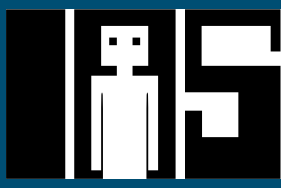
Particle-Based Adaptive Sampling for Curriculum Learning

Master thesis by Heiko José Helmut Carrasco Huertas
Date of submission: April 25, 2022

1. Review: Pascal Klink
2. Review: Carlo d'Eramo
3. Review: Jan Peters

Darmstadt



To my family and my boyfriend

Erklärung zur Abschlussarbeit gemäß §22 Abs. 7 APB TU Darmstadt

Hiermit versichere ich, Heiko José Helmut Carrasco Huertas, die vorliegende Masterarbeit gemäß §22 Abs. 7 APB der TU Darmstadt ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 25. April 2022

H. Carrasco

Abstract

Inspired by nature, curriculum learning (CL) has been shown to be a promising approach to increase the performance and sample efficiency of a reinforcement learning (RL) agent on difficult tasks. In combination with transfer learning it enables the agent to leverage previous experience on a series of task variants with increasing difficulty, allowing for a more guided training process and a more versatile agent. Often, these *curricula* are represented as series of distributions, from which concrete tasks can be sampled and trained on. Although this representation allows for a lot of flexibility when generating curricula, the nature of distributions lead to some inefficiencies during the sampling and training process. Depending on the task, newly sampled tasks might end up in regions already covered by previous curriculum distributions, leading to less informative agent runs and training. In this thesis, we propose the usage of particle-based variational inference methods (ParVIs) to more effectively sample new tasks from less covered areas of the task space. The agent is only evaluated on these new tasks but is trained on both previous and new experiences. We specifically investigate two methods in the space of ParVIs called Stein variational gradient descent (SVGD) and Stein points (SP) and develop new sampling algorithms based on them to use for curriculum learning. These algorithms are later investigated on both pedagogical examples and also on a RL task using self-paced reinforcement learning (SPRL) for generating curricula. These experiments allow us to gain some insights into the inner workings of curriculum learning in the space of RL and how ParVIs can be used in the future to improve sampling in curriculum learning.

Acknowledgements

This thesis was written under the supervision of Pascal Klink, Carlo d'Eramo, and Jan Peters. I want to thank them for their good support and feedback they provided as part of their supervision and acknowledge their contribution to this work.

Many thanks to Fabian Damken and Claas Völcker for commenting and proofreading this thesis.

Contents

Abbreviations

List of Abbreviations

Notation	Description
ARD	automatic relevance determination
CL	curriculum learning
CMDP	contextual Markov decision process
DL	deep learning
ELBO	evidence lower bound
GAN	generative adverserial networks
i.i.d.	independently and identically distributed
ISPS	iterative Stein point sampler
KL	Kullback-Leibler
KSD	kernelized Stein discrepancy
MDP	Markov decision process
MIQCP	mixed-integer quadratically constrained programming problem
ML	machine learning

NF	normalizing flow
OSPS	optimization Stein point sampler
ParVIs	particle-based variational inference methods
RBF	radial basis function
REPS	relative entropy policy search
RKHS	reproducing kernel hilbert space
RL	reinforcement learning
SAC	soft actor critic
SP	Stein points
SPRL	self-paced reinforcement learning
SVGD	Stein variational gradient descent
VI	variational inference

1. Introduction

Although the field of reinforcement learning (RL) has evolved considerably over the last years, especially with the application of deep learning (DL) based approaches, some fundamental problems still remain. Current approaches require great amounts of interactions with the environment to learn the complex and multi-step tasks encountered in the real world. Not only does this lead to long training times but the resulting agents are often not able to generalize to variations in the tasks.

There are many approaches to alleviate such problems one of which is *curriculum learning (CL)*. Although this technique has its roots in supervised learning [1], its adoption in RL has been quite successful with many applications in a variety of different environments [2].

Curriculum learning takes its inspiration from nature, where both humans and animals have shown far better learning performance when presented with tasks in an increasing order of difficulty [1]. This can be modeled using probability distributions from which tasks are drawn. By choosing the series of distributions carefully, the agent can be guided in its learning process without the need of specifying every single task. While this formulation has been used with a lot of success in the past [1, 3, 4], it might not be as efficient as it could be due to the sequential nature of the sampling distributions.

To adequately cover the task space, some overlap between the curriculum distributions will necessarily occur over the course of training. In practice, this will lead to later task samples being close to previous ones. Although usually not a problem in the space of supervised machine learning where inference only takes a small amount of time, this can introduce difficulties in RL settings. Here, a full trajectory rollout on a task sample might take a considerable amount of time due to factors such as the simulation complexity of the environment. Therefore, rollouts should ideally be limited to cases where truly new experiences can be collected while previous recorded experiences can be used to prevent catastrophic forgetting [5]. This, however, raises the question how to determine when (and where) to use new task samples and when previous experience can be reused. In this thesis we propose to answer this questions be recasting it as a sampling problem, where we want to add new samples and remove old ones such that the resulting sample set more closely matches the current curriculum distribution.

One approach to this problem of sampling lies in the field of variational inference (VI), which provides several methods to match a posterior distribution using an often simpler proposal distribution by iteratively minimizing a statistical discrepancy between them [6]. Classically, this is often done by minimizing the Kullback-Leibler (KL) divergence between the a parametric proposal and a target distribution, however, a new subfield has recently gained the attention of the variational inference (VI) community: Particle-based variational inference methods (ParVIs) [7]. These methods provide the means of generating or moving a set of samples (often called *particles*) so that they closely match the target posterior distribution. Using these methods to generate (and remove) samples from the series of curriculum distributions is the main contribution of this thesis.

We focus on two specific approaches called Stein variational gradient descent (SVGD) and Stein points (SP), which both derive their main intuition from Stein's method. We show the necessary changes required to use these two ParVIs as curriculum learning samplers, which can fit an existing set of samples (such as previous task samples) to a target distribution proposed by a curriculum learning algorithm by adding or moving a set of proposal samples.

The rest of this thesis is structured as follows: We start with a description of our problem functioning as a motivation for the rest of the thesis in chapter 2 In chapter 3, the necessary fundamental methods and equations behind particle-based variational inference methods and curriculum learning are explained, which we then contextualize by presenting related work in chapter 4. We explain the two investigated methods for ParVIs in chapter 5 together with the necessary changes to fit them into a curriculum learning framework. Afterwards, we show some strengths and weaknesses of our new samplers, first on synthetic experiments in chapter 6 and afterwards on a reinforcement learning baseline in chapter 7. Finally, we conclude with a summary of our findings and outline potential prospects of our work in chapter 8.

Our contributions to the state-of-the-art are: First, we identify an inefficiency in current curriculum learning approaches and identify it as a problem of sequential sampling. Second, we propose a solution for this problem by adapting particle-based variational inference methods to be used as samplers for curriculum learning. Third, we evaluate our new samplers on pedagogical and CL problems in reinforcement learning.

2. Problem Statement

We start with a description of our investigated problem, which serves as a motivation for the following chapters detailing our methods. We also provide an example of a curriculum learning task in the space of reinforcement learning, which acts as a running example and will later be investigated as part of our RL experiments.

2.1. Inefficient Sampling in Curriculum Learning for Reinforcement Learning

The motivation for this thesis is derived from observations made in the training process of a curriculum learning algorithm for reinforcement learning called self-paced reinforcement learning [3, 8], although they generalize to other CL algorithms which model there curricula using a series of distributions [1, 4, 9]. These formulations tend to generate similar task samples due to the overlap of distributions and the sampling process, which leads to increased training times.

As a means to make the problem more intuitive, we are now defining a running example for this chapter, which is inspired by Klink *et al.* [3]. Assume that we want to train an agent to push a ball from a fixed starting point to a goal in a two dimensional space. The agent controls the position of the ball by applying a force to it, which can come from either spatial dimensions. To increase task complexity, we add a wall the agent must not touch between the ball and the goal position, only containing a small gap. We now want to train an agent, such that it can solve the task in the configuration shown on the right in Figure 2.1.

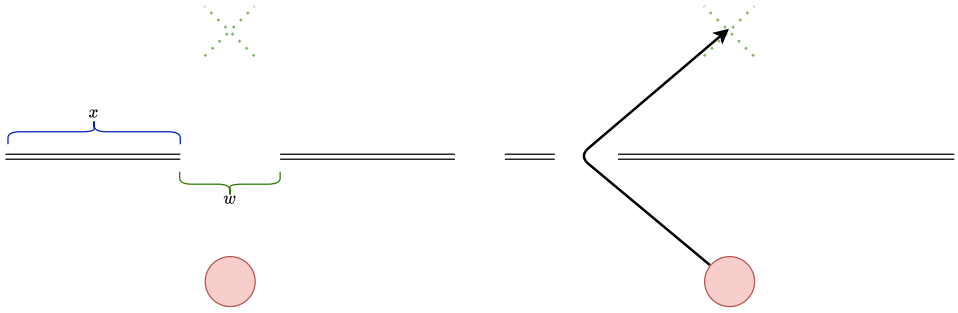
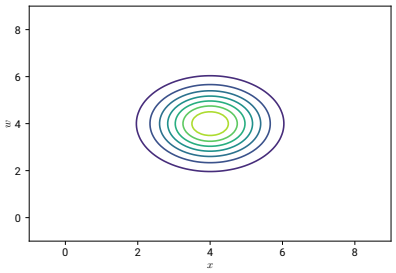


Figure 2.1.: The example environment with the two parameters x, w is depicted on the left, while a working trajectory is shown for a less trivial setting of x and w on the right.

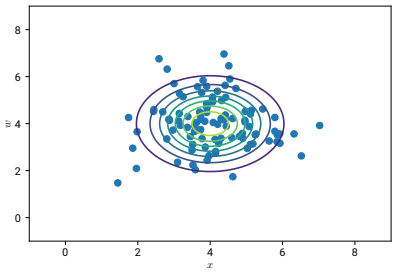
While it is most likely possible to directly train an agent in this environment, it might require many attempts for the agent to just discover the gate and move through it without touching the walls. It is, however, possible to use curriculum learning to first learn a simpler variant of the task and gradually increase complexity until the targeted configuration is reached. We can do this by parametrizing the environment and specifically the

position x and width w of the gap. This allows us to use a simpler initial configuration as shown on the left in Figure 2.1, before gradually moving towards the final configuration shown on the right.

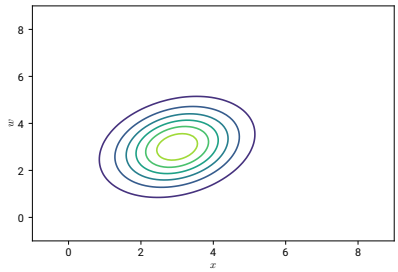
We do so by defining an initial normal bivariate distribution $q(\theta_0, \mathbf{c})$, from which we can sample the parameters $\mathbf{c} = (x, w)$. Then, according to a CL algorithm of our choosing, we update θ_i after every training iteration i of the agent. A visualization of this process is provided in Figure 2.2.



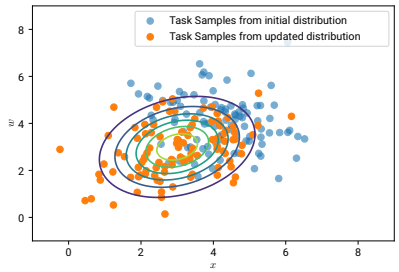
(a) Initial task distribution $q(\theta_0, \mathbf{c})$



(b) Added task samples $q(\theta_0, \mathbf{c})$



(c) Updated task distribution $q(\theta_1, \mathbf{c})$



(d) Updated CL distribution with previous samples $q(\theta_1, \mathbf{c})$

Figure 2.2.: One step in the curriculum learning process. Observe how the overlap between the distribution leads to "duplicate" samples.

If we look at the last step in the provided example Figure 2.2d, we can identify a potential inefficiency when using distributions to model curricula: Due to the nature of the sampling process, some of the new task samples drawn are close to already explored samples from the previous iteration. Training the agent on these new samples will most likely not yield any significant insights into the task (assuming a smooth value function as in our example) and therefore unnecessarily increases training time. This is especially true for off-policy RL algorithms such as soft actor critic (SAC) [10] and certain implementations of relative entropy policy search (REPS) [11], which use some form of *experience buffer* where previously generated trajectories or action-step-reward tuples are saved and re-used in the training process. While these buffers by design "forget" older experiences by removing samples the oldest samples when they fill up, it might not be sufficient in the case of CL where significant overlap can occur between two iterations as seen in Figure 2.2.

However, it is possible to leverage the experience buffer as a countermeasure against the aforementioned problem of duplicate samples. This requires a small extension of the buffer, where not only previous experiences are saved but also their associated task sample from which they were generated. Using this information, we can now take these older task samples into account when sampling from a new task distribution by only running the agent on samples which are distinct enough from previous experiences, however, the agent is still trained on the union of the new and old experiences. We show an example of such a set of samples in Figure 2.3.

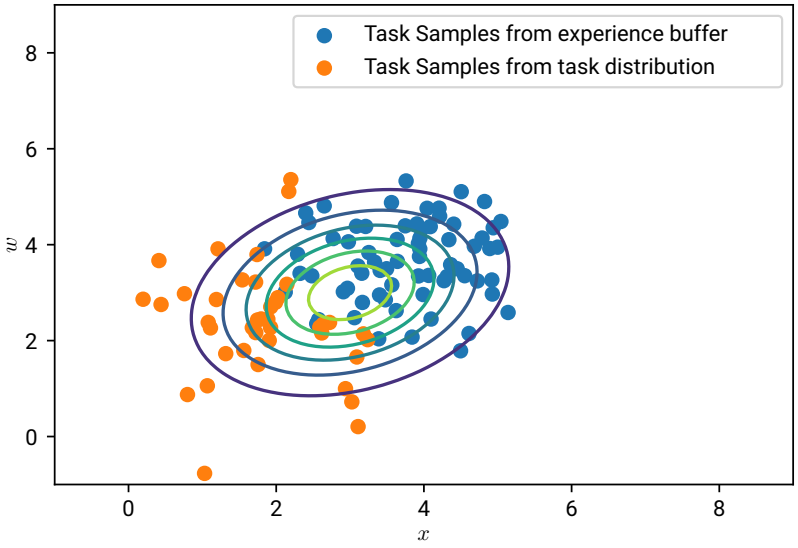


Figure 2.3.: Visualization of the combination of experience buffer and newly acquired samples. Here, we only evaluate the agent on the new samples (in orange) but perform the final training on the joint set. Compare this to the joint set shown in Figure 2.2d.

This leaves us with two main questions:

(**1**) How do we determine and acquire samples which are distinct enough from previous experiences to require runs of the agent?

(**2**) How do we remove older stale samples from the buffer according to the training process?

For the first question, we ideally want to adapt the sampling procedure to only produce samples in the non-covered part of the distribution. While we already saw a heuristic for the second question, i.e. removing the oldest samples when the buffer is full, its underlying assumption (that older samples are less relevant than newer ones) does not always hold: For example, we might want to start with a wide distribution covering large spaces of the task space and subsequently narrow down the variance until the target distribution size is reached. It might be beneficial in that case to not uniformly remove samples from the distribution by dropping the oldest samples, but concentrate on unlikely areas far from the target distribution first.

The proposal of this thesis is to cast both questions as a single *variational inference* problem. That is, our goal is to minimize the statistical distance between the empirical distribution of the experience buffer $\hat{q}$ and the current distribution produced by the curriculum learning algorithm. Since $\hat{q}$ consists solely of samples, we propose the usage of Particle-based variational inference methods, which are a recent development in the space of variational inference. They naturally use samples, which are often referred to as particles, to approximate a target distribution and do not rely on $\hat{q}$ to have a parametric form [7].

We can formalize this notion as follows: Given an experience buffer $\{\hat{q}_i\}_{i=1}^{n}$ containing task samples from previous iterations of the training process, a set of proposal samples $\{x_i\}_{i=1}^{m}$ and a target distribution p generated by a curriculum learning algorithm. Our goal is it to find a subset of both $\{\hat{q}_i\}_{i=1}^{t}, \{x_i\}_{i=1}^{k}$ such that their union q^* is close to p, i.e. by having a minimal statistical distance $D(q^*, p)$.

This leaves us with the task of finding a method to select the subsets of both old task samples and proposal samples and to find a statistical discrepancy which can be calculated between a set of samples and a parametric distribution. For the latter problem, we will see in the next chapter that we can derive such a discrepancy using Stein's method. Combining this method with techniques from variational inference, as we describe in chapter 5, will then allow us to tackle the former problem of selecting a fitting set of proposal samples x. There we also present our adaptions to extend these techniques such that they also select an appropriate subset of $\hat{q}$, allowing us to use the final samplers in our intended domain of curriculum learning.

3. Background

We provide some background into the underlying methods of curriculum learning and particle-based variational inference methods in this chapter. Starting with curriculum learning in section 3.1, we give an overview over the field and its application for reinforcement learning. We then continue with an account of variational inference methods before moving on to Stein's method, which together form the theoretical base for particle-based variational inference methods.

