given by:*

$$f_w(y) = \begin{cases} \frac{\overline{\sigma}}{y\sqrt{\pi \log \frac{A}{y}}} \exp\left(-\frac{1}{2}\overline{\mu}^2\right) \left(\frac{y}{A}\right)^{\overline{\sigma}^2} \cosh\left(\overline{\mu\sigma}\sqrt{2 \log \frac{A}{y}}\right), & \text{if } \sigma_Q^2 > \sigma_P^2,\ y \in [0, A], \\ \frac{\overline{\sigma}}{y\sqrt{\pi \log \frac{y}{A}}} \exp\left(-\frac{1}{2}\overline{\mu}^2\right) \left(\frac{A}{y}\right)^{\overline{\sigma}^2} \cosh\left(\overline{\mu\sigma}\sqrt{2 \log \frac{y}{A}}\right), & \text{if } \sigma_Q^2 < \sigma_P^2,\ y \in [A, \infty), \end{cases}$$

where $\overline{\mu} = \frac{\sigma_Q}{\sigma_Q^2 - \sigma_P^2}(\mu_P - \mu_Q)$ and $\overline{\sigma}^2 = \frac{\sigma_P^2}{|\sigma_Q^2 - \sigma_P^2|}$.

Proof. Let us consider $w(x)$ as a function of the random variable $x \sim Q$. We can define the following constants:

$$m = \frac{\sigma_Q^2 \mu_P - \sigma_P^2 \mu_Q}{\sigma_Q^2 - \sigma_P^2}, \quad \tau = \frac{\sigma_Q^2 - \sigma_P^2}{\sigma_Q^2 \sigma_P^2}.$$

Let us start computing the c.d.f.:

$$F_w(y) = \Pr\left(w(x) \leq y\right) = \Pr\left(A \exp\left(-\frac{1}{2}\tau(x - m)^2\right) \leq y\right)$$

$$= \Pr\left(\tau(x-m)^2 \geq -2\log\frac{y}{A}\right).$$

According to the sign of τ, we can study two different cases, and we observe that $x = \mu_Q + \sigma_Q z$ where $z \sim \mathcal{N}(0,1)$:

$\boldsymbol{\tau > 0}$:

$$\begin{aligned}
F_w(y) &= \Pr\left((x-m)^2 \geq \frac{2}{\tau}\log\frac{A}{y}\right) \\
&= \Pr\left(x \leq m - \sqrt{\frac{2}{\tau}\log\frac{A}{y}}\right) + \Pr\left(x \geq m + \sqrt{\frac{2}{\tau}\log\frac{A}{y}}\right) \\
&= \Pr\left(z \leq \frac{m-\mu_Q}{\sigma_Q} - \sqrt{\frac{2}{\tau\sigma_Q^2}\log\frac{A}{y}}\right) + \Pr\left(z \geq \frac{m-\mu_Q}{\sigma_Q} + \sqrt{\frac{2}{\tau\sigma_Q^2}\log\frac{A}{y}}\right).
\end{aligned}$$

We call $\overline{\mu} = \frac{m-\mu_Q}{\sigma_Q} = \frac{\sigma_Q}{\sigma_Q^2-\sigma_P^2}(\mu_P - \mu_Q)$ and $\overline{\sigma}^2 = \frac{1}{\tau\sigma_Q^2} = \frac{\sigma_P^2}{\sigma_Q^2-\sigma_P^2}$, thus we have:

$$\begin{aligned}
F_w(y) &= \Pr\left(z \leq \overline{\mu} - \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right) + \Pr\left(z \geq \overline{\mu} + \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right) \\
&= \Phi\left(\overline{\mu} - \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right) + 1 - \Phi\left(\overline{\mu} + \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right),
\end{aligned}$$

where Φ is the c.d.f. of a normal standard distribution. We can obtain the p.d.f by taking the derivative w.r.t. y:

$$\begin{aligned}
f_w(y) = \frac{\partial F_w(y)}{\partial y} &= -\sqrt{2\overline{\sigma}^2}\frac{1}{2\sqrt{\log\frac{A}{y}}}\frac{y}{A}\frac{-A}{y^2}\left(\phi\left(\overline{\mu} - \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right) + \phi\left(\overline{\mu} + \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right)\right) \\
&= \frac{\sqrt{2}\overline{\sigma}}{2y\sqrt{\log\frac{A}{y}}}\left(\phi\left(\overline{\mu} - \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right) + \phi\left(\overline{\mu} + \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right)\right) \\
&= \frac{\sqrt{2}\overline{\sigma}}{2y\sqrt{\log\frac{A}{y}}}\frac{1}{\sqrt{2\pi}}\left(\exp\left(-\frac{1}{2}\left(\overline{\mu} - \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right)^2\right) + \exp\left(-\frac{1}{2}\left(\overline{\mu} + \sqrt{2\overline{\sigma}^2\log\frac{A}{y}}\right)^2\right)\right) \\
&= \frac{\overline{\sigma}}{y\sqrt{\pi\log\frac{A}{y}}}\exp\left(-\frac{1}{2}\overline{\mu}^2\right)\exp\left(-\overline{\sigma}^2\log\frac{A}{y}\right)\frac{\exp\left(\overline{\mu}\overline{\sigma}\sqrt{2\log\frac{A}{y}}\right) + \exp\left(-\overline{\mu}\overline{\sigma}\sqrt{2\log\frac{A}{y}}\right)}{2} \\
&= \frac{\overline{\sigma}}{y\sqrt{\pi\log\frac{A}{y}}}\exp\left(-\frac{1}{2}\overline{\mu}^2\right)\left(\frac{y}{A}\right)^{\overline{\sigma}^2}\cosh\left(\overline{\mu}\overline{\sigma}\sqrt{2\log\frac{A}{y}}\right),
\end{aligned}$$

where ϕ is the p.d.f. of a normal standard distribution.

$\boldsymbol{\tau < 0}$: We can derive the p.d.f. using similar steps. First, let us define $\overline{\sigma}^2 = -\frac{1}{\tau\sigma_Q^2} = \frac{\sigma_P^2}{\sigma_P^2-\sigma_Q^2}$, then the c.d.f. becomes:

$$F_w(y) = \Phi\left(\overline{\mu} + \sqrt{2\overline{\sigma}^2\log\frac{y}{A}}\right) - \Phi\left(\overline{\mu} - \sqrt{2\overline{\sigma}^2\log\frac{y}{A}}\right),$$

and the p.d.f. is:

$$f_w(x) = \frac{\overline{\sigma}}{y\sqrt{\pi \log \frac{y}{A}}} \exp\left(-\frac{1}{2}\overline{\mu}^2\right) \left(\frac{A}{y}\right)^{\overline{\sigma}^2} \cosh\left(\overline{\mu\sigma}\sqrt{2\log\frac{y}{A}}\right).$$

We can consider both cases by defining $\overline{\sigma}^2 = \frac{\sigma_P^2}{|\sigma_Q^2 - \sigma_P^2|}$. □

Studying the behavior of the tail of the distribution when w is unbounded, we discover that it displays a fat-tail behavior.

Proposition 4.2.2. *If $\sigma_P^2 > \sigma_Q^2$ then there exists $c > 0$ and $y_0 > 0$ such that for any $y \geq y_0$, the p.d.f. f_w can be lower bounded as $f_w(y) \geq c y^{-1-\overline{\sigma}^2} (\log y)^{-\frac{1}{2}}$.*

Proof. Set $z = y/A$ and let $a > 0$ be a constant, then for sufficiently large y it holds that:

$$f_w(y) \geq a z^{-1-\overline{\sigma}^2} (\log z)^{-1/2} \exp\left(\sqrt{\log z}\right)^{\sqrt{2}\overline{\mu\sigma}}. \tag{4.5}$$

We can first observe that, for $z > 1$, we have $\exp\left(\sqrt{\log z}\right) \geq 1$. Then, by replacing z with y/A, we just need to change the constant a into $c > 0$. □

Thanks to this result, we can see that the α-th moment of $w(x)$ does not exist when $\alpha - 1 - \overline{\sigma}^2 \geq -1 \implies \alpha \geq \overline{\sigma}^2 = \frac{\sigma_P^2}{\sigma_P^2 - \sigma_Q^2}$. As a consequence, we can not bound in probability the importance weights using Bernstein-like inequalities. Note that also the fact that the α-Rényi divergence is defined only when $\sigma_\alpha^2 = \alpha\sigma_Q^2 + (1-\alpha)\sigma_P^2 > 0$, i.e., $\alpha < \frac{\sigma_P^2}{\sigma_P^2 - \sigma_Q^2}$ confirms the non-existence of the finite moments of importance weights.

4.2.2 Equal variances

If $\sigma_Q^2 = \sigma_P^2 = \sigma^2$, the importance weights assume the following expression:

$$w(x) = \exp\left(\frac{\mu_P - \mu_Q}{\sigma^2}\left(x - \frac{\mu_P + \mu_Q}{2}\right)\right), \tag{4.6}$$

for $x \sim Q$. Under this condition, the importance weights $w(x)$ are unbounded and assume infimum value at 0. Also in this case, we can derive the p.d.f. of the importance weights.

Proposition 4.2.3. *Let $Q \sim \mathcal{N}(\mu_Q, \sigma^2)$ be the behavioral distribution and $P \sim \mathcal{N}(\mu_P, \sigma^2)$ be the target distribution. The probability density function of $w(x) = q(x)/p(x)$ is given by:*

$$f_w(y) = \frac{|\widetilde{\sigma}|}{\sqrt{2\pi} y^{\frac{3}{2}}} \exp\left(-\frac{1}{2}\left(\widetilde{\mu}^2 + \widetilde{\sigma}^2 (\log y)^2\right)\right), \tag{4.7}$$

where $\widetilde{\mu} = \frac{\mu_P - \mu_Q}{2\sigma}$ *and* $\widetilde{\sigma} = \frac{\sigma}{\mu_P - \mu_Q}$.

Proof. We start computing the c.d.f.:

$$\begin{aligned} F_w(y) &= \Pr\left(\exp\left\{\frac{\mu_P - \mu_Q}{\sigma^2}\left(x - \frac{\mu_P + \mu_Q}{2}\right)\right\} \leq y\right) \\ &= \Pr\left(\frac{\mu_P - \mu_Q}{\sigma^2}\left(x - \frac{\mu_P + \mu_Q}{2}\right) \leq \log y\right). \end{aligned}$$

Let us consider first the case $\mu_P - \mu_Q > 0$ and observe that $x = \mu_Q + \sigma z$, where $z \sim \mathcal{N}(0,1)$:

$$\begin{aligned} F_w(y) &= \Pr\left(x \leq \frac{\mu_P + \mu_Q}{2} + \frac{\sigma^2}{\mu_P - \mu_Q}\log y\right) \\ &= \Pr\left(z \leq \frac{\mu_P - \mu_Q}{2\sigma} + \frac{\sigma}{\mu_P - \mu_Q}\log y\right). \end{aligned}$$

Set $\widetilde{\mu} = \frac{\mu_P - \mu_Q}{2\sigma}$ and $\widetilde{\sigma} = \frac{\sigma}{\mu_P - \mu_Q}$. Then we have:

$$F_w(y) = \Pr\left(z \leq \widetilde{\mu} + \widetilde{\sigma}\log y\right) = \Phi\left(\widetilde{\mu} + \widetilde{\sigma}\log y\right).$$

We can obtain the p.d.f by taking the derivative w.r.t. y:

$$\begin{aligned} f_w(y) &= \frac{\partial F_w(y)}{\partial y} \\ &= \frac{\widetilde{\sigma}}{y}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\widetilde{\mu} + \widetilde{\sigma}\log y\right)^2\right) \\ &= \frac{\widetilde{\sigma}}{\sqrt{2\pi}y^{\widetilde{\mu}\widetilde{\sigma}+1}}\exp\left(-\frac{1}{2}\left(\widetilde{\mu}^2 + \widetilde{\sigma}^2\left(\log y\right)^2\right)\right). \end{aligned}$$

When considering the case $\mu_P - \mu_Q < 0$, the p.d.f. differs only by a minus sign. We can consider both cases by defining $|\widetilde{\sigma}|$ in the final formula. □

We can note that, when the variance of the behavioral and target policies is equal, then the tail behavior is different.

Proposition 4.2.4. *If* $\sigma_P^2 = \sigma_Q^2$ *then for any* $\alpha > 0$ *there exist* $c > 0$ *and* $y_0 > 0$ *such that for any* $y \geq y_0$*, the p.d.f. can be upper bounded as* $f_w(y) \leq cy^{-\alpha}$.

