METRICS MATTER: A CLOSER LOOK ON SELF-PACED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Curriculum reinforcement learning (CRL) allows to solve complex tasks by generating a tailored sequence of learning tasks, starting from easy ones and subsequently increasing their difficulty. However, the generation of such task sequences is largely governed by application assumptions, often preventing a theoretical investigation of existing approaches. Recently, Klink et al. (2021) showed how self-paced learning induces a principled interpolation between task distributions in the context of RL, resulting in high learning performance. So far, this interpolation is unfortunately limited to Gaussian distributions. Here, we show that on one side, this parametric restriction is insufficient in many learning cases but that on the other, the interpolation of self-paced RL (SPRL) can be degenerate when not restricted to this parametric form. We show that the introduction of concepts from optimal transport into SPRL prevents aforementioned issues. Experiments demonstrate that the resulting introduction of metric structure into the curriculum allows for a well-behaving non-parametric version of SPRL that leads to stable learning performance across tasks.

1 INTRODUCTION

Reinforcement learning (RL) (Sutton & Barto, 1998) has celebrated great successes as a framework for autonomous acquisition of desired behavior. With ever-increasing computational power, this framework and the algorithms developed under it have allowed to create learning agents capable of solving non-trivial long-horizon planning (Mnih et al., 2015; Silver et al., 2017) and control tasks (Akkaya et al., 2019). However, these successes have also highlighted the need for certain forms of regularization, such as leagues in the context of boardgames (Silver et al., 2017), a gradual diversification of simulated training environments for robotic manipulation (Akkaya et al., 2019) or a tailored training pipeline in the context of humanoid control for soccer (Liu et al., 2021). These regularizations help to overcome shortcomings of modern RL agents, such as poor exploratory behavior – a problem that is an active topic of research (Bellemare et al., 2016; Ghavamzadeh et al., 2015; Machado et al., 2020).

One can view aforementioned regularizations under the umbrella term of curriculum reinforcement learning (Narvekar et al., 2020), where the idea is to avoid shortcomings of modern (deep) RL agents such as aforementioned poor exploration by learning on a tailored sequence of tasks. Such curricula can materialize in a variety of ways and are motivated from many perspectives in the literature (Andrychowicz et al., 2017; Florensa et al., 2017; Wöhlke et al., 2020). Although the resulting curricula can often be interpreted as a sequence of task distributions, these sequences typically lack a formal connection to the reinforcement learning objective of maximizing the expected reward under a given target task distribution. In a recent line of work, Klink et al. (2021) proposed the idea of self-paced reinforcement learning (SPRL), borrowing from the concept of self-paced learning that has been established in the supervised learning literature (Kumar et al., 2010; Jiang et al., 2015; Meng et al., 2017). Klink et al. showed a connection between a regularized RL objective and a sequence of task distributions that trade-off between yielding high expected reward and tasks likely under the target distribution. This interpolant has, however, so far been restricted to Gaussian distributions (Klink et al., 2020a;b; 2021). While successful in experimental evaluations, this Gaussian assumption clearly imposes a limitation on the flexibility of the curriculum and disconnects the algorithmic implementation from the established theory. This disconnect raises the question whether the observed performance of SPRL is due to the Gaussian approximation.

Contribution: The key insight presented in this paper is that the Gaussian approximation of existing SPRL implementations is indeed important for the practical performance of SPRL, as it masks the weakness of the KL divergence to express task distribution similarity in a CRL setting. In more detail, we show that

- Parametric assumptions in SPRL hinder the learning performance in task spaces in which only a non-euclidean subspace is learnable by the agent. Leaving these parametric assumptions behind allows to overcome this problem.
- In the non-parametric regime, SPRL can fail to facilitate learning on the target task distribution due to the usage of the KL divergence as a measure between task distributions.
- Wasserstein metrics – that take the metric structure of the task space into account – ensure a meaningful interpolation in aforementioned failure cases, providing higher performance.

Apart from achieving higher empirical performance in the SPRL framework, the results presented in this paper indicate that introducing a notion of metric structure on the task space may be an important next step for deriving a principled, yet practical, understanding of CRL.

2 Related Work

The main focus of this work is on self-paced reinforcement learning (SPRL) (Klink et al., 2020a;b; 2021) that takes the concept of self-paced curriculum learning (Kumar et al., 2010) from supervised learning to reinforcement learning (RL). Opposed to supervised learning, where there is ongoing discussion about the mechanics of curricula and their effect in different situations (Weinshall & Amir, 2020; Wu et al., 2021), the mechanics seem to be more agreed upon in RL. In RL, curricula typically alleviate the problem of challenging exploration and/or try to maximize the learning speed by focusing on tasks in which the policy is still capable of improving its performance. These mechanics reduce the typically high sample complexity of modern RL algorithms, which e.g. arises from simple heuristics used for exploration such as random Gaussian noise on actions (Silver et al., 2014; Schulman et al., 2015; Haarnoja et al., 2018). Although the reason for the benefit of curricula in RL tend to be agreed upon, curriculum RL (CRL) algorithms are motivated from many different perspectives. They typically follow a particular heuristic to generate task sequences, such as the notion of learning progress (Portelas et al., 2019) or intermediate task difficulty (Florensa et al., 2018). While effective in practice, such heuristics prevent a formal analysis of the proposed algorithms. SPRL has been shown to perform an interpolation between task distributions (Klink et al., 2021), allowing to relate the effect of a curriculum to the concept of annealing in statistics (Neal, 2001). We wish to add to this formal understanding by investigating the interpolation produced by SPRL more closely, particularly in the context of curriculum reinforcement learning.
As this investigation will lead us to the problem of optimal transport, we wish to point out important literature in this field. Dating back to the work by Monge in the 18th century, optimal transport has been understood as an important fundamental concept touching upon many fields in both theory and application (Liu et al., 2019; Peyré et al., 2019; Chen et al., 2021). In probability theory, optimal transport translates to the so-called Wasserstein metric (Kantorovich, 1942) between two distributions that compares them under a given metric on the sample space. From a computational perspective, the use of entropic regularization (Cuturi, 2013) has led to tangible speed-ups in computations revolving around optimal transport and is hence widely applied (Feydy et al., 2019).

3 Preliminaries

This section serves to introduce the necessary background on (contextual) RL, self-paced RL and optimal transport.

3.1 Contextual Reinforcement Learning

Contextual reinforcement learning can be seen as a conceptual extension to the (single task) reinforcement learning (RL) problem. (Single task) RL aims to maximize an expected reward objective by finding an optimal a policy $\pi : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ for a given MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, p, r, p_0 \rangle$

$$\max_{\pi} J(\pi) = \max_{\pi} \mathbb{E}_{p_0(\mathbf{s}_0), p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t), \pi(\mathbf{a}_t|\mathbf{s}_t)} \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \right]. \tag{1}$$

Contextual RL extends this objective to a space of MDPs $\mathcal{M}(\mathbf{c}) = \langle \mathcal{S}, \mathcal{A}, p_{\mathbf{c}}, r_{\mathbf{c}}, p_{0,\mathbf{c}} \rangle$ indexed by a parameter $\mathbf{c} \in \mathcal{C}$. The distributions p_0 and $p_{0,\mathbf{c}}$ in (1) represent distributions over initial states $\mathbf{s}_0$ and p as well as $p_{\mathbf{c}}$ the transition dynamics of the environment. In the contextual setting, the policy $\pi : \mathcal{S} \times \mathcal{C} \times \mathcal{A} \mapsto \mathbb{R}$ is further conditioned on the contextual parameter $\mathbf{c}$ and the objective is extended via an additional distribution μ:$\mathcal{C} \mapsto \mathbb{R}$ over contextual variables $\mathbf{c}$

$$\max_{\pi} J(\pi, \mu) = \max_{\pi} \mathbb{E}_{\mu(\mathbf{c})} \left[J(\pi, \mathbf{c}) \right]. \tag{2}$$

The distribution $\mu(\mathbf{c})$ encodes the tasks $\mathcal{M}(\mathbf{c})$ that the agent is expected to encounter. The objective $J(\pi, \mathbf{c})$ in Eq. (2) corresponds to the objective $J(\pi)$ in Eq. (1) where, however, the initial state distribution p_0, the transition dynamics p as well as the reward function r of $\mathcal{M}$ are replaced by their counterparts in $\mathcal{M}(\mathbf{c})$. This contextual model of optimal decision making is well-suited for learning in multiple related tasks as is the case in multi-task (Wilson et al., 2007), goal-conditioned (Schaul et al., 2015) or curriculum RL (Narvekar et al., 2020).