3.1. Curriculum Learning

Although initial attempts at using curriculum learning to train machine learning agents date back to the early 1990s [2], it received new attention due to the formalization by Bengio *et al.* for the space of supervised learning in 2009 [1]. They model the curriculum as a series of probability distributions $(q_0, \ldots, q_n)$ where the samples $x \sim q_i$ represent the different tasks the agent encounters. Depending on the context, the samples x can represent different things such as images from a dataset or simulation parameters. While the weights of the distributions are initially set to include a subset of the task space which can be considered "simple"[1], they are updated over the course of training until they match the weights of a target distribution. While this update can occur in fixed intervals (e.g. to encourage more exploration of the space), it can also be conditional on the agent's performance. However, designing such an algorithm has proven non-trivial, due to the difficulties involved in choosing an appropriate challenge for the agent that aids in learning but does not inhibit it. There are some attempts of this, especially in the supervised domain, which use approaches such as multi-armed bandits [9] or train a support vector machine to predict model performance and focus on low performance samples [9].

3.1.1. Curriculum Learning for Reinforcement Learning

While the generation of a curriculum already poses some challenges in the domains of supervised learning, additional hurdles need to be overcome to use curriculum learning in the space of reinforcement learning. Although generating a variety of tasks is often easier compared to supervised approaches due to the wide availability of parametrized environments, this is often counter-balanced by the difficulties involved in classifying different parts of the induced task space as challenging or easy to a constantly evolving agent [2]. Nevertheless, a number of methods have been developed in the recent years covering a wide variety of potential applications (see [2] for a comprehensive review of the field).

[1] Depending on the overall task the meaning of simple might differ. For example, it can mean that the selected subset of tasks is small such that the intra-set variance is bounded. Alternatively, a simple subset might refer to a region in the task space with the highest initial performance of the agent preceding any training.

For this thesis, we focus on a subfield of these efforts which are based on two closely connected fields of the general reinforcement learning research: *contextual reinforcement learning* and *transfer learning*. In transfer learning, the agent is not immediately trained on a target task but on a set of subtasks with the goal of *transferring* the acquired knowledge to the target task. Depending on the task setting, this might entail transferring different parts of the RL training such as previous experiences [12] or policies [8, 13]. It is possible to connect transfer learning with contextual reinforcement learning by interpreting $\mathbf{c}$ as a task variable encoding a specific subtask.

In comparison to "regular" RL, the goal of contextual reinforcement learning is to learn a viable policy over a set of Markov decision processs, where the dynamics and the rewards depend on a context parameter $\mathbf{c}$ [14]. The standard formulation of the Markov decision process (MDP) can easily be extended to this additional parameter: A contextual Markov decision process (CMDP) is a parametrized MDP $M(\mathbf{c}) = (\mathcal{S}, \mathcal{A}, p_{\mathbf{c}} r_{\mathbf{c}}, p_{0,\mathbf{c}})$, where the state-action space $\mathcal{S}, \mathcal{A}$ are shared between the instances, while the reward $r_{\mathbf{c}}$, initial state distribution $p_{0,\mathbf{c}}$ and the conditional transition probability $p_{\mathbf{c}}$ are dependent on a contextual variable $\mathbf{c} \in \mathcal{C}$. Given a distribution over contexts $\mu(\mathbf{c})$ and a conditional policy $\pi(\mathbf{a}|\mathbf{s}, \mathbf{c}, \omega)$ with parameters ω, the contextual RL objective is

$$\mathcal{J}(\omega, \mu) = \max_{\omega} \mathbb{E}_{\mathbf{c}\sim\mu} \left[\mathcal{J}(\omega, \mathbf{c})\right] = \max_{\omega} \mathbb{E}_{\mathbf{c}\sim\mu, \mathbf{s}\sim p_{0,\mathbf{c}}} \left[V_\omega(\mathbf{s}, \mathbf{c})\right],$$

where $V_\omega(\mathbf{s}, \mathbf{c})$ is the contextual value function

$$V_\omega(\mathbf{s}, \mathbf{c}) = \mathbb{E}_{\mathbf{a}\sim\pi(\cdot|\mathbf{s},\mathbf{c},\omega)} \left[r_{\mathbf{c}}(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}'\sim p_{\mathbf{c}}(\cdot|\mathbf{s},\mathbf{a})} \left[V_\omega(\mathbf{s}', \mathbf{c})\right]\right].$$

Returning to the topic of CL, we will use the aforementioned formulation and specifically investigate applications [3, 8] where the curriculum is modeled using a series of parametrized distributions $\mu(\theta_{\mathbf{i}}, \mathbf{c})$, where the parameter θ_i is updated according to the performance of the agent at iteration i. Compared to other approaches which just vary initial or terminal states [15, 16] or only modify the reward function [17], this formulation is less restrictive regarding the particular choice of MDP. Incidentally, this is quite similar to the approach by Bengio *et al.* in their application of curriculum learning to the supervised learning setting.

As we have discussed in chapter 2, this choice of modeling can lead to inefficiencies in the training when the curriculum distributions overlap. Our proposal in this thesis is to use methods of variational inference to sample from the current curriculum distribution while taking previous samples into account. For this, we need a basic notion of variational inference, which we will present in the next section.

3.2. Variational Inference

In variational inference (VI), the task is to approximate a target distribution $p(x)$ by finding a closely matching proposal distribution from a predefined set $q^*(x) \in \mathcal{Q}$, which is often chosen such that sampling from its members is simple. While there exist a number of methods to determine the most fitting distribution $q^*(x)$ (see [6] for a literature review), a common way is based on minimizing the KL divergence between q and p,

$$\arg\min_{q\in\mathcal{Q}} D_{KL}\left(q(x) \| p(x)\right) \tag{3.1}$$

While this formulation is intuitive, it is often not feasible to optimize directly due to the proposal distribution q usually not being able to capture the full complexity of the target p. Instead, using Jensen's inequality a

lower bound can be derived from (3.1):

$$\mathcal{L} := \mathbb{E}_{x\sim q}\left[\log \frac{p(x)}{q(x)}\right]. \tag{3.2}$$

In the literature, $\mathcal{L}$ is often referred to as the *evidence lower bound* or the *variational lower bound* [18]. There are multiple methods for optimizing the evidence lower bound (ELBO), however success heavily depends on the choice of the set $\mathcal{Q}$ and the target p.

Since we want to fit our set of samples q^* to our curriculum distribution p as described in our problem statement, we have less flexibility in our choice of $\mathcal{Q}$. Fortunately, the impact of choosing a specific set $\mathcal{Q}$ can be alleviated with a number of techniques [6]; one of which being *normalizing flow* [19, 20] as suggested by Rezende and Mohamed [21]. A normalizing flow is a series of (invertible) transformations applied to a probability density. As the name suggests, the final product after the density "flowed" through these mappings is a valid probability distribution, hopefully one potentially closer to the target distribution. Given an invertible transformation $f : \mathbb{R}^d \to \mathbb{R}^d$, we get the distribution of a random variable z with distribution $q(z)$ after applying the transformation $z' = f(z)$ as follows:

$$q(z') = q(z)\left|\det \frac{\partial f^{-1}}{\partial z'}\right| = q(z)\left|\frac{\partial f}{\partial z}\right|^{-1},$$

where the last equality follows from the chain rule.

One of the main challenge of normalizing flow is finding the right transformations. Although it is theoretically always possible to find a transformation for two distributions (under some reasonable assumptions, see [22, A 3.4.3]), a large set of transformation candidates might make the optimization of the ELBO infeasible in practical applications. We will investigate Stein variational gradient descent (SVGD) in section 5.1, which is able to create such transformations on a step-by-step basis, therefore iteratively generating a normalizing flow.

Since variational inference methods often rely on the KL divergence as a measure between the target and proposal distribution, it is difficult to apply them to situations where one of the distributions is only available as a set of samples. To overcome this problem, we will investigate the adapted variational inference methods known as particle-based variational inference methods, which use the kernelized Stein discrepancy instead of the KL divergence as a statistical discrepancy, which can be calculated between a set of samples and a distribution. To get a better understanding for this discrepancy, we will now look into its derivation from Stein's discrepancy in the next section.

3.3. Stein's Method and Stein's Discrepancy

In its original formulation by Stein, Stein's method was presented as a bound between the distribution of a sum of random variables and the normal distribution [23]. However, its usage as a computable sample quality measure and as a statistical discrepancy, was first established by Gorham and Mackey [24].

Consider a target distribution P with (continuously differentiable) density p over a measurable space $\mathcal{X} \in \mathbb{R}^n$. Assuming a set of samples $x_1, \ldots, x_i \in \mathcal{X}$, which induce the distribution Q, we want to calculate the quality of these samples according to our target P. An intuitive first step for this would be comparing the expectations

between these two distributions under some function $h : \mathcal{X} \to \mathbb{R}$. If we can calculate $\mathbb{E}_{x\sim P}[h(x)] = \int h(x)dP$ and $\mathbb{E}_{x\sim Q}[h(x)] = \sum_{i=1}^{n} q(x_i)h(x_i)$, we can define a discrepancy between Q and P as:

$$D_{\mathcal{H}}(Q, P) = \sup_{h\in\mathcal{H}} |\mathbb{E}_{x\sim Q}[h(x)] - \mathbb{E}_{x\sim P}[h(x)]| .$$

The intuition here is that we want to find the function which highlights the differences between the two distributions the most in terms of their expectation. Therefore, if our set of functions is sufficiently large and $D_{\mathcal{H}}(Q, P) = 0$, we can assume that $Q = P$. Depending on the choice of $\mathcal{H}$ some common discrepancies can be recovered, such as the *Wasserstein distance*, where

$$\mathcal{H} := \{h : \mathcal{X} \to \mathbb{R} | \ \|h\|_L \leq 1\}$$

is the set of 1-Lipschitz continuous functions.

This formulation is limited to cases where calculating the expectation over P is possible, i.e. integration under P is tractable. However, by limiting $\mathcal{H}$ such that $\mathbb{E}_{x\sim P}[h(x)] = 0$, the problem of intractable expectation calculation is side-stepped and $D_{\mathcal{H}}(Q, P)$ is only dependent on the expectation of Q. We can formalize this constraint by defining a functional operator $\mathcal{T}$, which for a set of functions $\mathcal{H}$ yields

$$\mathbb{E}_{x\sim P}[(\mathcal{T}h)(x)] = 0 \quad \text{for all} \quad h \in \mathcal{H}.$$

The operator $\mathcal{T}$ is called the *Stein Operator*, as it was first described by Stein [23]. Together with the set of functions $\mathcal{H}$ (which is also sometimes called the *Stein set*) it forms the *Stein discrepancy*

$$\mathbb{S}_{\mathcal{T},\mathcal{H}}(P, Q) := \sup_{h\in\mathcal{H}} |\mathbb{E}_{x\sim Q}[(\mathcal{T}h)(x)]| = \sup_{h\in\mathcal{H}} |\mathbb{E}_{x\sim Q}[(\mathcal{T}h)(x)] - \mathbb{E}_{x\sim P}[(\mathcal{T}h)(x)]| . \tag{3.3}$$

There exist infinitely many Stein operators which depend on the choice of distribution. To give a short example here, using *Stein's lemma*

$$\mathbb{E}_{x\sim P}[f(x)(x - \mu)] = \sigma^2 \, \mathbb{E}_{x\sim P}(f'(x)) \tag{3.4}$$

we can derive a Stein operator for the standard normal distribution $\mathcal{N}(0, 1)$ quite easily, since it follows from (3.4) that $\mathbb{E}_{x\sim P}[f'(x)-xf(x)] = 0$ and therefore we can set $\mathcal{T}$ such that $(\mathcal{T}f)(x) = f'(x)-xf(x)$. Other methods such as the one presented by Barbour [25] exist, which yield usable Stein operators. While these allow the practical usage of Stein's discrepancy as a quality measure as shown by Gorham and Mackey [24], the problem of evaluating the supremum in (3.3) over a necessarily large set of functions remains. However, through a particular choice of the function set, a closed form solution can be derived, which leads to the formulation of the *Kernelized Stein Discrepancy*.

3.4. Kernelized Stein Discrepancy

Building on the works of Gorham and Mackey, both Chwialkowski *et al.* and Liu *et al.* proposed an extension of the method using kernels to circumvent the problems described in the previous section.

Consider a reproducing kernel hilbert space (RKHS) $\mathcal{F}$ consisting of functions on $\mathbb{R}^d$ together with a reproducing kernel k and an inner product $\langle f, g\rangle_{\mathcal{F}^d} = \sum_{i=1}^{d}\langle f_i, g_i\rangle_{\mathcal{F}}$, where $\mathcal{F}^d$ is the product RKHS, with elements

$f := (f_1, \ldots, f_d);\ f_i \in \mathcal{F}$. Inspired by Gorham and Mackey, we can now define a Stein operator $\mathcal{T}$, which acts on the elements of the product RKHS as follows[2],

$$\mathcal{T} f := \sum_{i=1}^{d} \left(\frac{\partial \log p(x)}{\partial x_i} f_i(x) + \frac{\partial f_i(x)}{\partial x_i} \right).$$

This choice has the benefit that it only requires the calculation of $\nabla \log p(x) = \nabla p(x)/p(x)$ of the target density p, which does not require a tractable normalization constant. We further define the function

$$\xi_p(x, \cdot) := [\nabla \log p(x) k(x, \cdot) + \nabla k(x, \cdot)], \tag{3.5}$$

such that the inner product between f and ξ_p results in the expected value of the stein operator

$$\mathbb{E}_{x\sim P}[(\mathcal{T} f)(x)] = \langle f, \mathbb{E}_{x\sim P}[\xi_p(x, \cdot)] \rangle_{\mathcal{F}^d} = \sum_{i=1}^{d} \langle f_i, \mathbb{E}_{x\sim P,i}[\xi_p(x, \cdot)] \rangle_{\mathcal{F}}.$$

Under the given choice $\mathcal{T}$ and together with ξ_p we can now return to (3.3),

$$\begin{aligned} \mathbb{S}_{\mathcal{T}}(P, Q) &:= \sup_{\|f\|<1} \mathbb{E}_{x\sim Q}[(\mathcal{T} f)(x)] \\ &= \sup_{\|f\|<1} \langle f, \mathbb{E}_{x\sim P}[\xi_p(x, \cdot)] \rangle_{\mathcal{F}^d} \\ &= \|\mathbb{E}_{x\sim P}[\xi_p(x, \cdot)]\|_{\mathcal{F}^d}, \end{aligned}$$

where we limit our set of functions to the unit ball in the RKHS, since it allows us to circumvent the calculation of the supremum.

Finally, we can calculate a closed form solution for $\mathbb{S}_{\mathcal{T}}(P, Q)$. As shown by Chwialkowski *et al.*, $\xi_p(x, \cdot)$ is *Bochner integrable*[3], therefore the following inequality holds

$$\mathbb{E}_{x\sim P}[\|\xi_p(x, \cdot)\|_{\mathcal{F}^d}] \leq \mathbb{E}_{x\sim P}[\|\xi_p(x, \cdot)\|^2_{\mathcal{F}^d}]$$

Based on this observation, we can now calculate a closed-form solution for $\mathbb{S}_{\mathcal{T}}(P, Q)^2$:

$$\begin{aligned} \mathbb{S}_{\mathcal{T}}(P, Q)^2 &= \langle \mathbb{E}_{x\sim P}[\xi_p(x, \cdot)], \mathbb{E}_{x\sim Q}[\xi_p(x, \cdot)] \rangle \\ &= \mathbb{E}_{x\sim Q, y\sim Q}[\langle \xi_p(x, \cdot), \xi_p(y, \cdot) \rangle] \end{aligned}$$

[2] For a full proof by intergration of the validity of this operator choice, we defer to the works of Chwialkowski *et al.* [26] and Liu *et al.* [27].

[3] See Steinwart and Christmann: *Support Vector Machines*, Definition A.5.20

and since

$$\begin{aligned}\langle \xi_p(x,\cdot), \xi_p(y,\cdot)\rangle &= \nabla \log p(x)^\top \nabla \log p(x) k(x,y) \\ &+ \nabla \log p(x) \nabla_x k(x,y) \\ &+ \nabla \log p(x)^\top \nabla_y k(x,y) \\ &+ \langle \nabla_x k(x,\cdot), \nabla_y k(\cdot,y)\rangle \\ &:= k_0(x,y), \end{aligned} \tag{3.6}$$

we have the *Kernelized Stein Discrepancy*

$$\mathbb{S}_{\mathcal{T}}(P,Q) = \mathbb{E}_{x,y\sim Q}[\sqrt{k_0(x,y)}].$$

For the full proof that $\mathbb{S}_{\mathcal{T}}(P,Q)$ is a valid discrepancy between probability distributions, we refer to [26].

We can now derive an empirical version of this discrepancy by calculating the expectation over the samples, such that

$$\mathbb{S}_{\mathcal{T},p}(\{x_i\}_{i=1}^n) = \sqrt{\frac{1}{n^2}\sum_{i,j}^{n} k_0(x_i,x_j)}, \tag{3.7}$$

for which k_0 is defined as in (3.6). This allows us to use the kernelized Stein discrepancy (KSD) as a discrepancy between a distribution and a set of samples, which we will leverage in chapter 5 as the foundation of our investigated ParVIs.

Before we continue with our main contribution, we will look at similar work to ours in the upcoming related work chapter.