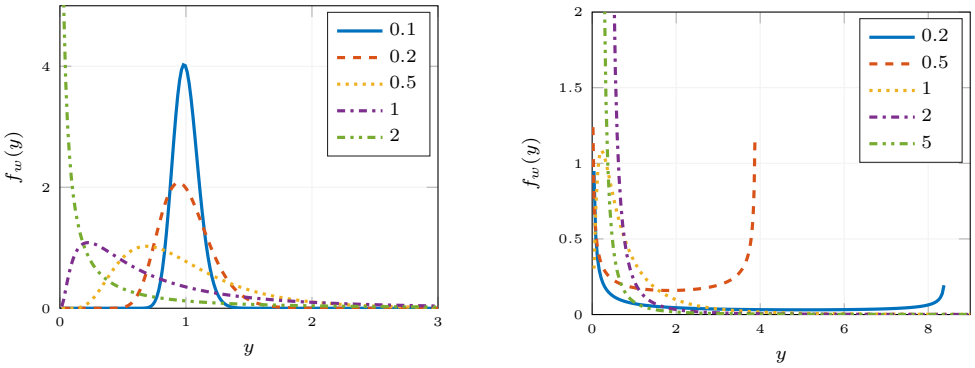
Proof. If we consider all constant terms as c, we can write the p.d.f. as:

$$f_w(y) = cy^{-3/2}\exp\left(\left(\log y\right)^2\right)^{-\frac{\widetilde{\sigma}^2}{2}}. \tag{4.8}$$

For any $\alpha > 0$, let us solve the following inequality:

$$y^{3/2}\exp\left(\left(\log y\right)^2\right)^{\frac{\widetilde{\sigma}^2}{2}} \geq y^{\alpha} \quad \Longrightarrow \quad y \geq \exp\left(\frac{2}{\widetilde{\sigma}^2}\left(\alpha - \frac{3}{2}\right)\right). \tag{4.9}$$

Thus, for $y \geq \exp\left(\frac{2}{\widetilde{\sigma}^2}\left(\alpha - \frac{3}{2}\right)\right)$ we have that $f_w(y) \leq cy^{-\alpha}$. □



(a) equal variance

(b) equal mean

Figure 4.1: Probability density function of the importance weights when the behavioral distribution is $\mathcal{N}(0,1)$ and the mean is changed keeping the variance equal to 1 (a) or the variance is changed keeping the target mean equal to 1 (b).

The corresponding Rényi divergence is: $\frac{\alpha(\mu_P-\mu_Q)^2}{2\sigma^2}$, therefore all moments of the importance weights exist. However, the distribution of $w(x)$ remains subexponential, as $\exp\left((\log y)^2\right)^{-\frac{\tilde{\sigma}^2}{2}} \geq e^{-\eta y}$ for sufficiently large y.

In Figure 4.1 the p.d.f. of the importance weights, for different values of mean and variance of the target distribution, is reported.

4.3 The Self-Normalized Importance Sampling Estimator

In the previous section, we saw that the IS estimator can exhibit a very high variance, when a target distribution is too far from the behavioral in terms of the Rényi divergence. In order to reduce the variance problem of the IS estimator, we can use the *self-normalized importance sampling* estimator (SN) [6]:

$$\widetilde{\mu}_{P/Q} = \frac{\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)}{\sum_{i=1}^{N} w_{P/Q}(x_i)} = \sum_{i=1}^{N} \widetilde{w}_{P/Q}(x_i) f(x_i), \tag{4.10}$$

where $\widetilde{w}_{P/Q}(x) = w_{P/Q}(x) / \sum_{i=1}^{N} w_{P/Q}(x_i)$ is the self-normalized importance weight. $\widetilde{\mu}_{P/Q}$ is a biased estimator but it is consistent [30] and, since $\left|\widetilde{\mu}_{P/Q}\right| \leq \|f\|_\infty$, it has always finite variance. This typically leads to a more desirable behavior. The SN estimator has the following interpretation: it can be seen

as the expected value of f under an approximation of the distribution P made by N deltas, i.e., $\widetilde{p}(x) = \sum_{i=1}^{N} \widetilde{w}_{P/Q}(x)\delta(x - x_i)$. The behavior of the SN estimator can be studied using diagnostic indices [30]. The *effective sample size* (ESS) was introduced in [19] and it represents the number of samples drawn from P such that the variance of the Monte Carlo estimator $\widetilde{\mu}_{P/P}$ is approximately equal to the variance of the SN estimator $\widetilde{\mu}_{P/Q}$ computed with N samples. The definition and its estimate are the following:

$$\mathrm{ESS}(P\|Q) = \frac{N}{\mathbb{V}\mathrm{ar}_{x\sim Q}\left[w_{P/Q}(x)\right] + 1} = \frac{N}{d_2(P\|Q)}, \tag{4.11}$$

$$\widehat{\mathrm{ESS}}(P\|Q) = \frac{1}{\sum_{i=1}^{N} \widetilde{w}_{P/Q}(x_i)^2}. \tag{4.12}$$

We can note that if $d_2(P\|Q) = 1$, i.e., $P = Q$ almost everywhere, then $\mathrm{ESS} = N$ since we are performing Monte Carlo estimation. If $P \neq Q$, then the more the dissimilarity of the distribution increases, the more the ESS decreases. In order to consider the properties of f, also other diagnostics similar to ESS have been proposed [22].

4.4 Analysis of the SN Estimator

In this section, some results regarding the bias and the variance of the self-normalized importance sampling estimator are provided. Thanks from the result in [7], we can bound the expected squared difference between non-self-normalized weight $w(x)$ and self-normalized weight $\widetilde{w}(x)$.

Lemma 4.4.1. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ and $d_2(P\|Q) < +\infty$. Let $x_1, x_2, \ldots, x_N$ i.i.d. random variables sampled from Q. Then, for $N > 0$ and for any $i = 1, 2, \ldots, N$ it holds that:*

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left(\widetilde{w}_{P/Q}(x_i) - \frac{w_{P/Q}(x_i)}{N}\right)^2\right] \leq \frac{d_2(P\|Q) - 1}{N}. \tag{4.13}$$

Proof. Since $\mathbb{V}\mathrm{ar}_{x\sim Q}\left[w_{P/Q}(x)\right] = d_2(P\|Q) - 1$, with basic algebraic manipulations, we have the following:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left(\widetilde{w}_{P/Q}(x_i) - \frac{w_{P/Q}(x_i)}{N}\right)^2\right] \\
&\quad = \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left(\frac{w_{P/Q}(x_i)}{\sum_{j=1}^{N} w_{P/Q}(x_j)}\right)^2 \left(1 - \frac{\sum_{j=1}^{N} w_{P/Q}(x_j)}{N}\right)^2\right]
\end{aligned}$$

$$\leq \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left(1-\frac{\sum_{j=1}^{N} w_{P/Q}(x_j)}{N}\right)^2\right] = \mathop{\mathbb{V}\mathrm{ar}}_{\mathbf{x}\sim Q}\left[\frac{\sum_{j=1}^{N} w_{P/Q}(x_j)}{N}\right]$$
$$= \frac{1}{N}\mathop{\mathbb{V}\mathrm{ar}}_{x_1\sim Q}\left[w_{P/Q}(x_1)\right] = \frac{d_2(P\|Q)-1}{N}.$$

□

Similarly, a bound on the bias of the SN estimator can be derived.

Proposition 4.4.1. *Let P and Q be two probability measures on the measurable space $(\mathcal{X},\mathcal{F})$ such that $P \ll Q$ and $d_2(P\|Q) < +\infty$. Let $x_1, x_2, \ldots, x_N$ i.i.d. random variables sampled from Q and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < \infty$). Then, the bias of the SN estimator can be bounded as:*

$$\left|\mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{x\sim P}\left[f(x)\right]\right]\right| \leq \|f\|_\infty \min\left\{2, \sqrt{\frac{d_2(P\|Q)-1}{N}}\right\}. \tag{4.14}$$

Proof. Since $|\widetilde{\mu}_{P/Q}| \leq \|f\|_\infty$, then the bias cannot be larger than $2\|f\|_\infty$. Exploiting the fact that the IS estimator is unbiased, i.e., $\mathbb{E}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{P/Q}\right] = \mathbb{E}_{x\sim P}\left[f(x)\right]$, a bound for the bias, which vanishes as $N \to \infty$, can be derived.

$$\begin{aligned}
\left|\mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{x\sim P}\left[f(x)\right]\right]\right| &= \left|\mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{P/Q}\right]\right]\right| = \left|\mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q} - \widehat{\mu}_{P/Q}\right]\right| \\
&\leq \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left|\widetilde{\mu}_{P/Q} - \widehat{\mu}_{P/Q}\right|\right] = \\
&= \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left|\frac{\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)}{\sum_{i=1}^{N} w_{P/Q}(x_i)} - \frac{\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)}{N}\right|\right] \\
&= \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left|\frac{\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)}{\sum_{i=1}^{N} w_{P/Q}(x_i)}\right|\left|1 - \frac{\sum_{i=1}^{N} w_{P/Q}(x_i)}{N}\right|\right] \qquad (4.15) \\
&\leq \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left(\frac{\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)}{\sum_{i=1}^{N} w_{P/Q}(x_i)}\right)^2\right]^{\frac{1}{2}} \mathop{\mathbb{E}}_{\mathbf{x}\sim Q}\left[\left(1 - \frac{\sum_{i=1}^{N} w_{P/Q}(x_i)}{N}\right)^2\right]^{\frac{1}{2}} \qquad (4.16) \\
&\leq \|f\|_\infty \sqrt{\frac{d_2(P\|Q)-1}{N}}, \qquad (4.17)
\end{aligned}$$

where (4.16) follows from (4.15) by applying Cauchy-Schwartz inequality and in (4.17) the fact that $\left(\frac{\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)}{\sum_{i=1}^{N} w_{P/Q}(x_i)}\right)^2 \leq \|f\|_\infty^2$ is used. □

Since the the normalization term makes all the samples interdependent, it is not trivial to bound the variance of the SN estimator. We can exploint the boundedness of $\widetilde{\mu}_{P/Q}$ and derive some trivial bounds like: $\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right] \leq \|f\|_\infty^2$. However, this bound does not shrink with the number of samples N.

The following approximation, derived using the delta method [55, 30], has been proposed in the literature:

$$\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\widetilde{\mu}_{P/Q}\right] = \frac{1}{N}\underset{x_1\sim Q}{\mathbb{E}}\left[w_{P/Q}^2(x_1)\left(f(x_1) - \underset{x\sim P}{\mathbb{E}}\left[f(x)\right]\right)^2\right] + o(N^{-2}). \quad (4.18)$$

The Mean Squared Error (MSE) of the SN estimator, which is the sum of the variance and the bias squared, can be dicectly bounded.

Proposition 4.4.2. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ and $d_2(P\|Q) < +\infty$. Let $x_1, x_2, \ldots, x_N$ i.i.d. random variables sampled from Q and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < +\infty$). Then, the* MSE *of the SN estimator can be bounded as:*

$$\mathrm{MSE}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right] \leq 2\|f\|_\infty^2 \min\left\{2, \frac{2d_2(P\|Q) - 1}{N}\right\}. \quad (4.19)$$

Proof. Since $\widetilde{\mu}_{P/Q}$ is bounded by $\|f\|_\infty$ then its MSE cannot be larger than $4\|f\|_\infty^2$. We can sum and subtract the IS estimator $\widehat{\mu}_{P/Q}$ and manipulate the MSE expression:

$$\begin{aligned}
\mathrm{MSE}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right] &= \underset{\mathbf{x}\sim Q}{\mathbb{E}}\left[\left(\widetilde{\mu}_{P/Q} - \underset{x\sim P}{\mathbb{E}}\left[f(x)\right]\right)^2\right] \\
&= \underset{\mathbf{x}\sim Q}{\mathbb{E}}\left[\left(\widetilde{\mu}_{P/Q} - \underset{x\sim P}{\mathbb{E}}\left[f(x)\right] \pm \widehat{\mu}_{P/Q}\right)^2\right] && (4.20)\\
&\leq 2\underset{\mathbf{x}\sim Q}{\mathbb{E}}\left[\left(\widetilde{\mu}_{P/Q} - \widehat{\mu}_{P/Q}\right)^2\right] + 2\underset{\mathbf{x}\sim Q}{\mathbb{E}}\left[\left(\widehat{\mu}_{P/Q} - \underset{x\sim P}{\mathbb{E}}[f(x)]\right)^2\right] && (4.21)\\
&\leq 2\underset{\mathbf{x}\sim Q}{\mathbb{E}}\left[\left(\frac{\sum_{i=1}^N w_{P/Q}(x_i)f(x_i)}{\sum_{i=1}^N w_{P/Q}(x_i)}\right)^2\left(1 - \frac{\sum_{i=1}^N w_{P/Q}(x_i)}{N}\right)^2\right] && (4.22)\\
&+ 2\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\widehat{\mu}_{P/Q}\right] && (4.23)\\
&\leq 2\|f\|_\infty^2\underset{\mathbf{x}\sim Q}{\mathbb{E}}\left[\left(1 - \frac{\sum_{i=1}^N w_{P/Q}(x_i)}{N}\right)^2\right] + 2\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\widehat{\mu}_{P/Q}\right] && (4.24)\\
&\leq 2\|f\|_\infty^2\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\frac{\sum_{i=1}^N w_{P/Q}(x_i)}{N}\right] + 2\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\widehat{\mu}_{P/Q}\right] && (4.25)\\
&\leq 2\|f\|_\infty^2\frac{d_2(P\|Q) - 1}{N} + 2\|f\|_\infty^2\frac{d_2(P\|Q)}{N} && (4.26)\\
&= 2\|f\|_\infty^2\frac{2d_2(P\|Q) - 1}{N}, && (4.27)
\end{aligned}$$

where line (4.21) follows from line (4.20) by applying the inequality $(a+b)^2 \leq 2(a^2+b^2)$, (4.24) follows from (4.23) by observing that $\left(\frac{\sum_{i=1}^N w_{P/Q}(x_i)f(x_i)}{\sum_{i=1}^N w_{P/Q}(x_i)}\right)^2 \leq \|f\|_\infty^2$. □

4.5 Importance Sampling and Natural Gradient

We can look at a parametric distribution $P_{\boldsymbol{\omega}}$, having $p_{\boldsymbol{\omega}}$ as a density function, as a point on a probability manifold with coordinates $\boldsymbol{\omega} \in \Omega$. If $p_{\boldsymbol{\omega}}$ is differentiable, we can define the Fisher Information Matrix (FIM) [36, 2] as: $\mathcal{F}(\boldsymbol{\omega}) = \int_{\mathcal{X}} p_{\boldsymbol{\omega}}(x) \nabla_{\boldsymbol{\omega}} \log p_{\boldsymbol{\omega}}(x) \nabla_{\boldsymbol{\omega}} \log p_{\boldsymbol{\omega}}(x)^T \, \mathrm{d}x$. The FIM is an invariant metric (up to a scale), [1] on parameter space Ω, i.e., $\kappa(\boldsymbol{\omega}' - \boldsymbol{\omega})^T \mathcal{F}(\boldsymbol{\omega})(\boldsymbol{\omega}' - \boldsymbol{\omega})$ is independent on the specific parameterization and provides a second order approximation of the distance between $p_{\boldsymbol{\omega}}$ and $p_{\boldsymbol{\omega}'}$ on the probability manifold up to a scale factor $\kappa \in \mathbb{R}$. We can define the natural gradient [1, 17] for a loss function $\mathcal{L}(\boldsymbol{\omega})$, as $\widetilde{\nabla}_{\boldsymbol{\omega}} \mathcal{L}(\boldsymbol{\omega}) = \mathcal{F}^{-1}(\boldsymbol{\omega}) \nabla_{\boldsymbol{\omega}} \mathcal{L}(\boldsymbol{\omega})$, which represents the steepest ascent direction in the probability manifold. Thanks to the invariance property, there is a connection between the geometry induced by the Rényi divergence and the Fisher information metric.

