Journal of Machine Learning Research 21 (2020) 1-37 Submitted 9/18; Revised 12/19; Published 9/20

Adaptive Smoothing for Path Integral Control

Dominik Thalmeier D.THALMEIER@SCIENCE.RU.NL
Radboud University Nijmegen
Nijmegen, The Netherlands

Hilbert J. Kappen B.KAPPEN@SCIENCE.RU.NL
Radboud University Nijmegen
Nijmegen, The Netherlands

Simone Totaro SIMONE.TOTARO@GMAIL.COM
Universitat Pompeu Fabra
Barcelona, Spain

Vicenç Gómez VICEN.GOMEZ@UPF.EDU
Universitat Pompeu Fabra
Barcelona, Spain

Editor: Benjamin Recht

Abstract

In Path Integral control problems a representation of an optimally controlled dynamical system can be formally computed and serve as a guidepost to learn a parametrized policy. The Path Integral Cross-Entropy (PICE) method tries to exploit this, but is hampered by poor sample efficiency. We propose a model-free algorithm called ASPIC (Adaptive Smoothing of Path Integral Control) that applies an inf-convolution to the cost function to speedup convergence of policy optimization. We identify PICE as the infinite smoothing limit of such technique and show that the sample efficiency problems that PICE suffers disappear for finite levels of smoothing. For zero smoothing, ASPIC becomes a greedy optimization of the cost, which is the standard approach in current reinforcement learning. ASPIC adapts the smoothness parameter to keep the variance of the gradient estimator at a predefined level, independently of the number of samples. We show analytically and empirically that intermediate levels of smoothing are optimal, which renders the new method superior to both PICE and direct cost optimization.

Keywords: Path Integral Control, Entropy-Regularization, Cost Smoothing

1. Introduction

How to choose an optimal action? For noisy dynamical systems, stochastic optimal control theory provides a framework to answer this question. Optimal control is framed as an optimization problem to find the control that minimizes an expected cost function. For non-linear dynamical systems that are continuous in time and space, this problem in general hard.

A method that has proven to work well is to introduce a parametrized policy like a neural network (Mnih et al., 2015; Levine et al., 2016; Duan et al., 2016; François-Lavet et al., 2018) and greedily optimize the expected cost using gradient descent (Williams, 1992; Peters and Schaal, 2008; Schulman et al., 2015; Heess et al., 2017). To achieve a robust decrease of the expected cost it is important to ensure that in each step, the updated policy stays in the proximity of the old policy (Duan et al., 2016). This can be achieved by enforcing a trust region constraint (Peters et al., 2010; Schulman et al., 2015) or using adaptive regularization that punishes strong deviations of the new policy from the old policy (Heess et al., 2017).

However the applicability of these methods is limited, as in each iteration of the algorithm, samples from the controlled system have to be computed, either from a simulator or from a real system. We want to increase the convergence rate of policy optimization to reduce the number of simulations needed.

To this end we consider path integral control problems (Kappen, 2005; Todorov, 2009; Kappen et al., 2012), that offer an alternative approach to direct cost optimization and explore if this allows to speed up policy optimization. This class of control problems permits arbitrary non-linear dynamics and state cost, but requires a linear dependence of the control on the dynamics and a quadratic control cost (Kappen, 2005; Bierkens and Kappen, 2014; Thijssen and Kappen, 2015). These restrictions allow to obtain an explicit expression for the probability density of optimally controlled system trajectories. Through this, an information-theoretical measure of the deviation of the current control policy from the optimal control can be calculated. The Path Integral Cross-Entropy (PICE) method (Kappen and Ruiz, 2016) proposes to use this measure as a pseudo-objective for policy optimization.

However, there is yet no comparative study on whether PICE actually offers an advantage over direct cost optimization; and, in its original form (Kappen and Ruiz, 2016), the PICE method does not scale well to complex problems because the PICE gradient is hard to estimate if the current controller is not close enough to the optimal control (Ruiz and Kappen, 2017). Furthermore the PICE method has been introduced with standard gradient descent and does not use trust regions to ensure robust updates, which has been shown to be effective for policy optimization (Duan et al., 2016).

In this work we propose and study a new kind of smoothing technique for the cost function that allows to interpolate between the optimization of the direct cost and the PICE objective. Optimizing this smoothed cost using a trust-region-based method yields an approach that is efficient and does not suffer from the feasibility issues of PICE. Our work is based on recently proposed smoothing techniques to speed up convergence in deep neural networks (Chaudhari et al., 2018). We adapt this smoothing technique to path integral control problems. In contrast to Chaudhari et al. (2018), smoothing for path integral control problems can be solved analytically and we obtain an expression of the gradient that can directly be computed from Monte Carlo samples. The strength of smoothing can be regulated by a parameter. Remarkably, this parameter can be determined independently of the number of samples. In the limits of this smoothing parameter we recover the PICE method for infinitely strong smoothing and direct cost optimization for zero smoothing, respectively. As in Chaudhari et al. (2018), the minimum of the smoothed cost, thus the optimal control policy, remains the same for all levels of smoothing.

We provide a theoretical argument why smoothing is expected to speed up optimization and conduct numerical experiments on different control tasks, which show this accelerative effect in practice. For this we develop an algorithm called ASPIC (Adaptive Smoothing for Path Integral Control) that uses cost smoothing to speed up policy optimization. The algorithm adjusts the smoothing parameter in each step to keep the variance of the gradient estimator at a predefined level. To ensure robust updates of the policy, ASPIC enforces a trust region constraint; similar to Schulman et al. (2015) this is achieved with natural gradient updates and an adaptive stepsize. Like other policy gradient based methods (Williams, 1992; Peters and Schaal, 2008; Schulman et al., 2015; Heess et al., 2017) ASPIC is model-free.

Many policy optimization algorithms update the control policy based on a direct optimization of the cost; examples are Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) or Path-Integral Relative Entropy Policy Search (PIREPS) (Gómez et al., 2014), where the later is particularly developed for path integral control problems. The main novelty of this work is the application to path integral control problems of the idea of smoothing, as introduced in Chaudhari et al. (2018). This technique outperforms direct cost optimization, achieving faster convergence rates with only a negligible amount of computational overhead.

2. Path Integral Control Problems

Consider the (multivariate) dynamical system

$$\dot{x}_t = f(x_t, t) + g(x_t, t)\left(u(x_t, t) + \xi_t\right), \tag{1}$$

with initial condition x_0. The control policy is implemented in the control function $u(x,t)$, which is additive to the white noise ξ_t which has variance $\frac{\nu}{dt}$.

Given a control function u and a time horizon T, this dynamical system induces a probability distribution $p_u(\tau)$ over state trajectories $\tau = \{x_t | \forall t : 0 < t \leq T\}$ with initial condition x_0.

We define the regularized expected cost

$$C(p_u) = \langle V(\tau)\rangle_{p_u} + \gamma KL(p_u||p_0), \tag{2}$$

with $V(\tau) = \int_0^T V(x_t, t)dt$, where the strength of the regularization $KL(p_u||p_0)$ is controlled by the parameter γ.

The Kullback-Leibler divergence $KL(p_u||p_0)$ puts high cost to controls u that bring the probability distribution p_u far away from the uncontrolled dynamics p_0 where $u(x_t,t) = 0$. We can also rewrite the regularizer $KL(p_u||p_0)$ directly in terms of the control function u by using the Girsanov theorem (compare Thijssen and Kappen (2015))

$$\log \frac{p_u(\tau)}{p_0(\tau)} = \frac{1}{\nu}\int_0^T \left(\frac{1}{2}u(x_t,t)^T u(x_t,t) + u(x_t,t)^T \xi_t\right) dt.$$

The regularization then takes the form of a quadratic control cost

$$\begin{aligned} KL(p_u||p_0) &= \left\langle \frac{1}{\nu}\int_0^T \left(\frac{1}{2}u(x_t,t)^T u(x_t,t) + u(x_t,t)^T \xi_t\right) dt \right\rangle_{p_u} \\ &= \left\langle \frac{1}{\nu}\int_0^T \frac{1}{2}u(x_t,t)^T u(x_t,t) dt \right\rangle_{p_u}, \end{aligned}$$

where we used that $\left\langle u(x_t,t)^T \xi_t \right\rangle_{p_u} = 0$. This shows that the regularization $KL(p_u||p_0)$ puts higher cost for large values of the controller u.

The path integral control problem is to find the optimal control function u^* that minimizes the regularized cost $C(p_u)$

$$u^* = \arg\min_u C(p_u). \tag{3}$$

For a more complete introduction to path integral control problems, see Thijssen and Kappen (2015); Kappen and Ruiz (2016).

2.1 Direct Cost Optimization Using Gradient Descent

A standard approach to find an optimal control function is to introduce a parametrized controller $u_\theta(x_t,t)$ (Williams, 1992; Schulman et al., 2015; Heess et al., 2017). This parametrizes the path probabilities p_{u_θ} and allows to optimize the expected cost $C(p_{u_\theta})$ (2) using stochastic gradient descent on the cost function:

$$\nabla_\theta C(p_{u_\theta}) = \left\langle S^\gamma_{p_{u_\theta}}(\tau) \nabla_\theta \log p_{u_\theta}(\tau) \right\rangle_{p_{u_\theta}}, \tag{4}$$

with the stochastic cost $S^\gamma_{p_{u_\theta}}(\tau) := V(\tau) + \gamma \log \frac{p_{u_\theta}(\tau)}{p_0(\tau)}$ (see Appendix A for details).

2.2 The Cross-Entropy Method for Path Integral Control Problems

An alternative approach to direct cost optimization was introduced as the PICE method in Kappen and Ruiz (2016). It uses that we can obtain an expression for p_{u^*}, the probability density of state trajectories induced by a system with the optimal controller u^*:

$$p_{u^*} = \arg\min_{p_u} C(p_u),$$

with $C(p_u)$ given by equation (2). Finding p_{u^*} is an optimization problem over the space of all probability distributions p_u that are induced by the controlled dynamical system (1). It has been shown (Bierkens and Kappen, 2014; Thijssen and Kappen, 2015) that we can solve this by replacing the minimization over p_u with a minimization over all path probability distributions p:

$$p_{u^*} \equiv p^* := \arg\min_p C(p) = \arg\min_p \langle V(\tau)\rangle_p + \gamma KL(p||p_0) = \frac{1}{Z} p_0(\tau) \exp\left(-\frac{1}{\gamma} V(\tau)\right), \tag{5}$$

with the normalization constant $Z = \left\langle \exp\left(-\frac{1}{\gamma}V(\tau)\right)\right\rangle_{p_0}$. Note that this is not a trivial statement, as we now take the minimum also over non-Markovian processes with non-Gaussian noise.

The PICE algorithm (Kappen and Ruiz, 2016) takes advantage of the existence of this explicit expression for the density of optimally controlled trajectories p_{u^*}. PICE does not directly optimize the expected cost, instead it minimizes the KL-divergence $KL\left(p^*||p_{u_\theta}\right)$ which measures the deviation of a parametrized distribution p_{u_θ} from the optimal one p^*. Although direct cost optimization and PICE are different methods, their global minimum is the same if the parametrization of u_θ can express the optimal control $u^* = u_{\theta^*}$. The parameters θ^* of the optimal controller are found using gradient descent:

$$\nabla_\theta KL\left(p^*||p_{u_\theta}\right) = \frac{1}{Z_{p_{u_\theta}}} \left\langle \exp\left(-\frac{1}{\gamma} S^\gamma_{p_{u_\theta}}(\tau)\right) \nabla_\theta \log p_{u_\theta}(\tau) \right\rangle_{p_{u_\theta}}, \tag{6}$$

where $Z_{p_{u_\theta}} \coloneqq \left\langle \exp\left(-\frac{1}{\gamma} S^\gamma_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}}$.

That PICE uses the optimal density as a guidepost for the policy optimization might give it an advantage compared to direct cost optimization. In practice however, this method only works properly if the initial guess of the controller u_θ does not deviate too much from the optimal control, as a high value of $KL\left(p^*||p_{u_\theta}\right)$ leads to a high variance of the gradient estimator and results in bootstrapping problems of the algorithm (Ruiz and Kappen, 2017; Thalmeier et al., 2016). In the next section we introduce a method that interpolates between direct cost optimization and the PICE method. This allows us to take advantage of the analytical solution of the optimal density without being hampered by the same bootstrapping problems as PICE.

3. Interpolating Between the two Methods: Smoothing Stochastic Control Problems

Cost function smoothing was recently introduced as a way to speed up optimization of neural networks (Chaudhari et al., 2018): optimization of a general cost function $f(\theta)$ can be speeded up by smoothing $f(\theta)$ using an inf-convolution with a distance kernel $d(\theta', \theta)$.[1] The smoothed function

$$J^\alpha(\theta) = \inf_{\theta'} \alpha d(\theta', \theta) + f(\theta') \tag{7}$$

preserves the global minima of the function $f(\theta)$. Chaudhari et al. (2018) showed that gradient descent optimization on $J^\alpha(\theta)$ instead of $f(\theta)$ may significantly speed up convergence. For that, the authors used a stochastic optimal control interpretation of the smoothing process of the cost function. In particular, they looked at the smoothing process as the solution to a non-viscous Hamiltion-Jacobi partial differential equation.