4. Related Work

Although still a relatively new field, particle-based variational inference methods have seen a strong uptick in publications in the past two years with many papers focusing on both new methods and applications. Similarly, many new and interesting methods and investigations into the application of curriculum learning in the space of RL have been published in recent years, indicating a strong interest in this field of research. Since our work draws from both of these research domains, we will investigate similar and related work in this chapter, starting with ParVIs in section 4.1 before moving on to the field of curriculum learning for RL in section 4.2. However, since our investigated problem of efficient adaptive sampling is, to the best of our knowledge, a novel contribution, we will focus on showing potential connections to other work.

4.1. Particle-Based Variational Inference Methods

Although SVGD is one of the more popular methods in the space of ParVIs, it is far from the only one developed in the recent years. For example, Wang *et al.* [29] derive a gradient update rule for particles based on the principles of energy dissipation in physical systems. Drawing from the insight that the KL-divergence can be viewed as the Helmholtz free energy, they present a gradient for the KL-divergence, which only slightly differs from the one presented by Liu and Wang for SVGD [30]. While the authors report a better performance compared to SVGD on their empirical benchmarks, the core algorithm still relies on the same gradient descent steps. Therefore, we expect the same problems we observed with our modification of SVGD to also occur with this method. For similar methods which use different derivations for the gradient updates such as Wasserstein variational gradient descent [31], DPVI [32] or Message Passing Stein Variational Gradient Descent [33] the same logic applies.

There have been some applications of SVGD in the space of reinforcement learning and control. Lambert and Boots use SVGD to sample trajectories as part of Gaussian process motion planing [34]. Their presented inference procedure called Stein variational motion planning allows for a smooth optimization of the proposal distribution compared to other methods such as Markov Chain Monte Carlo, which simplifies the process of trajectory optimization. The paper shows promising results, albeit the authors defer a detailed comparison of their algorithm to similar approaches to potential future work. A more thorough review is done by Barcelos *et al.* in their presentation of their usage of Stein variational inference for model predictive control [35]. Here, SVGD is used to perform sampling on the joint distribution over actions and dynamics model parameters in an online fashion, which not only results in a continuing refinement of the agents actions but also makes recovery from sudden changes in the environment possible. While the investigated domain of this work is quite similar to the one in our work, the general approach differs from the methods presented in this thesis. More specifically, the authors do not selectively move certain particles as it is required for our application. This is also true for other work in the field of RL, such as Stein variational policy gradient [36] or deep energy-based policies [37].

4.2. Curriculum Learning for Reinforcement Learning

We will now look at different curriculum learning methods in reinforcement learning to investigate their approaches to reduce sample inefficiency.

One method of generating a curriculum in reinforcement learning is performing the training in “reverse”, as proposed by Florensa *et al.* [15]. That is, the agent is initially placed close to the goal state and the curriculum consists of further and further removed start states over the course of training until the intended initial state is reached. On an algorithm level these new states are generated from previous iterations by predicting the attained reward and removing all start sample above or below a certain treshold, therefore ensuring the right amount of challenge for the agent. Using sampled actions, these states are then updated by performing small rollouts, which generate the next set of start states. An “forward” version of this method was later provided by the authors using generative adverserial networks (GAN)s to generate goal states as part of a min-max game. In both cases sample efficiency is ensured using the aforementioned pruning method, where samples with a high predicted reward are removed from the buffer. However, this requires a good approximation of the expected return for a given starting state, which might not always be available.

Narvekar *et al.* propose to create curricula by casting the generation process as a meta MDP, which they call curriculum MDP [38]. The states of the MDP represent different policies, with the action space consisting of the tasks the agent can train on next. The training process updates the policy, leading to a transition in the state space of the curriculum MDP. This process is repeated until the final policy “state” is reached, which is defined as the policy being able to solve the target task as fast as possible while still reaching some performance threshold; this metric is also used as the reward function of the curriculum MDP. However, in its original formulation this algorithm requires that the policy is trained on every potential task, since the authors propose policy change in addition to attained reward as metrics to determine whether a certain task should be added to the curriculum. While this process is potentially feasible in the simple grid worlds investigated by the authors, it can not be used in continuous domains. Furthermore, while the resulting curriculum only contains the minimum amount of task samples required to learn the task efficiently, the goal of this thesis is to reduce the amount of unnecessary evaluations, which should also ideally lead to a minimal curriculum but with less required rollouts.

5. Adaptive Sampling Using Stein's Method

In this chapter two sampling methods are presented, which use the KSD to draw samples from a proposal set such that they fit a target distribution. In both cases, the approaches utilise the KSD, however in two different ways: While the first method, Stein variational gradient descent, is a gradient-based approach iteratively moving the particles towards the target distribution according to the KSD, the second one, Stein points, directly selects samples from a proposal distribution according to the KSD between them and the target distribution. We also present the required changes and adaptions necessary to use both methods for our problem of effective curriculum sampling.

5.1. Stein Variational Gradient Descent

In their paper "Stein Variational Gradient Descent" [30], Liu and Wang draw a connection between the KSD and the derivative of the KL-divergence, which allows them to derive a gradient estimator for usage in variational inference.

As in section 3.2, let p be the density of a target distribution with $\mathcal{Q}$ being a set of of proposal distributions generated by applying a transformation $z = T(x)$ to a reference distribution $q_0(x)$. As per the change of variables formula (see section 3.2 for more details), the resulting density $q_{[T]}(z)$ is

$$q_{[T]}(z) = q(T^{-1}(z)) \left| \det \frac{\partial T^{-1}}{\partial z} \right|. \tag{5.1}$$

As discussed in section 3.2, selecting a good set of transformations $\{T\}$ is not trivial. However, as shown by Liu and Wang [30], it is possible to side-step this problem by defining T as a identity map with a small perturbation $T(x) = x + \epsilon f(x)$, where $f(x)$ is a smooth function used as the perturbation direction and ϵ is a scalar functioning as the perturbation magnitude. It follows from change of variable that

$$D_{KL}(q_{[T]} \| p) = D_{KL}(q \| p_{[T^{-1}]}),$$

which now allows us to take the derivative of the KL-divergence with respect to the perturbation factor ϵ using (3.2) and (5.1):

$$\nabla_\epsilon D_{KL}(q_{[T]} \| p) = \nabla_\epsilon D_{KL}(q \| p_{[T^{-1}]}) = -\mathbb{E}_{x \sim q}[\nabla_\epsilon \log p_{[T^{-1}]}(x)].$$

We can now calculate $\log p_{[T^{-1}]}(x)$ using the same schema as in (5.1), yielding

$$\nabla_\epsilon \log p_{[T^{-1}]}(x) = \nabla_x \log p(T(x))^\top \nabla_\epsilon T(x) + \text{trace}(\nabla_x T(x)^{-1} \cdot \nabla_\epsilon \nabla_x T(x)).$$

An interesting connection to the Stein operator emerges for the particular choice of $\epsilon = 0$, for which follows

$$\nabla_\epsilon \log p_{[T^{-1}]}(x)|_{\epsilon=0} = -\mathbb{E}_{x\sim q}[\text{trace}(\mathcal{T} f(x))], \tag{5.2}$$

where the Stein operator is defined as $(\mathcal{T} f(x)) = \nabla_x \log p(x) f(x)^\top + \nabla_x f(x)$. We already saw the benefits for this particular choice for the Stein operator in section 3.4 (see equation (3.5)), which makes the following connection to the KSD quite intuitive: If we choose $f \in \mathcal{F}^d$, where $\mathcal{F}^d$ is a reproducing kernel hilbert space following the definition in section 3.4, we know that the function f^* which maximizes (5.2) is

$$f^*_{q,p}(\cdot) = \mathbb{E}_{x\sim q}[\nabla_x \log p(x) k(x,\cdot) + \nabla_x k(x,\cdot)], \tag{5.3}$$

from which follows that

$$\nabla_\epsilon \log p_{[T^{-1}]}(x)|_{\epsilon=0} = -\mathbb{S}_{\mathcal{T}}(P,Q)^2.$$

Given the above insight, it is now possible to iteratively construct a one-step normalizing flow between a starting distribution q_0 and a target distribution p, where at every step ℓ we can calculate $q_{\ell+1}$ via

$$q_{\ell+1} = q_{l[T^*_\ell]}, \quad \text{where} \quad T^*_\ell(x) = x + \epsilon_\ell f^*_{q_l,p}(x).$$

Given enough iterations and a sufficiently small step-size ϵ_l this will converge to the target distribution p, such that $f^*_{q_\infty,p} = 0$, resulting in T^*_∞ being the identity map. Note that $f^*_{q_\infty,p} = 0$ if and only if $q_\infty = p$.

If we want to apply this procedure in practice, it would be necessary to calculate (or at least approximate) the expectation in (5.3) for all intermediate distributions q_ℓ. This can be circumvented, however, by sampling from the initial distribution q_0 and subsequently apply the iterative procedure on the samples as outlined in algorithm 1. In this case, the expectation of the empirical distribution of the particles[1] can be calculated using the empirical mean.

Algorithm 1: Iterative Stein Variational Gradient Descend

Input: Target distribution $p(x)$, an initial set of particles $\{x^0_i\}^n_{i=1}$ and an amount of iterations L
Output: A set of particles $\{x_i\}^n_{i=1}$, which approximates $p(x)$
for $\ell \leftarrow 1$ *to* L **do**
  for $i \leftarrow 1$ *to* n **do**
    $x^{\ell+1}_i \leftarrow x^\ell_i + \epsilon_\ell f^*(x^\ell_i) \quad \text{where} \quad f^*(x^\ell_j) = \frac{1}{n}\sum_{j=1}^n [k(x^\ell_j, x)\nabla_x x^\ell_j \log p(x^\ell_j) + \nabla_{x^\ell_j} k(x^\ell_j, x)]$
  end
end

A closer look at the dynamics of the gradient update $f^*(x)$ in algorithm 1 reveals some intuitive insights into the inner workings of SVGD. The first term, $k(x^\ell_j, x)\nabla_x x^\ell_j \log p(x^\ell_j)$, acts as a kernel-weighted *driving force*, which moves the particles towards the high-density areas of $p(x)$, while the second term, $\nabla_{x^\ell_j} k(x^\ell_j, x)$, acts as a *repulsive force*, which drives particles away from each other and avoiding the collapse of all particles into the modes of p.

[1] We refer to these samples from the proposal distribution as particles to avoid confusion later when describing the RL experiments.

5.1.1. Bandwidth Selection

The final step to apply SVGD in practice is the choice of an appropriate kernel, which might also require a selection of kernel parameters. For example, the well-known radial basis function kernel $k(x, y) = \exp\{-\frac{1}{h}\|x - y\|^2\}$ has a bandwidth parameter h. While the choice of kernel is important, it has been shown that for a variety of domains the radial basis function (RBF) kernel performs quite well [24, 30, 39]. This leaves the choice of the bandwidth, which not only can be difficult to tune depending on the task but is often crucial to the performance of the algorithm (see [40] for an example in the domain of support vector machines).

In the case of SVGD, Liu and Wang [30] choose the RBF kernel for all their experiments together with a heuristic to estimate the bandwidth h. Based on the intuition that the gradient contribution of every particle x_i should be balanced with the contributions from all other particles, i.e. by using the median such as $\sum_j k(x_i, x_j) \approx n \exp\{-\frac{1}{h}\text{med}^2\} = 1$, they select the bandwidth at every step as $h = \text{med}^2 / \log n$ [30]. Note that with this choice the bandwidth is not fixed but rather changes between iterations.

While this choice of bandwidth selection has been adopted with minor changes by a wide range of publications in the space of SVGD [36, 41, 42], it might not be the best option given all circumstances [7, 43]. An alternative heuristic was suggested by Liu *et al.*, which they derived as part of their investigations into the gradient flow nature of particle-based variational inference methods. They point out that SVGD follows the same gradient flow as Langevin dynamics, which can be used to sample from unnormalized density functions [44], since those describe the gradient flow of the KL divergence. Because these dynamics arise in physics in the form of Brownian motion, which is describes the movement of heated particles, they use this equivalency to derive a new method for selecting the bandwidth h:

$$\begin{aligned} &\arg\min_h \quad \frac{1}{h^{D+2}} \sum_k \lambda(x_{(k)})^2 \\ &\text{where } \lambda(x) := \Delta\tilde{q}(\{x_i\}_{i=1}^n) + \sum_j \nabla_{x_j} \tilde{q}(\{x_i\}_{i=1}^n) \cdot \nabla \log \tilde{q}(\{x_i\}_{i=1}^n) \\ &\text{and } \tilde{q}(x) := \frac{1}{n} \sum_j k(x_i, x_j), \end{aligned} \tag{5.4}$$

where $\Delta\tilde{q}$ denotes the second derivative of $\tilde{q}$. As with the median-based method, the bandwidth is not fixed but might change in every iteration.

A visual comparison between the two methods can be found in Figure 5.1. Note how the change in bandwidth selection affects the placement of the particles. With the HE method, samples are placed almost exclusively in the high probability regions of the distribution and are also distributed much more evenly. Compare this to the median method, which generated samples are much more clustered together with more samples outside the high density regions of the distributions.

In the rest of this thesis, we will refer to the first method as the *median method* and the second one as the *heat equation* or *HE method* for short.

5.2. Adapting SVGD for Partial Variational Inference

Our first approach is based on SVGD, which is one of the first particle-based variational inference methods. As presented in section 5.1, SVGD makes it possible to calculate the steepest gradient direction for a set of

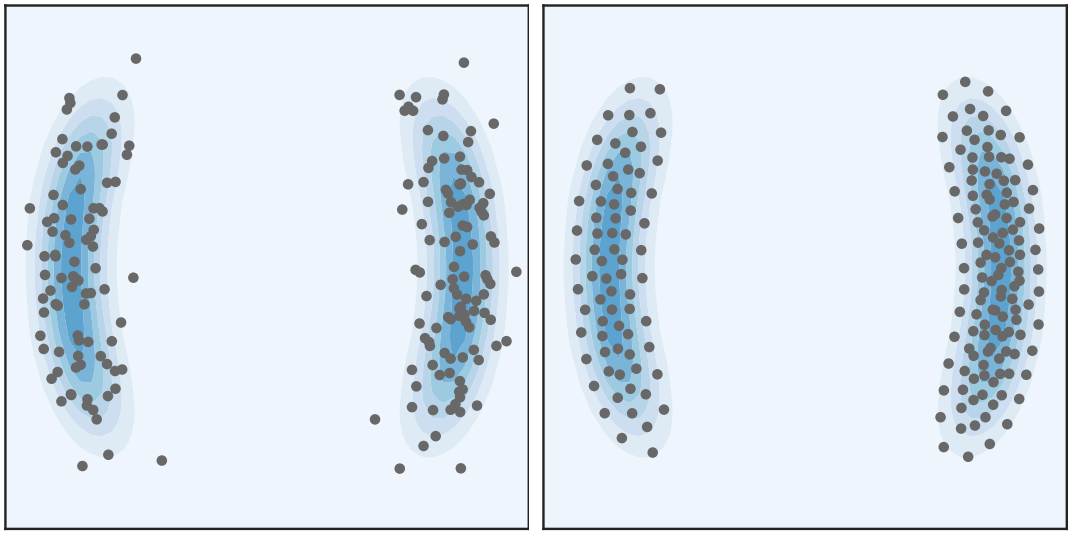


Figure 5.1.: Comparison between the median method (left) and the HE method (right) for selecting the bandwidth in SVGD. Taken from [7].

particles to better fit a given target distribution. We can apply this to the task of efficiently sampling from the curriculum distribution as follows:

1. Take the task samples $\{x_i\}_{i=1}^n$ from the experience buffer.
2. Add new proposal task samples $\{y_i\}_{i=1}^m$.
3. Perform SVGD on the union of these two sets using the CL distribution as the target.

The key difference to the original formulation [30] can be found in the third step: Here, we only perform gradient descend on the set of samples $\{y_i\}_{i=1}^m$ generated in step two but keep the particles from the experience buffer $\{x_i\}_{i=1}^n$ fixed; however, the gradient is still calculated using the full set of particles (see algorithm 2 for the modified algorithm). Intuitively, while this approach should lead to a less effective coverage of the target distribution (since the fixed samples will stay in non-optimal positions), it nevertheless reduces the KL-divergence between the particle and target distribution by moving the proposal particles towards less covered areas of the target distribution. We will evaluate this hypothesis in section 6.3.

Since we have access to the target distribution p, we can sample our proposal samples $\{y_i\}_{i=1}^m$ directly from it. Note that while in this case $\{y_i\}_{i=1}^m$ would already match p to some extent, the union of proposal and experience buffer samples $\{z_i\}^{n+m}_{i=1} = \{x_i\}_{i=1}^n \cup \{y_i\}_{i=1}^m$ would not due to the overlap. This allows us to still use SVGD and result in non-zero gradient values for the samples in $\{z_i\}_{i=1}^{n+m}$ which are also in $\{y_i\}_{i=1}^m$.

5.3. Stein Points

While SVGD uses a gradient-based approach to minimize the KL-divergence between two distribution using the KSD as an intermediate stepping stone, it is also possible to directly optimize the KSD between a set of samples and a distribution through selection using a method called *Stein points* [45].