Theorem 4.5.1. *Let $p_{\boldsymbol{\omega}}$ be a p.d.f. differentiable w.r.t. $\boldsymbol{\omega} \in \Omega$. Then, it holds that, for the Rényi divergence:*

$$D_\alpha(p_{\boldsymbol{\omega}'} \| p_{\boldsymbol{\omega}}) = \frac{\alpha}{2} \left(\boldsymbol{\omega}' - \boldsymbol{\omega}\right)^T \mathcal{F}(\boldsymbol{\omega}) \left(\boldsymbol{\omega}' - \boldsymbol{\omega}\right) + o(\|\boldsymbol{\omega}' - \boldsymbol{\omega}\|_2^2),$$

and for the exponentiated Rényi divergence:

$$d_\alpha(p_{\boldsymbol{\omega}'} \| p_{\boldsymbol{\omega}}) = 1 + \frac{\alpha}{2} \left(\boldsymbol{\omega}' - \boldsymbol{\omega}\right)^T \mathcal{F}(\boldsymbol{\omega}) \left(\boldsymbol{\omega}' - \boldsymbol{\omega}\right) + o(\|\boldsymbol{\omega}' - \boldsymbol{\omega}\|_2^2).$$

Proof. Let us compute the second-order Taylor expansion of the α-Rényi divergence. First, consider the term:

$$I(\boldsymbol{\omega}') = \int_{\mathcal{X}} \left(\frac{p_{\boldsymbol{\omega}'}(x)}{p_{\boldsymbol{\omega}}(x)}\right)^\alpha p_{\boldsymbol{\omega}}(x) \, \mathrm{d}x = \int_{\mathcal{X}} p_{\boldsymbol{\omega}'}(x)^\alpha p_{\boldsymbol{\omega}}(x)^{1-\alpha} \, \mathrm{d}x. \tag{4.28}$$

The gradient w.r.t. $\boldsymbol{\omega}'$ is:

$$\nabla_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}') = \int_{\mathcal{X}} \nabla_{\boldsymbol{\omega}'} p_{\boldsymbol{\omega}'}(x)^\alpha p_{\boldsymbol{\omega}}(x)^{1-\alpha} \, \mathrm{d}x = \alpha \int_{\mathcal{X}} p_{\boldsymbol{\omega}'}(x)^{\alpha-1} p_{\boldsymbol{\omega}}(x)^{1-\alpha} \nabla_{\boldsymbol{\omega}'} p_{\boldsymbol{\omega}'}(x) \, \mathrm{d}x.$$

Thus, $\nabla_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}')|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} = \mathbf{0}$. We can compute the Hessian matrix:

$$\begin{aligned}
\mathcal{H}_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}') &= \nabla_{\boldsymbol{\omega}'} \nabla_{\boldsymbol{\omega}'}^T I(\boldsymbol{\omega}') = \alpha \nabla_{\boldsymbol{\omega}'} \int_{\mathcal{X}} p_{\boldsymbol{\omega}'}(x)^{\alpha-1} p_{\boldsymbol{\omega}}(x)^{1-\alpha} \nabla_{\boldsymbol{\omega}'}^T p_{\boldsymbol{\omega}'}(x) \, \mathrm{d}x \\
&= \alpha \int_{\mathcal{X}} (\alpha - 1) p_{\boldsymbol{\omega}'}(x)^{\alpha-2} p_{\boldsymbol{\omega}}(x)^{1-\alpha} \nabla_{\boldsymbol{\omega}'} p_{\boldsymbol{\omega}'}(x) \nabla_{\boldsymbol{\omega}'}^T p_{\boldsymbol{\omega}'}(x) \, \mathrm{d}x + \\
&+ \alpha \int_{\mathcal{X}} p_{\boldsymbol{\omega}'}(x)^{\alpha-1} p_{\boldsymbol{\omega}}(x)^{1-\alpha} \mathcal{H}_{\boldsymbol{\omega}'} p_{\boldsymbol{\omega}'}(x) \, \mathrm{d}x.
\end{aligned}$$

Evaluating the Hessian in $\boldsymbol{\omega}$ we have:

$$\begin{aligned}\mathcal{H}_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}')|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} &= \alpha(\alpha-1)\int_{\mathcal{X}} p_{\boldsymbol{\omega}}(x)^{-1}\nabla_{\boldsymbol{\omega}} p_{\boldsymbol{\omega}}(x)\nabla_{\boldsymbol{\omega}}^T p_{\boldsymbol{\omega}}(x)\,\mathrm{d}x \\ &= \alpha(\alpha-1)\int_{\mathcal{X}} p_{\boldsymbol{\omega}}(x)\nabla_{\boldsymbol{\omega}}\log p_{\boldsymbol{\omega}}(x)\nabla_{\boldsymbol{\omega}}^T \log p_{\boldsymbol{\omega}}(x)\,\mathrm{d}x = \alpha(\alpha-1)\mathcal{F}(\boldsymbol{\omega}).\end{aligned}$$

Now, $D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}}) = \frac{1}{\alpha-1}\log I(\boldsymbol{\omega}')$. Thus:

$$\nabla_{\boldsymbol{\omega}'} D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} = \frac{1}{\alpha-1}\frac{\nabla_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}')}{I(\boldsymbol{\omega}')}\bigg|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} = \mathbf{0},$$

$$\begin{aligned}\mathcal{H}_{\boldsymbol{\omega}'} D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} &= \frac{1}{\alpha-1}\frac{I(\boldsymbol{\omega}')\mathcal{H}_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}') + \nabla_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}')\nabla_{\boldsymbol{\omega}'}^T I(\boldsymbol{\omega}')}{\left(I(\boldsymbol{\omega}')\right)^2}\bigg|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} \\ &= \frac{1}{\alpha-1}\mathcal{H}_{\boldsymbol{\omega}'} I(\boldsymbol{\omega}')|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} = \alpha\mathcal{F}(\boldsymbol{\omega}),\end{aligned}$$

since $I(\boldsymbol{\omega}) = 1$. Computing the gradient of $d_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})$ wrt w.r.t. $\boldsymbol{\omega}'$, we have:

$$\begin{aligned}\nabla_{\boldsymbol{\omega}'} d_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} &= \nabla_{\boldsymbol{\omega}'}\exp\left(D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})\right)|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} \\ &= \exp\left(D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})\right)\nabla_{\boldsymbol{\omega}'} D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} = \mathbf{0},\end{aligned}$$

$$\begin{aligned}&\mathcal{H}_{\boldsymbol{\omega}'} d_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} = \mathcal{H}_{\boldsymbol{\omega}'}\exp\left(D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})\right)|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} \\ &\quad= \exp\left(D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})\right)\left(\mathcal{H}_{\boldsymbol{\omega}'} D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}}) + \nabla_{\boldsymbol{\omega}'} D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})\nabla_{\boldsymbol{\omega}'}^T D_\alpha(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}})\right)|_{\boldsymbol{\omega}'=\boldsymbol{\omega}} \\ &\quad= \alpha\mathcal{F}(\boldsymbol{\omega}).\end{aligned}$$

□

Thanks to this result, we have an approximate expression for the variance of the importance weights, as $\mathbb{V}\mathrm{ar}_{x\sim p_{\boldsymbol{\omega}}}\left[w_{\boldsymbol{\omega}'/\boldsymbol{\omega}}(x)\right] = d_2(p_{\boldsymbol{\omega}'}\|p_{\boldsymbol{\omega}}) - 1 \simeq \frac{\alpha}{2}\left(\boldsymbol{\omega}'-\boldsymbol{\omega}\right)^T\mathcal{F}(\boldsymbol{\omega})\left(\boldsymbol{\omega}'-\boldsymbol{\omega}\right)$. Since performing a step in the natural gradient has a measurable effect on the variance of the importance weights, this result justifies the use of natural gradient in off-policy optimization.

Chapter 5

Optimization via Importance Sampling

5.1 Concentration Inequality

In the off-policy optimization problem [52], we want to find the best target policy π_T (or hyperpolicy ν_T), given samples collected by a behavioral policy π_B (or hyperpolicy ν_B). We can consider the following, more general, optimization problem: finding the target distribution P maximizing the expected value of a bounded function $\mathbb{E}_{x\sim P}[f(x)]$, using samples collected from a behavioral distribution Q. If we decide to optimize directly the importance sampling estimator $\widehat{\mu}_{P/Q}$ or $\widetilde{\mu}_{P/Q}$ the optimization process is going to assign as much probability mass as possible to the maximum value among $f(x_i)$. This would lead to an unreliable target policy, as the variance of the estimator could be high. In order to avoid this scenario, we decide to use a risk-averse approach, optimizing a statistical *lower bound* of the expected value $\mathbb{E}_{x\sim P}[f(x)]$ which holds with high confidence. The use of a lower bound can be seen as a penalty optimization method, as it maximizes the IS estimator but it penalizes for choices of the target distribution leading to a high variance. The connection between the variance of the estimator and the dissimilarity of the distributions can be expressed in terms of the Rényi divergence. Indeed, studying the behavior of the IS estimator, we derive the following bound on the variance of $\widehat{\mu}_{P/Q}$ in terms of the Renyi divergence.

Lemma 5.1.1. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$. Let $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ i.i.d. random variables sampled from Q and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < +\infty$). Then, for any $N > 0$, the variance of the IS estimator $\widehat{\mu}_{P/Q}$ can be*

upper bounded as:

$$\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\widehat{\mu}_{P/Q}\right] \leq \frac{1}{N}\|f\|_\infty^2 d_2\left(P\|Q\right). \tag{5.1}$$

Proof. Since x_i are i.i.d. we can write:

$$\begin{aligned}\underset{\mathbf{x}\sim Q}{\mathbb{V}\mathrm{ar}}\left[\widehat{\mu}_{P/Q}\right] &\leq \frac{1}{N}\underset{x_1\sim Q}{\mathbb{V}\mathrm{ar}}\left[\frac{p(x_1)}{q(x_1)}f(x_1)\right] \leq \frac{1}{N}\underset{x_1\sim Q}{\mathbb{E}}\left[\left(\frac{p(x_1)}{q(x_1)}f(x_1)\right)^2\right] \\ &\leq \frac{1}{N}\|f\|_\infty^2\underset{x_1\sim Q}{\mathbb{E}}\left[\left(\frac{p(x_1)}{q(x_1)}\right)^2\right] = \frac{1}{N}\|f\|_\infty^2 d_2\left(P\|Q\right).\end{aligned}$$

□

When $P = Q$ almost everywhere, we get $\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{Q/Q}\right] \leq \frac{1}{N}\|f\|_\infty^2$, the bound on the variance of a Monte Carlo estimator. Thanks to the relation between the Rényi divergence and the Effective Sample Size (ESS) (4.12), we can write the bound in terms of the ESS: $\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{P/Q}\right] \leq \frac{\|f\|_\infty^2}{\mathrm{ESS}(P\|Q)}$, i.e., the variance scales with ESS instead of N.

Several statistical lower bound have been proposed in the literature for offline policy evaluation [53] and for optimization [52]. However, studying the properties of the IS estimator, we discover that many of the assumptions required by these lower bounds are failing to hold, or can introduce unacceptable limitations.

Student-T inequality

A first possible choice could be to rely on the Central Limit Theorem (CLT). If we assume to have $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ i.i.d. random variables with finite variance $\mathbb{V}\mathrm{ar}[\widehat{\mu}_{P/Q}] = \sigma^2$, then, applying the CLT, for N sufficiently large, we have:

$$\widehat{\mu}_{P/Q} \sim \mathcal{N}\left(\mu_{P/Q}, \frac{\sigma^2}{N}\right)$$

This would lead to the following statistical lower bound:

Theorem 5.1.1. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ and $\mathbb{V}\mathrm{ar}[\widehat{\mu}_{P/Q}] = \sigma^2 < +\infty$. Let $x_1, x_2, \ldots, x_N$ be i.i.d. random variables sampled from Q, and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < +\infty$). Then, for N sufficiently large and for any $0 < \delta \leq 1$, with probability at least $1 - \delta$ it holds that:*

$$\underset{x\sim P}{\mathbb{E}}\left[f(x)\right] \geq \frac{1}{N}\sum_{i=1}^{N} w_{P/Q}(x_i)f(x_i) - \frac{\sigma}{\sqrt{N}}t_{1-\delta,N-1} \tag{5.2}$$

However, the value of σ is unknown and must be estimate using Importance Sampling. Choosing this bound would introduce further uncertainty in the optimization process and would lead to a not reliable estimate with high variance.