3.2 Self-Paced Reinforcement Learning

Self-paced reinforcement learning (SPRL) has been introduced by Klink et al. (2020a;b; 2021) as a curriculum RL algorithm that alters the context distribution $\mu(\mathbf{c})$ in the contextual RL objective (2) to increase the learning performance of an agent and/or make it less susceptible to local optima of the objective function. SPRL computes a surrogate distribution $p : \mathcal{C} \mapsto \mathbb{R}$ under which to train the RL agent, i.e.

$$\max_{\pi} J(\pi, p) = \max_{\pi} \mathbb{E}_{p(\mathbf{c})} \left[J(\pi, \mathbf{c}) \right].$$

This surrogate distribution is found by optimizing the KL divergence to the target distribution $\mu(\mathbf{c})$ subject to two constraints (see Klink et al. (2021, Section 8))

$$\min_{p} \; D_{\text{KL}} \left(p(\mathbf{c}) \parallel \mu(\mathbf{c}) \right) \;\; \text{s.t.} \;\; J(\pi, p) \geq \delta \qquad D_{\text{KL}} \left(p(\mathbf{c}) \parallel q(\mathbf{c}) \right) \leq \epsilon. \tag{3}$$

The distribution $p(\mathbf{c})$ balances between representing similar tasks as the (target) distribution $\mu(\mathbf{c})$ and encoding tasks in which the agent currently obtains large rewards. The KL divergence constraint w.r.t. the previous context distribution $q(\mathbf{c})$ is employed to avoid large changes in the distribution $p(\mathbf{c})$ during subsequent iterations. In practice, this regularization is required as the the expected performance $J(\pi, \mathbf{c})$ in context $\mathbf{c}$ needs to be estimated from a limited set of samples e.g. via a neural network.

A particularly interesting aspect of this work is that objective (3) can be interpreted to perform a specific interpolation between the distributions $\mu(\mathbf{c})$, $q(\mathbf{c})$ and a maximum entropy distribution $p_J(\mathbf{c}) \propto \exp(\eta J(\pi, \mathbf{c}))$ encoding high reward tasks. This interpolation is given by

$$p_{\alpha,\eta}(\mathbf{c}) \propto \mu(\mathbf{c})^{\frac{1}{1+\alpha}} q(\mathbf{c})^{\frac{\alpha}{1+\alpha}} \exp(\eta J(\pi, \mathbf{c}))^{\frac{1}{1+\alpha}}. \tag{4}$$

The two parameters α and η controlling the interpolation are the Lagrangian multipliers of the two constraints in objective (3). So far, the algorithmic realizations of (3) by Klink et al. (2020a;b; 2021) restricted the distribution $p_{\alpha,\eta}(\mathbf{c})$ to a parametric form with parameters $\boldsymbol{\nu}$, in particular Gaussian distributions $p_{\boldsymbol{\nu}}(\mathbf{c}) = \mathcal{N}(\mathbf{c}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In this case, optimizing (3) w.r.t. $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ of $p_{\boldsymbol{\nu}}$ corresponds to performing an I-projection of the analytic optimal distribution (4) to the Gaussian restriction

$$\min_{\boldsymbol{\nu}} D_{\text{KL}} \left(p_{\boldsymbol{\nu}}(\mathbf{c}) \;\middle\|\; \frac{1}{Z_{\alpha,\eta}} \mu(\mathbf{c})^{\frac{1}{1+\alpha}} q(\mathbf{c})^{\frac{\alpha}{1+\alpha}} \exp(\eta J(\pi, \mathbf{c}))^{\frac{1}{1+\alpha}} \right).$$

In above equation, $Z_{\alpha,\eta}$ is the normalizer of the analytic distribution $p_{\alpha,\eta}(\mathbf{c})$. In this work, we are interested in investigating the distribution $p_{\alpha,\eta}$ outside of this analytic restriction $p_{\boldsymbol{\nu}}$, i.e. truly employing the distribution (4) instead of its I-projection to a Gaussian.

3.3 Optimal Transport

The problem of optimally transporting mass between two distributions has been initially investigated by Monge (1781). As of today, generalizations established by Kantorovich (1942) have led to so

Algorithm 1 Self-Paced Reinforcement Learning Implementations

Input: Context dist. $p_0(\mathbf{c})$, target context dist. $\mu(\mathbf{c})$, performance threshold δ, distance bound ϵ
for $k = 0$ **to** K **do**
  Agent Improvement:
  Sample contexts $\mathbf{c}_i \sim p_k(\mathbf{c}),\ i \in [1, M]$
  Train policy π under $\mathbf{c}_i$ and observe episodic rewards $R_i = \sum_{t=1}^{\infty} r_{\mathbf{c}_i}(\mathbf{s}_t, \mathbf{a}_t),\ i \in [1, M]$
  Estimate $J(\pi, \mathbf{c})$ from the dataset $\mathcal{D} = \{(\mathbf{c}_i, R_i) | i \in [1, M]\}$
  Context Distribution Update:
  G-SPRL Optimize (3) w.r.t. $\boldsymbol{\mu}_{k+1}$ and $\boldsymbol{\Sigma}_{k+1}$, where $p_{k+1}(\mathbf{c}) = \mathcal{N}(\mathbf{c}|\boldsymbol{\mu}_{k+1}, \boldsymbol{\Sigma}_{k+1})$
  NP-SPRL Optimize (3) w.r.t α and η using a discrete approximation $\bar{\mathbf{p}}_{\alpha,\eta} \approx p_{\alpha,\eta}(\mathbf{c})$ (4)
  WB-SPRL Optimize (7) w.r.t. $\boldsymbol{\beta}$ to obtain $p_{\boldsymbol{\beta}}(\mathbf{c})$
end for

called **Wasserstein distances** as metrics between probability distributions defined on a metric space $M = (d, \mathcal{C})$ with metric $d : \mathcal{C} \times \mathcal{C} \mapsto \mathbb{R}_{\geq 0}$

$$\mathcal{W}_p(p_1, p_2) = \left(\inf_{\gamma \in \Gamma(p_1, p_2)} \int_{\mathcal{C} \times \mathcal{C}} d(\mathbf{c}_1, \mathbf{c}_2)^p \, \mathrm{d}\gamma(\mathbf{c}_1, \mathbf{c}_2) \right)^{1/p}, \qquad p \geq 1$$

$$\Gamma(p_1, p_2) = \left\{ \gamma : \mathcal{C} \times \mathcal{C} \mapsto \mathbb{R}_{\geq 0} \middle| p_1(\mathbf{c}_1) = \int \gamma(\mathbf{c}_1, \mathbf{c}_2) \, \mathrm{d}\mathbf{c}_2,\ p_2(\mathbf{c}_2) = \int \gamma(\mathbf{c}_1, \mathbf{c}_2) \, \mathrm{d}\mathbf{c}_1 \right\}$$

The distance between p_1 and p_2 results from solving an optimization problem that finds a so-called plan γ. This plan encodes how to move density from p_1 to p_2 taking into account the cost of moving density between between parts of the space $\mathcal{C}$. This cost is encoded by the metric d. In the following, we will always assume to work with 2-Wasserstein distances under euclidean metric, i.e. $p = 2$ and $d(\mathbf{c}_1, \mathbf{c}_2) = \|\mathbf{c}_1 - \mathbf{c}_2\|_2$. Particularly interesting for the investigation conducted in this paper are so called **Wasserstein barycenters**

$$p_{\mathcal{W}_2}(\mathbf{c}) = \arg\min_p \sum_{k=1}^{K} w_k \mathcal{W}_2(p, p_k), \qquad \sum_{k=1}^{K} w_k = 1, \tag{5}$$

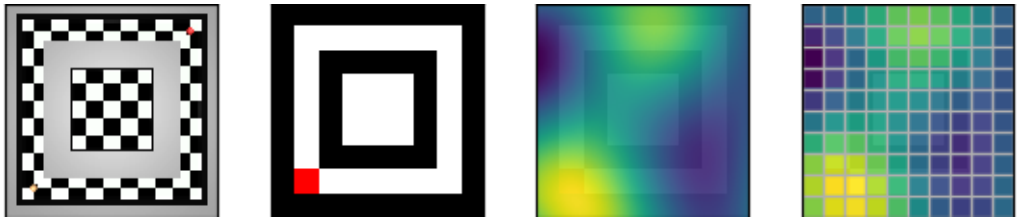
that perform a weighted interpolation between a set of distributions p_k by computing a distribution p that minimizes the weighted sum of Wasserstein distances to them.