In this work, we want to use this accelerative effect to find the optimal parametrization of the controller u_θ. Therefore, we smooth the cost function $C(p_{u_\theta})$ as a function of the parameters θ. As $C(p_{u_\theta}) = \langle V(\tau)\rangle_{p_{u_\theta}} + \gamma KL(p_{u_\theta}||p_0)$ is a functional on the space of probability distributions p_{u_θ}, the natural distance[2] is the KL-divergence $KL(p_{u_{\theta'}}||p_{u_\theta})$. So we replace

$$\begin{aligned} f(\theta) &\to C(p_{u_\theta}) \\ d(\theta', \theta) &\to KL(p_{u_{\theta'}}||p_{u_\theta}) \end{aligned}$$

1. This is a generalized description. Chaudhari et al. (2018) used $d(\theta', \theta) = |\theta' - \theta|^2$.
2. Remark: Strictly speaking the KL is not a distance, but a directed divergence.

and obtain the smoothed cost $J^\alpha(\theta)$ as

$$\begin{aligned} J^\alpha(\theta) &= \inf_{\theta'} \alpha KL(p_{u_{\theta'}}||p_{u_\theta}) + C(p_{u_{\theta'}}) \\ &= \inf_{\theta'} \alpha KL(p_{u_{\theta'}}||p_{u_\theta}) + \gamma KL(p_{u_{\theta'}}||p_0) + \langle V(\tau)\rangle_{p_{u_{\theta'}}} . \end{aligned} \tag{8}$$

Note the different roles of α and γ: α penalizes the deviation of $p_{u_{\theta'}}$ from p_{u_θ}, while γ penalizes the deviation of $p_{u_{\theta'}}$ from the uncontrolled dynamics p_0.

3.1 Computing the Smoothed Cost and its Gradient

The smoothed cost J^α is expressed as a minimization problem that has to be solved. Here we show that for path integral control problems this can be done analytically. To do this we first show that we can replace $\inf_{\theta'} \to \inf_{p'}$ and then solve the minimization over p' analytically. We replace the minimization over θ' by a minimization over p' in two steps: first we state an assumption that allows us to replace $\inf_{\theta'} \to \inf_{u'}$ and then proof that for path integral control problems we can replace $\inf_{u'} \to \inf_{p'}$.

We assume that for every u_θ and any $\alpha > 0$, the minimizer $\theta^*_{\alpha,\theta}$ over the parameter space

$$\theta^*_{\alpha,\theta} := \arg\min_{\theta'} \alpha KL(p_{u_{\theta'}}||p_{u_\theta}) + C(p_{u_{\theta'}}) \tag{9}$$

is the parametrization of the minimizer $u^*_{\alpha,\theta}$ over the function space

$$u^*_{\alpha,\theta} := \arg\min_{u'} \alpha KL(p_{u'}||p_{u_\theta}) + C(p_{u'}),$$

such that $u^*_{\alpha,\theta} \equiv u_{\theta^*_{\alpha,\theta}}$. We call this assumption *full parametrization*. Naturally it is sufficient for full parametrization if $u_\theta(x,t)$ is a universal function approximator with a fully observable state space x and the time t as input, although this may be difficult to achieve in practice. With this assumption we can replace $\inf_{\theta'} \to \inf_{u'}$.

Analogously we replace $\inf_{u'} \to \inf_{p'}$: in Appendix B we proof that for path integral control problems the minimizer $u^*_{\alpha,\theta}$ over the function space induces the minimizer $p^*_{\alpha,\theta}$ over the space of probability distributions

$$p^*_{\alpha,\theta} := \arg\min_{p'} \alpha KL(p'||p_{u_\theta}) + C(p'), \tag{10}$$

such that $p^*_{\alpha,\theta} \equiv p_{u^*_{\alpha,\theta}}$. This step is similar to the derivation of equation (5) in Section 2.2, but now we have added an additional term $\alpha KL(p_{u'}||p_{u_\theta})$.

Hence, given a path integral control problem and a controller u_θ that satisfies full parametrization we can replace $\inf_{\theta'} \to \inf_{p'}$ and equation (8) becomes

$$J^\alpha(\theta) = \inf_{p'} \alpha KL(p'||p_{u_\theta}) + \gamma KL(p'||p_0) + \langle V(\tau)\rangle_{p'} . \tag{11}$$

This can be solved directly: first we compute the minimizer (see Appendix C for details)

$$p^*_{\alpha,\theta}(\tau) = \frac{1}{Z^\alpha_{p_{u_\theta}}} p_{u_\theta}(\tau) \exp\left(-\frac{1}{\gamma+\alpha} S^\gamma_{p_{u_\theta}}(\tau)\right) \tag{12}$$

with the normalization constant $Z^{\alpha}_{p_{u_\theta}} = \left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}}$. We plug this back in equation (11) and get an expression of the smoothed cost

$$J^{\alpha}(\theta) = -(\gamma+\alpha)\log\left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}} \tag{13}$$

and its gradient (for details see Appendix D)

$$\nabla_\theta J^{\alpha}(\theta) = -\frac{\alpha}{Z^{\alpha}_{p_{u_\theta}}}\left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\nabla_\theta \log p_{u_\theta}(\tau)\right\rangle_{p_{u_\theta}}, \tag{14}$$

which both can be estimated using samples from the distribution p_{u_θ}.

3.2 PICE, Direct Cost Optimization and Risk Sensitivity as Limiting Cases of Smoothed Cost Optimization

The smoothed cost and its gradient depend on the two parameters α and γ, which come from the smoothing equation (7) and the definition of the control problem (2), respectively. Although at first glance the two parameters seem to play a similar role, they change different properties of the smoothed cost $J^{\alpha}(\theta)$ when they are varied.

In the expression for the smoothed cost (13), the parameter α only appears in the sum $\gamma+\alpha$. Varying it changes the effect of the smoothing but leaves the optimum $\theta^* = \arg\min_\theta J^{\alpha}(\theta)$ of the smoothed cost invariant (see Appendix E). We therefore call α the *smoothing parameter*.

The larger α, the weaker the smoothing; in the limiting case $\alpha \to \infty$, smoothing is turned off as we can see from equation (13): for very large α, the exponential and the logarithmic function linearise, $J^{\alpha}(\theta) \to C(p_{u_\theta})$ and we recover direct cost optimization. For the limiting case $\alpha \to 0$, we recover the PICE method: the optimizer $p^*_{\alpha,\theta}$ becomes equal to the optimal density p^* and the gradient on the smoothed cost (14) becomes proportional to the PICE gradient (6):

$$\lim_{\alpha\to 0} \frac{1}{\alpha}\nabla_\theta J^{\alpha}(\theta) = \nabla_\theta KL(p^* || p_{u_\theta}).$$

Varying γ changes the control problem and thus its optimal solution. For $\gamma \to 0$, the control cost becomes zero. In this case the cost only consists of the state cost and arbitrary large controls are allowed. We get

$$J^{\alpha}(\theta) = -\alpha \log\left\langle \exp\left(-\frac{1}{\alpha}V(\tau)\right)\right\rangle_{p_{u_\theta}}.$$

This expression is identical to the risk sensitive control cost proposed by Fleming and Sheu (2002); Fleming and McEneaney (1995); van den Broek et al. (2010). Thus, for $\gamma = 0$, the smoothing parameter α controls the risk-sensitivity, resulting in risk seeking objectives for $\alpha > 0$ and risk avoiding objectives for $\alpha < 0$. In the limiting case $\gamma \to \infty$, the problem becomes trivial; the optimal controlled dynamics becomes equal to the uncontrolled dynamics: $p^* \to p_0$, see equation (5), and $u^* \to 0$.

If both parameters α and γ are small, the problem is hard (see Ruiz and Kappen (2017); Thalmeier et al. (2016)) as many samples are needed to estimate the smoothed cost. The problem becomes feasible if either α or γ is increased. Increasing γ however, changes the control problem, while increasing α weakens the effect of smoothing. In the remainder of this article we analyze, first theoretically in Section 4 and then numerically in Section 6, the effect that a finite $\alpha > 0$ has on the iterative optimization of the control u_θ for a fixed value γ.

4. The Effect of Cost Function Smoothing on Policy Optimization

We introduced smoothing as a way to speed up policy optimization compared to a direct optimization of the cost. In this section we analyze policy optimization with and without smoothing and show analytically how smoothing can speed up policy optimization. To simplify notation, we overload $p_{u_\theta} \to \theta$ so that we get $C(p_{u_\theta}) \to C(\theta)$ and $KL(p_{u_{\theta'}}||p_{u_\theta}) \to KL(\theta'||\theta)$.

We use a trust region constraint to robustly optimize the policy (compare Peters et al. (2010); Schulman et al. (2015); Gómez et al. (2014)). There are two options. On the one hand, we can directly optimize the cost C:

Definition 1 *We define the direct update with stepsize $\mathcal{E}$ as an update $\theta \to \theta'$ with $\theta' = \Theta_{\mathcal{E}}^{C}(\theta)$ and*

$$\Theta_{\mathcal{E}}^{C}(\theta) := \underset{\substack{\theta' \\ s.t.\ KL(\theta'||\theta)\leq\mathcal{E}}}{\arg\min} C(\theta').$$

The direct update results in the minimal cost that can be achieved after one single update. We define the optimal one-step cost

$$C_{\mathcal{E}}^{*}(\theta) := \min_{\substack{\theta' \\ s.t.\ KL(\theta'||\theta)\leq\mathcal{E}}} C(\theta').$$

On the other hand we can optimize the smoothed cost J^α:

Definition 2 *We define the smoothed update with stepsize $\mathcal{E}$ as an update $\theta \to \theta'$ with $\theta' = \Theta_{\mathcal{E}}^{J^\alpha}(\theta)$ and*

$$\Theta_{\mathcal{E}}^{J^\alpha}(\theta) := \underset{\substack{\theta' \\ s.t.\ KL(\theta'||\theta)\leq\mathcal{E}}}{\arg\min} J^\alpha(\theta'). \tag{15}$$

While a direct update achieves the minimal cost that can be achieved after a single update, we show below that a smoothed update can result in a faster cost reduction if more than one update step is performed.

Definition 3 *We define the optimal two-step update $\theta \to \Theta' \to \Theta''$ as an update that results in the lowest cost that can be achieved with a two-step update $\theta \to \theta' \to \theta''$ with fixed stepsizes $\mathcal{E}$ and $\mathcal{E}'$ respectively:*

$$\Theta', \Theta'' := \underset{\substack{\theta', \theta'' \\ s.t.\ KL(\theta''||\theta')\leq\mathcal{E}' \\ KL(\theta'||\theta)\leq\mathcal{E}}}{\arg\min} C(\theta'')$$

and the corresponding optimal two-step cost

$$C^*_{\mathcal{E},\mathcal{E}'}(\theta) := \min_{\substack{\theta' \\ s.t.\ KL(\theta'||\theta)\leq\mathcal{E}}} \ \min_{\substack{\theta'' \\ s.t.\ KL(\theta''||\theta')\leq\mathcal{E}'}} C(\theta'')$$

$$= \min_{\substack{\theta' \\ s.t.\ KL(\theta'||\theta)\leq\mathcal{E}}} C\left(\Theta^C_{\mathcal{E}'}(\theta')\right). \tag{16}$$

Figure 1 illustrates how such an optimal two-step update leads to a faster decrease of the cost than two consecutive direct updates.

Theorem 1 Statement 1: *For all $\mathcal{E}$, α there exists an $\mathcal{E}'$, such that a smoothed update with stepsize $\mathcal{E}$ followed by a direct update with stepsize $\mathcal{E}'$ is an optimal two-step update:*

$$\Theta' = \Theta^{J^\alpha}_{\mathcal{E}}(\theta)$$
$$\Theta'' = \Theta^C_{\mathcal{E}'}(\Theta')$$

$$\Rightarrow C\left(\Theta''\right) = C^*_{\mathcal{E},\mathcal{E}'}(\theta)$$

The size of the second step $\mathcal{E}'$ is a function of θ and α.

Statement 2: *$\mathcal{E}'$ is monotonically decreasing in α.*

While it is evident from equation (16) that the second step of the optimal two-step update must be a direct update, the statement that the first step is a smoothed update is non-trivial. We proof this and statement 2 in Appendix F.

Direct updates are myopic and do not take into account successive steps and are thus suboptimal when more than one update is needed. Smoothed updates on the other hand, as we see on Theorem 1, anticipate a subsequent step and minimize the cost that results from this two-step update. Hence smoothed updates favour a greater cost reduction in the future over maximal cost reduction in the current step. The strength of this anticipatory effect depends on the smoothing strength, which is controlled by the smoothing parameter α: For large α, smoothing is weak and the size $\mathcal{E}'$ of this anticipated second step becomes small. Figure 1(B) illustrates that for this case, when $\mathcal{E}'$ becomes small, smoothed updates become more similar to direct updates. In the limiting case $\alpha \to \infty$ the difference between smoothed and direct updates vanishes completely, as $J^\alpha(\theta) \to C(\theta)$ (see Section 3.2).