Hoeffding Inequality
Hoeffding inequality assumes that we know the maximum of the estimator. Under this assumption, we have the following lower bound:

Theorem 5.1.2. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ and $d_\infty(P\|Q) = M < +\infty$. Let $x_1, x_2, \ldots, x_N$ be i.i.d. random variables sampled from Q, and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < +\infty$). Then, for any $0 < \delta \leq 1$ and $N > 0$ with probability at least $1 - \delta$ it holds that:*

$$\mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \frac{1}{N}\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i) - M\sqrt{\frac{\log(1/\delta)}{2N}} \tag{5.3}$$

Constraining this value to be finite, i.e., $||\widehat{\mu}_{P/Q}||_\infty = M < +\infty$, is equivalent to constraining $d_\infty(P\|Q) < +\infty$. Unfortunately, even in the particular case of univariate Gaussian distributions, this lead to an unacceptable limitation: as we reported in Section 4.1, this constraint does not allow the optimization process to reach a target policy whose variance is higher than the behavioral one. Hence, the variance of the distribution is going to decrease at every optimization step, which might be a problem if we want our policy to be able to explore the environment.

Bernstein Inequality
Bernstein-like inequalities, which can also be used under Hoeffding's assumptions, assume that the estimator has a sub-exponential distribution, which means, in particular, that its tail has to converge to zero at least as fast as an exponential distribution. Thanks to the analysis on the IS distribution in Section 4.1, we discover that this is not the case. Indeed, the tail of the distribution can be lower bounded by a logarithmic function, hence displaying a fat-tail behavior. This prevents from using bounds which rely on Bernstein inequality.

Cantelli's inequality
We decide to rely on Chebyshev-like inequalities, in particular on Cantelli's inequality, which assumes that the variance of the importance weights has to be finite, i.e., $d_2(P\|Q) < +\infty$. Under this condition, when using univariate Gaussian distributions, the target distribution can have a standard deviation

which is at most two times the behavioral one, thus the policy can still explore sufficiently the environment.

Theorem 5.1.3. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ and $d_2(P\|Q) < +\infty$. Let $x_1, x_2, \ldots, x_N$ be i.i.d. random variables sampled from Q, and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < +\infty$). Then, for any $0 < \delta \leq 1$ and $N > 0$ with probability at least $1 - \delta$ it holds that:*

$$\mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \frac{1}{N}\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i) - \|f\|_\infty \sqrt{\frac{(1-\delta) d_2(P\|Q)}{\delta N}}. \tag{5.4}$$

Proof. We take the random variable $\widehat{\mu}_{P/Q} = \frac{1}{N}\sum_{i=1}^{N} w_{P/Q}(x_i) f(x_i)$ and apply Cantelli's inequality:

$$\Pr\left(\widehat{\mu}_{P/Q} - \mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \lambda\right) \leq \frac{1}{1 + \frac{\lambda^2}{\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{P/Q}\right]}}. \tag{5.5}$$

Set $\delta = \frac{1}{1+\frac{\lambda^2}{\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{P/Q}\right]}}$ and consider the complementary event. Then, with probability at least $1 - \delta$, we have:

$$\mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \widehat{\mu}_{P/Q} - \sqrt{\frac{1-\delta}{\delta}\mathop{\mathbb{V}\mathrm{ar}}_{\mathbf{x}\sim Q}\left[\widehat{\mu}_{P/Q}\right]}. \tag{5.6}$$

By replacing the variance with the bound in Theorem 5.1.3 we get the result. □

In this bound, we can see a trade-off between the estimated performance and the variability introduced by changing the distribution. The main intuition is that we should optimize the unbiased IS estimator $\widehat{\mu}_{P/Q}$, but we should penalize for optimization steps leading to a target distribution which is too dissimilar w.r.t. the behavioral one. In such cases, the estimator is not reliable and the learning process could be highly unstable. The penalization term is expressed in terms of the Rényi divergence of order two between the target distribution P and the behavioral distribution Q. When using the SN estimator, accounting for the bias, we can derive a similar lower bound, hence, the optimization process is the same. We can condense all the constant of the lower bound of Theorem 5.1.3 in $\lambda = \|f\|_\infty \sqrt{(1-\delta)/\delta}$ and obtain a surrogate objective function.

Thanks to this result, a high confidence bound for the SN estimator, using Cantelli's inequality, can be derived.

Proposition 5.1.1. *Let P and Q be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ such that $P \ll Q$ and $d_2(P\|Q) < +\infty$. Let $x_1, x_2, \ldots, x_N$ i.i.d. random variables sampled from Q and $f : \mathcal{X} \to \mathbb{R}$ be a bounded function ($\|f\|_\infty < +\infty$). Then, for any $0 < \delta \leq 1$ and $N > 0$ with probability at least $1 - \delta$:*

$$\mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \frac{1}{N}\sum_{i=1}^{N} \widetilde{w}_{P/Q}(x_i) f(x_i) - 2\|f\|_\infty \min\left\{1, \sqrt{\frac{d_2(P\|Q)(4-3\delta)}{\delta N}}\right\}.$$

Proof. In order to obtain this result we have to apply Cantelli's inequality and to account for the bias. Consider the random variable $\widetilde{\mu}_{P/Q} = \frac{1}{N}\sum_{i=1}^{N} \widetilde{w}_{P/Q}(x_i) f(x_i)$ and let $\widetilde{\lambda} = \lambda - \left|\mathbb{E}_{x\sim P}[f(x)] - \mathbb{E}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right|$:

$$\begin{aligned}
\Pr\left(\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \lambda\right) &= \Pr\left(\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right] \geq \lambda + \mathop{\mathbb{E}}_{x\sim P}[f(x)] - \mathop{\mathbb{E}}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right) \\
&\leq \Pr\left(\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right] \geq \lambda - \left|\mathop{\mathbb{E}}_{x\sim P}[f(x)] - \mathop{\mathbb{E}}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right|\right) \\
&= \Pr\left(\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right] \geq \widetilde{\lambda}\right).
\end{aligned}$$

We can apply Cantelli's inequality:

$$\begin{aligned}
\Pr\left(\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \lambda\right) &\leq \Pr\left(\widetilde{\mu}_{P/Q} - \mathop{\mathbb{E}}_{x\sim P}\left[\widetilde{\mu}_{P/Q}\right] \geq \widetilde{\lambda}\right) && (5.7) \\
&\leq \frac{1}{1 + \frac{\widetilde{\lambda}^2}{\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right]}} \\
&= \frac{1}{1 + \frac{\left(\lambda - \left|\mathbb{E}_{x\sim P}[f(x)] - \mathbb{E}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right|\right)^2}{\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right]}}. && (5.8)
\end{aligned}$$

Set $\delta = \frac{1}{1+\frac{\left(\lambda - \left|\mathbb{E}_{x\sim P}[f(x)] - \mathbb{E}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right|\right)^2}{\mathbb{V}\mathrm{ar}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right]}}$ and consider the complementary event:
then, with probability at least $1 - \delta$, we have:

$$\mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \widetilde{\mu}_{P/Q} - \left|\mathop{\mathbb{E}}_{x\sim P}[f(x)] - \mathop{\mathbb{E}}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right| - \sqrt{\frac{1-\delta}{\delta}\mathop{\mathbb{V}\mathrm{ar}}_{\mathbf{x}\sim Q}\left[\widetilde{\mu}_{P/Q}\right]} \quad (5.9)$$

The bias term $\left|\mathbb{E}_{x\sim P}[f(x)] - \mathbb{E}_{\mathbf{x}\sim P}\left[\widetilde{\mu}_{P/Q}\right]\right|$ can be bounded using equation (4.14) and the variance term can be bounded using the MSE in equation (4.19). Then we have:

$$\mathop{\mathbb{E}}_{x\sim P}[f(x)] \geq \widetilde{\mu}_{P/Q} - \|f\|_\infty\sqrt{\frac{d_2(P\|Q) - 1}{N}} - \|f\|_\infty\sqrt{\frac{1-\delta}{\delta}\frac{2(2d_2(P\|Q) - 1)}{N}}$$

$$
\begin{aligned}
&\geq \widetilde{\mu}_{P/Q} - \|f\|_\infty \sqrt{\frac{d_2(P\|Q)}{N}} - \|f\|_\infty \sqrt{\frac{1-\delta}{\delta}\frac{4d_2(P\|Q)}{N}} \\
&= \widetilde{\mu}_{P/Q} - \|f\|_\infty \sqrt{\frac{d_2(P\|Q)}{N}} \left(1 + 2\sqrt{\frac{1-\delta}{\delta}}\right) \\
&\geq \widetilde{\mu}_{P/Q} - 2\|f\|_\infty \sqrt{\frac{d_2(P\|Q)}{N}} \sqrt{1 + \frac{4(1-\delta)}{\delta}} \\
&\geq \widetilde{\mu}_{P/Q} - 2\|f\|_\infty \sqrt{\frac{d_2(P\|Q)(4-3\delta)}{\delta N}},
\end{aligned}
$$

when in the last line the fact that $\sqrt{a} + \sqrt{b} \leq 2\sqrt{a+b}$ for any $a, b \geq 0$ is used. Finally, since the range of the SN estimator is $2\|f\|_\infty$, we obtain the result. □

Note that the bound has the same dependence on d_2 as in Theorem 5.1.3. This allows to optimize the same surrogate objective function for both IS and SN estimators.

Algorithm 1 Action-based POIS

Initialize $\boldsymbol{\theta}_0^0$ arbitrarily
for $j = 0, 1, 2, ...,$ until convergence **do**
  Collect N trajectories with $\pi_{\boldsymbol{\theta}_0^j}$
  for $k = 0, 1, 2, ...,$ until convergence **do**
    Compute $\mathcal{G}(\boldsymbol{\theta}_k^j)$, $\nabla_{\boldsymbol{\theta}_k^j}\mathcal{L}(\boldsymbol{\theta}_k^j/\boldsymbol{\theta}_0^j)$ and α_k
    $\boldsymbol{\theta}_{k+1}^j = \boldsymbol{\theta}_k^j + \alpha_k \mathcal{G}(\boldsymbol{\theta}_k^j)^{-1}\nabla_{\boldsymbol{\theta}_k^j}\mathcal{L}(\boldsymbol{\theta}_k^j/\boldsymbol{\theta}_0^j)$
  end for
  $\boldsymbol{\theta}_0^{j+1} = \boldsymbol{\theta}_k^j$
end for

5.2 Policy Optimization via Importance Sampling

The bound we found in Theorem 5.1.3 can be adapted to the Reinforcement Learning setting. In particular, in this section, we see how we can customize it for policy optimization. This leads to a novel, model-free, actor-only, policy search algorithm, which we call *Policy Optimization via Importance Sampling* (POIS). The proposed bound is adapted both in the Action-based and Parameter-based settings. We call these two variants A-POIS and P-POIS, respectively.

5.2.1 Action-based POIS

In Action-based methods, we want to find the policy parameters $\boldsymbol{\theta}^*$ maximizing $J_D(\boldsymbol{\theta})$ within a parametric space $\Pi_\Theta = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p\}$ of stochastic differentiable policies. In order to use the results obtained in the previous section, and to adapt the surrogate objective function, we substitute the behavioral (resp. target) distribution Q (resp. P) with the behavioral (resp. target) distribution over the trajectories $p(\cdot|\boldsymbol{\theta})$ (resp. $p(\cdot|\boldsymbol{\theta}')$) generated by the behavioral policy $\pi_{\boldsymbol{\theta}}$ (resp. target policy $\pi_{\boldsymbol{\theta}'}$). In this setting, the uniformly bounded function f becomes the trajectory return $R(\tau)$ (it is uniformly bounded since $|R(\tau)| \leq R_{\max}\frac{1-\gamma^H}{1-\gamma}$).

5.2.2 Estimation of the Rényi divergence

The surrogate objective function in A-POIS contains the Rényi divergence between the target distribution $p(\cdot|\boldsymbol{\theta}')$ and the behavioral distribution $p(\cdot|\boldsymbol{\theta})$

in terms of the trajectories. Computing this term is intractable as it requires to integrate over all the space of trajectories. When the agent is acting in stochastic environments, computing this term requires also to know the transition model $\mathcal{P}$, which is unknown in the model-free setting. We can provide an exact bound on the Rényi divergence over trajectories when the horizon of the tasks is finite.