4 Non-Parameteric Self-Paced Reinforcement Learning

We now describe the two implementations of SPRL that will be evaluated in the experimental section of this paper. The first implementation – NP-SPRL– is a faithful implementation of the SPRL objective (3) that computes and uses the distribution $p_{\alpha,\eta}(\mathbf{c})$ (4) by discretizing the context space $\mathcal{C}$ to avoid Gaussian approximations. Furthermore, we instantiate Wasserstein barycenters in the SPRL framework under the name WB-SPRL. As shown in Algorithm 1, all implementations (including the Gaussian one from Klink et al. (2021)) only differ in the computation of the context distributions. Hence, contrasting these implementations sheds light on the importance of metrics in the context of CRL.

4.1 NP-SPRL

Computing the analytic distribution $p_{\alpha,\eta}(\mathbf{c})$ and adjusting the parameters α and η via the SPRL objective (3) will require approximations in the general case, as even sampling arbitrary continuous distributions is already an open research problem (Brooks et al., 2011; Liu & Wang, 2016; Liu et al., 2019; Wibisono, 2018). Using approximate methods resulting from this research would, however, directly interfere with our intent of evaluating the behavior of $p_{\alpha,\eta}(\mathbf{c})$ as exactly as possible. Consequently, we discretize the continuous context spaces in our experiments in order to faithfully sample and evaluate expectations w.r.t. $p_{\alpha,\eta}(\mathbf{c})$. This is schematically shown in Figure 1 for one of the evaluation environments. Although such a discretization will clearly not scale gracefully to higher dimensions, it allows us to investigate the true behavior of SPRL, which, as we will show, can be sub-optimal even for low-dimensional context spaces.
With a context space $\mathcal{C} \subseteq \mathbb{R}^d$ discretized into N cells, we can represent $p_{\alpha,\eta}$ via a vector $\bar{\mathbf{p}}_{\alpha,\eta} \in \mathbb{R}^N_{\geq 0}$.



(a) Maze Environment (b) Context Space $\mathcal{C}$ (c) Function f on $\mathcal{C}$ (d) f discretized on $\mathcal{C}$

Figure 1: (a) The first environment for evaluation of SPRL is a maze task simulated in MuJoCo (Todorov et al., 2012), in which a point-mass needs to move in a maze of circular shape to reach desired target positions. (b) The target position is encoded via the contextual variable $\mathbf{c} \in \mathcal{C} \subseteq \mathbb{R}^2$. The area highlighted in red visualizes the initial position of the point mass and the walls of the environment are shown in black. (c + d) In order to faithfully evaluate expectations over functions or compute KL divergences e.g. in the SPRL objective (3), we discretize the context space $\mathcal{C}$.

To sample a continuous context $\mathbf{c} \in \mathcal{C}$, we can then first sample an index of a cell from $\bar{\mathbf{p}}_{\alpha,\eta}$ and then sample uniformly within this cell to obtain a context $\mathbf{c}$. Further, the KL divergences in the SPRL objective (3) are straightforward to evaluate when working with $\bar{\mathbf{p}}_{\alpha,\eta}$. To evaluate the expected performance, we simply evaluate the performance measure $J(\pi, \mathbf{c})$ at the N cell centers $\bar{\mathbf{c}}_n$ to obtain the vector $\bar{\mathbf{J}}(\pi) \in \mathbb{R}^N$. With that we obtain $J(\pi, p_{\alpha,\eta}) = \bar{\mathbf{p}}_{\alpha,\eta}^T \bar{\mathbf{J}}(\pi)$.

4.2 WB-SPRL

Before discussing **how** to incorporate Wasserstein barycenters into SPRL, we first want to answer the question **why** this may be useful. As mentioned in Section 3.3, the distinguishing feature of optimal transport is the ability to take metrics defined on the sample space, i.e. the context space $\mathcal{C}$ in the case of curriculum RL, into account. Given that the contextual variables $\mathbf{c} \in \mathcal{C}$ are representing MDPs $\mathcal{M}(\mathbf{c})$, we can introduce a metric on MDPs into SPRL by employing optimal transport approaches on $\mathcal{C}$. This metric can ensure that the curriculum only gradually changes the learning tasks, and with that stabilize learning. As we show in the experiments, this effect is indeed existent and crucial to generate meaningful interpolations between task distributions. To leverage optimal transport in an SPRL style algorithm we realize the interpolation between current distribution $q(\mathbf{c})$, target distribution $\mu(\mathbf{c})$ and "value distribution" $p_J(\mathbf{c}) \propto \exp(\eta J(\pi, \mathbf{c}))$ by Wasserstein barycenters

$$p_{\boldsymbol{\beta}}(\mathbf{c}) = \arg\min_{p} (1-\beta_1-\beta_2)\mathcal{W}(p(\mathbf{c}), \mu(\mathbf{c})) + \beta_1 \mathcal{W}(p(\mathbf{c}), p_J(\mathbf{c})) + \beta_2 \mathcal{W}(p(\mathbf{c}), q(\mathbf{c})). \quad (6)$$

The weights $\boldsymbol{\beta}$ of the interpolation are adjusted with the goal of minimizing $\mathcal{W}_2(p_{\boldsymbol{\beta}}, \mu)$ while ensuring a constraint on expected performance and distance to the previous distribution $q(\mathbf{c})$

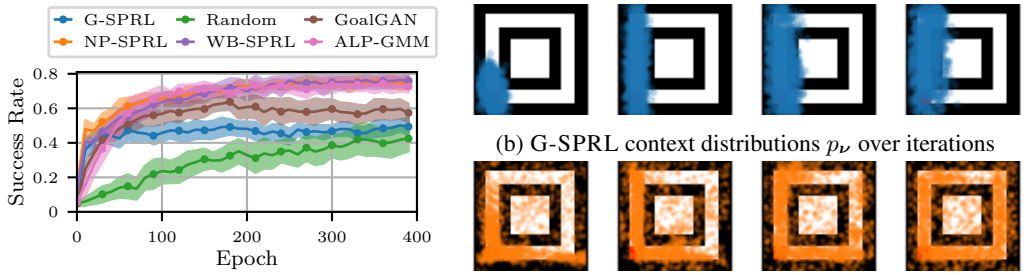
$$\min_{\boldsymbol{\beta}} \mathcal{W}_2(p_{\boldsymbol{\beta}}(\mathbf{c}), \mu(\mathbf{c})) \text{ s.t } J(\pi, p_{\boldsymbol{\beta}}) \geq \delta \quad \mathcal{W}_2(p_{\boldsymbol{\beta}}(\mathbf{c}), q(\mathbf{c})) \leq \epsilon. \quad (7)$$

A difference w.r.t. SPRL is that the parameter η of the value distribution $p_J(\mathbf{c})$ is not adjusted in optimization problem (7). Instead we adjust η before optimizing (7) such that $J(\pi, p_J) \geq \delta_H$, where $\delta_H > \delta$ is another performance threshold. This ensures that with $\beta_1 \to 1$, it holds that $J(\pi, p_{\boldsymbol{\beta}}) \geq \delta$. We resort to particle-based representations of the distributions when implementing (6) and (7). This allows us to make use of Monge maps to compute $p_{\boldsymbol{\beta}}(\mathbf{c})$ efficiently. More details on the implementation of WB-SPRL are provided in appendix B.

5 Experiments

The experiments in this section serve to show the need for a realization of SPRL without parametric restrictions on $p_{\alpha,\eta}$ but to also highlight that NP-SPRL is not necessarily well-suited for this endeavour due its ignorance w.r.t. the metric on $\mathcal{C}$. Experimental details can be found in appendix C [1]. We refer to the Gaussian realization of SPRL by Klink et al. (2021) as G-SPRL. Although our focus is on evaluating the behavior of SPRL, we also evaluate current state-of-the art algorithms GoalGAN and ALP-GMM (Florensa et al., 2018; Portelas et al., 2019) in all environments to contrast the results with currently well-performing algorithms. With the final environment defined over a discrete space $\mathcal{C}$, we also investigate the algorithm by (Graves et al., 2017), naming it ACL.