Algorithm 2: Partial Iterative Stein Variational Gradient Descent

Input: Target distribution $p(x)$, set of fixed particles $\{x_i\}_{i=1}^n$, set of proposal particles $\{y_i\}_{i=1}^m$ and amount of iterations L

Output: A set of particles $\{z_i\}^{n+m}{}_{i=1}$, which approximates $p(x)$

$\{z_i\}^{n+m}{}_{i=1} = \{x_i\}_{i=1}^n \cup \{y_i\}_{i=1}^m$, **for** $\ell \leftarrow 1$ *to* L **do**

  for $i \leftarrow 1$ *to* $n+m$ **do**

    if $z_i \in \{y_i\}_{i=1}^m$ **then**

      $z_i^{\ell+1} \leftarrow z_i^\ell + \epsilon_\ell f^*(z_i^\ell) \quad$ where $\quad f^*(z_j^\ell) = \frac{1}{n}\sum_{j=1}^{n+m}[k(z_j^\ell, z)\nabla_z z_j^\ell \log p(z_j^\ell) + \nabla_{z_j^\ell} k(z_j^\ell, z)]$

    else

      Continue

    end

  end

end

We already presented an empirical formulation for the KSD in (3.7). We can use this formulation to sample from a distribution p by leveraging the weak convergence of the KSD; that is that $\mathbb{S}_{\mathcal{T},p}(\{x_i\}_{i=1}^n) \approx 0$ implies that the set $\{x_i\}_{i=1}^n$ has the same distribution as p. Therefore, given a set of proposal samples $\{x_i\}_{i=1}^m$ we can "sample" from p by selecting a subset $\{x_i\}_{i=1}^n$ for which the KSD is minimal. Unfortunately, it is usually not possible to perform this optimization directly: Such problems are known in the literature as mixed-integer quadratically constrained programming problem (MIQCP) and are known to be NP-hard [46].[2] However, Chen *et al.* show that a full optimization is not necessary: Instead by proving that first $\mathbb{S}_{\mathcal{T},p}(\{x_i\}_{i=1}^n) \to 0$ for $n \to \infty$ and second that $\mathbb{S}_{\mathcal{T},p}(\{x_i\}_{i=1}^n) \to 0$ implies that the empirical distribution of the samples converges to p.

Using these results we can now formulate an algorithm that iteratively optimizes the KSD. We select the first sample x_1 as the global maximum of p. At every iteration the sample x_n from the proposal set X is added, for which $\mathbb{S}_{\mathcal{T},p}(\{x_i\}_{i=1}^n)$ is minimal; the already selected samples $\{x_i\}_{i=1}^{n-1}$ are kept fixed. Therefore, at iteration n, x_n is selected as follows

$$x_n = \arg\min_{x \in X} \frac{k_0(x,x)}{2} + \sum_{i=1}^{n-1} k_0(x_i, x),$$

where the first part is the KSD of the set with single element x and the second part is a reduced calculation of the KSD, which only calculates the change after adding x.[3]

This algorithm is referred to by the original authors as *Stein Greedy-n*, due to its nature of selecting the n points with the lowest KSD. An alternative variant can be derived from the concept of *kernel herding* proposed by Chen *et al.* [47]. In this variant, at iteration n, x_n is selected as follows

$$x_n = \arg\min_{x \in X} \sum_{i=1}^{n-1} k_0(x_i, x),$$

which the authors call *Stein Herding-n*. Comparing it to the previous algorithm it can be seen that Stein Greedy-n is a regularized variant of Stein Herding-n with regularizer $0.5\ k_0(x,x)$. While both versions produce similar results, Stein Herding-n generally produces samples with a slightly higher variance compared

[2] We show this objective and a reduced version in subsection 5.4.1.

[3] Since k_0 is strictly positive, we can omit the square root.

to the greedy variant, which generally shows less outlier samples in its output. We will see in our RL experiments in chapter 7 that additional variance in the samples proves to be beneficial to the agents performance. For this reason, we choose the herding version as our used method for the rest of this thesis. The general Stein points procedure for both greedy and non-greedy variants can be found in algorithm 3.

It is possible to further refine the results of the algorithm by repeated application, which the authors refer to as block coordinate descent. Here, after generating an initial set of points $\{x_i\}_{i=1}^n$ using the Stein points algorithm is further adapted by iteratively calculating

$$\text{for } j = 1, \ldots, n \quad x_j = \arg\min_{x \in X} \mathbb{S}_{\mathcal{T},p}(\{x_i\}_{i=1, i \neq j}^n \cup \{x\}).$$

Algorithm 3: Stein Points

Input: Target distribution $p(x)$, proposal set of samples X, and target amount of samples N
Output: A set of particles $\{x_i\}_{i=1}^n$, approximating $p(x)$
$x_1 = \arg\max_{x \in X} p(x)$ **for** $n \leftarrow 2$ *to* N **do**
    if *greedy* **then**
        $x_n = \arg\min_{x \in X} \frac{k_0(x,x)}{2} + \sum_{i=1}^{n-1} k_0(x_i, x)$
    else
        $x_n = \arg\min_{x \in X} \sum_{i=1}^{n-1} k_0(x_i, x)$
    end
    $\{x_i\}_{i=1}^n = \{x_i\}_{i=1}^{n-1} \cup \{x_n\}$
end

5.4. Sampling From the Target Distribution Using Stein Points

The second approach investigated as part of this thesis is based on Stein points [45]. Compared to SVGD no major changes are required to the inner workings of the algorithm itself; however, some minor differences do exist due to the specific requirements of CL.

As described in section 5.3, the Stein points algorithm iteratively builds a set of samples $\{x_i\}_{i=1}^n$ by selecting the sample from a proposal set $x_n \in X$ which minimizes the KSD between the target distribution and the set of samples previously selected $\{x_i\}_{i=1}^{n-1} \cup x_n$. While in the original publication [45] the initial sample is selected as the maximum of the target distribution p, in this thesis we use the task samples from the experience buffer as the initial selection of samples. Later iterations then proceed according to the original formulation as described in algorithm 3 until the total amount of samples reaches the desired threshold after which the RL policy is solely evaluated on the newly added samples.

As it is the case with SVGD, we can again use the target distribution p as the source of the proposal samples X. However, we discuss the choice of proposal distribution in section 7.1 and show that it can be beneficial to use a slightly modified distribution $\hat{p}$ as opposed to p.

5.4.1. Pruning the Experience Buffer With Stein Points

While in most applications including the initial presentation Stein points are used to add samples, we propose to use them as a method to *remove* samples as well. This will allow us to prune the experience buffer more effectively according to the current curriculum distribution, removing less relevant samples instead of just a percentage of the oldest ones.

For this approach, we calculate the union of the proposal sample and experience buffer as done in SVGD. We then proceed according to algorithm 3 until we have enough samples to fully fill the buffer. Since samples where chosen from the union of proposal and previous task samples, the resulting set will consist of a mix between these two sets[4]. However, due to the iterative nature of the algorithm, it is difficult to enforce some balance between the selection of new samples and the retainment of old ones.

To circumvent this problem, it would be beneficial to view the process of Stein points as a global optimization problem instead of an iterative one. While we have seen in section 5.3 that the global optimization is NP-hard (due to it being a MIQCP), it might be possible in this case to use a heuristic to simplify the optimization.

The objective of the global optimization of Stein points can be written as

$$\min x^T C x, \tag{5.5}$$

where $x \in \{0,1\}^n$ is the binary selection vector and $C \in \mathbb{R}^{n \times n}_{>0}$ is the symmetric matrix of all KSD values with $C_{ij} = k_0(x_i, x_j)$. This can be rewritten as

$$x^T C x = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i C_{ij} x_j = \sum_{i,j \in A(x)} C_{ij},$$

where $A(x) = \{k | x_k = 1, k \in [0, n]\}$ is the set of "active" indices. This sum can be factorized

$$\sum_{i,j \in A(x)} C_{ij} = \sum_{i \in A(x)} C_{ii} + \sum_{i,j \in A(x), i \neq j} C_{ij}.$$

Since $C_{ij} > 0$, it follows that

$$\sum_{i,j \in A(x), i \neq j} C_{ij} \leq \sum_{i \in A(x)} C_{ii} \sum_{i,j \in A(x), i \neq j}^{n} C_{ij}.$$

We therefore conclude that

$$\begin{aligned} \sum_{j=1}^{n} (x^T C)_j = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i C_{ij} &= \sum_{i \in A(x)} C_{ii} + \sum_{i \in A(x)} C_{ii} \sum_{i,j \in A(x), i \neq j}^{n} C_{ij} \\ &\geq \sum_{i \in A(x)} C_{ii} + \sum_{i,j \in A(x), i \neq j} C_{ij} = x^T C x \end{aligned}$$

Based on this derivation, we propose to use the reduced objective instead

$$\min \sum x^T C. \tag{5.6}$$

[4] Given that there is some overlap between the distribution of the task samples from the experience buffer and the current target distribution.

The optimization problem with (5.6) is linear (in its objective), which modern optimizers can solve comparatively fast using heuristics. Since deriving an upper error bound for this problem is out of the scope of this thesis, we will content ourselves with an empirical analysis of this objective (for our specific use case) in section 6.3. We expect that the impact of our reduced objective changes according to the ratio between unselected to selected samples, with higher amount of proposal samples leading to less accurate estimations of the optimal sample set. This is caused by the higher chance of outlier selection due to the noise added by the additional sum terms from the unselected samples.

Using (5.6) it is now possible to solve the global Stein points optimization problem approximately using modern optimizers such as Gurobi. This makes the application of Stein points (SP) more flexible, since it allows for the usage of different constraints on the optimization process. For example, if we now add the constraint $\sum_i^n x_i = r$, assuming that the first n entries of x represent the values from the experience buffer, we can now define some constant r, which stands for the amount of samples from the experience buffer we want to retain.

5.4.2. Selecting a Proposal Distribution

So far we have not talked about the source of proposal samples for our SP samplers. While the original presentation [45] evaluated multiple sources, such as a grid over the whole space or a continuously optimized proposal distribution using methods such as Monte Carlo sampling or similar numerical methods, we do have access to the target distribution in curriculum learning. However, directly using samples from this distribution can introduce a bias which is amplified by the usage of SP samplers, since selected samples will tend to be close to the modes of the curriculum distribution. While this bias vanishes in the limit as more proposal samples are used, we investigate a method to combat this problem by artificially increasing the variance of the curriculum distribution.

The distribution is stretched by calculating the eigendecomposition of the covariance matrix

$$\Sigma = Q\Lambda Q^{-1},$$

where Q is a square matrix with columns corresponding to the eigenvectors of Σ and Λ is a diagonal matrix, where the diagonal elements are the eigenvalues associated with the eigenvectors in Q. We then multiply Λ with the stretch factor, creating a new target distribution Σ^* which is subsequently used for the sample process. This increases the lower sample variance encountered when using Stein points based samplers. The stretching parameter is a hyper parameter of our samplers in the RL experiments, as shown in section 7.1.

We now investigate several pedagogical examples to highlight the behavior and shortcomings of the three presented methods.

6. Empirical Comparison of Methods

In this chapter we investigate our proposed modifications of self-paced reinforcement learning (SPRL) and SP on a set of specifically crafted examples with the goal of highlighting the advantages and problems introduced by our changes. Additionally, we perform an empirical evaluation of our proposed reduced optimization objective for the Stein points sampler.

6.1. Sample Movement With Fixed Samples

We test the impact of the changes outlined in section 5.2 on a set of pedagogical examples. Specifically, we want to estimate the impact of the fixed particles on the movement of the movable particles for both SVGD and SP based approaches.

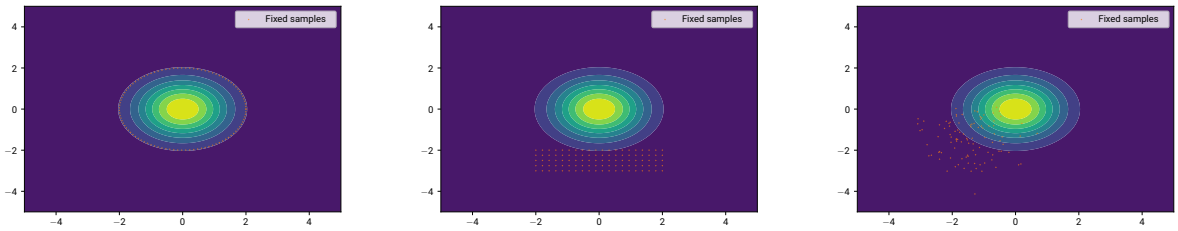
For this we use a simple bivariate normal distribution $\mathcal{N}(0, \mathbf{I})$ as the target distribution, where 0 is the zero vector and $\mathbf{I}$ is the identity matrix. We surround the distribution with three different patterns of samples (depicted in Figure 6.1), which will be set as fixed during the run of SVGD. While these patterns are highly unlikely to occur during a normal curriculum learning run (except for the random distribution samples in Figure 6.1c), they nevertheless highlight the behavior with the proposed modifications.

For SVGD, to better visualize the impact of the fixed samples, we consider a grid over the space from which we take the proposal samples one-by-one and execute SVGD on. This allows us to compare how different initial sample positions are affected by the fixed samples and how they end up relative to each other after running SVGD. We perform the SVGD runs using the RBF kernel, where the bandwidth is set either by the HE or the median method presented in subsection 5.1.1. Compared to the original publication [30], we use Adam [48] for the step size instead of AdaGrad due to its wide spread adoption in other SVGD implementations.

To compare the Stein points approach to the one based on SVGD, we use a similar technique to visualize the inner workings of the algorithm. However, since Stein points do not rely on gradient descend to generate new samples, we will instead calculate the KSD for a grid of samples. This is equivalent to a single step in the Stein points algorithm and visualizes where new samples would be placed.

Additionally, we will show how different kernels affect the selection of new samples. The additional kernels (besides the RBF kernel) are:

1. **Inverse Log Kernel**: $k(x, y) = (\alpha + \log(1 + \|x - y\|_2^2))^{-1}$
2. **Mahalanobis Kernel**: $k(x, y) = \exp\left(-\frac{(x-y)^\top \Sigma^{-1}(x-y)}{h}\right)$,



(a) Circular fixed sample placement (b) Rectangular fixed sample placement (c) Random distribution fixed samples

Figure 6.1.: Differently arranged sample patterns used in the SVGD experiments. While the circular and the rectangular placements are clearly artificial, the random distribution samples could stem from a previous curriculum distribution.

where the parameters α, h are kernel parameters. In the case of the Mahalanobis kernel, we use the covariance of the target distribution as Σ. Finally, all experiments were performed using the iterative formulation of SP, since it allows for a better insight into the one-step execution of the algorithm. To show the differences between the RBF and the Mahalanobis kernel, we use a normal distribution with a non-diagonal covariance and where the dimension differ in their order of magnitude. For $\mathcal{N}(0, \Sigma)$, with 0 being the zero vector and $\Sigma = \left(\begin{smallmatrix} 2 & 0.8 \\ 0.8 & 0.5 \end{smallmatrix}\right)$ we present our results in the upcoming sections.

6.1.1. Results

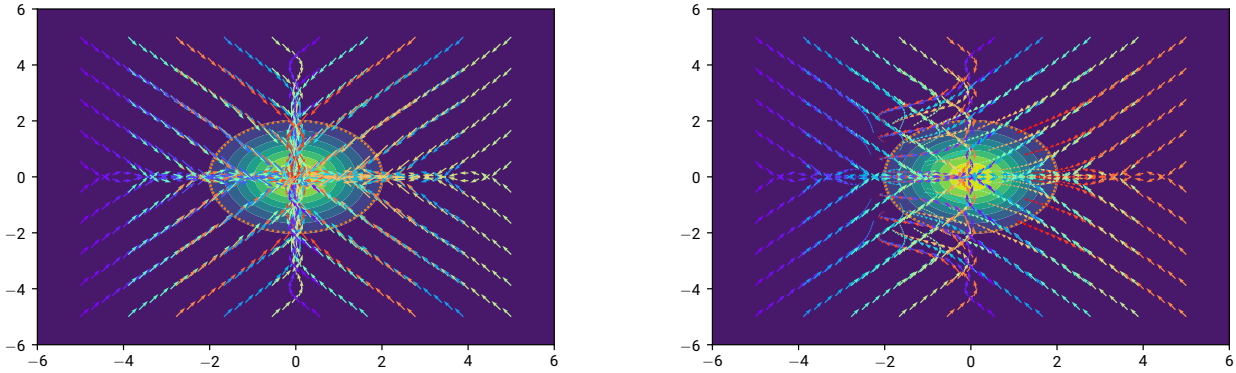
We start with SVGD before moving on to Stein points. To better differentiate between the fixed and movable samples, we will refer to the first ones as samples and to the second ones as particles.

SVGD

First, we look at the circular arranged fixed samples (Figure 6.1a) to investigate, whether the fixed samples prevent the movable samples to reach the high-density area of the distribution. If the median method is employed, we can see in Figure 6.2a that all samples converge to the mode of the distribution. Conversely, if the HE method is employed as its done in Figure 6.2b, the final positions are more distributed over the high density regions of the distribution. However, the final position of any particle is heavily dependent on the starting position, such that distant particles rarely manage to enter the high density regions.