We expect that also with multiple update steps due to this anticipatory effect, iterating smoothed updates leads to a faster decrease of the cost than iterating direct updates. We will confirm this by numerical studies. Furthermore, we expect that this accelerating effect of smoothing is stronger for smaller values of α. On the other hand, as we will discuss in the next section, for smaller values of α it is harder to accurately perform the smoothed updates. Therefore we expect an optimal performance for an intermediate value of α. Based on this we build an algorithm in the next section that aims to accelerate policy optimization by cost function smoothing.

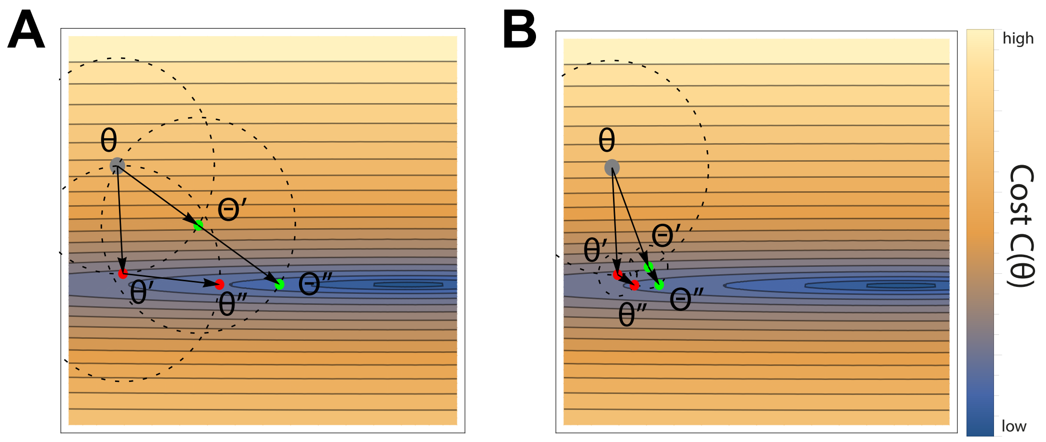

Figure 1: Illustration of optimal two-step updates compared with two consecutive direct updates. Illustrated is a two-dimensional cost landscape $C(\theta)$ parametrized by θ. Dark colors represent low cost, while light colors represent high cost. Green dots indicate the optimal two-step update $\theta \to \Theta' \to \Theta''$ while red dots indicate two consecutive direct updates $\theta \to \theta' \to \theta''$ with $\theta' = \Theta^C_{\mathcal{E}}(\theta)$ and $\theta'' = \Theta^C_{\mathcal{E}'}(\theta')$. The dashed circles indicate trust regions. θ', θ'' and Θ'' are the minimizers of the cost in the trust regions around θ, θ' and Θ' respectively. Θ' is chosen such that the cost $C(\Theta'')$ after the subsequent direct update is minimized. In both panels, the final cost after an optimal two-step update $C(\Theta'')$ is smaller than the final cost after two direct updates $C(\theta'')$. **(A)** Equal sizes of the update steps, $\mathcal{E} = \mathcal{E}'$. **(B)** When the size of the second step becomes small $\mathcal{E}' \ll \mathcal{E}$, the smoothed update $\theta \to \Theta'$ becomes more similar to the direct update $\theta \to \theta'$.

5. Numerical Method

In this section we develop an algorithm that takes a parametrized control function u_θ with initial parameters θ_0 and updates these parameters in each iteration n using smoothed updates.

5.1 Smoothed and Direct Updates Using Natural Gradients

So far we have specified the smoothed updates $\theta_{n+1} = \Theta^{J^\alpha}_{\mathcal{E}}(\theta_n)$ (15) in an abstract manner and left open how to perform this optimization step. To compute an explicit expression we introduce a Lagrange multiplier β and express the constraint optimization (15) as an unconstrained optimization

$$\theta_{n+1} = \arg\min_{\theta'} J^\alpha(\theta') + \beta KL(\theta'||\theta_n) \tag{17}$$

Following Schulman et al. (2015) we assume that the trust region size $\mathcal{E}$ is small. For small $\mathcal{E} \ll 1$ we get $\beta \gg 1$ and can expand $J^\alpha(\theta')$ to first and $KL(\theta'||\theta_n)$ to second order (see

Appendix G for the details). This gives

$$\theta_{n+1} = \theta_n - \beta^{-1} F^{-1} \left.\nabla_{\theta'} J^{\alpha}(\theta')\right|_{\theta'=\theta_n}, \tag{18}$$

a natural gradient update with the Fisher-matrix $F = \left.\nabla_\theta \nabla_\theta^T KL(\theta' || \theta_n)\right|_{\theta'=\theta_n}$ (we use the conjugate gradient method to approximately compute the natural gradient for high dimensional parameter spaces. See Appendix J or Schulman et al. (2015) for details). The parameter β is determined using a line search such that[3]

$$KL(\theta_n || \theta_{n+1}) = \mathcal{E}. \tag{19}$$

Note that for direct updates this derivation is the same, just replace J^α by C.

5.2 Reliable Gradient Estimation Using Adaptive Smoothing

To compute smoothed updates using equation (18) we need the gradient of the smoothed cost. We assume full parametrization and use equation (14), which can be estimated using N weighted samples drawn from the distribution p_{u_θ}:

$$\nabla_\theta J^\alpha(\theta) \approx \alpha \sum_{i=1}^{N} w^i \nabla_\theta \log p_{u_\theta}(\tau^i), \tag{20}$$

with weights given by

$$w^i = \frac{1}{\tilde{Z}} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right), \qquad \tilde{Z} = \sum_{i=1}^{N} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right).$$

The variance of this estimator depends sensitively on the entropy of the weights

$$H_N(w) = -\sum_{i=1}^{N} w^i \log(w^i).$$

If the entropy is low, the total weight is concentrated on a few particles. This results in a poor gradient estimator where only a few of the particles actually contribute. This concentration is dependent on the smoothing parameter α: for small α, the weights are very concentrated in a few samples, resulting in a large weight-entropy and thus a high variance of the gradient estimator. As small α corresponds to strong smoothing, we want α to be as small as possible, but large enough to allow a reliable gradient estimation. Therefore, we set a bound to the weight entropy $H_N(w)$. To get a bound that is independent of the number of samples N, we use that in the limit of $N \to \infty$ the weight entropy is monotonically related to the KL-divergence $KL(p^*_{\alpha,u_\theta} || p_{u_\theta})$

$$KL(p^*_{\alpha,u_\theta} || p_{u_\theta}) = \lim_{N\to\infty} \log N - H_N(w)$$

3. For practical reasons, we reverse the arguments of the KL-divergence, since it is easier to estimate it from samples drawn from the first argument. For very small values, the KL is approximately symmetric in its arguments. Also, the equality in (19) differs from Schulman et al. (2015), which optimizes a value function *within* the trust region, e.g., $KL(\theta_n || \theta_{n+1}) \leq \mathcal{E}$.

(see Appendix I). This provides a method for choosing α *independently of the number of samples*: we set the constraint $KL(p^*_{\alpha,u_\theta}||p_{u_\theta}) \leq \Delta$ and determine the smallest α that satisfies this condition using a line search. Large values of Δ correspond to small values of α (see Appendix H) and therefore strong smoothing, we thus call Δ the *smoothing strength.*

5.3 Formulating a Model-Free Algorithm

We can compute the gradient (20) and the KL-divergence while treating the dynamical system as a black-box. For this we write the probability distribution p_{u_θ} over trajectories τ as a Markov process:

$$p_{u_\theta}(\tau) = \prod_{0<t<T} p_{u_\theta}(x_{t+dt}|x_t, t),$$

where the product runs over the time t, which is discretized with time step dt. We define the noisy action $a_t = u(x_t, t) + \xi_t$ and formulate the Markov transitions $p_{u_\theta}(x_{t+dt}|x_t)$ for the dynamical system (1) as

$$p_{u_\theta}(x_{t+dt}|x_t) = \delta\left(x_{t+dt} - \mathcal{F}\left(x_t, a_t, t\right)\right) \cdot \pi_\theta(a_t|t, x_t),$$

with $\delta(\cdot)$ the Dirac delta function. This splits the transitions up into the deterministic dynamical system $\mathcal{F}\left(x_t, a_t, t\right)$[4] and a Gaussian policy $\pi_\theta(a_t|t, x_t) \sim \mathcal{N}\left(a_t|u_\theta(x_t, t), \frac{\nu}{dt}\right)$ with mean $u_\theta(x_t, t)$ and variance $\frac{\nu}{dt}$. Using this we get a simplified expression for the gradient of the smoothed cost (20) that is independent of the system dynamics, given samples drawn from the controlled system p_{u_θ}:

$$\nabla_\theta J^\alpha(\theta) \approx \alpha \sum_{i=1}^{N} \sum_{0<t<T} w^i \nabla_\theta \log \pi_\theta(a_t^i|t, x_t^i).$$

Similarly we obtain an expression for the estimator of the KL divergence

$$KL(\theta_n||\theta_{n+1}) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{0<t<T} \log \frac{\pi_{\theta_n}(a_t^i|t, x_t^i)}{\pi_{\theta_{n+1}}(a_t^i|t, x_t^i)}.$$

With this we formulate ASPIC (Algorithm 1), a model-free algorithm which optimizes the parametrized policy π_θ by iteratively drawing samples from the controlled system.

6. Numerical Experiments

We now analyze empirically the convergence speed of policy optimization with and without smoothing and show that smoothing accelerates convergence. For the optimization with smoothing, we use ASPIC (Algorithm 1) and for the optimization without smoothing, we use a version of ASPIC where we replaced the gradient of the smoothed cost with the gradient of the cost itself. We first consider a simple linear-quadratic (LQ) control problem and then focus on non-linear control tasks, for which we analyze the dependence of ASPIC on the hyper-parameters. We also compare ASPI to other related RL algorithms. Further details about the numerical experiments are found in Appendix L.

4. Using the Euler method, we get $\mathcal{F}\left(x_t, a_t\right) = \left(x_t + dt \cdot \left(f(x_t, t) + g(x_t, t)a_t\right)\right)$.

Algorithm 1 ASPIC - Adaptive Smoothing for Path Integral Control

```
Require: State cost function $V(x,t)$
         control cost parameter $\gamma$
         base policy that defines uncontrolled dynamics $\pi_0$
         real system or simulator to compute dynamics using a parametrized policy $\pi_\theta$
         trust region sizes $\mathcal{E}$
         smoothing strength $\Delta$
         number of samples per iteration $N$
  initialize $\theta_0$
  $n = 0$
  repeat
    draw state trajectories $\tau^i, i = 1, \dots, N$, using parametrized policy $\pi_{\theta_n}$
    for each sample $i$ compute $S^{\gamma}_{p_{u_{\theta_n}}}(\tau^i) = \sum_{0<t<T} V(x^i_t, t) + \gamma \log \frac{\pi_{\theta_n}(a^i_t|t, x^i_t)}{\pi_0(a^i_t|t, x^i_t)}$
    {Find minimal $\alpha$ such that $KL \leq \Delta$}
    $\alpha \leftarrow 0$
    repeat
      increase $\alpha$
      $S^i_\alpha \leftarrow S^{\gamma}_{p_{u_{\theta_n}}}(\tau^i) \cdot \frac{1}{\gamma+\alpha}$
      compute weights $w_i \leftarrow \exp(-S^i_\alpha)$
      normalize weights $w_i \leftarrow \frac{w_i}{\sum_i (w_i)}$
      compute sample size independent weight entropy $KL \leftarrow \log N + \sum_i w_i \log(w_i)$
    until $KL \leq \Delta$
    {whiten the weights}
    $\hat{w}_i \leftarrow \frac{w_i - \mathrm{mean}(w_i)}{\mathrm{std}(w_i)}$
    {compute the gradient on the smoothed cost}
    $g \leftarrow \sum_i \sum_t \hat{w}_i \frac{\partial}{\partial \theta} \log \pi_\theta(a^i_t|t, x^i_t)\big|_{\theta=\theta_n}$
    {compute Fisher matrix}
    use conjugate gradient to approximate the natural gradient $g_F = F^{-1} g$ (Appendix J)
    do line search to compute step size $\eta$ such $KL(\theta_n||\theta_{n+1}) = \mathcal{E}$
    update parameters $\theta_{n+1} \leftarrow \theta_n + \eta \cdot g_F$
    $n = n + 1$
  until convergence
```

6.1 A Simple Linear-Quadratic Control Problem: Brownian Viapoints

We analyze the convergence speed for different values of the smoothing strength Δ in the task of controlling a one-dimensional Brownian particle

$$\dot{x} = u(x,t) + \xi. \tag{21}$$

We define the state cost as a quadratic penalty for deviating from the viapoints x_i at the different times t_i: $V(x,t) = \sum_i \delta(t - t_i) \frac{(x-x_i)^2}{2\sigma^2}$ with $\sigma = 0.1$. As a parametrized controller we use a time varying linear feedback controller, i.e., $u_\theta(x,t) = \theta_{1,t} x + \theta_{0,t}$. This controller fulfils the requirement of full parametrization for this task (see Appendix K). For further details of the numerical experiment see Appendix L.1.