Proposition 5.2.1. *Let $p(\cdot|\boldsymbol{\theta})$ and $p(\cdot|\boldsymbol{\theta}')$ be the behavioral and target trajectory probability density functions. Let $H < \infty$ be the task-horizon. Then, it holds that:*

$$d_\alpha\left(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\right) \leq \left(\sup_{s\in\mathcal{S}} d_\alpha\left(\pi_{\boldsymbol{\theta}'}(\cdot|s)\|\pi_{\boldsymbol{\theta}}(\cdot|s)\right)\right)^H.$$

Proof. The proposition is proven by induction on the horizon H. Let us define $d_{\alpha,H}$ as the α-Rényi divergence at horizon H. For $H = 1$ we have:

$$\begin{aligned}
d_{\alpha,1}\left(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\right) &= \int_{\mathcal{S}} D(s_0) \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_0|s_0) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_0|s_0)}{\pi_{\boldsymbol{\theta}}(a_0|s_0)}\right)^\alpha \int_{\mathcal{S}} P(s_1|s_0,a_0)\,\mathrm{d}s_1\,\mathrm{d}a_0\,\mathrm{d}s_0 \\
&= \int_{\mathcal{S}} D(s_0) \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_0|s_0) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_0|s_0)}{\pi_{\boldsymbol{\theta}}(a_0|s_0)}\right)^\alpha \mathrm{d}a_0\,\mathrm{d}s_0 \\
&\leq \int_{\mathcal{S}} D(s_0)\,\mathrm{d}s_0 \sup_{s\in\mathcal{S}} \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_0|s) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_0|s)}{\pi_{\boldsymbol{\theta}}(a_0|s)}\right)^\alpha \mathrm{d}a_0 \\
&\leq \sup_{s\in\mathcal{S}} d_\alpha\left(\pi_{\boldsymbol{\theta}'}(\cdot|s)\|\pi_{\boldsymbol{\theta}}(\cdot|s)\right),
\end{aligned}$$

where in the last but one passage Holder's inequality is used. Assume that the proposition holds for any $H' < H$. We need to prove that it holds for H.

$$\begin{aligned}
&d_{\alpha,H}\Big(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\Big) = \\
&\quad= \int_{\mathcal{S}} D(s_0) \cdots \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_{H-2}|s_{H-2}) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_{H-2}|s_{H-2})}{\pi_{\boldsymbol{\theta}}(a_{H-2}|s_{H-2})}\right)^\alpha \int_{\mathcal{S}} P(s_{H-1}|s_{H-2},a_{H-2}) \\
&\qquad\times \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_{H-1}|s_{H-1}) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_{H-1}|s_{H-1})}{\pi_{\boldsymbol{\theta}}(a_{H-1}|s_{H-1})}\right)^\alpha \int_{\mathcal{S}} P(s_H|s_{H-1},a_{H-1})\,\mathrm{d}s_0 \ldots \mathrm{d}s_{H-1} \\
&\qquad\times \mathrm{d}a_{H-2}\,\mathrm{d}s_{H-1}\,\mathrm{d}a_{H-1}\,\mathrm{d}s_H \\
&\quad= \int_{\mathcal{S}} D(s_0) \cdots \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_{H-2}|s_{H-2}) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_{H-2}|s_{H-2})}{\pi_{\boldsymbol{\theta}}(a_{H-2}|s_{H-2})}\right)^\alpha \int_{\mathcal{S}} P(s_{H-1}|s_{H-2},a_{H-2}) \\
&\qquad\times \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_{H-1}|s_{H-1}) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_{H-1}|s_{H-1})}{\pi_{\boldsymbol{\theta}}(a_{H-1}|s_{H-1})}\right)^\alpha \mathrm{d}s_0 \ldots \mathrm{d}s_{H-1}\,\mathrm{d}a_{H-2}\,\mathrm{d}s_{H-1}\,\mathrm{d}a_{H-1} \\
&\quad\leq \int_{\mathcal{S}} D(s_0) \cdots \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_{H-2}|s_{H-2}) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_{H-2}|s_{H-2})}{\pi_{\boldsymbol{\theta}}(a_{H-2}|s_{H-2})}\right)^\alpha \int_{\mathcal{S}} P(s_{H-1}|s_{H-2},a_{H-2}) \\
&\qquad\times \mathrm{d}s_0 \ldots \mathrm{d}s_{H-1}\,\mathrm{d}a_{H-2}\,\mathrm{d}s_{H-1} \times \sup_{s\in\mathcal{S}} \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a_{H-1}|s) \left(\frac{\pi_{\boldsymbol{\theta}'}(a_{H-1}|s)}{\pi_{\boldsymbol{\theta}}(a_{H-1}|s)}\right)^\alpha \mathrm{d}a_{H-1} \\
&\quad\leq d_{\alpha,H-1}\left(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\right) \sup_{s\in\mathcal{S}} d_\alpha\left(\pi_{\boldsymbol{\theta}'}(\cdot|s)\|\pi_{\boldsymbol{\theta}}(\cdot|s)\right)
\end{aligned}$$

$$\leq \left(\sup_{s\in\mathcal{S}} d_\alpha\left(\pi_{\boldsymbol{\theta}'}(\cdot|s)\|\pi_{\boldsymbol{\theta}}(\cdot|s)\right)\right)^H,$$

where, as before, Holder's inequality is applied and, in the last step, we use the inductive hypothesis. □

Unfortunately, this bound is very conservative and it requires to compute the supremum among all the states. For A-POIS, we need to find an estimate of the Rényi divergence. We can obtain an estimator using a sample-based version of the definition (4.2) of the Rényi divergence:

$$\widehat{d}_\alpha\left(P\|Q\right) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{p(x_i)}{q(x_i)}\right)^\alpha = \frac{1}{N}\sum_{i=1}^{N} w_{P/Q}^\alpha(x_i), \tag{5.10}$$

where $x_i \sim Q$. This estimator is unbiased, but it requires to use distribution over trajectories. We can express in exact form these probabilities in terms of the policies, providing the following formula for the α-Rényi divergence:

$$d_\alpha\left(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\right) = \int_{\mathcal{T}} p(\cdot|\boldsymbol{\theta})(\tau)\left(\frac{p(\tau|\boldsymbol{\theta}')}{p(\tau|\boldsymbol{\theta})}\right)^\alpha \mathrm{d}\tau =$$
$$= \int_{\mathcal{T}} D(s_{\tau,0})\prod_{t=0}^{H-1} P(s_{\tau,t+1}|s_{\tau,t},a_{\tau,t})\prod_{t=0}^{H-1}\pi_{\boldsymbol{\theta}}(a_{\tau,t}|s_{\tau,t})\left(\frac{\pi_{\boldsymbol{\theta}'}(a_{\tau,t}|s_{\tau,t})}{\pi_{\boldsymbol{\theta}}(a_{\tau,t}|s_{\tau,t})}\right)^\alpha \mathrm{d}\tau.$$

The term $d_\alpha\left(\pi_{\boldsymbol{\theta}'}(\cdot|s)\|\pi_{\boldsymbol{\theta}}(\cdot|s)\right)$ can be computed exactly since both $\pi_{\boldsymbol{\theta}}$ and $\pi_{\boldsymbol{\theta}'}$ are known. Thus, we propose the following estimate for the Rényi divergence between two distributions on trajectories:

$$\widehat{d}_\alpha\left(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\right) = \frac{1}{N}\sum_{i=1}^{N}\prod_{t=0}^{H-1} d_\alpha\left(\pi_{\boldsymbol{\theta}'}(\cdot|s_{\tau_i,t})\|\pi_{\boldsymbol{\theta}}(\cdot|s_{\tau_i,t})\right). \tag{5.11}$$

Thus, we obtain the following surrogate objective:

$$\mathcal{L}_\lambda^{\mathrm{A-POIS}}(\boldsymbol{\theta}'/\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(\tau_i)R(\tau_i) - \lambda\sqrt{\frac{\widehat{d}_2\left(p(\cdot|\boldsymbol{\theta}')\|p(\cdot|\boldsymbol{\theta})\right)}{N}}, \tag{5.12}$$

where $w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(\tau_i) = \frac{p(\tau_i|\boldsymbol{\theta}')}{p(\tau_i|\boldsymbol{\theta})} = \prod_{t=0}^{H-1}\frac{\pi_{\boldsymbol{\theta}'}(a_{\tau_i,t}|s_{\tau_i,t})}{\pi_{\boldsymbol{\theta}}(a_{\tau_i,t}|s_{\tau_i,t})}$. We focus in the particular case in which $\pi_{\boldsymbol{\theta}}(\cdot|s)$ is a Gaussian distribution whose mean is a function of the state and the diagonal covariance is state-independent: $\mathcal{N}(u_{\boldsymbol{\mu}}(s), \mathrm{diag}(\boldsymbol{\sigma}^2))$, where $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\sigma})$. Online and offline optimization steps alternate in learning. At each online iteration j, we use the current behavioral policy $\pi_{\boldsymbol{\theta}_0^j}$ to interact with the environment and collect a batch of trajectories. These trajectories

Algorithm 2 Parameter-based POIS

Initialize $\boldsymbol{\rho}_0^0$ arbitrarily
for $j = 0, 1, 2, ...,$ until convergence **do**
  Sample N policy parameters $\boldsymbol{\theta}_i^j$ from $\nu_{\boldsymbol{\rho}_0^j}$
  Collect a trajectory with each $\pi_{\boldsymbol{\theta}_i^j}$
  for $k = 0, 1, 2, ...,$ until convergence **do**
    Compute $\mathcal{G}(\boldsymbol{\rho}_k^j)$, $\nabla_{\boldsymbol{\rho}_k^j}\mathcal{L}(\boldsymbol{\rho}_k^j/\boldsymbol{\rho}_0^j)$ and α_k
    $\boldsymbol{\rho}_{k+1}^j = \boldsymbol{\rho}_k^j + \alpha_k \mathcal{G}(\boldsymbol{\rho}_k^j)^{-1}\nabla_{\boldsymbol{\rho}_k^j}\mathcal{L}(\boldsymbol{\rho}_k^j/\boldsymbol{\rho}_0^j)$
  end for
  $\boldsymbol{\rho}_0^{j+1} = \boldsymbol{\rho}_k^j$
end for

are used offline to optimize the surrogate objective function $\mathcal{L}_\lambda^{\text{A-POIS}}$ via gradient ascent: $\boldsymbol{\theta}_{k+1}^j = \boldsymbol{\theta}_k^j + \alpha_k \mathcal{G}(\boldsymbol{\theta}_k^j)^{-1}\nabla_{\boldsymbol{\theta}_k^j}\mathcal{L}(\boldsymbol{\theta}_k^j/\boldsymbol{\theta}_0^j)$, where $\alpha_k > 0$ is the step size, selected using a line search method (see Section 5.3.1) and $\mathcal{G}(\boldsymbol{\theta}_k^j)$ is a general positive semi-definite matrix (e.g., $\mathcal{F}(\boldsymbol{\theta}_k^j)$, the FIM, for natural gradient). The pseudo-code of A-POIS is reported in Algorithm 1.

5.2.3 Parameter-based POIS

In Parameter-based methods, we want to find the hyperpolicy parameters ρ^* maximizing $J_D(\boldsymbol{\rho})$ within a parametric hyperpolicy space, which we call $\mathcal{N}_\mathcal{P} = \{\nu_{\boldsymbol{\rho}} : \boldsymbol{\rho} \in \mathcal{P} \subseteq \mathbb{R}^r\}$ of stochastic differentiable hyperpolicies. In this setting, the policy $\pi_{\boldsymbol{\theta}'}$, which is assumed to be deterministic ($\pi_{\boldsymbol{\theta}}(a|s) = \delta(a - u_{\boldsymbol{\theta}}(s))$, where $u_{\boldsymbol{\theta}}$ is a deterministic function of the state s[42, 14]), is not required to be differentiable. In order to use the results obtained in the previous section, and to adapt the surrogate objective function, we substitute the behavioral (resp. target) distribution Q (resp. P) with the behavioral (resp. target) hyperpolicy $\nu_{\boldsymbol{\rho}}$ (resp. $\nu_{\boldsymbol{\rho}'}$. As in A-POIS, the uniformly bounded function f becomes the trajectory return $R(\tau)$. At the beginning of each episode, the the policy parameters $\boldsymbol{\theta}$ are sampled from the hyperpolicy. The importance weight for P-POIS becomes τ: $w_{\boldsymbol{\rho}'/\boldsymbol{\rho}}(\boldsymbol{\theta}) = \frac{\nu_{\boldsymbol{\rho}'}(\boldsymbol{\theta})p(\tau|\boldsymbol{\theta})}{\nu_{\boldsymbol{\rho}}(\boldsymbol{\theta})p(\tau|\boldsymbol{\theta})} = \frac{\nu_{\boldsymbol{\rho}'}(\boldsymbol{\theta})}{\nu_{\boldsymbol{\rho}}(\boldsymbol{\theta})}$. Thus, we obtain the following surrogate objective:

$$\mathcal{L}_\lambda^{\text{P-POIS}}(\boldsymbol{\rho}'/\boldsymbol{\rho}) = \frac{1}{N}\sum_{i=1}^{N} w_{\boldsymbol{\rho}'/\boldsymbol{\rho}}(\boldsymbol{\theta}_i)R(\tau_i) - \lambda\sqrt{\frac{d_2\left(\nu_{\boldsymbol{\rho}'}\|\nu_{\boldsymbol{\rho}}\right)}{N}}. \tag{5.13}$$

We focus in the particular case in which $\nu_{\boldsymbol{\rho}}$ is a Gaussian with diagonal

covariance matrix, i.e., $\mathcal{N}(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2))$ with $\boldsymbol{\rho} = (\boldsymbol{\mu}, \boldsymbol{\sigma})$.

As in A-POIS, learning uses online and offline optimization steps. At each online iteration j, we use the current behavioral hyperpolicy $\nu_{\boldsymbol{\rho}_0^j}$ to interact with the environment and collect a batch of trajectories. These trajectories are used offline to optimize the surrogate objective function $\mathcal{L}_\lambda^{\text{P-POIS}}$ via gradient ascent: $\boldsymbol{\rho}_{k+1}^j = \boldsymbol{\rho}_k^j + \alpha_k \mathcal{G}(\boldsymbol{\rho}_k^j)^{-1} \nabla_{\boldsymbol{\rho}_k^j} \mathcal{L}(\boldsymbol{\rho}_k^j / \boldsymbol{\rho}_0^j)$, where $\alpha_k > 0$ is the step size, selected using a line search method (see Section 5.3.1) and $\mathcal{G}(\boldsymbol{\rho}_k^j)$ is a general positive semi-definite matrix (e.g., $\mathcal{F}(\boldsymbol{\rho}_k^j)$, the FIM, for natural gradient). The pseudo-code of P-POIS is reported in Algorithm 2.