Moving on to the rectangularly distributed samples as shown in Figure 6.1b, we can observe stronger rejection behavior when the fixed samples are more clustered together. While the median method shows only a moderate impact (Figure 6.3a) on the final positions, we can see a strong rejection force from the clustered samples when using the HE method in Figure 6.3a. Even particles which are relatively far away from the sample cluster still get affected and move further away from the distribution.

Finally, we take a look at the random distribution samples. While the previous examples served as extreme cases to highlight the shortcomings of either bandwidth estimation method and are highly unlikely to occur in a real world setting, we chose this example to be closer to our targeted application of curriculum learning.



(a) Median method

(b) HE method

Figure 6.2.: Particle movement after the running SVGD on hundred samples arranged around a bivariate normal distribution.

Here, the samples stem from a distribution with different variance and mean compared to the target distribution, with the latter potential being the product of a CL algorithm. After the application of SVGD, we can see in Figure 6.4a that the influence of the fixed samples on the particles is quite pronounced when using the median method compared to the previous two examples. However, most particles are still collapsed into the same small area. With the HE method we observe a similar albeit shifted pattern in particle movement in Figure 6.4b compared to the rectangular pattern from before.

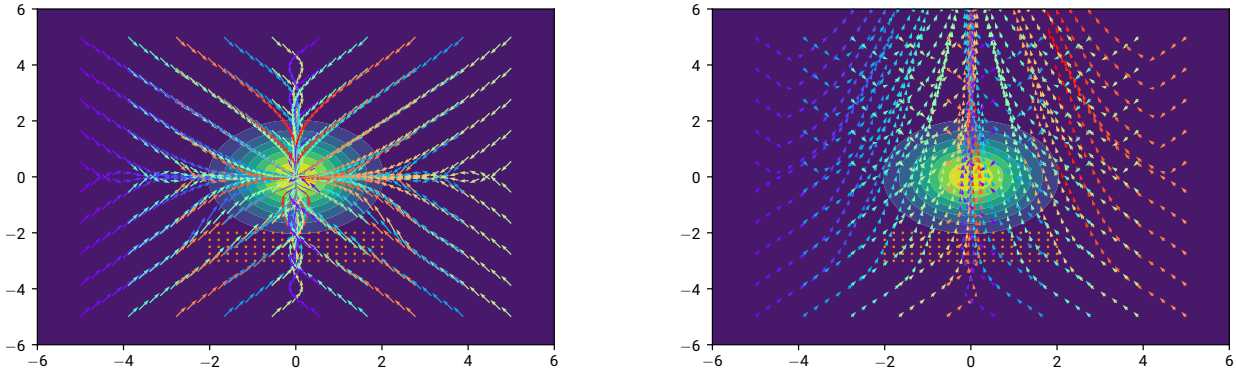
Comparing the different outcomes of the SVGD experiments a general pattern can easily be identified: While the usage of the median method tends to lead to clustered end positions for the particles not strongly affected by the fixed samples, the HE method seems to lead to a the opposite result, with the fixed samples having a strong impact on the final positions of the particles. In terms of the output, the median method seems to propose lower bandwidth values, while the HE method suggests very high values for the kernel hyper parameter.

Although its magnitude is much smaller, this behavior can already be observed in the vanilla comparison of the two methods shown in Figure 5.1. Here, the particles are also much more evenly spread out when using the HE method compared to the median method, leading to a more uniformly looking final distribution coverage.

Discussion of Gradient Descent Dynamic

The reason for the stronger response in our application of SVGD can be traced back to how the modification of having fixed samples affects the general calculation of the gradient update. As described in section 5.1, the equation to calculate the gradient (see algorithm 1) can be split in two parts, the driving and the repulsive term. If we look at the repulsive term, we can see that it is symmetric such that for every particle pair x, y x is pushed in the opposite direction from y but by the same distance; that is $\nabla_x k(x, y) = -\nabla_y k(y, x)$. Therefore, by keeping some particles fixed, the repulsive force is significantly reduced. Although this explains the behavior of SVGD when using the median method (which already creates less repulsive force between the particles due to its lower bandwidth estimate), it does not explain the behavior of the particles when the HE method is employed.

For this algorithm, the reason for the diverging behavior can be found in (5.4). Here, the first term $\Delta\tilde{q}(\{x_i\}_{i=1}^n)$



(a) Median method

(b) HE method

Figure 6.3.: Particle movement after the running SVGD on hundred samples arranged in a rectangle next to a bivariate normal distribution.

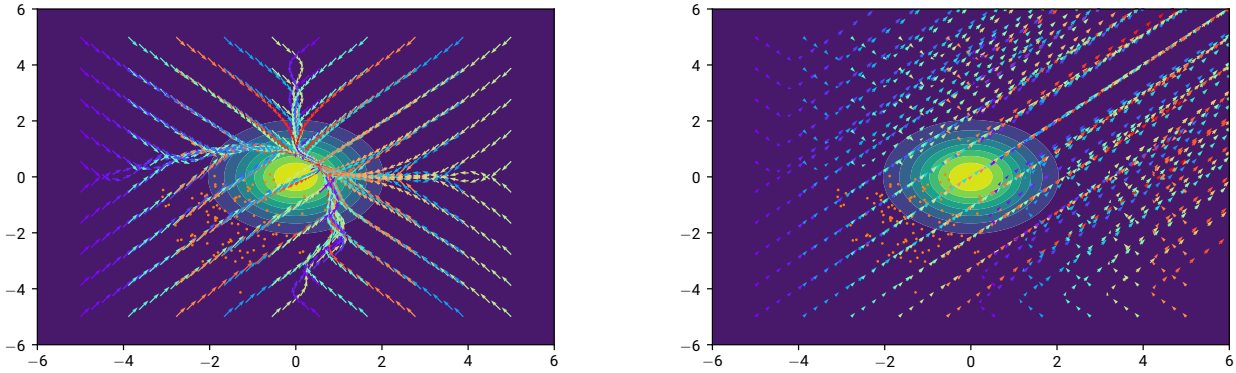
is sensitive to the variances in the distance between particles (compared to the more temperate second part of the equation). We can see this in the visualizations, where particles starting closer to the fixed samples end up at more sensible locations compared to particles starting farther away. This is in line with the assumptions being made by the authors of the HE method [7], who assume that the particles in the approximating distribution are reasonably close.

In summary, both investigated methods show shortcomings when being applied to the modified version of SVGD. Specifically, the final result seems to heavily correspond to the initial movable particle positions in relation to the fixed ones. Determining the right initial particle positions is therefore crucial, albeit quite difficult as can be estimated when looking at the viable starting points visible in our experiments. Further complexity is introduced when not one (as done in the experiments) but multiple movable particles should be fitted to a target distribution, since this would require that intra-particle effects have to be accounted for when setting their initial position. While it might be possible to side-step some of these problems by introducing a particle-dependent step size, it lies outside the scope of this thesis to formulate the necessary fundamental changes to the theory. Instead, we investigate a gradient-free approach in the form of Stein points and apply it to an RL task.

Stein Points

We now present the results of applying the Stein points based sampling on the pedagogical examples shown in Figure 6.1 to compare their performance to SVGD. Note that we do not calculate the true KSD but rely on the same approximation used in the iterative Stein point sampler. This is due to the lower variance between two different points when the true KSD is calculated, which can be attributed to the square root and averaging terms in the formulation. However, as discussed in section 5.3, this does not affect the final result of the optimization and is only employed here to make the visualizations more distinctive.

Starting with the circular arrangement of samples (as shown in Figure 6.1a), we visualize the results in Figure 6.5. Note that the kernel choice is only marginally important in terms of sample placement. We



(a) Median method

(b) HE method

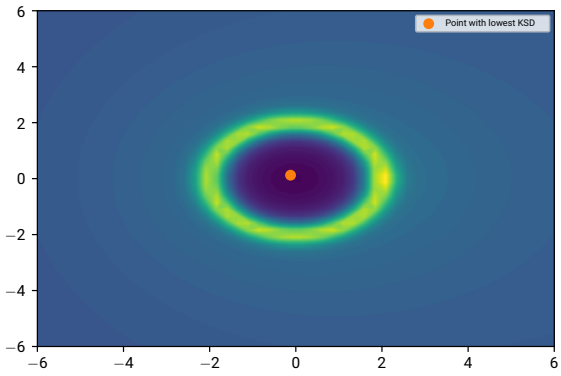
Figure 6.4.: Particle movement after the running SVGD on hundred samples sampled from a distribution with slightly different means and variance than the targeted distribution.

further present the results of the rectangular arrangement in Figure 6.6 and for the random distribution in Figure 6.7.

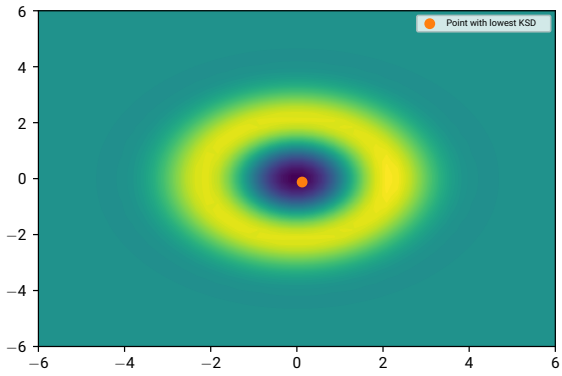
We begin our discussion with the initial baselines to discuss the effects of the kernel choice on the final result. Observing the inverse-log kernel, we can see that the performance differs a lot from the RBF kernel. While the inverse-log kernel show strong changes in the KSD around the locations of the fixed samples as seen in Figure 6.8b, the RBF kernel result in a more smoothed end result. A similar sensitivity can be observed in the kernel hyper parameters, where the inverse-log kernel required a manual tuning of its α parameter for almost every experiment compared to the RBF kernel, which used the same value for all experiments. This need for precise kernel parameter adjustment based on the current distribution of samples makes the inverse-log kernel a suboptimal choice for our intended application in curriculum learning. We therefore focus on the two other kernels, which seem to be more robust in terms of their parameters, although a downside of the vanilla RBF kernel appears when the distributions variance is not diagonal.

We investigated the differences in KSD estimation for the RBF and the Mahalanobis kernel in Figure 6.9, where the target distribution has a non-diagonal covariance and differs in its output dimensions in the order of magnitude. Here, we see that the Mahalanobis kernel captures this difference quite well compared to the RBF kernel. This result can be related back to the fixed bandwidth h employed in the kernel. In modern applications of kernels, such as Gaussian processes, a different bandwidth per dimension is often used which is set during training using techniques such as automatic relevance determination (ARD) [49]. However, since we have access to the target distribution and its covariance we can directly employ it as the Σ parameter of the Mahalanobis kernel. As we will see in chapter 7, where we encounter the exact problem of having differences in the orders of magnitude in the dimensions, this greatly improves the performance of Stein points-based methods without the needs for tuning the bandwidth over the course of training.

Comparing the results from the Stein points-based approach to our SVGD-based sampler we can see that many of the problems involving the fixed samples simply do not occur when direct sampling instead of moving particles is used. Compared to SVGD, the SP-based samplers are less affected by adversely placed samples. Since no particle movement is involved, “barriers” like in the circle or rectangle example pose no problem to the sampler when placing new samples, especially if the more smoothing kernels as the RBF or the Mahalanobis kernel are used. Additionally, Stein points are more resilient against non-optimal proposal

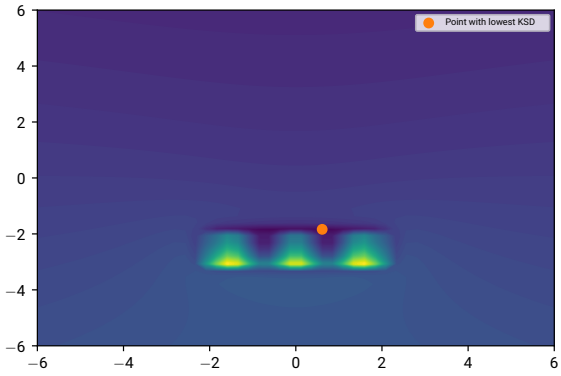


(a) Inverse Log Kernel

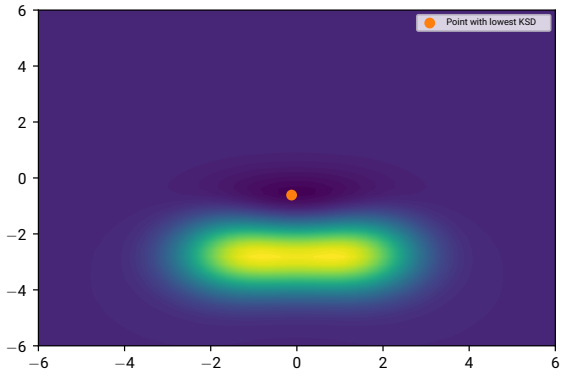


(b) RBF kernel

Figure 6.5.: KSD using different kernels with a circularly arranged fixed samples (shown in Figure 6.1a). The bandwidth for the RBF-based kernels was set to 2, while the α parameter of the inverse-log kernel was set to 0.4.

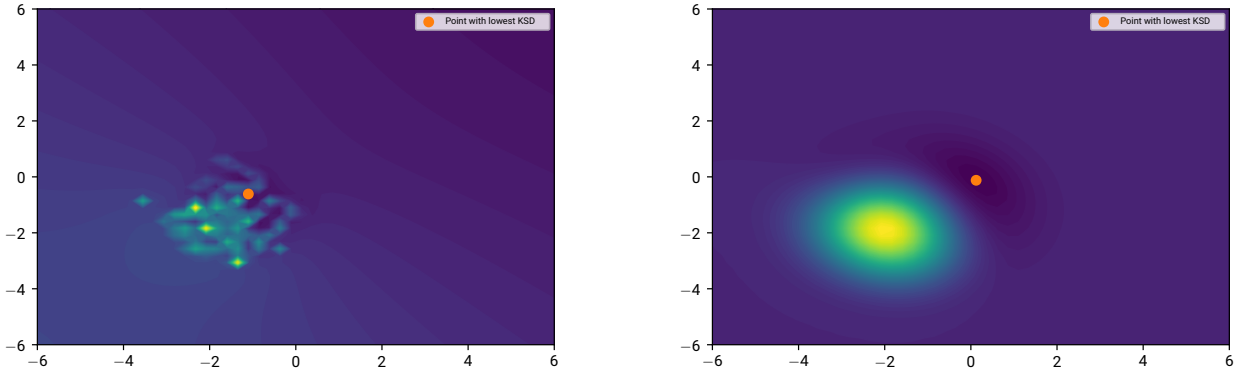


(a) Inverse Log Kernel



(b) RBF kernel

Figure 6.6.: KSD using different kernels with a rectangularly arranged fixed samples (shown in Figure 6.1b). The bandwidth for the RBF-based kernels was set to 2, while the α parameter of the inverse-log kernel was set to 0.02.



(a) Inverse Log Kernel

(b) RBF kernel

Figure 6.7.: KSD using different kernels with fixed samples from a simple distribution (shown in Figure 6.1c). The bandwidth for the RBF-based kernels was set to 2, while the α parameter of the inverse-log kernel was set to 0.05.

sample sets, which do not lead to catastrophic failures as it is the case with SVGD-based approaches. The gradient fields, especially for the RBF-based kernels, are relatively smooth, which makes a less ideal set of proposal samples degrade the final output gracefully instead of collapsing into the highly noisy results observed in Figure 6.2. Since our changes to the core formulation are much less severe for the SP based samplers in the sense of changing core parts of the associated math, this matches our initial expectation regarding the performance of both methods.

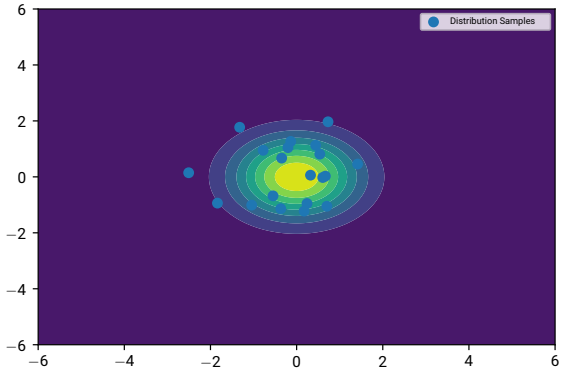
In summary, Stein points based samplers overcome some of the challenges which can occur when trying to sample from a curriculum learning distribution. While the kernel choice is important depending on the targeted problem, we found a good candidate in the form of the Mahalanobis kernel. We therefore will continue by applying the Stein points samplers on an reinforcement learning problem together with the CL algorithm self-paced reinforcement learning. Before we focus our attention on these experiments however, we will discuss our particular kernel choice in the upcoming section.