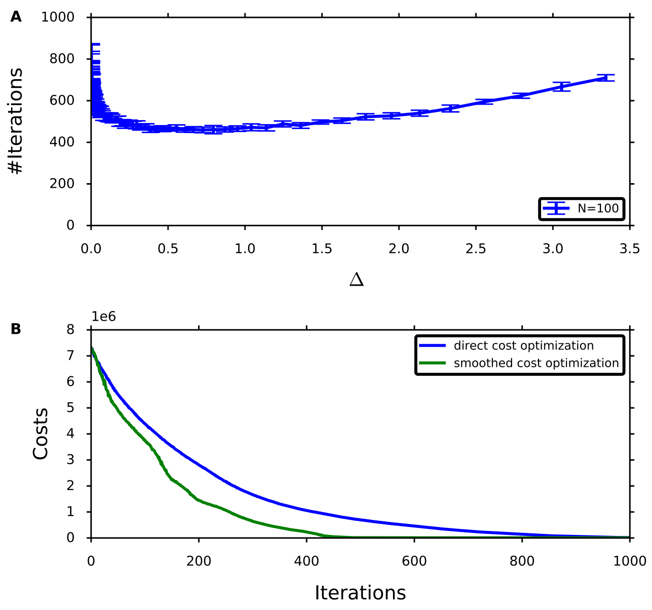

Figure 2: LQ control problem: Brownian viapoints. For each iteration we used $N = 100$ rollouts to compute the gradient. **(A)** Number of iterations needed for the cost to cross a threshold $C \leq 2 \cdot 10^4$ versus the smoothing strength Δ. For $\Delta = 0$ there is no smoothing. Increasing the smoothing strength results in a faster decrease of the cost; when Δ is increased further the performance decreases again. Errorbars denote mean and standard deviation over 10 runs of the algorithm. **(B)** Cost versus the iterations of the algorithm. Direct optimization of the cost exhibits a slower convergence rate than optimization of the smoothed cost with $\Delta = 0.2 \log 100$.

We apply ASPIC to this control problem and compare its performance for different sizes of the smoothing strength Δ (see Figure 2). The results confirm our expectations from our theoretical analysis (sections 4 and 5.2). As predicted by theory we observe an acceleration of the policy optimization when smoothing is switched on. This acceleration becomes more pronounced when Δ is increased, which we attribute to an increase of the anticipatory effect of the smoothed updates as smoothing becomes stronger (see Section 4). When Δ is too large the performance of the algorithm deteriorates again, in agreement with our discussion of gradient estimation problems that arise for strong smoothing (see Section 5.2).

6.2 Nonlinear Control Problems

We now consider non-linear control problems, which violate the full parametrization assumption. We focus on the pendulum swing-up task, the Acrobot task, and a 2D Walker task. The latter was simulated using the OpenAI gym (Brockman et al., 2016). For pendulum swing-up and the Acrobot tasks we used time-varying linear feedback controllers, whereas for the 2D Walker task we parametrized the control u_θ using a neural network.

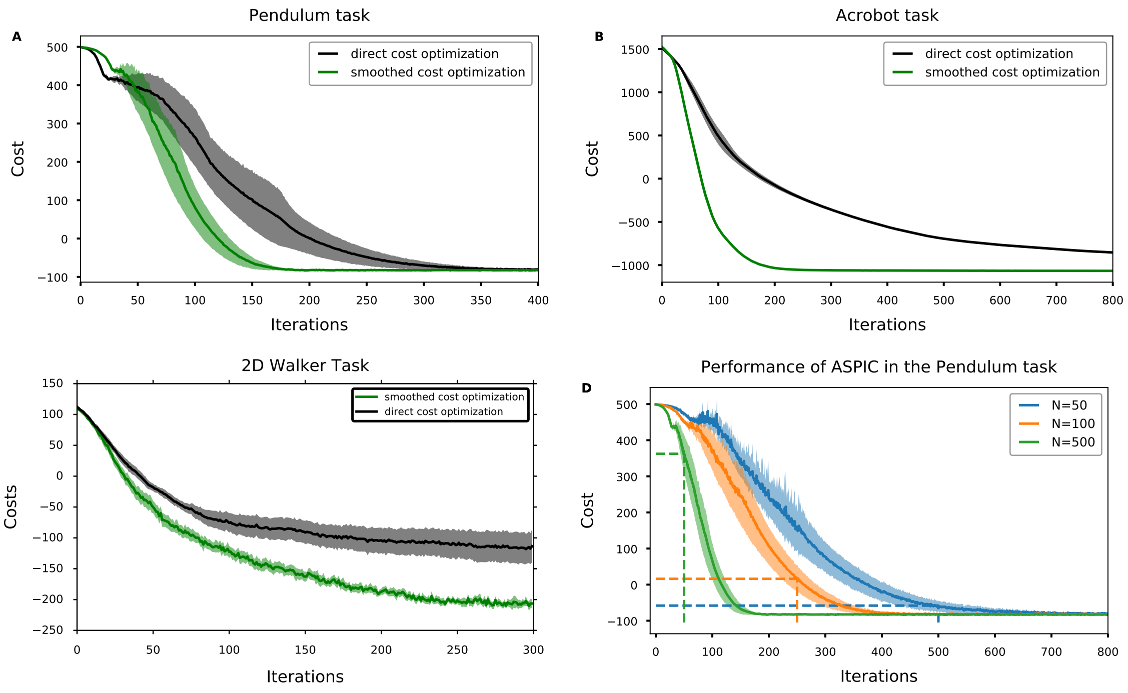

Figure 3: **(A-C)** Smoothed cost optimization (ASPIC) exhibits faster convergence than direct cost optimization in a variety of tasks. Plots show mean and standard deviation of the cost per iteration for 10 runs of the algorithm. In all tasks except 2D Walker, we used $N = 500$ rollouts and a trust region size $\mathcal{E} = 0.1$. For ASPIC, the smoothing strength was set to $\Delta = 0.5$. In the 2D Walker task **(C)** we used $N = 100$ rollouts and $\Delta = 0.05 \log N$. **(D)** Performance as a function of the number of iterations for different values of $N \in \{50, 100, 500\}$. Dashed lines denote the solution for a total fixed budget of 25K rollouts, i.e., 500, 250, and 50 iterations, respectively. In this case, $N = 50$ achieves near optimal performance whereas using larger values of N leads to worse solutions.

Further details are given in Appendix L.2 for the pendulum, L.3 for the Acrobot and L.4 for the 2D Walker.

Convergence Rate of Policy Optimization

Figure 3(A-C) shows the comparison of ASPIC algorithm with smoothing against direct-cost optimization. In all three tasks, smoothing improves the convergence rate of policy optimization. Smoothed cost optimization requires less iterations to achieve the same cost reduction as direct cost optimization, with only a negligible amount of additional computational steps that do not depend on the complexity of the simulation runs.

We can thus conclude that even in cases when the parametrized controller does not strictly meet the requirement of full parametrization, a strong performance boost can also be achieved.

Dependence on the Number of Rollouts per Iteration N

We now analyze the dependence of the performance of ASPIC on the number of rollouts per iteration N. In general, using larger values of N allows for more reliable gradient estimates and achieves convergence in fewer iterations. However, too large N may be inefficient and lead to suboptimal solutions in the presence of a fixed budget of rollouts.

Figure 3(D) illustrates this trade-off in the Pendulum swing-up task for three values of N, including the previous one $N = 500$. For a total budget of 25K rollouts (dashed lines) the lowest value of $N = 50$ achieves near optimal performance and is preferable to the other choices, despite resulting in higher variance estimates and requiring more iterations until convergence.

Interplay Between Smoothing Strength Δ and Trust Region Size $\mathcal{E}$

To understand better the relation between the smoothing strength and the trust region sizes, we analyze empirically the performance of ASPIC as a function of both Δ and $\mathcal{E}$ parameters. We focus on the Acrobot task and in the setting of $N = 500$ and intermediate smoothing strength, when smoothing is most beneficial.

Figure 4 shows the cost as a function of Δ and $\mathcal{E}$ averaged over the first 500 iterations of the algorithm, and for 10 different runs. Larger (averaged) costs correspond runs where the algorithm fails to converge. Conversely, the lower the cost, the fastest the convergence. In general, larger values of $\mathcal{E}$ lead to faster convergence. However, the convergence is less stable for smaller values of Δ. For stronger smoothing, the algorithm is less sensitive to $\mathcal{E}$.

Comparison with other model-free RL algorithms

We finish this experimental analysis with a comparison between ASPIC and other related model-free RL algorithms. We consider trajectory-based algorithms that use the return of the entire trajectories, instead of evaluating the gradient at every state within a trajectory. This setting allows us to disentangle the effect of smoothing in the optimization from other factors, such as the use of state-dependent baselines. In particular, we compare ASPIC with following methods:

Policy Gradient (PG): this is the vanilla policy gradient method (Sutton et al., 2000), and can be seen as direct cost optimization without a trust region constraint.

Policy Gradient with a trust region constraint (PG-TR): this is again a direct cost optimization method, similar to natural gradient descent (Kakade, 2002), with the main difference that it can perform multiple gradient steps inside the trust region.

Trajectory-based Trust Region Policy Optimization (TRPO-TB): we consider the original TRPO (Schulman et al., 2015) without the state-dependent baseline, that is, computing the gradient estimate over trajectories instead of state-action pairs. We use the same controller architecture and hyper-parameters as in the original paper.

We evaluate the performance of these algorithms on a set of six tasks from Pybullet, an open source real-time physics engine (see Appendix L.5 for more details). Figure 5

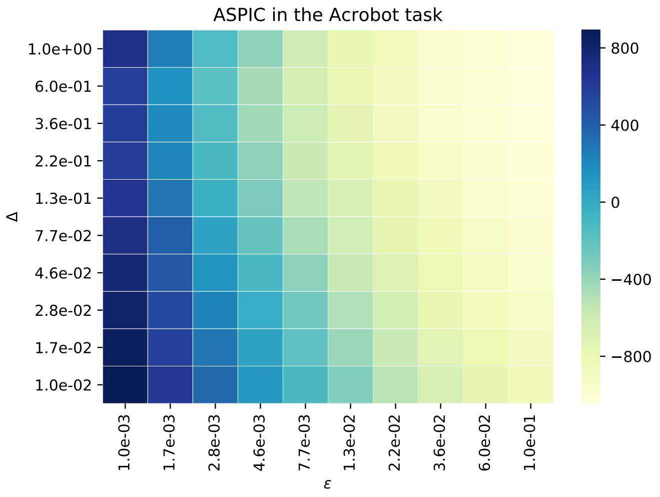

Figure 4: Solution cost as a function of the smoothing strength Δ and the trust region size $\mathcal{E}$ in the Acrobot task. Shown is the cost averaged over the first 500 iterations of the algorithm, and for 10 different runs. Blue indicates failure to convergence. White indicates the solutions which converged fastest.

shows the results. For the six tasks, ASPIC systematically converges faster than the other methods. Remarkably, in the tasks with higher dimensions (Walker2D and Half-Cheetah) the differences in performance is more pronounced, indicating that ASPIC can also scale well to higher-dimensional problems.

7. Discussion

For path integral control problems the optimal control policy can serve as a guidepost for policy optimization. This is used in the PICE algorithm (Kappen and Ruiz, 2016). One might hope that a representation of optimal control can help to find a parametrized policy and surpass the more general approach of direct cost optimization. In practice however, the PICE algorithm suffers from problems with sample efficiency (Ruiz and Kappen, 2017). We introduced a smoothing technique using an inf-convolution which preserves global minima. Remarkably, for path integral control problems, minimization in the inf-convolution can be solved analytically. We used this result to interpolate between direct cost optimization and the PICE method. In between these extremes we have found a method that is superior to direct cost optimization while remaining feasible.