Differently from A-POIS, here the term $d_2\left(\nu_{\boldsymbol{\rho}'} \| \nu_{\boldsymbol{\rho}}\right)$ can be computed exactly. Moreover, when the hyperpolicy is a Gaussian with diagonal covariance matrix, the FIM $\mathcal{F}(\boldsymbol{\rho})$ is also diagonal and can be computed exactly [25]. This makes the use of the natural gradient much easier in P-POIS.

5.3 Implementation details

In this Section, we provide some aspects about our implementation of POIS.

5.3.1 Line Search

At every iteration k performed offline, we change the parameters in the direction of the natural gradient $\mathcal{G}(\boldsymbol{\theta}_k^j)^{-1} \nabla_{\boldsymbol{\theta}_k^j} \mathcal{L}(\boldsymbol{\theta}_k^j / \boldsymbol{\theta}_0^j)$ with a step size α_k such that the improvement is maximized. Most gradient-based methods use the step size of the optimizer as an hyperparameter to tune. We adopt a different approach and derive a line search method for finding an adaptive step-size. The main idea of this method is to locally approximate the objective function $\mathcal{L}(\boldsymbol{\theta})$, in the direction of the gradient, $\mathcal{G}^{-1}(\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$, as a concave parabola in the Riemann manifold with $\mathcal{G}(\boldsymbol{\theta})$ as Riemann metric tensor. Suppose we know an initial point $\boldsymbol{\theta}_0$, the gradient in that point $\mathcal{G}(\boldsymbol{\theta}_0)^{-1} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)$ and another point: $\boldsymbol{\theta}_l = \boldsymbol{\theta}_0 + \alpha_l \mathcal{G}(\boldsymbol{\theta}_0)^{-1} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)$. Then we can compute for both points the loss function: $\mathcal{L}_0 = \mathcal{L}(\boldsymbol{\theta}_0)$ and $\mathcal{L}_l = \mathcal{L}(\boldsymbol{\theta}_l)$ and define as $\Delta\mathcal{L}_l = \mathcal{L}_l - \mathcal{L}_0$ the objective function improvement. In this approximation setting, thanks to this assumption, we can compute the global maximum, which corresponds to the vertex of the parabola. We can define the parabola $l(\alpha) = \mathcal{L}\left(\boldsymbol{\theta}_0 + \alpha \mathcal{G}^{-1}(\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\right) - \mathcal{L}(\boldsymbol{\theta}_0)$, which can be written as $l(\alpha) = a\alpha^2 + b\alpha + c$. Since, by definition of $l(\alpha)$, we have that $c = 0$, we can compute the value of a and b as follows:

$$b = \left.\frac{\partial l}{\partial \alpha}\right|_{\alpha=0} = \frac{\partial}{\partial \alpha} \mathcal{L}\left(\boldsymbol{\theta}_0 + \alpha \mathcal{G}^{-1}(\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\right) - \mathcal{L}(\boldsymbol{\theta}_0)|_{\alpha=0} =$$

$$= \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)^T \mathcal{G}^{-1}(\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0) =$$
$$= \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)},$$

$$l(\alpha_l) = a\alpha_l^2 + b\alpha_l = a\alpha_l^2 + \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l = \Delta\mathcal{L}_l \quad \Longrightarrow$$
$$\Longrightarrow \quad a = \frac{\Delta\mathcal{L}_l - \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l}{\alpha_l^2}.$$

Therefore, the parabola has the form:

$$l(\alpha) = \frac{\Delta\mathcal{L}_l - \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l}{\alpha_l^2} \alpha^2 + \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha. \tag{5.14}$$

The parabola is concave only if $\Delta\mathcal{L}_l < \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l$. We can compute the position of the vertex as:

$$\alpha_{l+1} = \frac{\|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l^2}{2\left(\|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l - \Delta\mathcal{L}_l\right)}. \tag{5.15}$$

As in [23], we can define $\alpha_l = \epsilon_l / \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)}$ and have a simplified expression. Then we have:

$$\epsilon_{l+1} = \frac{\epsilon_l^2}{2(\epsilon_l - \Delta\mathcal{L}_l)}. \tag{5.16}$$

In the case where the parabola is convex, i.e., $\Delta\mathcal{L}_l \geq \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l$, we can distinguish between two cases: i) $\Delta\mathcal{L}_l > \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l$, the function is sublinear and in this case we use (5.16) to determine the new step size $\alpha_{l+1} = \epsilon_{l+1} / \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)}$; ii) $\Delta\mathcal{L}_l \geq \|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)} \alpha_l$, the function is superlinear, in this case we increase the step size multiplying by $\eta > 1$, i.e., $\alpha_{l+1} = \eta\alpha_l$. We can condense the update procedure as:

$$\epsilon_{l+1} = \begin{cases} \eta\epsilon_l & \text{if } \Delta\mathcal{L}_l > \frac{\epsilon_l(2\eta-1)}{2\eta} \\ \frac{\epsilon_l^2}{2(\epsilon_l - \Delta\mathcal{L}_l)} & \text{otherwise} \end{cases}. \tag{5.17}$$

We iterate this process for a fixed amount of iterations, or until the objective function improvement is too small. Pseudocode is reported in Algorithm 3.

Algorithm 3 Parabolic Line Search

Input: $\text{tol}_{\Delta\mathcal{L}} = 1e-4$, $M_{\text{ls}} = 30$, $\mathcal{L}_0$
Output : α^*

$\alpha_0 = 0$
$\epsilon_1 = 1$
$\Delta\mathcal{L}_{k-1} = -\infty$
for $l = 1, 2, \dots, M_{\text{ls}}$ **do**
  $\alpha_l = \epsilon_l / \|\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)\|^2_{\mathcal{G}^{-1}(\boldsymbol{\theta}_0)}$
  $\boldsymbol{\theta}_l = \alpha_l \mathcal{G}^{-1}(\boldsymbol{\theta}_0)\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)$
  $\Delta\mathcal{L}_l = \mathcal{L}_l - \mathcal{L}_0$
  if $\Delta\mathcal{L}_l < \Delta\mathcal{L}_{l-1} + \text{tol}_{\Delta\mathcal{L}}$ **then**
    return α_{l-1}
  end if
  $\epsilon_{l+1} = \begin{cases} \eta\epsilon_l & \text{if } \Delta\mathcal{L}_l > \frac{\epsilon_l(1-2\eta)}{2\eta} \\ \frac{\epsilon_l^2}{2(\epsilon_l - \Delta\mathcal{L}_l)} & \text{otherwise} \end{cases}$
end for

5.3.2 Computation of the Fisher Matrix

In A-POIS we cannot obtain the exact expression of the Fisher Information Matrix and we need to estimate it off-policy from samples. Using the IS estimator, we obtain the following estimate for the FIM:

$$\widehat{\mathcal{F}}(\boldsymbol{\theta}'/\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(\tau_i)\left(\sum_{t=0}^{H-1}\nabla_{\boldsymbol{\theta}'}\log\pi_{\boldsymbol{\theta}'}(a_{\tau_i,t}|s_{\tau_i,t})\right)^T\left(\sum_{t=0}^{H-1}\nabla_{\boldsymbol{\theta}'}\log\pi_{\boldsymbol{\theta}'}(a_{\tau_i,t}|s_{\tau_i,t})\right).$$

We can replace $w_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(\tau_i)$ with $\widetilde{w}_{\boldsymbol{\theta}'/\boldsymbol{\theta}}(\tau_i)$ in order to obtain an estimate of the FIM using the SN estimator. Unfortunately, when $\boldsymbol{\theta}'$ is far from $\boldsymbol{\theta}$, these estimators are not reliable due to the high variance induced by the IS process. On the contrary, in P-POIS, when Gaussian hyperpolicies are used, we can compute the FIM exactly [47]. If the hyperpolicy has diagonal covariance matrix, i.e., $\nu_{\boldsymbol{\mu},\boldsymbol{\sigma}} = \mathcal{N}(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2))$, the FIM is also diagonal:

$$\mathcal{F}(\boldsymbol{\mu},\boldsymbol{\sigma}) = \left(\begin{array}{c|c} \text{diag}(1/\boldsymbol{\sigma}^2) & \mathbf{0} \\ \hline \mathbf{0} & 2\mathbf{I} \end{array}\right),$$

where $\mathbf{I}$ is a properly-sized identity matrix.

5.3.3 Practical surrogate objective functions

The exact Rényi divergence for P-POIS and the approximated version for A-POIS tend to be very conservative. In order to mitigate this problem, we

can rely on the fact that $d_2(P\|Q)/N = 1/\mathrm{ESS}(P\|Q)$, from equation (4.12). We can use an estimate $\widehat{\mathrm{ESS}}(P\|Q)$, as presented in equation (4.12), in order to estimate the Rényi divergence. This leads to the following approximated surrogate objective functions:

$$\widetilde{\mathcal{L}}_{\lambda}^{\mathrm{A-POIS}}(\boldsymbol{\theta'}/\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} w_{\boldsymbol{\theta'}/\boldsymbol{\theta}}(\tau_i)R(\tau_i) - \frac{\lambda}{\sqrt{\widehat{\mathrm{ESS}}\left(p(\cdot|\boldsymbol{\theta'})\|p(\cdot|\boldsymbol{\theta})\right)}},$$

$$\widetilde{\mathcal{L}}_{\lambda}^{\mathrm{P-POIS}}(\boldsymbol{\rho}'/\boldsymbol{\rho}) = \frac{1}{N}\sum_{i=1}^{N} w_{\boldsymbol{\rho'}/\boldsymbol{\rho}}(\boldsymbol{\theta}_i)R(\tau_i) - \frac{\lambda}{\sqrt{\widehat{\mathrm{ESS}}\left(\nu_{\boldsymbol{\rho'}}\|\nu_{\boldsymbol{\rho}}\right)}}.$$

Moreover, in all the experiments, we use the empirical maximum reward in place of the true $R_{\max}$.

5.3.4 Practical P-POIS for Deep Neural Policies (N-POIS)

When using Deep policies, P-POIS suffers from a curse of dimensionality because of the high number of parameters (in the Deep Neural Network we use as policy, the number of parameters are 10^3). The dimensionality of the corresponding Gaussian hyperpolicy (with diagonal covariance matrix) is very high, producing very unstable results when sampling. This can cause a very conservative behavior in the optimization process, preventing any learning. We decide to group the policy parameters in smaller blocks, and to learn each group independently. We can see each neuron of the network as a function:

$$U_i(\mathbf{x}|\boldsymbol{\theta}_m) = g(\mathbf{x}^T\boldsymbol{\theta}_m),$$

where $\mathbf{x}$ is the vector of weights an biases connected to the input of the neuron and $g(\cdot)$ is an activation function. We associate a set of parameters Θ_m for each unit U_m such that $\boldsymbol{\theta}_m \in \Theta_m$. Then, for each corresponding hyperparameter subspace $\mathcal{P}_m$, we can compute, as before, a surrogate objective function which we optimize using natural gradient ascent, with the step size found by the line search:

$$\widetilde{\mathcal{L}}_{\lambda}^{\mathrm{N-POIS}}(\boldsymbol{\rho}'_m/\boldsymbol{\rho}_m) = \frac{1}{N}\sum_{i=1}^{N} \widetilde{w}_{\boldsymbol{\rho}'_m/\boldsymbol{\rho}_m}(\boldsymbol{\theta}^i_m)R(\tau_i) - \frac{\lambda}{\sqrt{\widehat{\mathrm{ESS}}\left(\nu_{\boldsymbol{\rho}'_m}\|\nu_{\boldsymbol{\rho}_m}\right)}},$$

where $\boldsymbol{\rho}_m, \boldsymbol{\rho}'_m \in \mathcal{P}_m, \boldsymbol{\theta}_m \in \Theta_m$. This trick reduces the dimension of the multivariate Gaussian hyperpolicies from $\sim 10^3$ to $\sim 10^2$. This variant of the algorithm is called Neuron-Based POIS (N-POIS).

Chapter 6

Experimental Evaluation

6.1 Description of the Simulated Environments

The algorithm is tested in a suite of continuous control tasks [11] and compared with state-of-the-art policy gradient algorithms. The suite contains classical problems like Mountain Car [26], Cart-Pole Balancing [3] , Swimmer [33, 20], double-inverted pendulum [12] and acrobot [9, 28, 10], as well as higher dimensional problems. In the following section the tasks used to evaluate both versions of POIS are described.

6.1.1 Cart-Pole Balancing

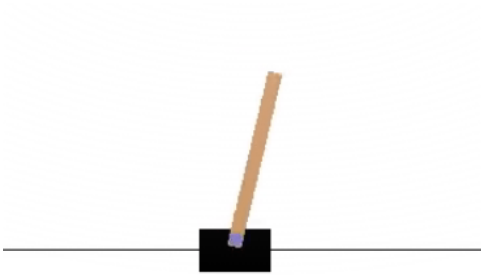


Figure 6.1: Cart-Pole Balancing environment.
Source: https://gym.openai.com/envs/CartPole-v1

In this task, a cart with an inverted pendulum mounted on a pivot point is constrained to linear movement. The goal is to apply a horizontal force to the cart in order to keep the pendulum in a vertical position. For this problem, the state of the agent is a vector whose elements are the position

of the cart x, its velocity $\dot{x}$, the pole angle θ and its velocity $\dot{\theta}$. The action the agent has to choose, at each interaction with the environment, is the horizontal force, which consists of a continuous value. The reward function is defined as $r(s, a) = 10 - (1 - cos(\theta)) - 10^{-5}||a||_2^2$ and an episode ends when the cart is too far from its initial point ($|x| > 2.4$) or when the agent fails to keep the pendulum upright ($|\theta| > 0.2$). Due to the high instability of the problem, the action performed by the agent has to be continuous.