6.2. Kernel Choices for Stein Points

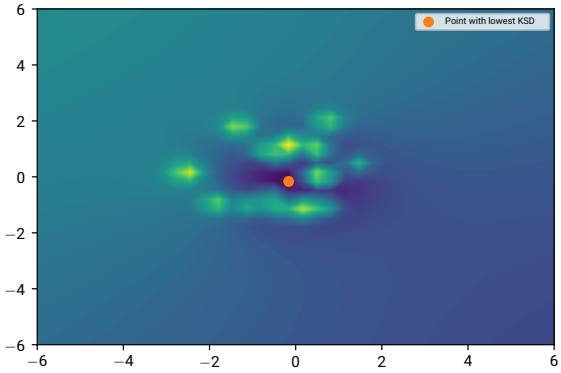
We show how the different kernel affect the sample placement on a standard normal distribution with a small amount of fixed samples. As can be seen in Figure 6.8, the RBF (Figure 6.8c) and the Mahalanobis kernel (Figure 6.8c) are quite similar in their output, especially when compared to the inverse log kernel (Figure 6.8b).

6.3. Stein Points Optimization With Approximate Objective

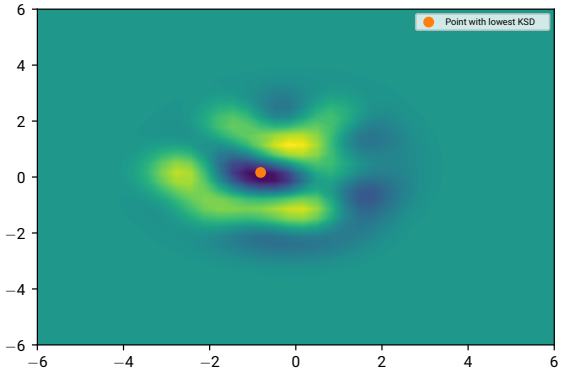
To verify that the proposed objective (5.6) is comparable to the original objective (5.5) (at least for our use case), we perform an empirical evaluation of the two variants.



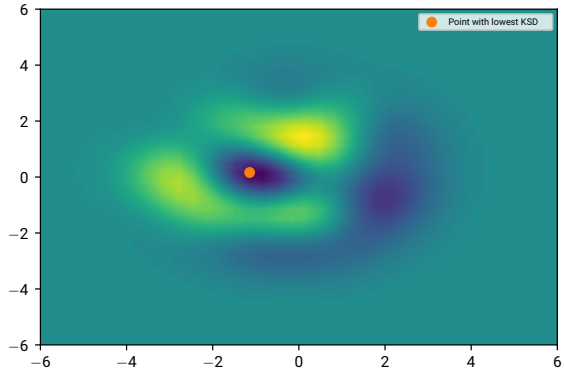
(a) Distribution and fixed samples



(b) Inverse Log Kernel kernel



(c) RBF kernel



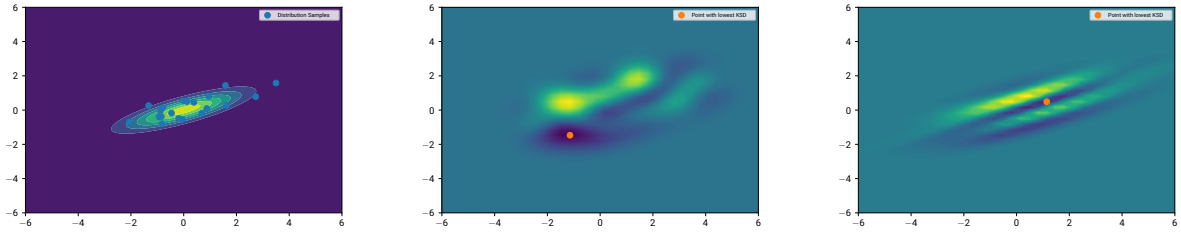
(d) Mahalanobis kernel

Figure 6.8.: KSD for different kernel choices with 20 fixed samples. The KSD is visualized as the gradient background, where darker areas symbolize lower values compared to lighter areas. The bandwidth for the RBF-based kernels was set to 2, while $\alpha = 0.4$ in the case of the inverse-log kernel.

For this, we simulate a curriculum by transforming an initial normal distribution to a target distribution by means of simple interpolation of the parameters (see Figure 6.10 for a visualization). Starting from an initial set of samples directly sampled from the distribution we update this set at every iteration using one of four methods:

1. **Iterative Stein point sampler**
2. **Full optimization Stein point sampler**
3. **Reduced optimization Stein point sampler**
4. **Random sampling from the distribution**

In the full optimization Stein point sampler variant we use the full quadratic objective (5.5) but limit computation time in addition to tuning some other parameters of the sampler, which we report in Table A.1. We compare this to the reduced optimization Stein point sampler, which uses the reduced objective (5.6) and the iterative Stein point sampler. Additionally, we use the same technique as done in SPRL where new



(a) Distribution and fixed samples (b) RBF kernel (c) Mahalanobis kernel

Figure 6.9.: Kernelized Stein discrepancy for the RBF and the Mahalanobis kernel on a normal distribution with a non-diagonal covariance.

samples are added to experience buffer by sampling from the current curriculum distribution. In the case of the iterative Stein point sampler (ISPS) and the random sampling, we remove ten percent of the oldest samples compared to both optimization Stein point sampler (OSPS) variants where a constraint is given to the optimizer requiring that exactly ninety percent of the previous samples have to be used. We also test a variant of the random sampling technique where ten percent of the sample set are randomly replaced at every iteration (instead of the oldest ten percent).

We perform our tests on different amounts of proposal samples provided to the Stein points samplers, and calculate the KSD between the samples and the current curriculum distribution. We keep the total amount of samples fixed at 100 due to computational constraints.

6.3.1. Results

We show our results in Figure 6.11, where we compare the Stein points based samplers to random sampling with two different sample removal methods.

We described the negative impact of the reduced objective in subsection 5.4.1 and our experimental results in Figure 6.11 show the impact of this approximation in practice. We first note that sampling with replacement of the oldest samples, which is the method employed by vanilla SPRL, consistently achieves a high KSD, meaning the distribution is not matched well, compared to almost all other samplers. It is also close in performance to the random replacement sampler which further shows the relative inefficiency of this approach.

Both the full optimization and ISPS perform very well in terms of achieved KSD and are not affected by changes in the amount of proposal samples. We especially want to highlight how similar the performance of ISPS is compared to its full optimization counterpart, since it shows the effectiveness of the iterative process compared to the true optimization (which takes considerably more time to calculate).

As for the performance of the reduced OSPS, it always achieves a better KSD compared to its random sampling counterparts. While the effects of higher amounts of proposal samples are visible, they only become a problem if the amount becomes very high. Even in that case, the OSPS still performs better compared to the sampling method employed in vanilla SPRL.

We also have to consider the run time required to calculate the new sets of samplers, which differs a lot. This is especially a problem with the full optimization sampler, which takes over fifty minutes to perform a single step on a Intel Core i5-8250U CPU from 2016. While the execution time can be lowered significantly

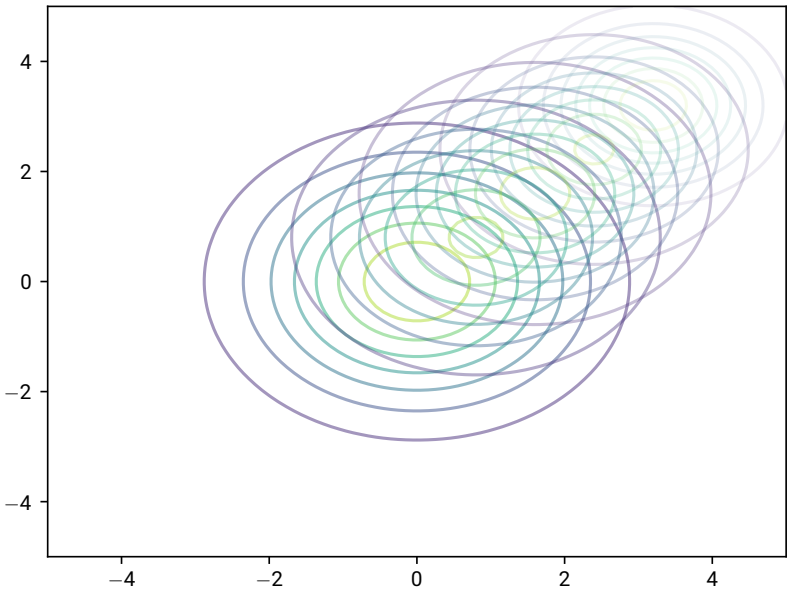
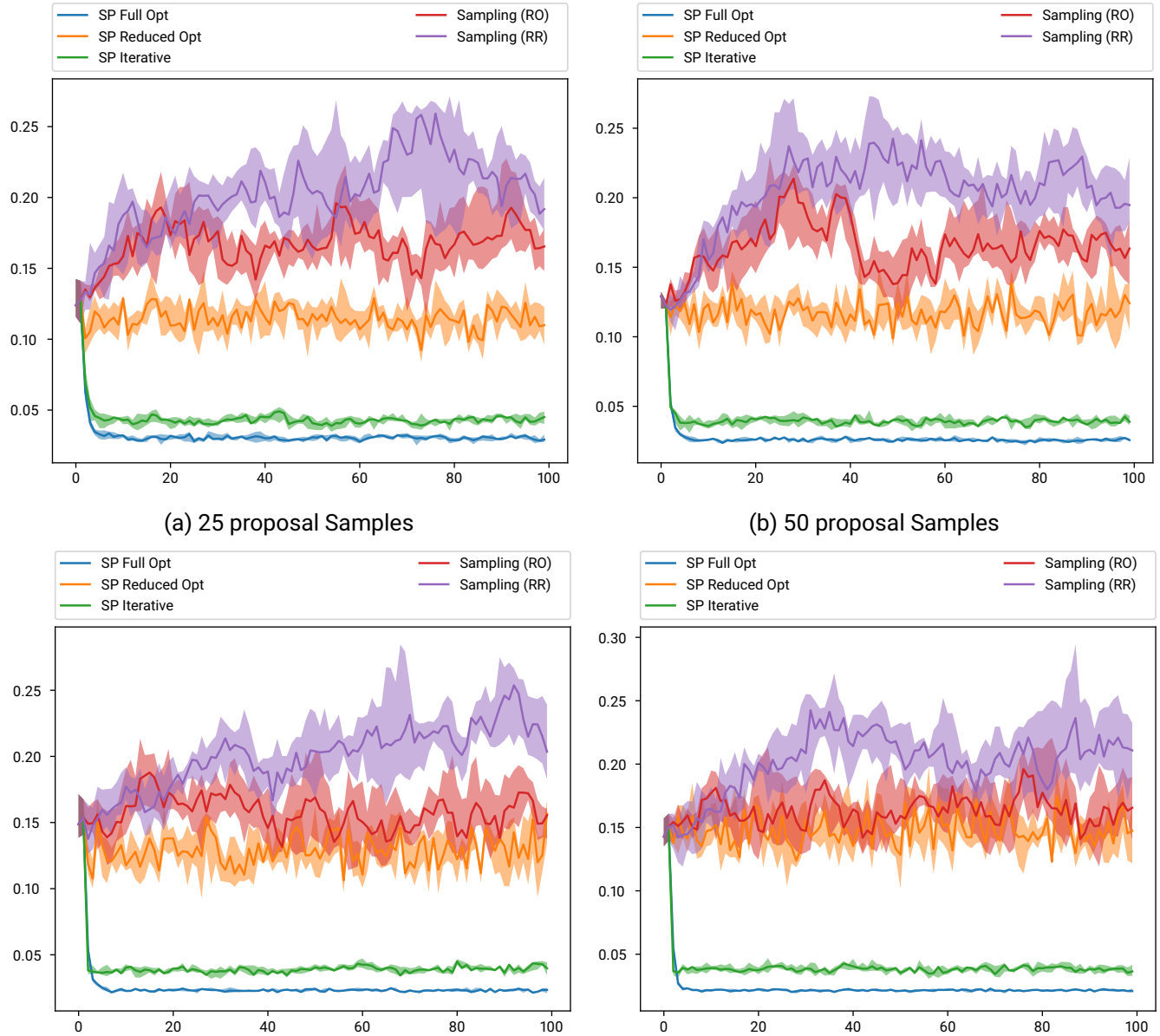


Figure 6.10.: A simulated curriculum, with the distribution changing its mean and variance over a fixed time frame. Later distributions are shown with increasing levels of transparency.

with stronger hardware the required computational power is still relatively higher compared to the reduced objective variant, which takes less than twenty seconds on the same hardware for a single step.

To summarize, our proposed reduced optimization Stein point sampler outperforms both random sampling methods, at least for reasonable proposal sample sizes between one to five times the amount of removed samples. While the final performance of the reduced OSPS is not on-par with its full or iterative counterparts, it nevertheless has its uses, since it allows additional constraints on the optimization problem to be used without sacrificing too much run time performance. Since vanilla SPRL employs the same technique as the "remove oldest" random sampling strategy to add and remove samples and the aforementioned ratios are not exceeded in our RL experiments, we will therefore test both of our Stein points-based samplers, ISPS and OSPS, on the RL task.



(a) 25 proposal Samples

(b) 50 proposal Samples

(c) 100 proposal Samples

(d) 200 proposal Samples

Figure 6.11.: KSD per sample method over a hundred step mock curriculum. We compare full and reduced optimization Stein points samplers with the iterative Stein point sampler and random sampling, where we either remove ten percent of the oldest samples (RO) or a random ten percent of the samples (RR) before sampling anew.

7. RL Experiments

In this chapter we present the application of our Stein points based samplers on an reinforcement learning experiment. We describe our experimental setup in section 7.1 and present and discuss our results in section 7.2.

7.1. Setup

We apply the Stein points algorithm to curriculum learning for RL using the self-paced reinforcement learning framework [8] and specifically the presented *gate* environment. This environment is very similar to the one presented in chapter 2, however instead of directly controlling the movement of the point-mass the policy generates the parameters for two PD-controllers. The first controller is used to move the point-mass towards the gate, while the second controller takes over as soon as the gate is passed. The dynamic system is linear with a small amount of Gaussian noises added to increase the overall difficulty. As the reward function, the distance to the goal position in addition to an L2-Regularization on the actions is used. The task is considered solved if the point-mass reaches the goal within some margin of error.

The reasoning for this specific choice of environment is the relative simplicity of SPRL together with the high sample usage of 100 newly added samples per iteration in the original presentation. As in the original paper [8], we use a buffer containing the task samples including the associated trajectories from the last ten iterations. However, we vary the amount of added samples per iteration between fifty and twenty to verify that our approach is indeed more sample efficient. This range is chosen since vanilla SPRL can still reliably learn the given task with about forty to fifty samples per iterations but starts to break down below this number.

We compare the performance of vanilla SPRL with our modified variant, which uses a Stein points sampler to sample from the curriculum distribution. Two variants of the Stein points algorithm are evaluated, which we call *iterative Stein point sampler* and *optimization Stein point sampler*: While the ISPS follows closely to the original iterative approach outlined in section 5.4, the OSPS is based on the hypothesis presented in Equation 5.6. In either case, the target distribution is set to the current task distribution as calculated by SPRL.

Both the ISPS and OSPS have two hyper parameters which influence the behavior of the sampler. They are:

- **Auxiliary Sample Factor**
- **Proposal Distribution Stretch Factor**

The first parameter controls the amount of proposal samples provided to the sampler as a ratio to the kept samples, while the second parameter functions as a stretching factor for the target distribution as described in subsection 5.4.2. Both hyper parameters where refined from an initial estimate using the Ax hyper parameter optimizer [50].

7.2. Results

We now investigate the gate task by replacing the sampling procedure from the original formulation of self-paced reinforcement learning with both our iterative Stein point sampler and optimization Stein point sampler (with the reduced objective) and comparing their performance with the baseline.

We start by comparing the received rewards for different sample sizes per iteration $i = [50, 40, 30, 20]$ in Figure 7.1. Although both the unmodified variant of SPRL and our version using the ISPS perform quite similar in terms of median reward, with the vanilla SPRL having a better peak performance, the OSPS performs consistently worse in terms of reward over iterations. It never reaches a median return of over one hundred, which is achieved for every sample count by the other two samplers. However, as seen by the 75 and 25 quantiles the OSPS has a very low variance, especially when compared to the ISPS.