We conducted a theoretical analysis of the optimization of smoothed cost-functions and showed that minimizing the smoothed cost can accelerate policy optimization by having less myopic updates that favour stronger cost reduction in subsequent updates over immediate cost reduction in the current step. This prediction is confirmed by our numerical experiments, which show that smoothing the cost accelerates the convergence of policy optimization. While the theoretical analysis only makes statements for optimizations with

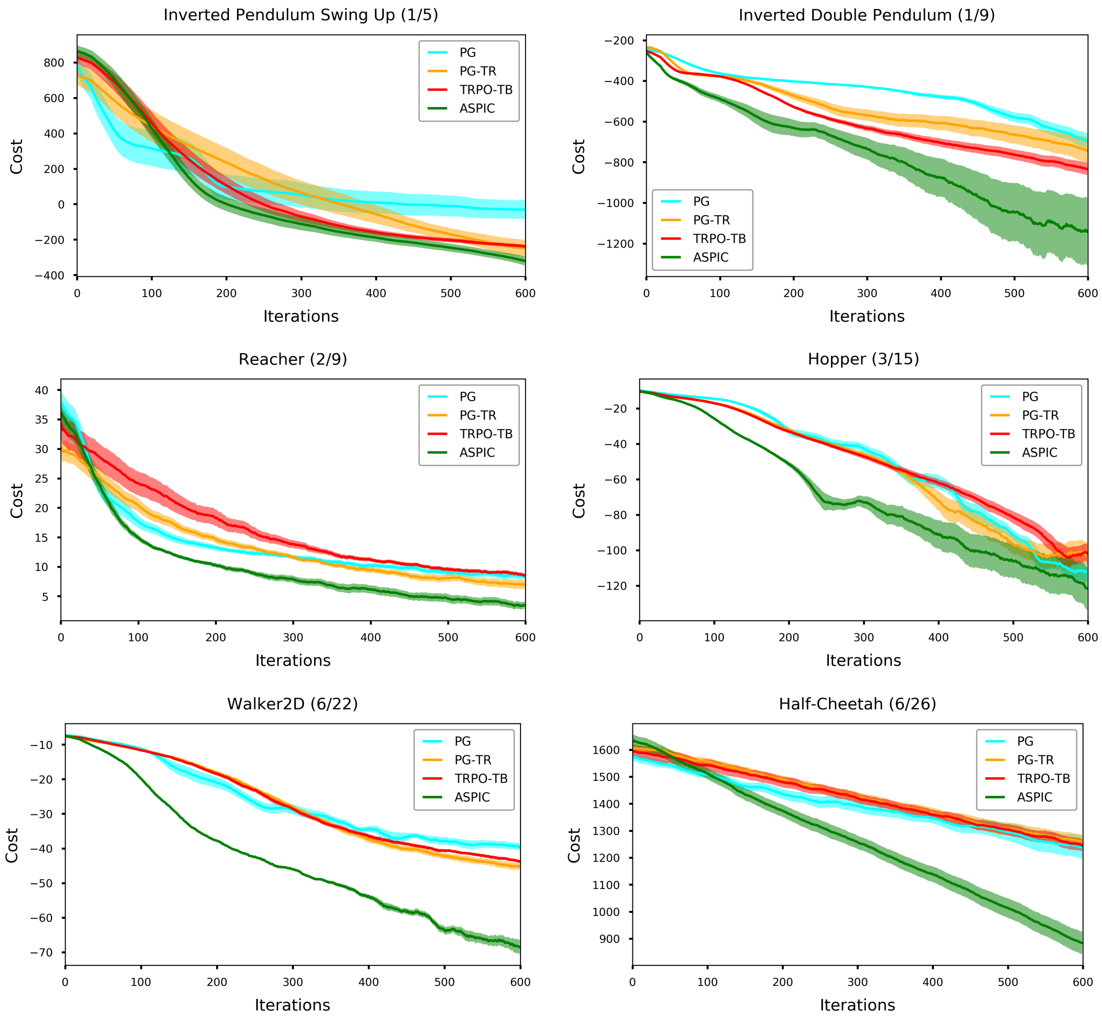

Figure 5: Pybullet experiments: comparison between ASPIC and other related methods (see main text for details). Curves show mean and standard deviation of the cost per iteration for 5 different runs. The panel titles show the task name as well as the number of action/state dimensions in parenthesis. ASPIC converges faster than the other methods in all tasks, specially in high-dimensional ones.

a total of two update steps, the numerical experiments show that the acceleration effect persists when more than two update steps are performed.

Because direct cost optimization and the PICE method are recovered in the limits of weak and strong smoothing respectively, we examined smoothed cost optimization for different levels of smoothing. The result shows in both limits the performance of the algorithm deteriorates. For weak smoothing this can be explained with the disappearance of the accelerating effect that is caused by smoothing. The deterioration of performance for strong smoothing may be attributed to the higher variance of the sample weights that result in gradient estimation problems which also appear in PICE (Ruiz and Kappen, 2017). These problems appear for strong smoothing, while the accelerative effect stays noticeable when smoothing is weak.

The explanatory power of the theoretical results regarding the numerical experiments is limited through the fact that our derivation of the smoothed cost and its gradient requires an assumption on the representational power of the parametrized control policy. In principle, a universal function approximator, like an infinitely large neural network, would be sufficient to satisfy this *full parametrization* assumption. However in practice, where we have to rely on function approximators with a finite number of parameters, this is difficult to obtain. Nevertheless, the qualitative behaviour, that smoothing speeds up policy optimization, persists despite this deviation of the numerical methods from the theoretical assumptions.

To conduct the numerical studies we used the algorithm ASPIC that we developed based on our theoretical results. ASPIC uses robust updates and an adaptive adjustment of the smoothing parameter to ensure that the gradient on the smoothed cost stays computable with a finite amount of samples. This procedure bears similarities to an adaptive annealing scheme, with the smoothing parameter playing the role of an artificial temperature. In contrast to classical annealing schemes, such as simulated annealing, changing the smoothing parameter does not change the optimization target: the minimum of the smoothed cost remains the optimal control solution for all levels of smoothing.

In the weak smoothing limit, ASPIC directly optimizes the cost using trust region constrained updates, similar to the TRPO algorithm (Schulman et al., 2015). TRPO differs from ASPIC's weak smoothing limit by additionally using certain variance reduction techniques for the gradient estimator: they replace the stochastic cost in the gradient estimator by the easier-to-estimate advantage function, which has a state dependent baseline and only takes into account future expected cost. Since this depends on the linearity of the gradient in the stochastic cost and this dependence is non-linear for the gradient of the smoothed cost, we cannot directly incorporate these variance reduction techniques in ASPIC.

In the strong smoothing limit ASPIC becomes a version of PICE (Kappen and Ruiz, 2016) that—unlike the plain PICE algorithm—uses a trust region constraint to achieve robust updates. The gradient estimation problem that appears in the PICE algorithm was previously addressed in Ruiz and Kappen (2017): they proposed a heuristic that allows to reduce the variance of the gradient estimator by adjusting the particle weights used to compute the policy gradient. Ruiz and Kappen (2017) introduced this heuristic as an ad hoc fix of the sampling problem and the adjustment of the weights introduces a bias with possible unknown side effects. Our study sheds a new light on this, as adjusting the particle weights corresponds to a change of the smoothing parameter in our case. The theoretical results we derived can however not directly be transferred to Ruiz and Kappen (2017), since we assume the use of trust regions to bound the updates of the policy optimization.

Especially when samples are expensive to compute it is important to squeeze out as much information from them as possible. We showed that for path integral control problems a smoothed version of the cost function and its gradient can directly be computed from the samples and allows to make less myopic policy updates than cost-greedy methods (like TRPO and PIREPS) and thereby accelerate convergence. We believe this can potentially be useful for variational inference in other areas of machine learning (Arenz et al., 2018). To fully benefit from this, it is important future work to develop variance reduction techniques for the gradient of the smoothed cost similar to the techniques already used for methods that directly optimize the cost. A possible way to achieve this would be control variates that are tailored to the gradient estimator of the smoothed cost (Papini et al., 2018; Ranganath et al., 2014; Glasserman, 2013). Another important future work is to develop a deeper understanding of the full parametrization assumption and how its violation impacts the performance of the algorithm. Minimizing this impact might be an important lever to boost the performance of policy optimization for path integral control problems.

Acknowledgments

The research leading to these results has received funding from “La Caixa” Banking Foundation. Vicenç Gómez is supported by the Ramon y Cajal program RYC-2015-18878 (AEI / MINEICO / FSE, UE).

Appendix A. Derivation of the Policy Gradient

Here we derive equation (4). We write $C(p_{u_\theta}) = \langle S^\gamma_{u_\theta}(\tau)\rangle_{p_{u_\theta}}$, with $S^\gamma_{u_\theta}(\tau) := V(\tau) + \gamma \log \frac{p_{u_\theta}(\tau)}{p_0(\tau)}$ and take the derivative of equation (2):

$$\nabla_\theta \left\langle S^\gamma_{u_\theta}(\tau)\right\rangle_{p_{u_\theta}} = \nabla_\theta \left\langle V(\tau) + \gamma \log \frac{p_{u_\theta}(\tau)}{p_0(\tau)} \right\rangle_{p_{u_\theta}}$$

Now we introduce the importance sampler $p_{u_{\theta'}}$ and correct for it.

$$\nabla_\theta \left\langle S^\gamma_{u_\theta}(\tau)\right\rangle_{p_{u_\theta}} = \nabla_\theta \left\langle \frac{p_{u_\theta}(\tau)}{p_{u_{\theta'}}(\tau)} \left(V(\tau) + \gamma \log \frac{p_{u_\theta}(\tau)}{p_0(\tau)} \right) \right\rangle_{p_{u_{\theta'}}}$$

This is true for all θ' as long as $p_{u_\theta}(\tau)$ and $p_{u_{\theta'}}(\tau)$ are absolutely continuous to each other. Taking the derivative we get:

$$\begin{aligned}
\nabla_\theta \left\langle S^\gamma_{u_\theta}(\tau)\right\rangle_{p_{u_\theta}} &= \left\langle \frac{\nabla_\theta p_{u_\theta}(\tau)}{p_{u_{\theta'}}(\tau)} \left(V(\tau) + \gamma \log \frac{p_{u_\theta}(\tau)}{p_0(\tau)} \right) \right\rangle_{p_{u_{\theta'}}} + \left\langle \frac{p_{u_\theta}(\tau)}{p_{u_{\theta'}}(\tau)} \left(\gamma \frac{1}{p_{u_\theta}(\tau)} \nabla_\theta p_{u_\theta}(\tau) \right) \right\rangle_{p_{u_{\theta'}}} \\
&= \left\langle (\nabla_\theta \log p_{u_\theta}(\tau)) \left(V(\tau) + \gamma \log \frac{p_{u_\theta}(\tau)}{p_0(\tau)} \right) \right\rangle_{p_{u_\theta}} + \gamma \nabla_\theta \left\langle \frac{1}{p_{u_{\theta'}}(\tau)} p_{u_\theta}(\tau) \right\rangle_{p_{u_{\theta'}}} \\
&= \left\langle S^\gamma_{u_\theta}(\tau) \nabla_\theta \log p_{u_\theta}(\tau) \right\rangle_{p_{u_\theta}} + \gamma \nabla_\theta \langle 1 \rangle_{p_{u_\theta}} \\
&= \left\langle S^\gamma_{u_\theta}(\tau) \nabla_\theta \log p_{u_\theta}(\tau) \right\rangle_{p_{u_\theta}}.
\end{aligned}$$

Appendix B. Replacing Minimization Over u by Minimization Over p'

Here we show that for

$$J^\alpha(\theta) = \inf_{u'} \alpha KL(p_{u'}||p_{u_\theta}) + \gamma KL(p_{u'}||p_0) + \langle V(\tau)\rangle_{p'} \tag{22}$$

we can replace the minimization over u by a minimization over p' to obtain equation (11). For this, we need to show that the minimizer $p^*_{\alpha,\theta}$ of equation (11) is induced by $u^*_{\alpha,\theta}$, the minimizer of equation (22):

$$p^*_{\alpha,\theta} \equiv p_{u^*_{\alpha,\theta}}.$$

The solution to (11) is given by (see Appendix C)

$$\begin{aligned}
p^*_{\alpha,\theta} &= \frac{1}{Z} p_{u_\theta}(\tau) \exp\left(-\frac{1}{\gamma+\alpha} S^\gamma_{p_{u_\theta}}(\tau) \right) \\
&= \frac{1}{Z} p_{u_\theta}(\tau) \left(\frac{p_0(\tau)}{p_{u_\theta}(\tau)} \right)^{\frac{\gamma}{\gamma+\alpha}} \exp\left(-\frac{1}{\gamma+\alpha} V(\tau) \right).
\end{aligned}$$

We rewrite

$$p_0(\tau) \left(\frac{p_{u_\theta}(\tau)}{p_0(\tau)} \right)^{1-\frac{\gamma}{\gamma+\alpha}} = p_0(\tau) \exp\left(\left(1 - \frac{\gamma}{\gamma+\alpha} \right) \int_0^T \left(\frac{1}{2} u_\theta(x_t,t)^T u_\theta(x_t,t) + u_\theta(x_t,t)^T \xi_t \right) dt \right),$$

where we used the Girsanov theorem (Bierkens and Kappen, 2014; Thijssen and Kappen, 2015) (and set $\nu = 1$ for simpler notation). With $\tilde{u}_\theta(x_t, t) := \left(1 - \frac{\gamma}{\gamma+\alpha}\right) u_\theta(x_t, t)$ this gives

$$\begin{aligned} p_0(\tau)\left(\frac{p_{u_\theta}(\tau)}{p_0(\tau)}\right)^{1-\frac{\gamma}{\gamma+\alpha}} &= p_0(\tau)\exp\left(\int_0^T \left(\frac{1}{2}\tilde{u}_\theta(x_t,t)^T\tilde{u}_\theta(x_t,t) + \tilde{u}_\theta(x_t,t)^T\xi_t\right) dt\right) \cdot \\ &\quad \cdot \exp\left(\int_0^T \left(\frac{1}{2}\frac{\gamma}{\alpha}\tilde{u}_\theta(x_t,t)^T\tilde{u}_\theta(x_t,t)\right) dt\right) \\ &= p_{\tilde{u}_\theta}(\tau)\exp\left(\int_0^T \left(\frac{1}{2}\frac{\gamma}{\alpha}\tilde{u}_\theta(x_t,t)^T\tilde{u}_\theta(x_t,t)\right) dt\right). \end{aligned}$$