6.1.2 Mountain Car

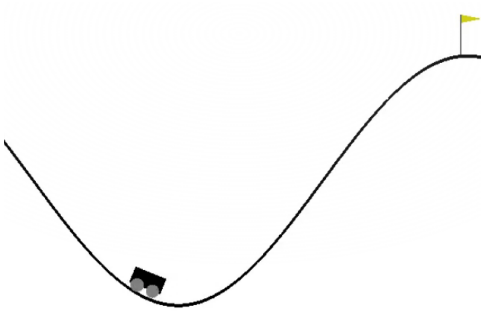


Figure 6.2: Mountain Car environment.
Source: https://gym.openai.com/envs/MountainCarContinuous-v0

In this task, a car starts in the middle of a valley and has to reach a certain altitude. Since the force applied tangent to the car is limited, the agent has to alternatively drive up both slopes in order to gain inertia. Without enough exploration, an agent would apply the force in the direction of the goal and then stop the car at the maximum high that can be reached. Hence it would be stuck in a local minimum. The state of the agent in the problem consists of the horizontal position x and velocity $\dot{x}$ of the car. The action is a tangential force applied to the car and the reward is defined as $r(s, a) = -1 + height$. The episode ends when the car reaches the height of 0.6.

6.1.3 Double Inverted Pendulum

In this task, which can be seen as an extension of the Cart-Pole Balancing problem, a double inverted pendulum is mounted on a cart through a pivot point. The goal is to keep the whole system balanced such that the two-link pendulum is maintained in an upright position. The action of the agent is the horizontal force applied to the cart. The state is a vector consisting of the position of the cart x, the joint angles of the double pendulum,

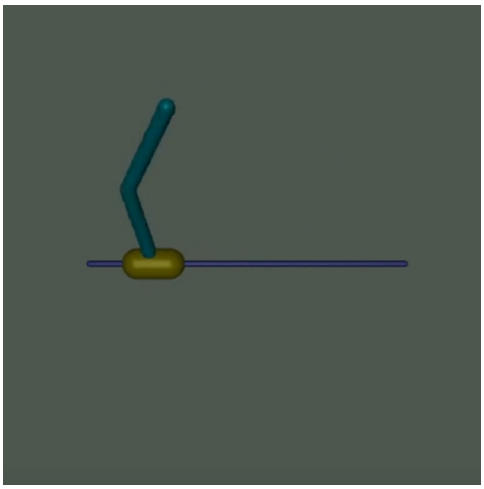


Figure 6.3: Double Inverted Pendulum environment.
Source: https://gym.openai.com/envs/InvertedDoublePendulum-v2

θ_1, θ_2, and their velocities, $\dot{\theta_1}$ and $\dot{\theta_2}$. The reward function is defined a $r(a, s) = 10 - 0.01x_{tip}^2 - (y_{tip} - 2)^2 - 10^{-3}\dot{\theta_1}^2 - 5 \cdot 10^{-3}\dot{\theta_2}^2$, where x_{tip}, y_{tip} denote the coordinates of the highest point of the pole. The episode ends when the vertical coordinate of the highest point of the pole is too low ($y_{tip} \leq 1$).

6.1.4 Acrobot

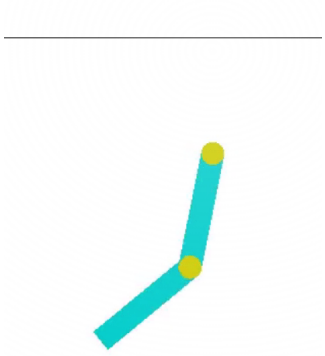


Figure 6.4: Acrobot environment.
Source: https://gym.openai.com/envs/Acrobot-v1

In this task, a two-link underactuated robot has to oscillate until it reaches an upright position and stabilizes. The robot consists of two joints: the first has a fixed position and can only rotate, while the second can exert torque.

The action of the agent is the value of the torque applied to the second joint and the state is a vector including the two angles of the joints, θ_1 and θ_2 and their velocities, $\dot{\theta}_1$ and $\dot{\theta}_2$. The reward function is $r(s,a) = -||tip(s) - tip_{target}||_2$, where $tip(s)$ denotes the coordinate of the tip of the robot.

6.1.5 Swimmer

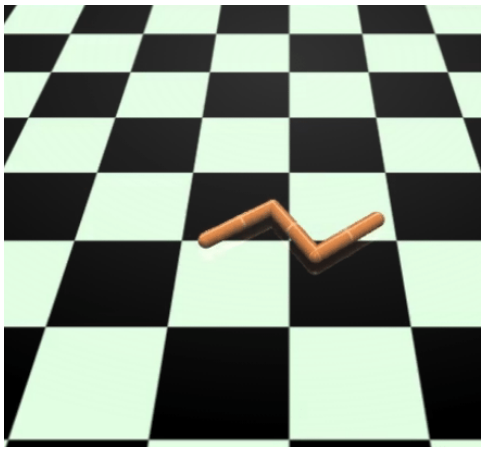


Figure 6.5: Swimmer environment.
Source: https://gym.openai.com/envs/Swimmer-v2

In this task, a robot, composed by three links and two joints, has to swim in a viscous fluid with the goal of moving forward as fast as possible. The action of the agent is the torque on each joint, while the state is a 13 dimensional vector including the angles and velocities of the joints, as well as the position of the center of mass. The reward function is defined as $r(s,a) = v_x - 0.005||a||_2^2$, where v_x indicates the forward velocity.

6.2 Results for Linear Policy

Although many state-of-the-art results for policy gradient methods have been achieved using Deep Neural Networks, comparable performances can also be realized by linear parametrized gaussian policies [35]. In this section, POIS, in both parameter-based and action-based versions, is compared to TRPO [39] and PPO [41], using the following policy:

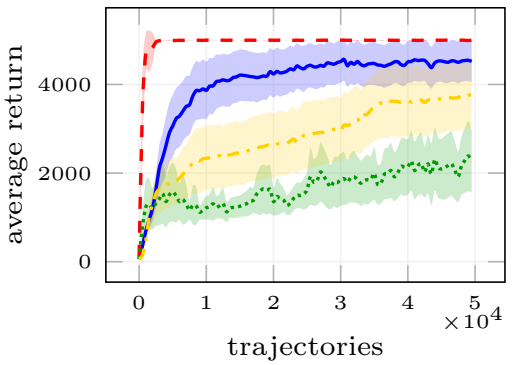
$$\pi(a|s) = \mathcal{N}(u_M(s), e^{2\Omega}), \tag{6.1}$$

where the mean $u_M(s)$ is a linear function of the state and the variance $e^{2\Omega}$ is a diagonal matrix not dependent on the state.

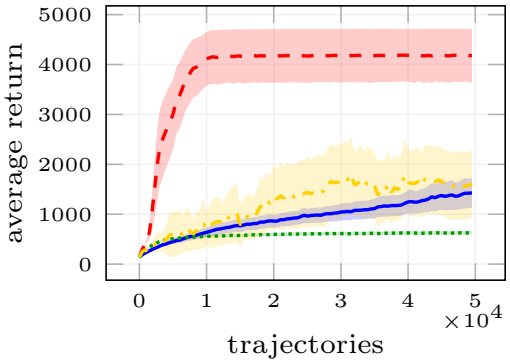
Task	A-POIS	P-POIS	TRPO	PPO
(a)	0.4	0.4	0.1	0.01
(b)	0.1	0.1	0.1	1
(c)	0.7	0.2	1	1
(d)	0.9	1	0.01	1
(e)	0.9	0.8	0.01	0.01

Table 6.1: Table representing the best hyperparameters (δ for POIS and the step size for TRPO and PPO)

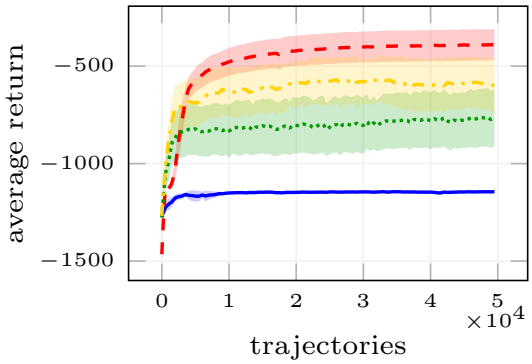
In Figure 6.6 it is possible to see that the learning curves for both versions of POIS significantly outperform TRPO and PPO in Cartpole. In Acrobot and Inverted Double Pendulum, the performance of P-POIS is remarkable, while A-POIS is not able to learn efficiently the tasks. In the Mountain Car environment, the learning curves of A-POIS, TRPO and PPO are almost one-shot, while A-POIS is much slower and needs more trajectories for learning. An insightful task is Inverted Pendulum, where we can observe that both versions of POIS fail to learn the task. This problem highlights one of the limitations of the algorithm: since POIS computes the importance weights at trajectory level, it is not able to assign credit to good actions in bad trajectories, while TRPO and PPO operate at step level. This problem could be overcome by using per-decision Importance Sampling [32]. Table 6.1 reports the best hyperparameters for POIS, TRPO and PPO across all the tasks.



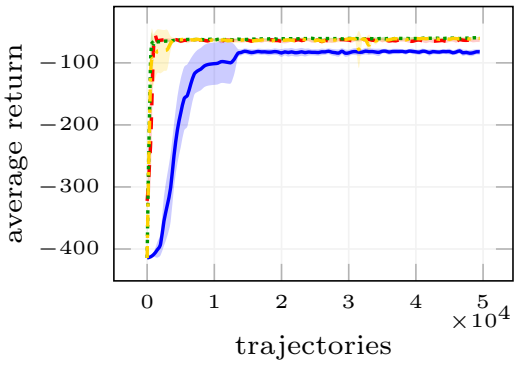
(a) Cartpole



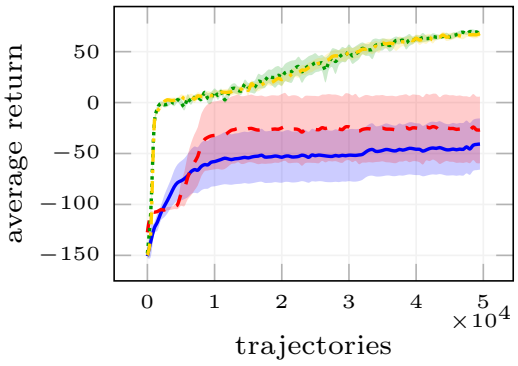
(b) Inverted Double Pendulum



(c) Acrobot



(d) Mountain Car



(e) Inverted Pendulum

Figure 6.6: Learning curves of A-POIS, P-POIS, TRPO and PPO, representing the average return as a function of the number of trajectories used for learning (average across 20 runs, 95% c.i.).

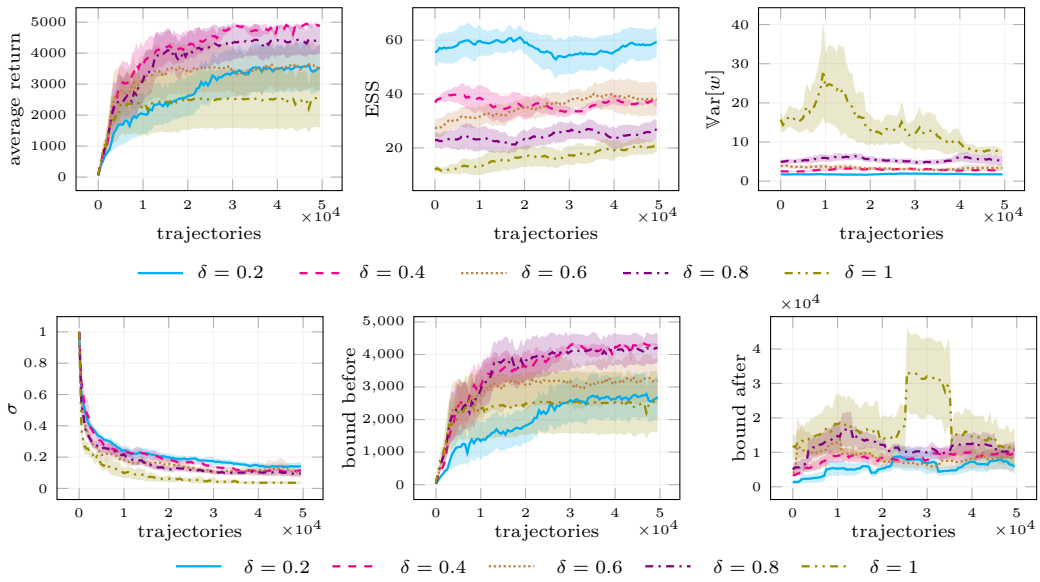


Figure 6.7: From left to righ, top to bottom: Plot of the average return, the Effective Sample Size (ESS), the variance of the importance weights ($\mathbb{V}\mathrm{ar}[w]$), the standard deviation σ of the policy, the value of the bound before and afer the optimization process, as a function of the number of trajectories, and for different values of δ in the Cartpole environment (average across 20 runs, 95% c.i.)