[1] Code is provided in the supplementary material and will be made available upon acceptance



(a) Learning Performance

(b) G-SPRL context distributions $p_{\boldsymbol{\nu}}$ over iterations

(c) NP-SPRL context distributions $p_{\alpha,\eta}$ over iterations

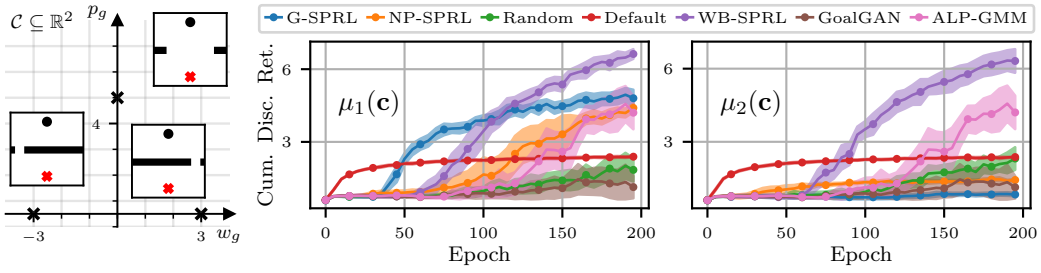
Figure 2: a) Achieved success rate of different curricula and a uniform sampling baseline (Random) over iterations. Mean and standard error are computed from 10 runs. b) Parametric and non-parametric context distributions $p_{\boldsymbol{\nu}}(\mathbf{c})$ and $p_{\alpha,\eta}(\mathbf{c})$ for a run of G-SPRL and NP-SPRL respectively. The distributions are represented by 2000 samples drawn from them.

5.1 MAZE

The first environment that we turn to is a sparse-reward, maze-like environment depicted in Figure 1, in which an agent needs to reach a desired goal. Such environments have e.g. been investigated by Florensa et al. (2018). The contexts $\mathbf{c} \in \mathcal{C}$ of this environment encode the goal position to be reached and hence contains unsolvable contexts (goals inside of a wall or in the inner circle of the maze). Defining $\mu(\mathbf{c})$ to be uniform over the context space, the curriculum needs to identify and train the agent on the subspace of feasible tasks in order to achieve a good learning performance. This subspace of feasible tasks is highly non-Gaussian, making it an interesting testbed for NP- and WB-SPRL. We discretize the context space $\mathcal{C} \subseteq \mathbb{R}^2$ of goal positions into a 50×50 grid. Figure 2 compares the performance of the different CRL algorithms. We see that both NP- and WB-SPRL perform better than G-SPRL. This is because the performance constraint $\mathbb{E}_{p(\mathbf{c})}\left[J(\pi, \mathbf{c})\right] \geq \delta$ in objective (3) at some point prevents the Gaussian context distribution from expanding, as otherwise too many infeasible tasks would be encoded in $p_{\boldsymbol{\nu}}(\mathbf{c})$ and hence the performance constraint violated. As shown in Figures 2 and 7, NP- and WB-SPRL can flexibly assign probability to feasible contexts, resulting in an increased learning speed compared to G-SPRL and training on $\mu(\mathbf{c})$ directly.

5.2 POINT-MASS

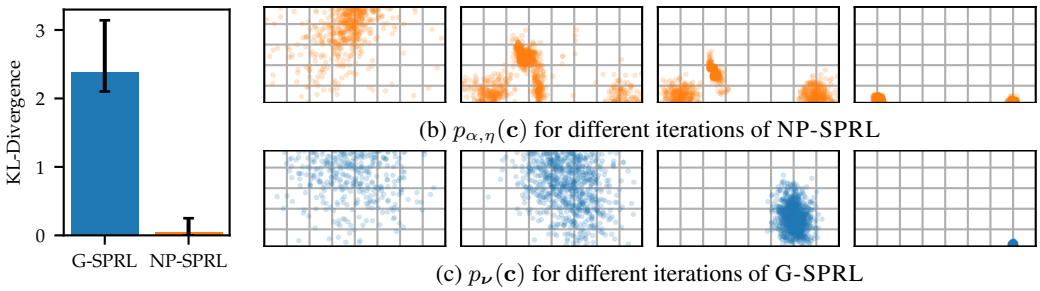
After this promising first demonstration of NP-SPRL in the previous section, we now turn towards the point-mass environment investigated by (Klink et al., 2020a;b; 2021) to highlight its challenges. As shown in Figure 3a, a point mass needs to be steered through a narrow gate to reach a goal position on the other side of a wall. While Klink et al. only considered a narrow gate at one specific position as the target task, we will investigate a version in which the gate is located at one of two opposing positions $\mathbf{c}_1$=[−3 0.5] and $\mathbf{c}_2$=[3 0.5], making $\mu(\mathbf{c})$ a bi-modal distribution. This again



(a) Environment

(b) Learning Performance

Figure 3: (a) The point mass environment with its two-dimensional context space. The target distributions $\mu_1(\mathbf{c})$ and $\mu_2(\mathbf{c})$ encode the two gates with width w_g=0.5, in which the agent (black dot) is required to navigate through a narrow gate at different positions to reach the goal (red cross). (b) Discounted cumulative return over iterations obtained under different curricula for both distributions. Statistics (mean and standard error) are computed from 10 seeds.



(a) $D_{\mathrm{KL}}\left(p(\mathbf{c}) \parallel \mu_1(\mathbf{c})\right)$

(b) $p_{\alpha,\eta}(\mathbf{c})$ for different iterations of NP-SPRL

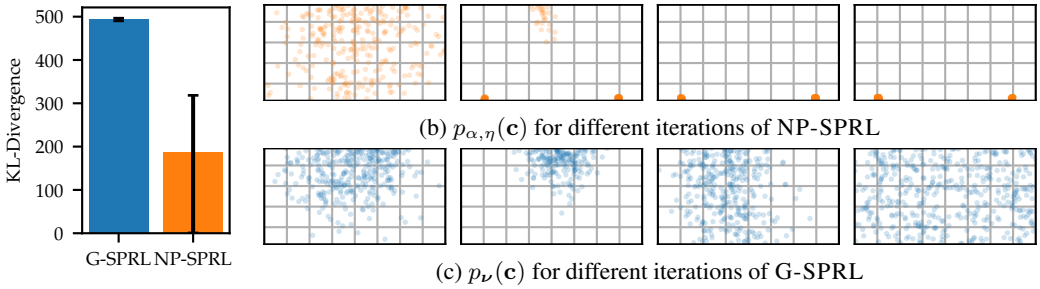
(c) $p_{\boldsymbol{\nu}}(\mathbf{c})$ for different iterations of G-SPRL

Figure 4: (a) Mean KL divergences between $\mu_1(\mathbf{c})$ and the final context distribution computed by G- and NP-SPRL as well as the minimum and maximum over 10 seeds. (b) and (c) visualize $p_{\alpha,\eta}(\mathbf{c})$ of NP-SPRL and $p_{\boldsymbol{\nu}}(\mathbf{c})$ of G-SPRL for increasing iterations of the algorithms (left to right).

challenges the Gaussian restriction of $p_{\boldsymbol{\nu}}(\mathbf{c})$ in the G-SPRL algorithm. We again discretize the context space $\mathcal{C} \subseteq \mathbb{R}^2$, encoding position and width of the gate, into a grid of 50 bins on each axis. We investigate two different target distributions

$$\mu_1(\mathbf{c}) = \frac{1}{2}\mathcal{N}\left(\mathbf{c}_1, 10^{-4}\mathbf{I}\right) + \frac{1}{2}\mathcal{N}\left(\mathbf{c}_2, 10^{-4}\mathbf{I}\right) \qquad \mu_2(\mathbf{c}) = \begin{cases} \frac{1}{2} & \text{, if } \mathbf{c} \in \{\mathbf{c}_1, \mathbf{c}_2\} \\ \approx 0 & \text{, else.} \end{cases}$$

Note that "$\approx$0" highlights that the probability is not exactly zero but has a value of $\exp(-1000)$, as this ensures that $\mu(\mathbf{c})$ is absolutely continuous w.r.t. $p_{\alpha,\eta}(\mathbf{c})$ and $p_{\boldsymbol{\nu}}(\mathbf{c})$. Absolute continuity is required to compute the KL divergence to $\mu(\mathbf{c})$. Samples from the two distributions are, however, visually hard to distinguish. Nonetheless, Figure 3 shows that the performance of G- and NP-SPRL depends drastically on the choice of target distribution. Further, we see that NP-SPRL does not outperform G-SPRL in this environment, although it should be able to match the bi-modal target distributions, which G-SPRL cannot. Finally, we see that WB-SPRL outperforms both other versions of SPRL and achieves a consistent performance for both target distributions. The illustrations in Figures 4 and 5 will help to understand the underlying problem. As evident in Figure 4, NP-SPRL is indeed able to match the target distribution correctly while G-SPRL ultimately covers only one mode of the target density $\mu_1(\mathbf{c})$. Consequently, it is suprising that the final performance of NP-SPRL only matches the performance of G-SPRL, although G-SPRL completely ignores one of the two target contexts. Investigating Figure 4b more closely reveals that NP-SPRL only generates a proper curriculum for one of the two contexts (the left one in the images) by gradually interpolating between easy tasks and the target task. As soon as it has gained sufficient proficiency on the (left) target task, NP-SPRL simply incorporates the second target task into $p_{\alpha,\eta}$, however **without** generating an appropriate curriculum for the agent to learn this task. As shown in appendix A, this leads to the agent only solving one target task under the NP-SPRL curriculum, although both target tasks are likely under the final context distribution $p_{\alpha,\eta}(\mathbf{c})$. Figure 5 shows that the curricula generated by G- and NP-SPRL for $\mu_2(\mathbf{c})$ are even more problematic. G-SPRL increases the variance of the Gaussian context distribution to match the constant (negligible) likelihood that is assigned to the non-target tasks. While NP-SPRL again encodes the two target distributions, it now