So we get

$$p^*_{\alpha,\theta} = \frac{1}{Z}p_{\tilde{u}_\theta}(\tau)\exp\left(\int_0^T \left(\frac{1}{2}\frac{\gamma}{\alpha}\tilde{u}_\theta(x_t,t)^T\tilde{u}_\theta(x_t,t)\right) dt\right)\exp\left(-\frac{1}{\gamma+\alpha}V(\tau)\right).$$

This has the form of an optimally controlled distribution with dynamics

$$\dot{x}_t = f(x_t,t) + g(x_t,t)\left(\tilde{u}_\theta(x_t,t) + \hat{u}(x_t,t) + \xi_t\right) \tag{23}$$

and cost

$$\left\langle \int_0^T \frac{1}{\gamma+\alpha}V(x_t,t) - \frac{1}{2}\frac{\gamma}{\alpha}\tilde{u}_\theta(x_t,t)^T\tilde{u}_\theta(x_t,t)dt + \int_0^T \left(\frac{1}{2}\hat{u}(x_t,t)^T\hat{u}(x_t,t) + \hat{u}(x_t,t)^T\xi_t\right) dt \right\rangle_{p_{\hat{u}}}.$$

This is a path integral control problem with state cost $\int_0^T \frac{1}{\gamma+\alpha}V(x_t,t) - \frac{1}{2}\frac{\gamma}{\alpha}\tilde{u}_\theta(x_t,t)^T\tilde{u}_\theta(x_t,t)dt$ which is well defined with $\tilde{u}_\theta(x_t,t) = \left(1 - \frac{\gamma}{\gamma+\alpha}\right)u_\theta(x_t,t)$.

Let $\hat{u}^*$ be the optimal control of this path integral control problem. Then $p^*_{\alpha,\theta}$ is induced by equation (23) with $\hat{u} = \hat{u}^*$. This is equivalent to say that $p^*_{\alpha,\theta}$ is induced by equation (1). As $p^*_{\alpha,\theta}$ is the density that minimizes equation (11), $\tilde{u}_\theta + \hat{u}^*$ is minimizing equation (22).

Appendix C. Minimizer of the Smoothed Cost

Here we want to proof equation (12):

$$\begin{aligned} p^*_{\alpha,\theta}(\tau) &:= \arg\min_{p'} \alpha KL(p'||p_{u_\theta}) + \left\langle S^\gamma_{p_{u_\theta}}(\tau)\right\rangle_{p'} \\ &= \arg\min_{p'} \left\langle \alpha\log\frac{p'(\tau)}{p_{u_\theta}(\tau)} + V(\tau) + \gamma\log\frac{p'(\tau)}{p_0(\tau)}\right\rangle_{p'}. \end{aligned}$$

For this we take the variational derivative and set it to zero:

$$0 = \frac{\delta}{\delta p'(\tau)}\left\langle \alpha\log\frac{p'(\tau)}{p_{u_\theta}(\tau)} + V(\tau) + \gamma\log\frac{p'(\tau)}{p_0(\tau)} + \kappa\right\rangle_{p'}\Bigg|_{p'=p^*_{\alpha,\theta}},$$

where we added a Lagrange multiplier κ to ensure normalization. We get

$$0 = \alpha\log\frac{p'(\tau)}{p_{u_\theta}(\tau)} + V(\tau) + \gamma\log\frac{p'(\tau)}{p_0(\tau)} + \kappa\Bigg|_{p'=p^*_{\alpha,\theta}},$$

from which follows

$$
\begin{aligned}
p^*_{\alpha,\theta}(\tau) &= \exp\left(\frac{\kappa}{\alpha+\gamma}\right) p_{u_\theta}(\tau)^{\frac{\alpha}{\alpha+\gamma}} p_0(\tau)^{\frac{\gamma}{\alpha+\gamma}} \exp\left(-\frac{1}{\gamma+\alpha}V(\tau)\right) \\
&= \exp\left(\frac{\kappa}{\alpha+\gamma}\right) p_{u_\theta}(\tau) \exp\left(-\frac{1}{\gamma+\alpha}V(\tau) - \frac{\gamma}{\alpha+\gamma}\log\frac{p_{u_\theta}(\tau)}{p_0(\tau)}\right) \\
&= \exp\left(\frac{\kappa}{\alpha+\gamma}\right) p_{u_\theta}(\tau) \exp\left(-\frac{1}{\gamma+\alpha}S^{\gamma}_{p_{u_\theta}}(\tau)\right),
\end{aligned}
$$

where κ is chosen such that the distribution is normalized.

Appendix D. Derivation of the Gradient of the Smoothed Cost Function

Here we derive equation (14) by taking the derivative of equation (13):

$$
\begin{aligned}
\nabla_\theta J^\alpha(\theta) &= -(\gamma+\alpha)\nabla_\theta \log\left\langle \exp\left(-\frac{1}{\gamma+\alpha}\left(V(\tau)+\gamma\log\frac{p_{u_\theta}(\tau)}{p_0(\tau)}\right)\right)\right\rangle_{p_{u_\theta}} \\
&= -\frac{\gamma+\alpha}{Z^\alpha_{p_{u_\theta}}}\nabla_\theta \left\langle \exp\left(-\frac{1}{\gamma+\alpha}\left(V(\tau)+\gamma\log\frac{p_{u_\theta}(\tau)}{p_0(\tau)}\right)\right)\right\rangle_{p_{u_\theta}}.
\end{aligned}
$$

Now we introduce the importance sampler $p_{u_{\theta'}}$ and correct for it.

$$
\begin{aligned}
\nabla_\theta J^\alpha(\theta) &= -\frac{\gamma+\alpha}{Z^\alpha_{p_{u_\theta}}}\nabla_\theta \left\langle \frac{p_{u_\theta}(\tau)}{p_{u_{\theta'}}(\tau)} \exp\left(-\frac{1}{\gamma+\alpha}\left(V(\tau)+\gamma\log\frac{p_{u_\theta}(\tau)}{p_0(\tau)}\right)\right)\right\rangle_{p_{u_{\theta'}}} \\
&= -\frac{\gamma+\alpha}{Z^\alpha_{p_{u_\theta}}}\nabla_\theta \left\langle \frac{p_0(\tau)^{\frac{\gamma}{\gamma+\alpha}}}{p_{u_{\theta'}}(\tau)} \left(p_{u_\theta}(\tau)\right)^{\frac{\alpha}{\gamma+\alpha}} \exp\left(-\frac{1}{\gamma+\alpha}V(\tau)\right)\right\rangle_{p_{u_{\theta'}}} \\
&= -\frac{\alpha}{Z^\alpha_{p_{u_\theta}}} \left\langle \frac{1}{p_{u_{\theta'}}(\tau)} \left(\frac{p_{u_\theta}(\tau)}{p_0(\tau)}\right)^{-\frac{\gamma}{\gamma+\alpha}} \exp\left(-\frac{1}{\gamma+\alpha}V(\tau)\right)\nabla_\theta p_{u_\theta}\right\rangle_{p_{u_{\theta'}}} \\
&= -\frac{\alpha}{Z^\alpha_{p_{u_\theta}}} \left\langle \exp\left(-\frac{1}{\gamma+\alpha}S^{\gamma}_{p_{u_\theta}}(\tau)\right)\nabla_\theta \log p_{u_\theta}(\tau)\right\rangle_{p_{u_\theta}}.
\end{aligned}
$$

Appendix E. Global Minimum is Preserved Under Full Parametrization

Here we show that smoothing leaves the global optimum of the cost $C(p_{u_\theta})$ invariant.
Proof As $KL(p_{u_{\theta'}}||p_{u_\theta}) \geq 0$ we have that

$$
J^\alpha(\theta) = \inf_{\theta'} C(p_{u_{\theta'}}) + \alpha KL(p_{u_{\theta'}}||p_{u_\theta}) \geq \inf_{\theta'} C(p_{u_{\theta'}}) = C(p_{u_{\theta^*}}).
$$

To show that the global minimum θ^* of C is also the global minimum of J^α, it is thus sufficient to show that

$$
J^\alpha(\theta^*) \leq C(p_{u_{\theta^*}}).
$$

We have

$$J^{\alpha}(\theta^*) = \inf_{\theta'} C(p_{u_{\theta'}}) + \alpha KL(p_{u_{\theta'}}||p_{u_{\theta^*}}).$$

Using that the minimum of a sum of terms is never larger than the sum of the minimum of terms, we get

$$\begin{aligned} J^{\alpha}(\theta^*) &\leq \left(\inf_{\theta'} C(p_{u_{\theta'}})\right) + \left(\inf_{\theta'} \alpha KL(p_{u_{\theta'}}||p_{u_{\theta^*}})\right) \\ &= C(p_{u_{\theta^*}}) + \alpha KL(p_{u_{\theta^*}}||p_{u_{\theta^*}}) \\ &= C(p_{u_{\theta^*}}). \end{aligned}$$

■

We also expect local minima to be also preserved for large-enough smoothing parameter α. This would correspond to small time smoothing by the associated Hamilton-Jacobi partial differential equation (Chaudhari et al., 2018).

Appendix F. Smoothing Theorem

Here we proof Theorem 1. We split the proof into three subsections: in the first subsection, we state and proof a lemma that we need to proof statement 1. In the second subsection, we proof statement 1 and in the third subsection, we proof statement 2.

F.1 Lemma

Lemma 2 *With $\theta^*_{\alpha,\theta}$ defined as in equation (9) and $\mathcal{E}_\alpha(\theta) = KL(\theta^*_{\alpha,\theta}||\theta)$ we can rewrite $J^\alpha(\theta)$:*

$$J^{\alpha}(\theta) = C\left(\Theta^{C}_{\mathcal{E}'}(\theta)\right)\big|_{\mathcal{E}'=\mathcal{E}_\alpha(\theta)} + \alpha\mathcal{E}_\alpha(\theta). \tag{24}$$

Proof With the definition of $\theta^*_{\alpha,\theta}$ as the minimizer of $C(\theta') + \alpha KL(\theta'||\theta)$ (see (9)) we have

$$\begin{aligned} J^{\alpha}(\theta) &= C\left(\theta^*_{\alpha,\theta}\right) + \alpha KL(\theta^*_{\alpha,\theta}||\theta) \\ &= C\left(\theta^*_{\alpha,\theta}\right) + \alpha\mathcal{E}_\alpha(\theta). \end{aligned}$$

What is left to show is that

$$\theta^*_{\alpha,\theta} \equiv \Theta^{C}_{\mathcal{E}_\alpha(\theta)}(\theta).$$

As $\Theta^{C}_{\mathcal{E}_\alpha(\theta)}(\theta)$ is the minimizer of the cost C within the trust region defined by $\{\theta' : KL(\theta'||\theta) \leq \mathcal{E}_\alpha(\theta)\}$ we have to show that

1. $\theta^*_{\alpha,\theta}$ lies within this trust region,

2. $C(\theta^*_{\alpha,\theta})$ is a minimizer of the cost C within this trust region.