In Figure 6.7 it is possible to see how the parameter δ (significance level) is able to influence the tradeoff between maximizing the expected return and minimizing the penalty due to a step leading to a policy far from the behavioral one. In particular, for a small value of δ the Effective Sample Size is very high, leading to over-conservative updates. On the other hand, a big value of δ is responsible for a low ESS, producing an update in the gradient step with a very high variance. In the bottom-left figure we can see that the value of the parameter δ is affecting the speed of convergence to zero of the variance of the policy. For small values of δ, the penalization term in the objective function is very high, therefore, as policies with a small variance induce a large Rényi divergence, it is more difficult to reduce the variance of the policy. Large values of δ (very low penalization) can produce estimators whose variance is very high (large Rényi divergence). In the bottom-right figure, it is shown that a policy with a high δ will produce a higher bound after optimization, since it will account for a large uncertainty. The optimal value for δ in this environment is 0.4.

seeds	10, 109, 904, 160, 570, 662, 963, 100, 746, 236, 247, 689, 153, 947, 307, 42, 950, 315, 545, 178
Task horizon	500
Iterations	500
Max line search attempts	30
Max iterations offline	10
Episodes for each iteration	100
IW estimator	Importance Sampling for A-POIS Self-Normalized Importance Sampling for P-POIS
Natural Gradient	Yes for P-POIS No for A-POIS
Policy	$\pi(a\|s) = \mathcal{N}(u_M(s), e^{2\Omega})$ $u_M(s)$ is a linear function of the state $e^{2\Omega}$ is a diagonal matrix not dependent on the state.
Policy initialization	mean sampled from $\mathcal{N}(0, 0.01^2)$ variance initialized to the value of 1

Table 6.2: Table representing the hyperparameters used in the experiments with linear policies

6.2.1 Experiments Details

The following table 6.2 shows the hyperparameter values used in the experiments for A-POIS and P-POIS with linear policy, while Table 6.3 reports the hyperparameters tuned for POIS, TRPO and PPO:

Table 6.3: Hyperparameters tuned for each environment: in A-POIS and P-POIS δ represents the significance level, while in TRPO and PPO it represents the step-size. Hyperparameters in **bold** are the best found.

Environment	A-POIS (δ)
Cart-Pole Balancing	0.1, 0.2, 0.3, **0.4**, 0.5
Inverted Pendulum	0.8, **0.9**, 0.99, 1
Mountain Car	0.8, **0.9**, 0.99, 1
Acrobot	0.1, 0.3, 0.5, **0.7**, 0.9
Double Inverted Pendulum	**0.1**, 0.2, 0.3, 0.4, 0.5

Environment	P-POIS (δ)
Cart-Pole Balancing	0.1, 0.2, 0.3, **0.4**, 0.5, 0.6, 0.7, 0.8, 0.9 1
Inverted Pendulum	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, **0.8**, 0.9 1
Mountain Car	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, **1**
Acrobot	0.1, **0.2**, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 1
Double Inverted Pendulum	**0.1**, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 1

Environment	TRPO (δ)	PPO (δ)
Cart-Pole Balancing	0.001, 0.01, **0.1**, 1	0.001, **0.01**, 0.1 , 1
Inverted Pendulum	0.001, **0.01**, 0.1, 1	0.001, **0.01**, 0.1, 1
Mountain Car	0.001, **0.01**, 0.1, 1	0.001, 0.01, 0.1, **1**
Acrobot	0.001, 0.01, 0.1, **1**	0.001, 0.01, 0.1, **1**
Double Inverted Pendulum	0.001, 0.01, **0.1**, 1	0.001, 0.01, 0.1, **1**

Table 6.4: Performance of POIS compared with [11] using *deep neural policies* (average across 5 runs, 95% c.i.). For each task, the performances not statistically significantly different w.r.t. the best performance are reported in **bold**.

Algorithm	Cart-Pole Balancing	Mountain Car	Double Inverted Pendulum	Swimmer
Random	77.1 ± 0.0	-415.4 ± 0.0	149.7 ± 0.1	-1.7 ± 0.1
REINFORCE	4693.7 ± 14.0	-67.1 ± 1.0	4116.5 ± 65.2	92.3 ± 0.1
TNPG	$\mathbf{3986.4 \pm 748.9}$	$\mathbf{-66.5 \pm 4.5}$	$\mathbf{4455.4 \pm 37.6}$	$\mathbf{96.0 \pm 0.2}$
RWR	$\mathbf{4861.5 \pm 12.3}$	-79.4 ± 1.1	3614.8 ± 368.1	60.7 ± 5.5
REPS	565.6 ± 137.6	-275.6 ± 166.3	446.7 ± 114.8	3.8 ± 3.3
TRPO	$\mathbf{4869.8 \pm 37.6}$	$\mathbf{-61.7 \pm 0.9}$	$\mathbf{4412.4 \pm 50.4}$	$\mathbf{96.0 \pm 0.2}$
DDPG	4634.4 ± 87.6	-288.4 ± 170.3	2863.4 ± 154.0	85.8 ± 1.8
A-POIS	$\mathbf{4842.8 \pm 13.0}$	-63.7 ± 0.5	$\mathbf{4232.1 \pm 189.5}$	88.7 ± 0.55
CEM	4815.4 ± 4.8	-66.0 ± 2.4	2566.2 ± 178.9	68.8 ± 2.4
CMA-ES	2440.4 ± 568.3	-85.0 ± 7.7	1576.1 ± 51.3	64.9 ± 1.4
P-POIS	4428.1 ± 138.6	-78.9 ± 2.5	3161.4 ± 959.2	76.8 ± 1.6

6.3 Results for Deep Policy

In this section, POIS is compared with the state-of-the-art algorithms when using a deep neural network to represent policy. The action performed by the agent is still sampled from a Gaussian distribution:

$$\pi(a|s) = \mathcal{N}(u_M(s), e^{2\Omega}), \tag{6.2}$$

but in order to be fully compatible with the results in [11], the mean u_M is an Artificial Neural Network with 3 layers (100, 50 and 25 neurons respectively). Each hidden layer has a tanh activation function, while the output layer is linear. The variance $e^{2\Omega}$ is still a state-independent diagonal matrix. Table 6.4 reports the results of P-POIS and A-POIS w.r.t. the other state-of-the-art algorithms. In particular, A-POIS is able to obtain performances comparable with TRPO and to beat DDPG, while P-POIS reaches a performance comparable w.r.t. CEM, the best among the parameter-based algorithms.

seeds	10, 109, 904, 160, 570
Task horizon	500
Iterations	500
Max line search attempts	30
Max iterations offline	20
Timesteps for each iteration	50000
IW estimator for P-POIS	Importance Sampling for A-POIS Self-Normalized Importance Sampling
Natural Gradient	Yes for P-POIS No for A-POIS
Policy	$\pi(a\|s) = \mathcal{N}(u_M(s), e^{2\Omega})$ $u_M(s)$ is a 3-layers Multi-Layer Perceprton $e^{2\Omega}$ is a diagonal matrix not dependent on the state.
Policy initialization	Xavier initialization [13]

Table 6.5: Table representing the hyperparameters used in the experiments with deep policies

6.3.1 Experiments Details

The following table 6.5 shows the hyperparameter values used in the experiments for A-POIS and P-POIS with deep policy, while Table 6.6 reports the hyperparameters tuned for POIS:

Table 6.6: Hyperparameters tuned for each environment: in A-POIS and P-POIS δ represents the significance level. Hyperparameters in **bold** are the best found.

Environment	A-POIS (δ)
Cart-Pole Balancing	0.9, **0.99**, 0.999
Mountain Car	0.9, **0.99**, 0.999
Double Inverted Pendulum	0.9, **0.99**, 0.999
Swimmer	0.9, **0.99**, 0.999

Environment	P-POIS (δ)
Cart-Pole Balancing	0.4, 0.5, **0.6**, 0.7, 0.8
Mountain Car	0.1, 0.2, **0.3**, 0.4, 0.5, 0.6, 0.7, 0.8
Double Inverted Pendulum	0.4, 0.5, 0.6, 0.7, **0.8**
Swimmer	0.4, 0.5, **0.6**, 0.7, 0.8

Chapter 7

Conclusion

In this work, we started by addressing a general problem in Reinforcement Learning, which is "how to use the information contained in a batch of trajectories in the most efficient way". We showed how this can be done by using a technique, called Importance Sampling. The main theoretical contributions in this work are the studying of the IS and SN estimators, the study of why some statistical lower bounds are not appropriate for this optimization problem, the derivation of a novel concentration inequality for off-distribution learning. The algorithmic and experimental contribution are also significant and include the derivation of a novel, model-free, actor only, policy gradient algorithm, which can be used both in action-based and parameter-based setting, and which achieves results comparable with state-of-the-art RL algorithms in continuous control tasks. There are several possible future contributions on this work, which include the use of per-decision importance sampling, the study of other variance-reduction techniques (control variate such as baselines), the extension to the multi-goal case and the use of multiple importance sampling for trajectory reuse. We believe that this work is a good starting point for a deeper understanding of Policy Optimization.

Bibliography

[1] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[2] Shun-ichi Amari. *Differential-geometrical methods in statistics*, volume 28. Springer Science & Business Media, 2012.

[3] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):834–846, Sept 1983.

[4] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

[5] Jacob Burbea. The convexity with respect to gaussian distributions of divergences of order α. *Utilitas Mathematica*, 26:171–192, 1984.

[6] William G Cochran. *Sampling techniques*. John Wiley & Sons, 2007.

[7] Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pages 442–450, 2010.

[8] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1–2):1–142, 2013.

[9] G. DeJong and M. W. Spong. Swinging up the acrobot: an example of intelligent control. In *Proceedings of 1994 American Control Conference - ACC '94*, volume 2, pages 2158–2162 vol.2, June 1994.

[10] K. Doya. Reinforcement learning in continuous time and space. *Neural Comput.*, 12(1):219âĂŞ245, 2000.

[11] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[12] Okutani T. Furuta, K. and H. Sone. Computer control of a double inverted pendulum. *Comput. Electr. Eng.*, 5(1):67âĂŞ84, 1978.

[13] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[14] Mandy Grttner, Frank Sehnke, Tom Schaul, and Juergen Schmidhuber. Multi-dimensional deep memory go-player for parameter exploring policy gradients.

[15] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

[16] Timothy Classen Hesterberg. *Advances in importance sampling*. PhD thesis, Stanford University, 1988.

[17] Sham M Kakade. A natural policy gradient. In *Advances in neural information processing systems*, pages 1531–1538, 2002.

[18] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[19] Augustine Kong. A note on importance sampling using standardized weights. *University of Chicago, Dept. of Statistics, Tech. Rep*, 348, 1992.

[20] Sergey Levine and Vladlen Koltun. Guided policy search. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1–III–9. JMLR.org, 2013.

[21] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[22] Luca Martino, Víctor Elvira, and Francisco Louzada. Effective sample size for importance sampling based on discrepancy measures. *Signal Processing*, 131:386–401, 2017.

[23] Takamitsu Matsubara, Tetsuro Morimura, and Jun Morimoto. Adaptive step-size policy gradients with average reward metric. In *Proceedings of 2nd Asian Conference on Machine Learning*, pages 285–298, 2010.

[24] Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 5443–5455. Curran Associates, Inc., 2018.

[25] Atsushi Miyamae, Yuichi Nagata, Isao Ono, and Shigenobu Kobayashi. Natural policy gradient methods with parameter-based exploration for control tasks. In *Advances in neural information processing systems*, pages 1660–1668, 2010.

[26] Andrew Moore. *Efficient Memory-based Learning for Robot Control*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, March 1991.

[27] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

[28] R. M. Murray and J. Hauser. A case study in approximate linearization: The acrobot example. *Technical report, UC Berkeley, EECS Department*, 1991.

[29] Andrew Y Ng and Michael Jordan. Pegasus: A policy search method for large mdps and pomdps. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 406–415. Morgan Kaufmann Publishers Inc., 2000.

[30] Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[31] Jing Peng and Ronald J Williams. Incremental multi-step q-learning. In *Machine Learning Proceedings 1994*, pages 226–232. Elsevier, 1994.

[32] Doina Precup, Richard S Sutton, and Satinder P Singh. Eligibility traces for off-policy policy evaluation. In *ICML*, pages 759–766. Citeseer, 2000.

[33] E. M. Purcell. Life at low reynolds number. *Am. J. Phys*, 45(1):3âĂŞ11, 1977.

[34] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[35] Aravind Rajeswaran, Kendall Lowrey, Emanuel V Todorov, and Sham M Kakade. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems*, pages 6553–6564, 2017.

[36] C Radhakrishna Rao. Information and the accuracy attainable in the estimation of statistical parameters. In *Breakthroughs in statistics*, pages 235–247. Springer, 1992.

[37] Alfréd Rényi. On measures of entropy and information. Technical report, HUNGARIAN ACADEMY OF SCIENCES Budapest Hungary, 1961.

[38] Reuven Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1(2):127–190, 1999.

[39] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[40] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[41] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[42] F. Sehnke, C. Osendorfer, T. Rückstiess, A. Graves, J. Peters, and J. Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 21(4):551–559, May 2010.

[43] Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Policy gradients with parameter-based exploration for control. In *International Conference on Artificial Neural Networks*, pages 387–396. Springer, 2008.

[44] Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

[45] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

[46] Kenneth O Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

[47] Yi Sun, Daan Wierstra, Tom Schaul, and Juergen Schmidhuber. Efficient natural evolution strategies. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 539–546. ACM, 2009.

[48] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[49] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[50] István Szita and András Lörincz. Learning tetris using the noisy cross-entropy method. *Neural computation*, 18(12):2936–2941, 2006.

[51] Russ Tedrake, Teresa Weirui Zhang, and H Sebastian Seung. Stochastic policy gradient reinforcement learning on a simple 3d biped. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2849–2854. IEEE, 2004.

[52] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence policy improvement. In *International Conference on Machine Learning*, pages 2380–2388, 2015.

[53] Philip S Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *AAAI*, pages 3000–3006, 2015.

[54] Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[55] Jay M Ver Hoef. Who invented the delta method? *The American Statistician*, 66(2):124–127, 2012.

[56] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[57] Daan Wierstra, Tom Schaul, Jan Peters, and Juergen Schmidhuber. Natural evolution strategies. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 3381–3387. IEEE, 2008.

[58] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer, 1992.