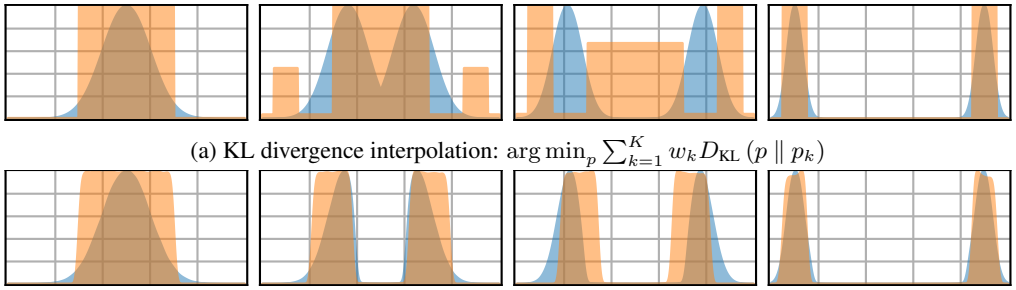


(a) $D_{\mathrm{KL}}\left(p(\mathbf{c}) \parallel \mu_2(\mathbf{c})\right)$

(b) $p_{\alpha,\eta}(\mathbf{c})$ for different iterations of NP-SPRL

(c) $p_{\boldsymbol{\nu}}(\mathbf{c})$ for different iterations of G-SPRL

Figure 5: (a) Mean KL divergences between $\mu_2(\mathbf{c})$ and the final context distribution computed by G- and NP-SPRL as well as the minimum and maximum over 10 seeds. (b) and (c) visualize $p_{\alpha,\eta}(\mathbf{c})$ of NP-SPRL and $p_{\boldsymbol{\nu}}(\mathbf{c})$ of G-SPRL for increasing iterations of the algorithms (left to right).



(a) KL divergence interpolation: $\arg\min_p \sum_{k=1}^{K} w_k D_{\mathrm{KL}}(p \parallel p_k)$

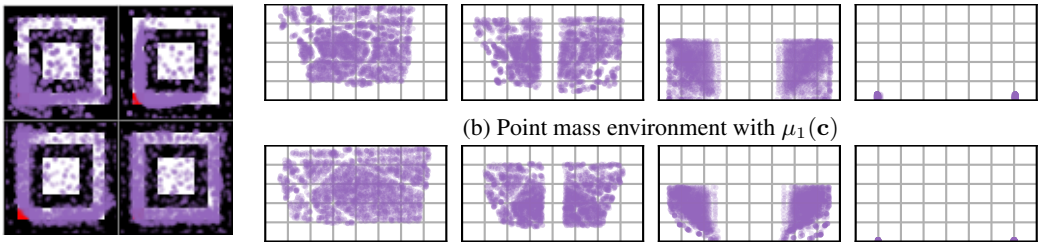
(b) Wasserstein barycenter (Equation 5)

Figure 6: Interpolations between unimodal (left) and bi-model (right) distributions $p_1(\mathbf{c})$ and $p_2(\mathbf{c})$ via a KL divergence-based interpolation and Wasserstein barycenters (5). In one case, all distribution are Gaussians or mixture of Gaussians (blue). In the other case, the distributions are uniform distributions (orange). The Wasserstein barycenters are computed using a particle-based approximation (Feydy et al., 2019). The visualized PDFs are then estimated using a kernel density estimation. This results in a small amount of smoothing for the uniform distribtions.

interpolates between none of them. Consequently, neither the agent under G- nor NP-SPRL learns to solve any of the tasks. These results highlight an important shortcoming of the SPRL distribution (4) in a curriculum reinforcement learning setting. This shortcoming is visualized in Figure 6, comparing Wasserstein barycenters and an interpolation between distributions based on KL divergences. The latter one allows probability mass to directly move between distant parts of the context space, because the KL divergence between two probability distributions $p_1 : \mathcal{C} \mapsto \mathbb{R}$ and $p_2 : \mathcal{C} \mapsto \mathbb{R}$ does not incorporate any notion of a metric defined on $\mathcal{C}$. This lack of metric is at the heart of the observed problems, as the SPRL objective (3) basically represents such a KL divergence-based interpolation. Looking at Figure 6b and the curricula generated by WB-SPRL in Figure 7, we see that the notion of a (Euclidean) metric prevents these jumps and leads to a gradual change in the tasks encoded by the curriculum. But why do G-SPRL and (sometimes) NP-SPRL create meaningful interpolations? The reason is in the usage of Gaussian distributions, as visualized in Figure 6a. The log-likelihood of a Gaussian distribution and with that the KL divergence between two Gaussians are both defined via Mahalanobis distances, i.e. a Euclidean distance in a transformed space. Consequently, the Gaussian distributions encode a (Euclidean) metric via their log-likelihood. This further explains the performance drop of G- and NP-SPRL as soon as we switch from $\mu_1(\mathbf{c})$ to $\mu_2(\mathbf{c})$. Without the implicit metric information contained in the Gaussian distribution $\mu_1(\mathbf{c})$, NP-SPRL completely breaks down and does not perform any reasonable interpolation. G-SPRL is challenged with matching the shape of $\mu(\mathbf{c}_2)$ due to its Gaussian restriction.

5.3 Pick and Place

As a final experiment, we consider the pick-and-place task in the OpenAI gym environment suite (Brockman et al., 2016) in which a robot is tasked to grasp a block on a table and move it to a desired position (see Figure 8). The sparse reward of this environment, only rewarding the robot upon completing the desired task, makes it a very challenging exploration problem, as can be seen in Figure 8 in which (default) SAC does not learn this pick and place task. One way to alleviate

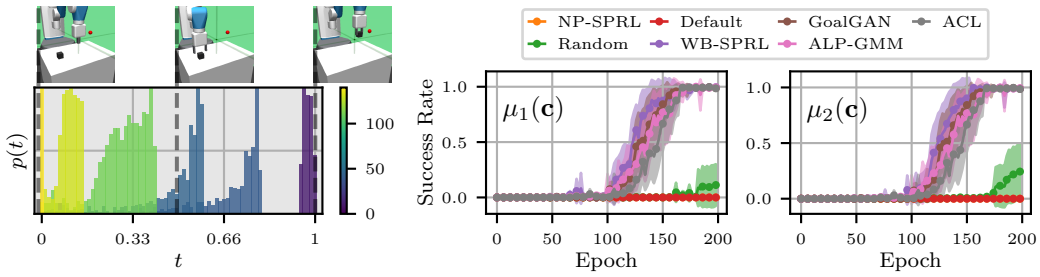


(a) Maze environment

(b) Point mass environment with $\mu_1(\mathbf{c})$

(c) Point mass environment with $\mu_2(\mathbf{c})$

Figure 7: Visualizations of the empirical distributions $p_{\boldsymbol{\beta}}$ generated by WB-SPRL in the maze- (a) and point-mass environments for both target distributions $\mu_1(\mathbf{c})$ and $\mu_2(\mathbf{c})$ ((b) + (c)).