The first point is trivially true as $KL(\theta^*_{\alpha,\theta}||\theta) = \mathcal{E}_\alpha(\theta)$ by definition. Hence $\theta^*_{\alpha,\theta}$ lies at the boundary of this trust region and therefore in it, as the boundary belongs to the trust region. The second point we proof by contradiction: Given $\theta^*_{\alpha,\theta}$ is not minimizing the cost within the trust region, then there exists a $\hat{\theta}$ with $C(\hat{\theta}) < C(\theta^*_{\alpha,\theta})$ and $KL(\hat{\theta}||\theta) \leq \mathcal{E}_\alpha(\theta) = KL(\theta^*_{\alpha,\theta}||\theta)$. Therefore it must hold that

$$C(\hat{\theta}) + \alpha KL(\hat{\theta}||\theta) < C(\theta^*_{\alpha,\theta}) + \alpha KL(\theta^*_{\alpha,\theta}, \theta)$$

which is a contradiction, as $\theta^*_{\alpha,\theta}$ is the minimizer of $C(\theta') + \alpha KL(\theta'||\theta)$. ■

F.2 Proof of Statement 1

Here we show that for every α and θ there exists an $\mathcal{E}' = \mathcal{E}^*_\alpha(\theta)$ such that

$$C\left(\Theta^C_{\mathcal{E}'}\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right)\right)\Big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} = C^*_{\mathcal{E},\mathcal{E}'}\Big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)}. \tag{25}$$

Proof As $J^\alpha(\theta)$ is the infimum of $C(\theta') + \alpha KL(\theta'||\theta)$, we have for any $\mathcal{E}' > 0$

$$J^\alpha(\theta) \leq C\left(\Theta^C_{\mathcal{E}'}(\theta)\right) + \alpha KL\left(\Theta^C_{\mathcal{E}'}(\theta)||\theta\right).$$

Further, as $\Theta^C_{\mathcal{E}'}(\theta)$ lies in the trust region $\{\theta' : KL(\theta'||\theta) \leq \mathcal{E}'\}$ we have that $KL\left(\Theta^C_{\mathcal{E}'}(\theta)||\theta\right) \leq \mathcal{E}'$, so we can write

$$C\left(\Theta^C_{\mathcal{E}'}(\theta)\right) + \alpha KL\left(\Theta^C_{\mathcal{E}'}(\theta)||\theta\right) \leq C\left(\Theta^C_{\mathcal{E}'}(\theta)\right) + \alpha\mathcal{E}'$$

and thus

$$J^\alpha(\theta) \leq C\left(\Theta^C_{\mathcal{E}'}(\theta)\right) + \alpha\mathcal{E}'.$$

Next we minimize both sides of this inequality within the trust region $\{\theta' : KL(\theta'||\theta) \leq \mathcal{E}\}$. We use that

$$J^\alpha\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right) = \min_{\substack{\theta' \\ \text{s.t. } KL(\theta'||\theta)\leq\mathcal{E}}} J^\alpha(\theta')$$

and get

$$J^\alpha\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right) \leq \min_{\substack{\theta' \\ \text{s.t. } KL(\theta'||\theta)\leq\mathcal{E}}} \left(C\left(\Theta^C_{\mathcal{E}'}(\theta')\right) + \alpha\mathcal{E}'\right). \tag{26}$$

Now we use Lemma 2 and rewrite the left hand side of this inequality.

$$J^\alpha\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right) = C\left(\Theta^C_{\mathcal{E}'}\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right)\right)\Big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} + \alpha\mathcal{E}^*_\alpha(\theta)$$

with $\mathcal{E}^*_\alpha(\theta) := \mathcal{E}_\alpha(\Theta^{J^\alpha}_{\mathcal{E}}(\theta))$. Plugging this back to (26) we get

$$C\left(\Theta^C_{\mathcal{E}'}\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right)\right)\Big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} + \alpha\mathcal{E}^*_\alpha(\theta) \leq \min_{\substack{\theta' \\ \text{s.t. } KL(\theta'||\theta)\leq\mathcal{E}}} \left(C\left(\Theta^C_{\mathcal{E}'}(\theta')\right) + \alpha\mathcal{E}'\right).$$

As this inequality holds for any $\mathcal{E}' > 0$ we can plug in $\mathcal{E}^*_\alpha(\theta)$ on the right hand side of this inequality and obtain

$$C\left(\Theta^C_{\mathcal{E}'}\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right)\right)\big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} + \alpha\mathcal{E}^*_\alpha(\theta) \leq \min_{\substack{\theta' \\ \text{s.t. } KL(\theta'||\theta)\leq\mathcal{E}}} C\left(\Theta^C_{\mathcal{E}'}(\theta')\right)\big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} + \alpha\mathcal{E}^*_\alpha(\theta).$$

We subtract $\alpha\mathcal{E}^*_\alpha(\theta)$ on both sides

$$C\left(\Theta^C_{\mathcal{E}'}\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right)\right)\big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} \leq \min_{\substack{\theta' \\ \text{s.t. } KL(\theta'||\theta)\leq\mathcal{E}}} C\left(\Theta^C_{\mathcal{E}'}(\theta')\right)\big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)}.$$

Using equation (16) gives

$$C\left(\Theta^C_{\mathcal{E}'}\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right)\right)\big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)} \leq C^*_{\mathcal{E},\mathcal{E}'}(\theta)\big|_{\mathcal{E}'=\mathcal{E}^*_\alpha(\theta)},$$

which concludes the proof. ■

F.3 Proof of Statement 2

Here we show that $\mathcal{E}' = \mathcal{E}^*_\alpha(\theta)$ is a monotonically decreasing function of α. $\mathcal{E}^*_\alpha(\theta)$ is given by

$$\mathcal{E}^*_\alpha(\theta) = \mathcal{E}_\alpha\left(\Theta^{J^\alpha}_{\mathcal{E}}(\theta)\right) = KL(\theta^*_{\alpha,\theta'}||\theta')\big|_{\theta'=R^{J^\alpha}_{\mathcal{E}}(\theta)}.$$

We have

$$\begin{aligned}\left(\alpha KL(\theta^*_{\alpha,\theta'}||\theta') + C\left(\theta^*_{\alpha,\theta'}\right)\right)\big|_{\theta'=R^{J^\alpha}_{\mathcal{E}}(\theta)} &= \left(\inf_{\theta''}\alpha KL(\theta''||\theta') + C(\theta'')\right)\Bigg|_{\theta'=R^{J^\alpha}_{\mathcal{E}}(\theta)} \\ &= \min_{\substack{\theta' \\ \text{s.t. } KL(\theta'||\theta)\leq\mathcal{E}}} \inf_{\theta''}\alpha KL(\theta''||\theta') + C(\theta'').\end{aligned}$$

For convenience we introduce a shorthand notation for the minimizers

$$\begin{aligned}\theta_\alpha &:= \Theta^{J^\alpha}_{\mathcal{E}}(\theta) \\ \theta'_\alpha &:= \theta^*_{\alpha,\theta'}\big|_{\theta'=\Theta^{J^\alpha}_{\mathcal{E}}(\theta)}.\end{aligned}$$

We compare $\alpha_1 \geq 0$ with $\mathcal{E}^*_{\alpha_1}(\theta) := KL(\theta'_{\alpha_1}||\theta_{\alpha_1})$ and $\alpha_2 \geq 0$ with $\mathcal{E}^*_{\alpha_2}(\theta) := KL(\theta'_{\alpha_2}||\theta_{\alpha_2})$ and assume that $\mathcal{E}^*_{\alpha_1}(\theta) < \mathcal{E}^*_{\alpha_2}(\theta)$. We show that from this it follows that $\alpha_1 > \alpha_2$.
Proof As θ'_{α_1},θ_{α_1} minimize $\alpha_1 KL(\theta'||\theta) + C(\theta')$ we have

$$\begin{aligned}\alpha_1 KL(\theta'_{\alpha_1}||\theta_{\alpha_1}) + C(\theta'_{\alpha_1}) &\leq \alpha_1 KL(\theta'_{\alpha_2}||\theta_{\alpha_2}) + C(\theta'_{\alpha_2}) \\ \Rightarrow \alpha_1\mathcal{E}_{\alpha_1}(\theta) + C(\theta'_{\alpha_1}) &\leq \alpha_1\mathcal{E}_{\alpha_2}(\theta) + C(\theta'_{\alpha_2})\end{aligned}$$

and analogous for α_2

$$\begin{aligned}\alpha_2 KL(\theta'_{\alpha_1}||\theta_{\alpha_1}) + C(\theta'_{\alpha_1}) &\geq \alpha_2 KL(\theta'_{\alpha_2}||\theta_{\alpha_2}) + C(\theta'_{\alpha_2}) \\ \Rightarrow \alpha_2\mathcal{E}_{\alpha_1}(\theta) + C(\theta'_{\alpha_1}) &\geq \alpha_2\mathcal{E}_{\alpha_2}(\theta) + C(\theta'_{\alpha_2})\end{aligned}$$

With $\mathcal{E}_{\alpha_1}(\theta) < \mathcal{E}_{\alpha_2}(\theta)$ we get

$$\alpha_1 \geq \frac{C(\theta'_{\alpha_1}) - C(\theta'_{\alpha_2})}{\mathcal{E}_{\alpha_2}(\theta) - \mathcal{E}_{\alpha_1}(\theta)} \geq \alpha_2.$$

■

We showed that from $\mathcal{E}_{\alpha_1}(\theta) < \mathcal{E}_{\alpha_2}(\theta)$ it follows that $\alpha_1 \geq \alpha_2$ which proofs that $\mathcal{E}_\alpha(\theta)$ is monotonously decreasing in α.

Appendix G. Smoothed Updates for Small Update Steps $\mathcal{E}$

We want to compute equation (17) for small $\mathcal{E}$ which corresponds to large β. Assuming a smooth dependence of p_{u_θ} on θ, bounding $KL(\theta||\theta_n)$ to a very small value allows us to do a Taylor expansion which we truncate at second order:

$$\begin{aligned}
&\arg\min_{\theta'} J^\alpha(\theta') + \beta KL(\theta'||\theta_n) \approx \\
&\quad \approx \arg\min_{\theta'} \; (\theta' - \theta_n)^T \nabla_{\theta'} J^\alpha(\theta') + \frac{1}{2}(\theta' - \theta_n)^T \left(H + \beta F\right)(\theta' - \theta_n) \\
&\quad = \theta_n - \beta^{-1} F^{-1} \left.\nabla_{\theta'} J^\alpha(\theta')\right|_{\theta'=\theta_n} + \mathcal{O}(\beta^{-2})
\end{aligned}$$

with

$$\begin{aligned}
H &= \left.\nabla_{\theta'} \nabla_{\theta'}^T J^\alpha(\theta')\right|_{\theta'=\theta_n} \\
F &= \left.\nabla_{\theta'} \nabla_{\theta'}^T KL(\theta'||\theta_n)\right|_{\theta'=\theta_n}.
\end{aligned}$$

See also Martens (2014). We used that $\mathcal{E} \ll 1 \Leftrightarrow \beta \gg 1$. With this the Fisher information F dominates over the Hessian H and thus the Hessian does not appear anymore in the update equation. This defines a natural gradient update with stepsize β^{-1}.

Appendix H. Δs Monotonic in α

Now we show that $\Delta = KL(p^*_{\alpha,\theta}||p_{u_\theta})$ is a monotonic function of α.

$$\begin{aligned}
\frac{\partial}{\partial\alpha} KL(p^*_{\alpha,\theta}||p_{u_\theta}) &= \frac{\partial}{\partial\alpha}\left\langle \ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p^*_{\alpha,\theta}} \\
&= \frac{\partial}{\partial\alpha}\left\langle \frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} \\
&= \left\langle \left(\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right)\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} + \left\langle \frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\frac{\partial}{\partial\alpha}\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} \\
&= \left\langle \left(\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right)\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} + \left\langle \frac{1}{p_{u_\theta}}\frac{\partial}{\partial\alpha}p^*_{\alpha,\theta}\right\rangle_{p_{u_\theta}} \\
&= \left\langle \left(\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right)\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} + \frac{\partial}{\partial\alpha}\langle 1\rangle_{p^*_{\alpha,\theta}} \\
&= \left\langle \left(\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right)\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} .
\end{aligned}$$

Now let us look at

$$\begin{aligned}
\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}} &= \frac{\partial}{\partial\alpha}\left(\frac{1}{Z^\alpha_{p_{u_\theta}}}\exp\left(-\frac{1}{\gamma+\alpha}S^\gamma_{p_{u_\theta}}(\tau)\right)\right) \\
Z^\alpha_{p_{u_\theta}} &= \left\langle \exp\left(-\frac{1}{\gamma+\alpha}S^\gamma_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}} .
\end{aligned}$$

we get

$$\begin{aligned}
\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}} &= \frac{1}{(\gamma+\alpha)^2}S^\gamma_{p_{u_\theta}}(\tau)\frac{p^*_{\alpha,\theta}}{p_{u_\theta}} - \frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\frac{1}{Z^\alpha_{p_{u_\theta}}}\frac{\partial}{\partial\alpha}Z^\alpha_{p_{u_\theta}} \\
\frac{\partial}{\partial\alpha}Z^\alpha_{p_{u_\theta}} &= \left\langle \frac{1}{(\gamma+\alpha)^2}S^\gamma_{p_{u_\theta}}\exp\left(-\frac{1}{\gamma+\alpha}S^\gamma_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}} .
\end{aligned}$$

and thus

$$\begin{aligned}
\frac{\partial}{\partial\alpha}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}} &= \frac{1}{(\gamma+\alpha)^2}S^\gamma_{p_{u_\theta}}(\tau)\frac{p^*_{\alpha,\theta}}{p_{u_\theta}} - \frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\frac{1}{(\gamma+\alpha)^2}\left\langle S^\gamma_{p_{u_\theta}}\right\rangle_{p^*_{\alpha,\theta}} \\
&= \frac{1}{(\gamma+\alpha)^2}\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\left(S^\gamma_{p_{u_\theta}}(\tau) - \left\langle S^\gamma_{p_{u_\theta}}\right\rangle_{p^*_{\alpha,\theta}}\right).
\end{aligned}$$

So finally we get

$$\begin{aligned}
\frac{\partial}{\partial\alpha} KL(p^*_{\alpha,\theta}||p_{u_\theta}) &= \frac{1}{(\gamma+\alpha)^2}\left\langle \frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\left(S^\gamma_{p_{u_\theta}}(\tau) - \left\langle S^\gamma_{p_{u_\theta}}\right\rangle_{p^*_{\alpha,\theta}}\right)\ln\frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\right\rangle_{p_{u_\theta}} \\
&= \frac{1}{(\gamma+\alpha)^2}\left\langle \frac{p^*_{\alpha,\theta}}{p_{u_\theta}}\left(S^\gamma_{p_{u_\theta}}(\tau) - \left\langle S^\gamma_{p_{u_\theta}}\right\rangle_{p^*_{\alpha,\theta}}\right)\left(-\frac{1}{\gamma+\alpha}S^\gamma_{p_{u_\theta}}(\tau) - \log Z^\alpha_{p_{u_\theta}}\right)\right\rangle_{p_{u_\theta}} \\
&= \frac{1}{(\gamma+\alpha)^2}\left\langle \left(S^\gamma_{p_{u_\theta}}(\tau) - \left\langle S^\gamma_{p_{u_\theta}}\right\rangle_{p^*_{\alpha,\theta}}\right)\left(-\frac{1}{\gamma+\alpha}S^\gamma_{p_{u_\theta}}(\tau) - \log Z^\alpha_{p_{u_\theta}}\right)\right\rangle_{p^*_{\alpha,\theta}} \\
&= -\frac{1}{(\gamma+\alpha)^3}\left(\left\langle\left(S^\gamma_{p_{u_\theta}}\right)^2\right\rangle_{p^*_{\alpha,\theta}} - \left\langle S^\gamma_{p_{u_\theta}}\right\rangle^2_{p^*_{\alpha,\theta}}\right) \\
&= -\frac{1}{(\gamma+\alpha)^3}\mathrm{Var}\left(S^\gamma_{p_{u_\theta}}\right) \le 0.
\end{aligned}$$

Therefore $\Delta = KL(p^*_{\alpha,\theta}||p_{u_\theta})$ is a monotonically decreasing function of α.