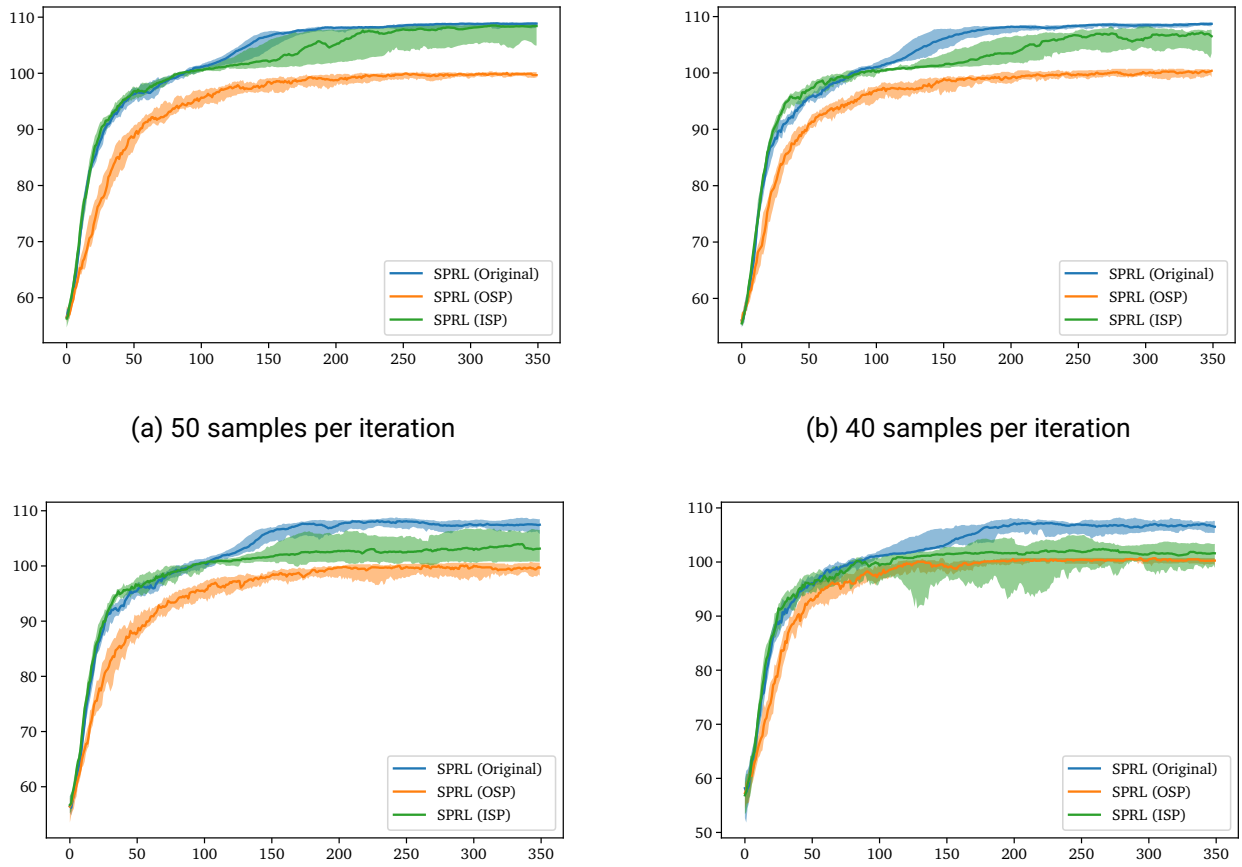
Given the relatively low reward achieved by the OSPS it is unsurprising that its performance when judged by the successes attained is zero, as seen in Figure 7.2. Here, the impact of the lower sample count is clearly visible with both the iterative Stein point and the vanilla SPRL samplers struggling to reach the final goal state. However, the vanilla sampler still reaches a median success rate of 0.5 in the lowest sample setting (Figure 7.2d), which is higher than the median success rate of the ISPS sampler in the second lowest setting.

To get a better insight into the training process we look at the KSD after every iteration for both the 50 and the 20 samples per iteration experiments. We calculated the KSD between all samples in the buffer and the current target in every iteration after sampling using the Mahalanobis kernel with a bandwidth of two. As we can see in Figure 7.3, the OSPS has the highest median KSD (therefore not fitting the target distribution well) but is comparable to the vanilla sampler in lower sample settings. The lowest KSD is consistently achieved by the ISPS, both in the 20 and the 50 samples per iteration experiments.

We start our discussion by first investigating potential shortcomings of our proposed sampling method before moving on to problems with SPRL and the RL environment under investigation.

We first focus on the OSPS which performs significantly worse than the ISPS. This performance can most likely be attributed to two effects, *sample reuse* and the general *shape and quality of samples*. Starting with sample reuse, we investigated OSPS in the hopes of a more effective method to replace older task samples according to how well they fit the current curriculum distribution. However when compared to the original version of SPRL, which just replaced a percentage of the oldest samples, this can lead to samples being reused over many iterations. Especially in scenarios like the gate environment, which either start or end on a distribution with high variance this scenario can occur frequently where early samples close to the modes of the distribution are not replaced until much later into the training. There are multiple ways to avoid this problem, either by employing the same sample replacement strategy as SPRL, incorporating the age of a sample as a constraint to be minimized or simply by choosing less overlapping start and end distributions for the curriculum.

This leaves us with the second effect, the shape and quality of the samples, which is more difficult to address since it is rooted in the core formulation of the OSPS. Samples generated by OSPS tend to clumped together



(a) 50 samples per iteration

(b) 40 samples per iteration

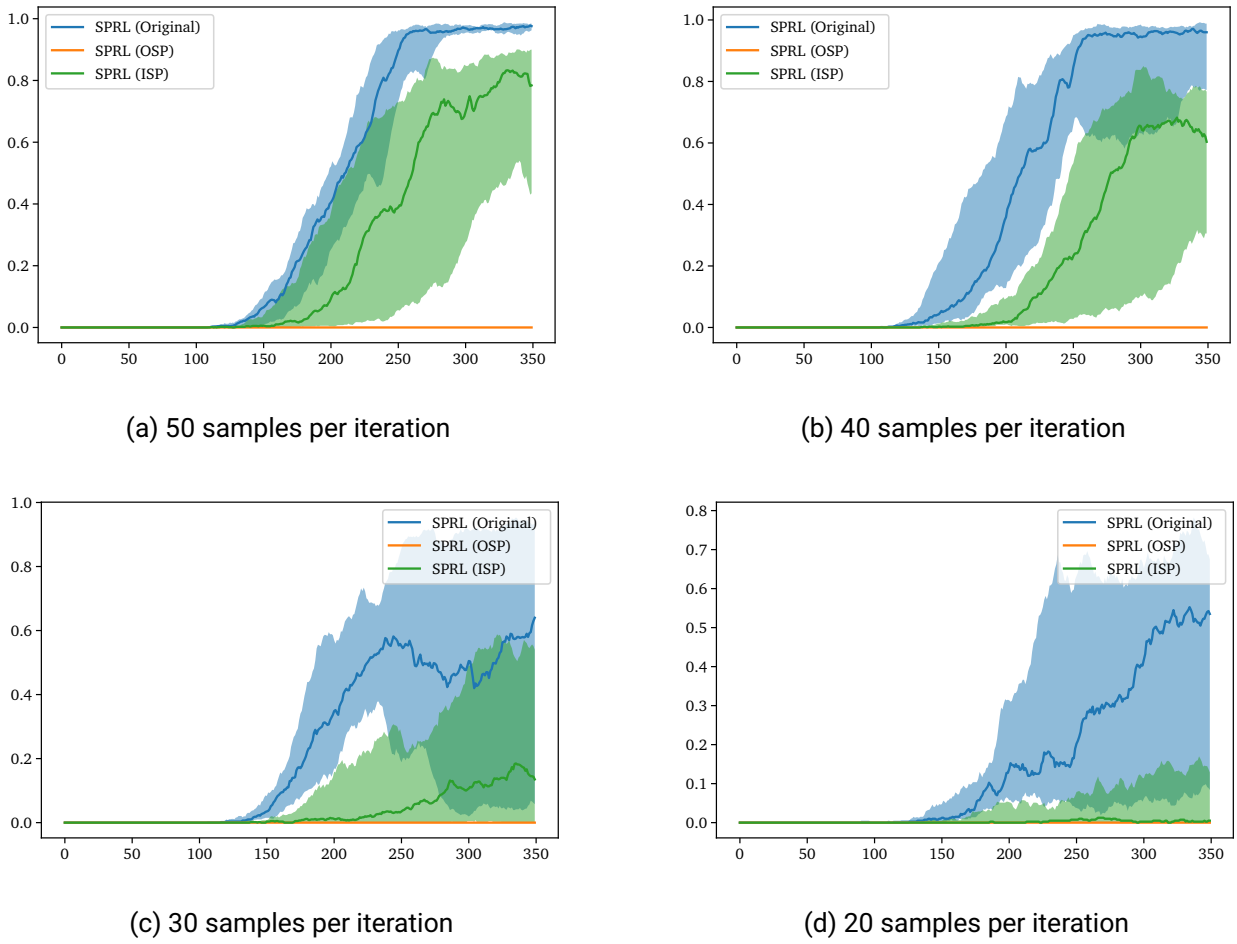
(c) 30 samples per iteration

(d) 20 samples per iteration

Figure 7.1.: Median reward per task sample on the gate task for twenty experiments. The shaded areas show the 25 and 75 quantiles.

compared to samples generated by ISPS. We provide an example of this behavior in Appendix B which also shows how the OSPS removes outlier samples and decreases the variance of the final sample distribution. While this behavior itself might still yield useful samples in some cases, it does lead to some problems when used in conjunction with SPRL.

One problem occurring in the original formulation of SPRL as presented by Klink *et al.* [8] is the collapse of the curriculum distribution variance. While this was already pointed out by the original authors and fixed in later iterations [51] by amending the curriculum calculation, it still a problem in the initial formulation which we base our experiment on. We initially focus on this specific version of SPRL since it uses *episodic reinforcement learning* for the policies and optimizes both the policy parameters and the parameters of the curriculum distribution jointly. While this approach allows us to apply our ParVIs methods to this joined distribution and sample both tasks and corresponding policies using our Stein points based samplers, we chose to focus first on the task distribution first before attempting a joint optimization. However, the variance collapse problem persisted, leading us to artificially increase the variance of the curriculum distribution before setting it as the target distribution for our samplers. We outlined this procedure in section 7.1, which reduced the variance collapse and lead to better results in terms of achieved reward and success. However, while it was not possible



(a) 50 samples per iteration

(b) 40 samples per iteration

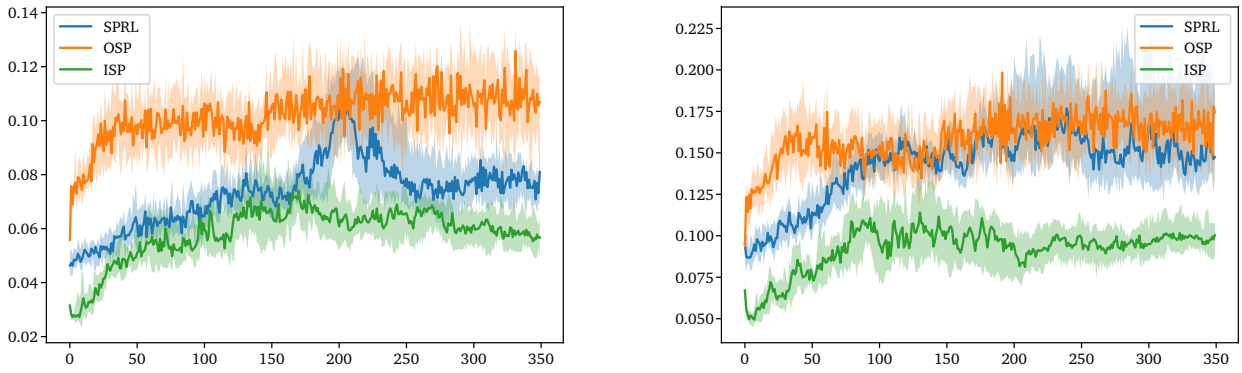
(c) 30 samples per iteration

(d) 20 samples per iteration

Figure 7.2.: Median success rate per task sample on the gate task for twenty experiments. The shaded areas show the 25 and 75 quantiles.

to do so in this thesis due to time constraints, we think that incorporating the later improvements done in the aforementioned paper could be even more beneficial.

We observed the positive impact of an increased variance in the target distribution also with the ISPS. Combined with a higher amount of proposal samples, which the ISPS can handle compared to the OSPS, this lead to the ISPS achieving a much higher median return compared to the OSPS. To explain why the vanilla SPRL still outperformed the ISPS in we take a look at the KSD per iteration and compare it to the reward received by both variants. Judging from Figure 7.1 and Figure 7.2, we can see that a local optimum in the RL training process occurs at about a reward of 100. This was already observed in the original paper [8] and is usually overcome by SPRL after a few iterations. Our hypothesis why this behavior is not reflected when using the ISPS is based on the lower KSD of the samples returned by this sampler as seen in Figure 7.3: Since these samples "match" the target distribution much better, the regularizing effect of the more varied samples produced by the original sampler is lost. This leads to the agent temporarily overfitting on the current sample selection, which prompts the curriculum generating part of SPRL to step faster through the curriculum for some iterations (which in this experiment decreases in variance). This in turn makes it more difficult for the agent to exit the local optima, slowing down the curriculum process until the end. We provide a visualization



(a) 50 samples per iteration

(b) 20 samples per iteration

Figure 7.3.: Median and 25/75 quantiles for the KSD after every sample iteration.

of this behavior in Figure C.1.

In total, we believe that these effects hinder the agent to overcome the local optima and reduce the attainable reward if the ISPS is used. Potential future work to alleviate these problems include using the higher variance distribution not only for the proposal samples but also as the target distribution. Additionally, different RL algorithms should be investigated, since these problems might be related to the usage of REPS as the method of choice in this experiment for finding the optimal policy.

8. Conclusion

In this thesis, we proposed a new method to sample from a curriculum distribution based on recent developments in the particle-based variational inference methods space. We validated our new samplers on both pedagogical and RL experiments, giving us both insights into our methods as well as the investigated curriculum learning algorithm.

Not only did our experiments show promising results, they also allowed us some deeper insights into the assumptions underlying SPRL. The initial hypothesis was that SPRL would benefit from a better coverage of the target distribution, but even though our proposed solution achieved this goal we did not observe performance gains in the low sample curriculum settings. This leads us to the conclusion, that the proposed target distributions are not optimal and that the design choices involving the experience buffer are more crucial than previously assumed. We can therefore offer some promising avenues which can pursued as part of future work.

8.1. Future Work

While the performance of our iterative Stein point sampler was at least comparable to the vanilla samplers used in SVGD, they still achieved a lower reward in the RL experiments compared to their unmodified version. One potential improvement for this could be found in a recent version of SPRL [51], which puts more emphasis on the agents performance when updating the current curriculum distribution. Since we observed some situations, where the distribution was changed to fast leaving the agent stuck in a local optima, switching to the newer variant might prove beneficial in such cases.

Some simpler changes to SPRL, which were out of scope of this thesis, involve the settings of the target distribution. We observed in our experiment that the ISPS was consistently achieving a lower KSD compared to all other investigated samplers; however, we pointed out that this might not be beneficial for the targeted environment. It would therefore be interesting to test different target distributions for the sampler to match, such that sample variance (and therefore the KSD) is increased. Another potential alleviation might be possible by changing the targeted task and environment, since it was shown by the original authors [8] to be difficult to solve for all agents except vanilla SPRL.

Another interesting avenue is the usage of deep learning methods for reinforcement learning such as soft actor critic (SAC) [10]. These methods might be able to better incorporate the different dynamics observed throughout the curriculum, therefore leading to a more stable learning process. In the case of the gate environment investigated in this thesis it was also shown that the deep learning version of SPRL performs better compared to its episodic counterpart [51]. Another interesting research direction in this space would be the incorporation of the policies experience buffer into the curriculum learning process. These buffers are part of some off-policy methods such as the aforementioned SAC and store previously made steps instead

of full trajectories. Using our presented methods, an intelligent pruning and resampling of these buffers could be added to the training process, which would favor relevant experience over newer one, which might positively affect the efficiency of these methods. Finally, deep learning methods take much longer to run and train compared to their episodic counter parts. Their relative benefit from an more effective curriculum sampling process would therefore be higher.

Finally, more investigations into the behavior of OSPS might lead to some potential solutions for the problems observed in our RL experiments. While we already tested one option by increasing the variance of our proposal distribution samples, other options are also available. For example, instead of sampling from the target distribution it might turn out more beneficial to sample from a grid overlaying the space or use some other form of proposal sample generation. Another way of improving performance could be found by investigating the differences between samples produced by the OSPS and the vanilla sampler in SPRL. Using these insights might also reveal some hidden "dependencies" of SPRL on the nature of the task samples, which is not satisfied by the samples produced using the OSPS.

8.2. Summary

Although still a relatively new subfield of variational inference, particle-based variational inference methods has already shown some promising new applications and future research opportunities. So does the usage of curriculum learning methods in the space of reinforcement learning open new and interesting avenues to train agents in a more versatile and potentially more effective manner. However, we have seen that the core formulation of such methods can lead to inefficiencies when there is not enough difference between tasks in these curricula, leading to superfluous and expensive agent evaluations. In this thesis we proposed to use ParVIs methods to reduce the amount of uninformative task rollouts by recasting sampling from the curriculum distribution as a problem of variational inference.

We investigated two ParVIs methods, Stein variational gradient descent and Stein points and proposed the necessary adaptions required for their application on curriculum learning problems. Validating our adaptations, we experimentally showed that an SVGD sampler introduces difficulties when applied to CL.

For the Stein points based approach, we presented two samplers, iterative Stein point sampler and optimization Stein point sampler, which directly optimize the kernelized Stein discrepancy by adding and removing samples according to our curriculum distribution. We were able to show that these samplers do not suffer from the same problems as the SVGD based approach and integrated them into an curriculum learning for reinforcement learning algorithm called self-paced reinforcement learning. While the application of this modified SPRL did not yield the expected results, we pointed out a few shortcomings of our method and proposed some future work, which could lead to improved results and a more efficient sample process for curriculum learning.

Bibliography

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning", in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09, New York, NY, USA: Association for Computing Machinery, Jun. 14, 2009, pp. 41–48, ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553380.