(a) WB-SPRL Curriculum

(b) Learning Performance

Figure 8: (a) A WB-SPRL curriculum for the pick-and-place task. The vertical bars and small images visualize the states corresponding to different time steps of the expert trajectory. (b) Success rates over iterations for different learning algorithms. Mean and standard error is computed from 10 algorithm runs.

such challenging exploration problems is to learn the task via a curriculum of starting states. Such starting states can be obtained from an expert demonstration, i.e. a trajectory

$$\boldsymbol{\tau}_{\text{EXPERT}} : [0, 1] \mapsto \mathcal{S}.$$

A curriculum over this trajectory is then formally defined by choosing the context space to be the unit interval $\mathcal{C} = [0, 1] \subseteq \mathbb{R}$. The contextual parameter c only influences the initial state distribution $p_{0,c} = \delta_{\mathbf{s}=\boldsymbol{\tau}_{\text{EXPERT}}(c)}$. We investigate the NP- and WB-SPRL algorithms in this setting by recording one execution of a hand-crafted controller that first moves the end-effector above the block to be grasped, then lowers the end-effector, grasps the block and moves to the target. We slightly randomize the position of the goal as well as the block to enforce that the agent learns a robust policy. In a realistic scenario, we only record the expert demonstration at a discrete set of states $\{t_i | i = 1, \dots, N\}$ and hence we have a truly discrete CRL setting in this experiment very well suited for NP-SPRL. Furthermore, the finite number of contexts avoids the necessity for function approximators to approximate the expected performance $J(\pi, \mathbf{c})$ in a context $\mathbf{c}$, as we can simply estimate the performance in each (discrete) context via a sliding window. We again investigate two target context distributions, one being a narrow Gaussian target distribution with mean at $c = 0$ and negligible variance ($\mu_1(\mathbf{c})$) and the other being again a Dirac-delta with a negligible amount of probability in any context to be absolutely continuous w.r.t. any distribution ($\mu_2(\mathbf{c})$). The results in Figure 8 show that WB-SPRL generates a curriculum that allows the agent to learn a reliable policy by smoothly moving probability from later steps of the expert demonstration towards earlier ones. This learned policy is more reliable (25% vs. 100% success rate) and faster (69 vs. 13 steps for task completion) than the demonstration. Both default learning (always starting from the initial state) and NP-SPRL do not lead to successful policies. In appendix A, we show that the non-effectiveness of NP-SPRL is again grounded in the degenerate interpolation that puts no probability mass at intermediate time-steps regardless of the target distribution.

6 Conclusion

We investigated self-paced reinforcement learning (SPRL) outside of its current parametric restrictions. While this allows to fit complicated target distribution in the framework of SPRL, it also reveals important shortcomings of the KL divergence for measuring the similarity of (task) distributions in curriculum reinforcement learning. Stressing the need for an incorporation of metric assumptions on the context space into the curriculum, we turned towards Wasserstein distances to replace the KL divergences originally employed in SPRL. As we have shown, this alleviates the observed conceptual problems of SPRL. These findings motivate a lot of future work, such as more elaborate implementations of WB-SPRL that e.g. directly move the individual particles of the context distribution via the gradient of the performance measure $\nabla_{\mathbf{c}} J(\pi, \mathbf{c})$ instead of using the value distribution $p_J(\mathbf{c})$ as a proxy for this. Another line of future work may investigate other metrics than the squared euclidean one employed in this paper. More appropriate metrics may increase the performance in settings in which the euclidean distance between two contextual variables is less well-suited to encode the similarity between the MDPs that these variables represent.

REFERENCES

Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. arXiv preprint arXiv:1910.07113, 2019.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In Neural Information Processing Systems (NeurIPS), 2017.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In Neural Information Processing Systems (NeurIPS), 2016.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. arXiv preprint arXiv:1606.01540, 2016.

Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. Handbook of markov chain monte carlo. CRC press, 2011.

Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Stochastic control liaisons: Richard sinkhorn meets gaspard monge on a schrödinger bridge. SIAM Review (SIREV), 63(2):249–313, 2021.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. Neural Information Processing Systems (NeurIPS), 2013.

Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. Interpolating between optimal transport and mmd using sinkhorn divergences. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2019.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In Conference on Robot Learning (CoRL), 2017.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In International Conference on Machine Learning (ICML), 2018.

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, and Aviv Tamar. Bayesian reinforcement learning: A survey. Foundations and Trends® in Machine Learning, 8(5-6):359–483, 2015.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In International Conference on Machine Learning (ICML), 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In International Conference on Machine Learning (ICML), 2018.

Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. Stable baselines. `https://github.com/hill-a/stable-baselines`, 2018.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In AAAI Conference on Artificial Intelligence (AAAI), 2015.

Leonid Kantorovich. On the transfer of masses (in russian). Doklady Akademii Nauk, 37(2):227–229, 1942.

Pascal Klink, Hany Abdulsamad, Boris Belousov, and Jan Peters. Self-paced contextual reinforcement learning. In Conference on Robot Learning (CoRL), 2020a.

Pascal Klink, Carlo D' Eramo, Jan R Peters, and Joni Pajarinen. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), Neural Information Processing Systems (NeurIPS), 2020b.

Pascal Klink, Hany Abdulsamad, Boris Belousov, Carlo D'Eramo, Jan Peters, and Joni Pajarinen. A probabilistic interpretation of self-paced learning with applications to reinforcement learning. Journal of Machine Learning Research (JMLR), 22(182):1–52, 2021.

M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In Neural Information Processing Systems (NeurIPS), 2010.

Chang Liu, Jingwei Zhuo, Pengyu Cheng, Ruiyi Zhang, Jun Zhu, and Lawrence Carin. Understanding and accelerating particle-based variational inference. In International Conference on Machine Learning (ICML), 2019.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In Neural Information Processing Systems (NeurIPS), 2016.

Siqi Liu, Guy Lever, Zhe Wang, Josh Merel, SM Eslami, Daniel Hennes, Wojciech M Czarnecki, Yuval Tassa, Shayegan Omidshafiei, Abbas Abdolmaleki, et al. From motor control to team play in simulated humanoid football. arXiv preprint arXiv:2105.12196, 2021.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. Count-based exploration with the successor representation. In AAAI Conference on Artificial Intelligence (AAAI), 2020.

Deyu Meng, Qian Zhao, and Lu Jiang. A theoretical understanding of self-paced learning. Information Sciences, 414:319–328, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529, 2015.

Gaspard Monge. Mémoire sur la théorie des déblais et des remblais. De l'Imprimerie Royale, 1781.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. Journal of Machine Learning Research (JMLR), 21(181):1–50, 2020.

Radford M Neal. Annealed importance sampling. Statistics and Computing, 11(2):125–139, 2001.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. Foundations and Trends® in Machine Learning, 11(5-6):355–607, 2019.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In Conference on Robot Learning (CoRL), 2019.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In International Conference on Machine Learning (ICML), 2015.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In International Conference on Machine Learning (ICML), 2015.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In International Conference on Machine Learning (ICML), 2014.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. Nature, 550(7676):354, 2017.

Richard S Sutton and Andrew G Barto. Introduction to Reinforcement Learning. MIT Press, 1998.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In International Conference on Intelligent Robots and Systems (IROS), 2012.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17:261–272, 2020.

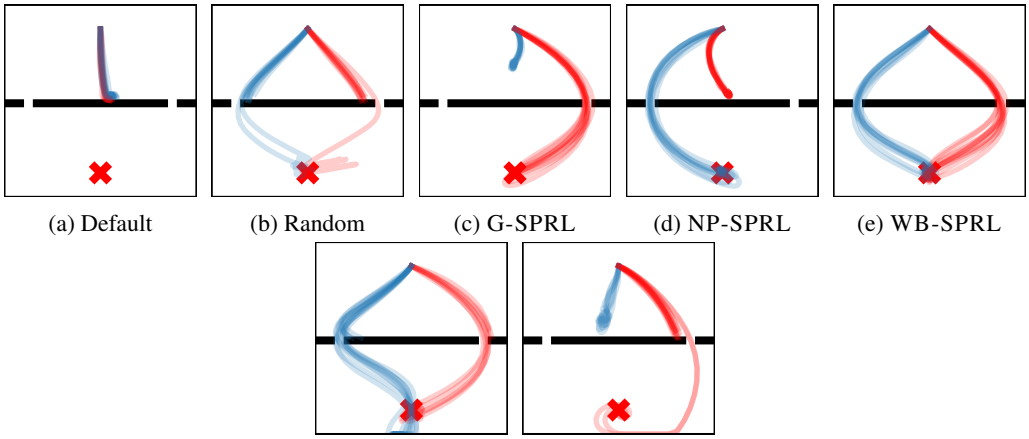
Daphna Weinshall and Dan Amir. Theory of curriculum learning, with convex loss functions. Journal of Machine Learning Research (JMLR), 21(222):1–19, 2020.

Andre Wibisono. Sampling as optimization in the space of measures: The langevin dynamics as a composite optimization problem. In Conference on Learning Theory (COLT), 2018.

Aaron Wilson, Alan Fern, Soumya Ray, and Prasad Tadepalli. Multi-task reinforcement learning: a hierarchical bayesian approach. In International Conference on Machine Learning (ICML), 2007.

Jan Wöhlke, Felix Schmitt, and Herke van Hoof. A performance-based start state curriculum framework for reinforcement learning. In International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1503–1511, 2020.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In International Conference on Learning Representations (ICLR), 2021.



(a) Default (b) Random (c) G-SPRL (d) NP-SPRL (e) WB-SPRL

(f) ALP-GMM (g) GOALGAN

Figure 9: Final trajectories generated by the different investigated curricula in the point mass environment. The color encodes the context: Blue represents gates positioned at the left and red gates positioned at the right.

A QUALITATIVE RESULTS

Figure 9 shows trajectories that have been generated by agents trained with different curricula. We see that directly learning on the two target tasks prevents the agent from finding the gates in the wall to pass through. Consequently, the agent minimizes the distance to the goal by moving right in front of the wall (but not crashing into it) to accumulate reward over time. We see that random learning indeed generates meaningful behavior. This behavior is, however, not precise enough to pass reliably through the wall. As mentioned in the main paper, G-SPRL and NP-SPRL only learn to pass through one of the gates. We see that in the particular run displayed in Figure 9, NP-SPRL stayed closer to the wall than G-SPRL in those contexts in which it could not pass the wall. This better behavior is probably a result of NP-SPRL ultimately sampling both target tasks during training while G-SPRL only focuses on one. Finally, WB-SPRL learns a policy that can pass through both gates reliably, showing that the gradual interpolation towards both target tasks allowed to learn both of them. ALP-GMM also learns good policies. The generated trajectories are, however, not as precise as the ones learned with WB-SPRL. The reason for this lack of precision is probably that ALP-GMM has no notion of target distribution and hence cannot focus learning on the target tasks at later iterations. Finally, GOALGAN partly creates meaningful behavior. However, this behavior is unreliable and hence leads to low returns due to the agent frequently crashing into the wall.

Figure 10 shows the distribution $p_{\alpha,\eta}(\mathbf{c})$ generated by NP-SPRL for different iterations in the pick and place environment. We see that regardless of the target distribution, the interpolation skips all intermediate states of the expert demonstration. This is surprising, given that the interpolation worked at least somewhat desirably in the point mass environment for a Gaussian target distribution. Comparing Figures 10b and 10a, we see that the Gaussian target distribution leads to $p_{\alpha,\eta}(\mathbf{c})$ putting probability mass on a few more states than only the initial one. However, these states are still so close to the initial state that no learning takes place.

B IMPLEMENTATION DETAILS OF WB-SPRL

The implementation of WB-SPRL relies on the computation of the Wasserstein barycenter $p_{\boldsymbol{\beta}}(\mathbf{c})$. To compute this barycenter, we represent the distributions $\mu(\mathbf{c})$, $p_J(\mathbf{c})$ and $q(\mathbf{c})$ via a set of particles. For a distribution $p(\mathbf{c})$, we denote the particle-based representation as $\tilde{p}(\mathbf{c})=\frac{1}{N}\sum_{n=1}^{N}\delta_{\mathbf{c}_n^p}(\mathbf{c})$ where $\mathbf{c}_n^p \sim p(\mathbf{c})$ and $\delta_{\mathbf{c}_n^p}$ is a Dirac delta located at $\mathbf{c}_n^p$. This representation allows to compute the so-called Monge maps $T_{\tilde{p}_1 \to \tilde{p}_k} : \mathcal{C} \mapsto \mathcal{C}$ that encode how to move the particles from the distribution $\tilde{p}_1$ to $\tilde{p}_k$. In the case of the Wasserstein-2 distances, we can use a linear combination of these Monge maps as a reasonable approximation to the exact barycenter $p_{\boldsymbol{\beta}}(\mathbf{c})$

$$\tilde{p}_{\boldsymbol{\beta}}(\mathbf{c}) \approx \left(\beta_1 T_{\tilde{q} \to \tilde{p}_J} + \beta_2 T_{\tilde{q} \to \tilde{\mu}}\right)_{\#} \tilde{q}(\mathbf{c})$$

where $T_{\#}\tilde{q}$ is the push-forward of $\tilde{q}$ by T, i.e. the application of the weighted Monge maps to the particles of $\tilde{q}$

$$T_{\#}\tilde{q}(\mathbf{c}) = \frac{1}{N}\sum_{n=1}^{N} \delta_{T(\mathbf{c}_n^q)}(\mathbf{c}).$$

The benefit of this approximation is that the maps $T_{\tilde{q}\to\tilde{p}_J}$ and $T_{\tilde{q}\to\tilde{\mu}}$ only need to be computed once in order to compute $\tilde{p}_{\boldsymbol{\beta}}(\mathbf{c})$ for arbitrary $\boldsymbol{\beta}$. This fast computation of $\tilde{p}_{\boldsymbol{\beta}}(\mathbf{c})$ allows to quickly optimize $\boldsymbol{\beta}$ via objective (7).
To make the optimization of $\tilde{p}_{\boldsymbol{\beta}}(\mathbf{c})$ even faster, we replace the two Wasserstein distances in (7) by approximations, resulting in

$$\min_{w_\mu, w_J} \sum_{n=1}^{N} \|T_{\tilde{q}\to\tilde{\mu}}(\mathbf{c}_n^q) - \mathbf{c}_n^{p_{\boldsymbol{\beta}}}\|^2 \;\; \text{s.t} \;\; \frac{1}{N}\sum_{n=1}^{N} J(\pi, \mathbf{c}_n^{p_{\boldsymbol{\beta}}}) \geq \delta \quad \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{c}_n^{p_{\boldsymbol{\beta}}} - \mathbf{c}_n^q\|^2 \leq \epsilon.$$

The objective term of above constrained optimization problem is a computationally cheap replacement for $\mathcal{W}_2(p_{\boldsymbol{\beta}}, \mu)$. Instead of computing the Wasserstein distance to μ, we simply ask the transported particles of $\tilde{p}_{\boldsymbol{\beta}}$ to follow the transport plan $T_{\tilde{q}\to\tilde{\mu}}$ as closely as possible. This avoids the (comparatively) expensive computation of $\mathcal{W}_2(p_{\boldsymbol{\beta}}, \mu)$ for each choice of $\boldsymbol{\beta}$. Finally, we also avoid to compute $\mathcal{W}_2(p_{\boldsymbol{\beta}}, q)$ but rather replace it with a pessimistic substitute $\frac{1}{N}\sum_{n=1}^{N} \|\mathbf{c}_n^{p_{\boldsymbol{\beta}}} - \mathbf{c}_n^q\|^2 \leq \epsilon$. This substitute is guaranteed to be larger than $\mathcal{W}_2(p_{\boldsymbol{\beta}}, q)^2$ since when using the particle representation, the Wasserstein distance $\mathcal{W}_2(p_{\boldsymbol{\beta}}, q)$ corresponds to finding a permutation $\pi : [1, N] \mapsto [1, N]$ that optimizes

$$\min_\pi \sqrt{\frac{1}{N}\sum_{n=1}^{N} \|\mathbf{c}_n^{p_{\mathcal{W}}} - \mathbf{c}_{\pi(n)}^q\|^2}.$$

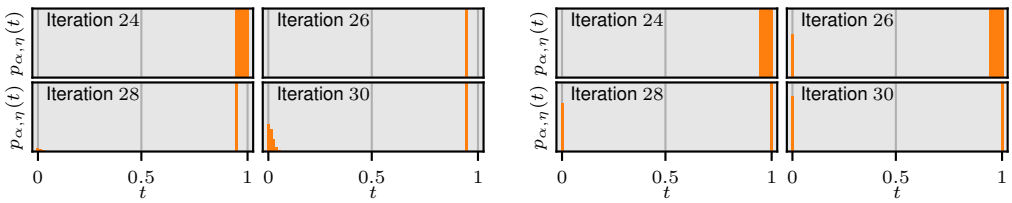
Note that the approximations to the Wasserstein distances in (7) do not alter the transport plan and with that $\tilde{p}_{\boldsymbol{\beta}}(\mathbf{c})$. The approximations may simply alter the optimal parameters $\boldsymbol{\beta}^*$ to (7). However, since the approximations have the same asymptotic behavior, these differences do not hinder good learning performance as can be seen in the experimental section. For the computations of the Monge maps, we use the GeomLoss library (Feydy et al., 2019) which allows to solve entropy regularized optimal transport problems (Cuturi, 2013). We then use the SciPy library (Virtanen et al., 2020) to adjust the weights $\boldsymbol{\beta}$ subject to the performance and distance constraints.