[2] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey", *Journal of Machine Learning Research*, vol. 21, no. 181, pp. 1–50, 2020, ISSN: 1533-7928.

[3] P. Klink, C. DEramo, J. R. Peters, and J. Pajarinen, "Self-Paced Deep Reinforcement Learning", in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 9216–9227.

[4] J. Wöhlke, F. Schmitt, and H. van Hoof, "A Performance-Based Start State Curriculum Framework for Reinforcement Learning", in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '20, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 5, 2020, pp. 1503–1511, ISBN: 978-1-4503-7518-4.

[5] R. M. French, "Catastrophic forgetting in connectionist networks", *Trends in Cognitive Sciences*, vol. 3, no. 4, pp. 128–135, Apr. 1, 1999, ISSN: 1364-6613. DOI: 10.1016/S1364-6613(99)01294-2.

[6] C. Zhang, J. Bütepage, H. Kjellström, and S. Mandt, "Advances in Variational Inference", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 2008–2026, Aug. 2019, ISSN: 1939-3539. DOI: 10.1109/TPAMI.2018.2889774.

[7] C. Liu, J. Zhuo, P. Cheng, R. Zhang, and J. Zhu, "Understanding and Accelerating Particle-Based Variational Inference", in *Proceedings of the 36th International Conference on Machine Learning*, PMLR, May 24, 2019, pp. 4082–4092.

[8] P. Klink, H. Abdulsamad, B. Belousov, and J. Peters, "Self-Paced Contextual Reinforcement Learning", in *Proceedings of the Conference on Robot Learning*, PMLR, May 12, 2020, pp. 513–529.

[9] A. Graves, M. G. Bellemare, J. Menick, R. Munos, and K. Kavukcuoglu, "Automated Curriculum Learning for Neural Networks", in *Proceedings of the 34th International Conference on Machine Learning*, PMLR, Jul. 17, 2017, pp. 1311–1320.

[10] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor", in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, Jul. 3, 2018, pp. 1861–1870.

[11] J. Peters, K. Mulling, and Y. Altun, "Relative Entropy Policy Search", in *Twenty-Fourth AAAI Conference on Artificial Intelligence*, Jul. 5, 2010.

[12] A. Lazaric, M. Restelli, and A. Bonarini, "Transfer of samples in batch reinforcement learning", in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08, New York, NY, USA: Association for Computing Machinery, Jul. 5, 2008, pp. 544–551, ISBN: 978-1-60558-205-4. DOI: 10.1145/1390156.1390225.

[13] F. Fernández, J. García, and M. Veloso, "Probabilistic Policy Reuse for inter-task transfer learning", *Robotics and Autonomous Systems*, Advances in Autonomous Robots for Service and Entertainment, vol. 58, no. 7, pp. 866–871, Jul. 31, 2010, ISSN: 0921-8890. DOI: 10.1016/j.robot.2010.03.007.

[14] A. Hallak, D. D. Castro, and S. Mannor, "Contextual markov decision processes", *CoRR*, vol. abs/1502.02259, 2015.

[15] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse Curriculum Generation for Reinforcement Learning", in *Proceedings of the 1st Annual Conference on Robot Learning*, PMLR, Oct. 18, 2017, pp. 482–495.

[16] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic Goal Generation for Reinforcement Learning Agents", in *International Conference on Machine Learning*, PMLR, Jul. 3, 2018, pp. 1515–1528.

[17] M. Riedmiller *et al.*, "Learning by Playing Solving Sparse Reward Tasks from Scratch", in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, Jul. 3, 2018, pp. 4344–4353.

[18] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes", in *Proceedings of ICLR, 2014*, May 1, 2014. arXiv: 1312.6114.

[19] E. G. Tabak and E. Vanden-Eijnden, "Density estimation by dual ascent of the log-likelihood", *Communications in Mathematical Sciences*, vol. 8, no. 1, pp. 217–233, 2010.

[20] E. G. Tabak and C. V. Turner, "A Family of Nonparametric Density Estimation Algorithms", *Communications on Pure and Applied Mathematics*, vol. 66, no. 2, pp. 145–164, 2013, ISSN: 1097-0312. DOI: 10.1002/cpa.21423.

[21] D. Rezende and S. Mohamed, "Variational Inference with Normalizing Flows", in *Proceedings of the 32nd International Conference on Machine Learning*, PMLR, Jun. 1, 2015, pp. 1530–1538.

[22] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing Flows: An Introduction and Review of Current Methods", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 3964–3979, Nov. 2021, ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.2992934.

[23] C. Stein, "A bound for the error in the normal approximation to the distribution of a sum of dependent random variables", in *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*, vol. 6, University of California Press, 1972, pp. 583–603.

[24] J. Gorham and L. Mackey, "Measuring Sample Quality with Stein' s Method", in *Advances in Neural Information Processing Systems*, vol. 28, Curran Associates, Inc., 2015.

[25] A. D. Barbour, "Stein's method and poisson process convergence", *Journal of Applied Probability*, vol. 25, no. A, pp. 175–184, 1988, ISSN: 0021-9002, 1475-6072. DOI: 10.2307/3214155.

[26] K. Chwialkowski, H. Strathmann, and A. Gretton, "A Kernel Test of Goodness of Fit", in *Proceedings of The 33rd International Conference on Machine Learning*, PMLR, Jun. 11, 2016, pp. 2606–2615.

[27] Q. Liu, J. Lee, and M. Jordan, "A Kernelized Stein Discrepancy for Goodness-of-fit Tests", in *Proceedings of The 33rd International Conference on Machine Learning*, PMLR, Jun. 11, 2016, pp. 276–284.

[28] I. Steinwart and A. Christmann, *Support Vector Machines*. Springer Science & Business Media, 2008.

[29] Y. Wang, J. Chen, C. Liu, and L. Kang, "Particle-based energetic variational inference", *Statistics and Computing*, vol. 31, no. 3, p. 34, Apr. 17, 2021, ISSN: 1573-1375. DOI: 10.1007/s11222-021-10009-7.

[30] Q. Liu and D. Wang, "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm", in *Advances in Neural Information Processing Systems*, vol. 29, Curran Associates, Inc., 2016.

[31] L. Ambrogioni, U. Güçlü, Y. Güçlütürk, and M. van Gerven, "Wasserstein variational gradient descent: From semi-discrete optimal transport to ensemble variational inference", *CoRR*, vol. abs/1811.02827, 2018. arXiv: 1811.02827.

[32] C. Zhang, Z. Li, H. Qian, and X. Du, "DPVI: A Dynamic-Weight Particle-Based Variational Inference Framework", Dec. 1, 2021. arXiv: 2112.00945 [cs].

[33] J. Zhuo, C. Liu, J. Shi, J. Zhu, N. Chen, and B. Zhang, "Message Passing Stein Variational Gradient Descent", in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, Jul. 3, 2018, pp. 6018–6027.

[34] A. Lambert and B. Boots, "Entropy Regularized Motion Planning via Stein Variational Inference", Jul. 11, 2021. arXiv: 2107.05146 [cs].

[35] L. Barcelos, A. Lambert, R. Oliveira, P. Borges, B. Boots, and F. Ramos, "Dual online stein variational inference for control and dynamics", *CoRR*, vol. abs/2103.12890, 2021. arXiv: 2103.12890.

[36] Y. Liu, P. Ramachandran, Q. Liu, and J. Peng, "Stein Variational Policy Gradient", Apr. 7, 2017. arXiv: 1704.02399 [cs].

[37] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement Learning with Deep Energy-Based Policies", in *Proceedings of the 34th International Conference on Machine Learning*, PMLR, Jul. 17, 2017, pp. 1352–1361.

[38] S. Narvekar, J. Sinapov, and P. Stone, "Autonomous Task Sequencing for Customized Curriculum Design in Reinforcement Learning.", in *IJCAI*, 2017, pp. 2536–2542.

[39] J. Gorham and L. Mackey, "Measuring Sample Quality with Kernels", in *Proceedings of the 34th International Conference on Machine Learning*, PMLR, Jul. 17, 2017, pp. 1292–1301.

[40] Q. Chang, Q. Chen, and X. Wang, "Scaling Gaussian RBF kernel width to improve SVM classification", in *2005 International Conference on Neural Networks and Brain*, vol. 1, Oct. 2005, pp. 19–22. DOI: 10.1109/ICNNB.2005.1614559.

[41] J. Han and Q. Liu, "Stein Variational Gradient Descent Without Gradient", in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, Jul. 3, 2018, pp. 1900–1908.

[42] Y. Feng, D. Wang, and Q. Liu, "Learning to Draw Samples with Amortized Stein Variational Gradient Descent", Oct. 30, 2017. arXiv: 1707.06626 [stat].

[43] Anonymous, "Understanding the Variance Collapse of SVGD in High Dimensions", presented at the Submitted to The Tenth International Conference on Learning Representations, Sep. 29, 2021.

[44] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics", in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, Citeseer, 2011, pp. 681–688.

[45] W. Y. Chen, L. Mackey, J. Gorham, F.-X. Briol, and C. Oates, "Stein Points", in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, Jul. 3, 2018, pp. 844–853.

[46] S. Burer and A. Saxena, "The MILP Road to MIQCP", in *Mixed Integer Nonlinear Programming*, J. Lee and S. Leyffer, Eds., New York, NY: Springer, 2012, pp. 373–405, ISBN: 978-1-4614-1927-3. DOI: 10.1007/978-1-4614-1927-3_13.

[47] Y. Chen, M. Welling, and A. Smola, "Super-samples from kernel herding", in *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'10, Arlington, Virginia, USA: AUAI Press, Jul. 8, 2010, pp. 109–116, ISBN: 978-0-9749039-6-5.

[48] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", Jan. 29, 2017. arXiv: 1412.6980 [cs].

[49] D. Duvenaud, "Automatic model construction with Gaussian processes", Thesis, University of Cambridge, Nov. 11, 2014. DOI: 10.17863/CAM.14087.

[50] E. Bakshy *et al.*, "AE: A domain-agnostic platform for adaptive experimentation", in *NeurIPS Systems for ML Workshop*, 2018, pp. 1–8.

[51] P. Klink, H. Abdulsamad, B. Belousov, C. D'Eramo, J. Peters, and J. Pajarinen, "A Probabilistic Interpretation of Self-Paced Learning with Applications to Reinforcement Learning", Feb. 25, 2021. arXiv: 2102.13176 [cs].

[52] J. Bradbury *et al.*, *JAX: Composable transformations of Python+NumPy programs*, version 0.2.5, 2018.

[53] Gurobi Optimization, LLC, *Gurobi optimizer reference manual*, 2022.

A. Gurobi Parameters for the Constrained Stein points sampler

Parameter	Value
NumericFocus	1
MIPFocus	3
MIPGap	0.30
WorkLimit	120

Table A.1.: Gurobi Parameters

B. Iterative vs Optimization Stein Points Sampler

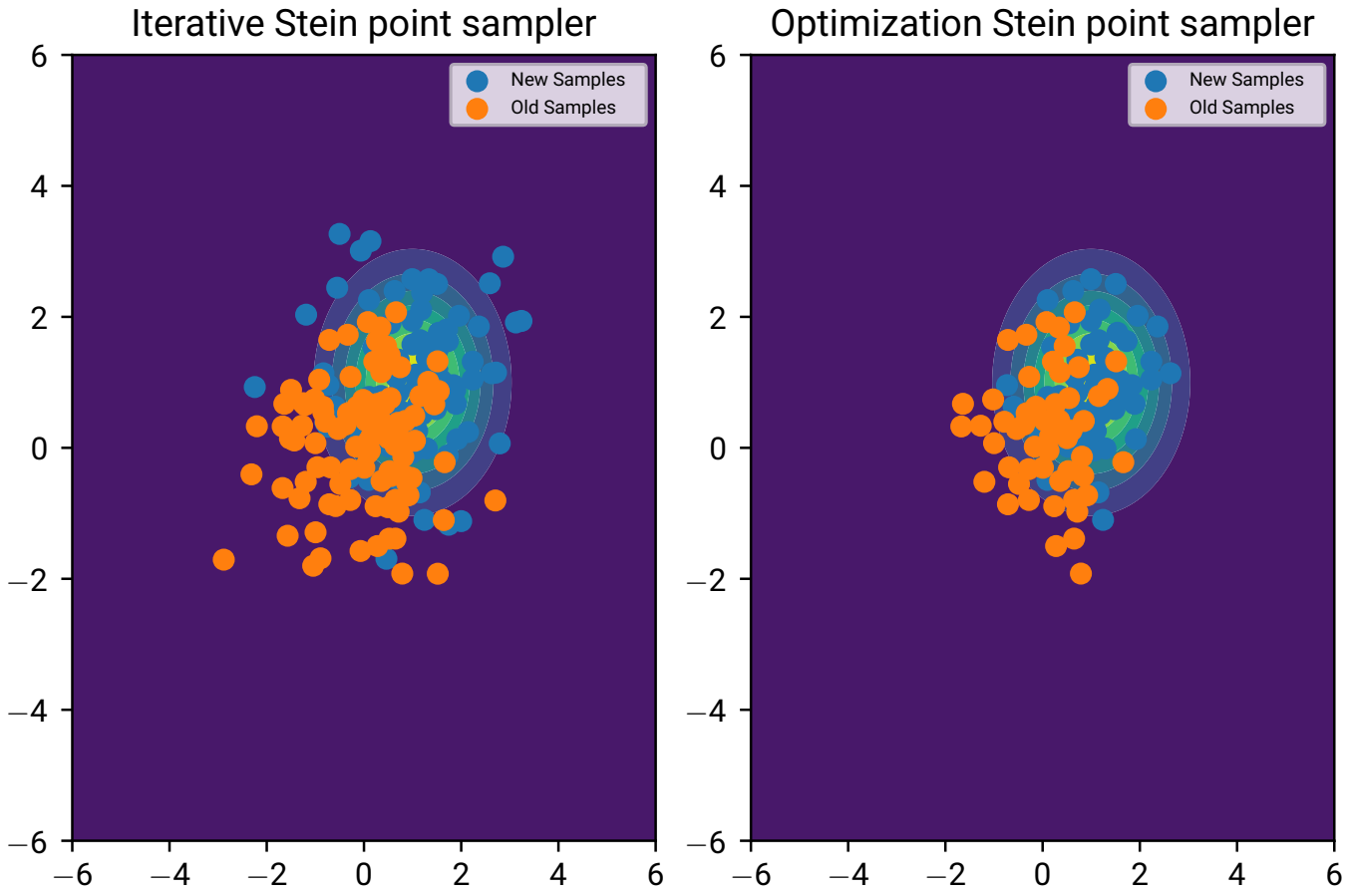
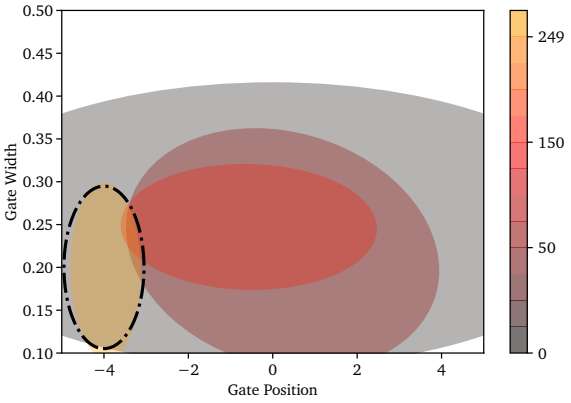
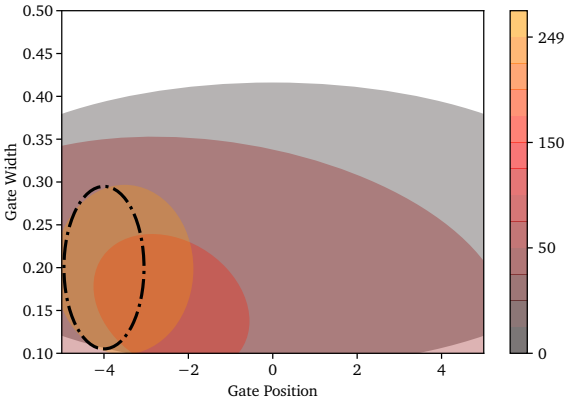


Figure B.1.: Comparing the final sample placement of ISPS and OSPS. Both samplers received the same 100 initial and 100 proposal samples as inputs and used the RBF kernel with a bandwidth set to one. Target sample size was 100, with samples being selected from the joined set of initial and proposal samples until the target size was reached.

C. Curriculum for SPRL and ISPS



(a) SPRL



(b) ISPS

Figure C.1.: Comparing the curricula generated using vanilla SPRL to the ones created when ISPS is used. The curriculum distribution are visualized at different iterations with the target highlighted with a dashed line. Compared to the vanilla variant, the distributions for shrink faster when the ISPS is used but take longer to reach the final position.

D. Technical Details

We implemented all samplers from scratch using the *jax* library [52]. For the optimization-based samplers, we use the proprietary *Gurobi* sampler [53]. All of our code is open-source and can be found on Github[1].

For the RL experiments, we use a modified version of the original code for the SPRL paper [8], with the added Stein points sampler. The code can be found on Github[2] as well.

[1] https://github.com/miterion/adaptive-sampler-baselines
[2] https://github.com/miterion/self-paced-rl/