C Experimental Details

This section discusses hyperparameters and additional details of the conducted experiments that could not be provided in the main text due to space limitations.

C.1 Algorithm Hyperparameters

The two main parameters of the three SPRL algorithms are the performance threshold δ, the allowed distance between subsequent distributions ϵ (note that we say "distance" as the interpretation is different for WB-SPRL than for G- and NP-SPRL). In practical realizations, two more parameters need to be taken into account: the number of steps per epoch n_{step} (i.e. between an update of the context distribution) and the number of epochs that take place before any context distribution update



(a) Target $\mu_1(c)$ (b) Target $\mu_2(c)$

Figure 10: Visualization of the curriculum distribution $p_{\alpha,\eta}(\mathbf{c})$ of NP-SPRL for different iterations in the pick and place tasks. (a) shows the curriculum generated for the Gaussian target distribution $\mu_1(\mathbf{c})$. (b) shows the curriculum in case of using the Dirac-delta-like distribution $\mu_2(\mathbf{c})$.

	GENERAL				G-SPRL		NP-SPRL		WB-SPRL
ENV.	δ	$\epsilon_{\text{KL}}\|\epsilon_{\text{M}}$	n_{STEP}	n_{OFFSET}	$\boldsymbol{\sigma}_{\text{LB}}$	$D_{\text{KL}_{\text{LB}}}$	$\boldsymbol{\sigma}_{\text{LB}}$	$D_{\text{KL}_{\text{LB}}}$	δ_H
MAZE	.65	.25|1	10000	5	-	-	-	-	.9
POINT-MASS	4	.25|.25	4096	5	[.2 .1875]	8000	4	1000	6
PNP	.5	.5|.13	3000	25	-	-	-	-	.8

Table 1: Hyperparameters of the investigated SPRL algorithms in the different learning environments, as described in appendix C. Note the differentiation between ϵ for G- and NP-SPRL (ϵ_{KL}) and WB-SPRL (ϵ_{M}).

is allowed n_{offset}. For G-SPRL, Klink et al. employed a lower bound on the standard deviation $\boldsymbol{\sigma}_{\text{lb}}$ of the context distribution that needs to be respected until the KL divergence w.r.t. $\mu(\mathbf{c})$ falls below a threshold D_{KL}. This avoids a collapse of the context distribution in the case of a very narrow target context distribution $\mu(\mathbf{c})$. In NP-SPRL, we translated this into a lower bound on the entropy of the context distribution. For WB-SPRL, this lower bound is not required. Only, the additional performance lower bound $\delta_H > \delta$ for computing the "value distribution" $p_J(\mathbf{c})$. Aforementioned parameters are listed in Table 1 for the different environments.
For ALP-GMM, the relevant hyperparameters are the percentage of random samples drawn from the context space p_{rand}, the number of completed learning episodes between the update of the context distribution n_{rollout} as well as the maximum buffer size of past trajectories to keep s_{buffer}.
For GOALGAN, we tuned the amount of random noise that is added on top of each sample δ_{noise}, the number of policy rollouts between the update of the context distribution n_{rollout} as well as the percentage of samples drawn from the success buffer p_{success}.
Finally, for ACL, the Exp3.S algorithm that ultimately realizes the curriculum requires two hyperparameters to be chosen: the scale factor for the updates of the arm probabilities η and the ϵ parameter of the ϵ-greedy exploration strategy. We combine ACL with the absolute learning progress (ALP) metric also used in ALP-GMM. In the discrete pick and place setting, the absolute learning progress in a context $\mathbf{c}$ can be estimated by keeping track of the last reward obtained in $\mathbf{c}$ and then computing the absolute difference between the return obtained from the current policy execution and the stored last reward. Implementing the ACL algorithm by Graves et al. (2017), we had numerical issues due to the normalization of the ALPs via quantiles computed from a representative sample. This is probably due to the sparse nature of the task. Consequently, we normalized via the maximum and minimum ALP seen over the entire history of tasks. The parameters of all employed baselines are given in Table 2.

C.2 TASK DESCRIPTIONS

We now detail details of the individual experiments, such as context-, state- and action spaces as well as the employed RL algorithms. As RL agents, we use PPO and SAC implemented in the `Stable Baselines` library (Hill et al., 2018). In all tasks, we represent policy- and/or value function networks using neural networks with three hidden layers of 128 neurons and $\tanh$ activations.

C.2.1 MAZE

The maze task is simulated using MuJoCo (Todorov et al., 2012) by defining a sphere with radius $.15$ that can move along two prismatic joints along x- and y-direction. The goal (i.e. the context) can be chosen within $\mathcal{C} = [-3, 15] \times [-3, 15] \subseteq \mathbb{R}$. The actually reachable space of positions (and with that goals) is a subset of $[-1, 13] \times [-1, 13]$ due to the "hole" because of the inner walls of the maze. The state of the environment is given by the x- and y-position and -velocity of the agent. The reward is sparse, only returning a reward of one if the goal is reached and zero otherwise. A goal is considered reached if the Euclidean distance between goal and position of the point-mass falls below 0.15.
We use the PPO algorithm for learning in this task, updating the policy every 10.000 environment steps for 10 epochs with 20 mini-batches. The parameter λ for the generalized advantage estimation is set to 0.995 and the entropy coefficient is set to zero. All other parameters are set to the default of the `Stable Baselines` implementation.
For algorithms that support the specification of an initial context distribution (SPRL variants and GOALGAN), we initialize the context distribution to be uniform over $[-1, 1] \times [-1, 1]$ for GOAL-

	ALP-GMM			GoalGAN			ACL	
Env.	p_{RAND}	n_{ROLLOUT}	s_{BUFFER}	δ_{NOISE}	n_{ROLLOUT}	p_{SUCCESS}	η	ϵ
Maze	.2	100	500	.1	100	.3	-	-
Point-Mass	.2	100	500	.1	200	.3	-	-
PnP	.2	50	500	.1	50	.2	.05	.05

Table 2: Hyperparameters of the investigated baseline algorithms in the different learning environments, as described in appendix C.

GAN, NP-SPRL and WB-SPRL and $\mathcal{N}(\mathbf{0}, .06\mathbf{I})$ for G-SPRL. This initial context distribution samples goals around the initial position of the agent.

C.3 Point-Mass

The environment setup is the same as the one investigated by Klink et al. (2020b; 2021) with the only difference in the target context distributions $\mu_1(\mathbf{c})$ and $\mu_2(\mathbf{c})$. We use PPO with 4.096 steps per policy update, four update epochs with 8 mini-batches in each policy update, λ=0.99 and an entropy coefficient of zero. All other parameters are left to the implementation defaults of the `Stable Baselines` implementation.

C.4 Pick and Place

The investigated environment is a slightly modified version of the `FetchPickAndPlace` environment from the `OpenAI Gym` Brockman et al. (2016) environment suite. The dynamics and reward functions are exactly the same as the ones from the original environment. We only modify the initial position of the cube as well as the goal to which the cube needs to be moved. The initial x- and y-position of the cube on the table is sampled uniformly within $[1.31, 1.37] \times [0.72, 0.78]$ and the goal position is sampled within $[1.31, 1.37] \times [0.87, 0.93] \times [0.59, 0.65]$. If the environment is reset to a state in which the cube is in contact with the gripper, its position is not randomized as otherwise the cube may e.g. fall out of the gripper due to randomization, which would counteract the idea behind the curriculum.
The demonstration is generated from a simple controller that first aligns the gripper with a position 10cm above the cube, then lowers the gripper until the cube is between its two fingers, closes the gripper and then moves the endeffector to the goal position. The control command for the robot endeffector is computed via

$$\mathbf{u} = \text{sign}(\Delta_{\mathbf{x}}) \max(0.2, |\Delta_{\mathbf{x}}|), \quad \Delta_{\mathbf{x}} = \mathbf{x}_{\text{des}} - \mathbf{x},$$

where $\mathbf{x}_{\text{des}}$ is the desired- and $\mathbf{x}$ the current endeffector position. Lower bounding the control command by 0.2 avoids the movements becoming very small as the endeffector approaches the target position, which would lead to unnecessarily long demonstrations. As soon as we are within a 0.01 distance to the target position we switch to the next step in the previously outlined sequence.
As an RL agent, we use SAC with 5 steps per policy update, a batch size of 512, a maximum replay buffer size of 100.000 and a learning rate 3×10^{-4}. All other parameters are left to the implementation defaults of the `Stable Baselines` implementation.
As in the maze task, we specify initial context distributions for GoalGAN and the investigated SPRL variants. These context distributions assign uniform probability to the last 4 time-steps of the trajectory.