Appendix I. Proof for Equivalence of Weight Entropy and KL-Divergence

We want to show that

$$\begin{aligned}
\lim_{N\to\infty} \log N - H_N(w) &= \lim_{N\to\infty} \log N + \sum_{i=1}^N w^i \log(w^i) \\
&= KL(p^*_{\alpha,\theta}||p_{u_\theta}),
\end{aligned}$$

where the samples i are drawn from p_{u_θ} and the w^i are given by

$$w^i = \frac{1}{\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha}S_{p_{u_\theta}}(\tau^i)\right)}\exp\left(-\frac{1}{\gamma+\alpha}S_{p_{u_\theta}}(\tau^i)\right).$$

We get

$$
\begin{aligned}
&\lim_{N\to\infty} \log N + \sum_{i=1}^{N} w^i \log(w^i) = \\
&= \lim_{N\to\infty} \log N + \sum_{i=1}^{N} \frac{1}{\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right) \cdot \\
&\qquad \cdot \log\left(\frac{1}{\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)\right) \\
&= \lim_{N\to\infty} \log N + \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\frac{1}{N}\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right) \cdot \\
&\qquad \cdot \log\left(\frac{\frac{1}{N}}{\frac{1}{N}\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)\right) \\
&= \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\frac{1}{N}\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right) \cdot \\
&\qquad \cdot \log\left(\frac{1}{\frac{1}{N}\sum_i^N \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau^i)\right)\right)
\end{aligned}
$$

Now we replace in the limit $N \to \infty$, $\frac{1}{N}\sum_i^N \to \langle\rangle_{p_{u_\theta}}$:

$$
\begin{aligned}
&= \left\langle \frac{1}{\left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}}} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right) \cdot \right. \\
&\left. \cdot \log\left(\frac{1}{\left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}}} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right)\right\rangle_{p_{u_\theta}}
\end{aligned}
$$

Using equation (12) this gives

$$
\begin{aligned}
&= \left\langle \log \left(\frac{1}{\left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}}} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right)\right\rangle_{p^*_{\alpha,\theta}} \\
&= \left\langle \log \left(\frac{1}{\left\langle \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right)\right\rangle_{p_{u_\theta}}} \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_\theta}}(\tau)\right) \frac{p_{u_\theta}(\tau)}{p_{u_\theta}(\tau)}\right)\right\rangle_{p^*_{\alpha,\theta}} \\
&= \left\langle \log \frac{p^*_{\alpha,\theta}(\tau)}{p_{u_\theta}(\tau)} \right\rangle_{p^*_{\alpha,\theta}} \\
&= KL(p^*_{\alpha,\theta}||p_{u_\theta}).
\end{aligned}
$$

Appendix J. Inversion of the Fisher Matrix

We compute an approximation to the natural gradient $g_f = F^{-1}g$ by approximately solving the linear equation $Fg_f = g$ using truncated conjugate gradient. With the standard gradient g and the Fisher matrix $F = \nabla_\theta \nabla_\theta^T KL(p_{u_\theta}||p_{u_{\theta_n}})$ (see Appendix G).

We use an efficient way to compute the Fisher vector product Fy (Schulman et al., 2015) using an automated differentiation package: first for each rollout i and timepoint t the symbolic expression for the gradient on the KL multiplied by a vector y is computed:

$$a_{i,t}(\theta_{n+1}) = \left(\nabla^T_{\theta_{n+1}} \log \frac{\pi_{\theta_n}(a^i_t|t, x^i_t)}{\pi_{\theta_{n+1}}(a^i_t|t, x^i_t)}\right) y.$$

Then we take the second derivative on this scalar quantity, sum over all times and average over the samples. This gives the Fisher vector

$$Fy = \frac{1}{N}\sum_{i=1}^{N} \sum_{0<t<T} \nabla_{\theta_{n+1}} a_{i,t}(\theta_{n+1}).$$

Appendix K. Full Parametrization in LQ Problems

Here we discuss why for a linear quadratic problem a time varying linear controller is a full parametrization. We want to show that for every

$$p^*_{\alpha,\theta_0} = \frac{1}{Z} p_{u_0}(\tau) \exp\left(-\frac{1}{\gamma+\alpha} S^{\gamma}_{p_{u_{\theta_0}}}(\tau)\right) \tag{27}$$

there is a time varying linear controller $u_{\theta^*_{\alpha,\theta_0}}$ such that $p_{u_{\theta^*_{\alpha,\theta_0}}} = p^*_{\alpha,\theta_0}$. We assume that u_{θ_0} is a time varying linear controller. In Appendix B we have shown that u^*_{α,θ_0} is the solution to the path integral control problem with dynamics

$$\dot{x}_t = f(x_t, t) + g(x_t, t)\left(\tilde{u}(x_t, t) + \hat{u}(x_t, t) + \xi_t\right)$$

and cost

$$\left\langle \int_0^T \frac{1}{\gamma} V(x_t, t) - \frac{1}{2}\frac{\gamma}{\alpha} \tilde{u}(x_t, t)^T \tilde{u}(x_t, t) dt + \int_0^T \left(\frac{1}{2} \hat{u}(x_t, t)^T \hat{u}(x_t, t) + \hat{u}(x_t, t)^T \xi_t \right) dt \right\rangle_{p_{\hat{u}}},$$

with $\tilde{u} = \left(1 - \frac{\gamma}{\gamma+\alpha}\right) u_{\theta_0}(x_t, t)$.

It is now easy to see that if u_{θ_0} is a time varying linear controller, thus a linear function of the state, the cost is a quadratic function of the state x (note that $V(x_t, t)$ is quadratic in the LQ case). Thus for all values of α, u^*_{α,θ_0} is the solution to a linear quadratic control problem and thus a time varying linear controller (see, e.g., Kwakernaak and Sivan (1972)). Therefore a time varying linear controller is a full parametrization.

Appendix L. Details for the Numerical Experiments

L.1 Linear-Quadratic Control Task

Dynamics: the dynamics are ODEs integrated by an Euler scheme (see Section 6.1). The differential equation is initialized at $x = 0$ and $dt = 0.1$.

Control problem: Regularization $\gamma = 1$. Time-Horizon $T = 10s$. State-Cost function: see Section 6.1. $(x_0, t_0) = (-10, 1)$, $(x_1, t_1) = (10, 2)$, $(x_2, t_2) = (-10, 3)$, $(x_3, t_3) = (-20, 4)$, $(x_4, t_4) = (-100, 5)$, $(x_5, t_5) = (-50, 6)$, $(x_6, t_6) = (10, 7)$, $(x_7, t_7) = (20, 8)$, $(x_8, t_8) = (30, 9)$. Variance of uncontrolled dynamics $\nu = 1$.

Algorithm: Batchsize: $N = 100$, trust region $\mathcal{E} = 0.1$, smoothing strength $\Delta = 0.2 \log 100$, conjugate gradient iterations: 2 (for each time step separately). The parametrized controller was initialized at $\theta_0 = 0$.

L.2 Pendulum Task

Dynamics: the differential equation for the pendulum is

$$\ddot{x} + c\omega_0 \dot{x} + \omega_0^2 \sin(x) = \lambda (u + \xi),$$

with $c\omega_0 = 0.1$ $[s^{-1}]$, $\omega_0^2 = 10$ $[s^{-2}]$, and $\lambda = 0.2$.

We implemented this differential equation as a first order differential equation and integrated it with an Euler scheme ($dt = 0.01$). The pendulum is initially resting at the bottom:

$$\dot{x} = 0, x = 0.$$

As a parametrized controller we use a time varying linear feedback controller:

$$u_\theta(x, \dot{x}, t) = \theta_{3,t} \cos(x) + \theta_{2,t} \sin(x) + \theta_{1,t} \dot{x} + \theta_{0,t}.$$

The parametrized controller was initialized at $\theta = 0$.

Control-problem: the regularization is set to $\gamma = 1$ and the time-horizon $T = 3.0s$. The state-cost function has end-cost only:

$$V(x, \dot{x}, t) = \delta(t - T) \left(-500Y + 10\dot{x}^2\right),$$

with $Y = -\cos(x)$ (height of tip). The variance of uncontrolled dynamics is $\nu = 1$.

Algorithm: batchsize: $N = 500$, trust region $\mathcal{E} = 0.1$, smoothing strength $\Delta = 0.5$. The Fisher-matrix was inverted for each time step separately using the scipy pseudo-inverse with rcond=10^{-4}.

L.3 Acrobot Task

Dynamics: we use the definition of the acrobot as in Spong (1995). The differential equations for the acrobot are

$$\begin{aligned} d_{11}(x)\ddot{x}_1 + d_{12}(x)\ddot{x}_2 + h_1(x,\dot{x}) + \phi_1(x) &= 0 \\ d_{21}(x)\ddot{x}_1 + d_{22}\ddot{x}_2 + h_2(x,\dot{x}) + \phi_2(x) &= \lambda \cdot (u + \xi) \end{aligned}$$

with

$$\begin{aligned} d_{11} &= m_1 l_{c1}^2 + m_2\left(l_1^2 + l_{c2}^2 + 2 l_1 l_{c2}\cos(x_2)\right) + I_1 + I_2 \\ d_{12} &= m_2\left(l_{c2}^2 + l_1 l_{c2}\cos(x_2)\right) + I_2 \\ d_{21} &= d_{12} \\ d_{22} &= m_2 l_{c2}^2 + I_2 \\ h_1 &= -m_2 l_1 l_{c2}\sin(x_2)\left(\dot{x}_2^2 + 2\dot{x}_1\dot{x}_2\right) \\ h_2 &= m_2 l_1 l_{c2}\sin(x_2)\dot{x}_1^2 \\ \phi_2 &= m_2 l_{c2} G\cos(x_1 + x_2) \\ \phi_1 &= (m_1 l_{c1} + m_2 l_1) g\cos(x_1) + \phi_2 \end{aligned}$$

and parameter values

- $G = 9.8$
- $l_1 = 1.$ [m]
- $l_2 = 2.$ [m]
- $m_1 = 1.$ [kg] mass of link 1
- $m_2 = 1.$ [kg] mass of link 2
- $l_{c1} = 0.5$ [m] position of the center of mass of link 1
- $l_{c2} = 1.0$ [m] position of the center of mass of link 2
- $I_1 = 0.083$ moments of inertia for both links
- $I_2 = 0.33$ moments of inertia for both links
- $\lambda = 0.2$

We implemented this differential equation as a first order differential equation and integrated it with an Euler scheme ($dt = 0.01$). The acrobot is initially resting at the bottom:

$$\dot{x}_1 = 0, \dot{x}_2 = 0, x_1 = -\frac{1}{2}\pi, x_2 = 0.$$

As a parametrized controller we use a time varying linear feedback controller:

$$\begin{aligned} u_\theta(x, \dot{x}, t) =& \theta_{8,t}\cos(x_1) + \theta_{7,t}\sin(x_2) + \theta_{6,t}\cos(x_2) + \theta_{5,t}\sin(x_2) + \\ &+ \theta_{4,t}\sin(x_1 + x_2) + \theta_{3,t}\cos(x_1 + x_2) + \theta_{2,t}\dot{x_1} + \theta_{1,t}\dot{x_2} + \theta_{0,t}. \end{aligned}$$

The parametrized controller was initialized at $\theta = 0$.

Control-problem: regularization $\gamma = 1$, time-horizon $T = 3.0s$, and state-cost function has end-cost only:

$$V(x, \dot{x}, t) = \delta(t - T)\left(-500Y + 10(\dot{x_1}^2 + \dot{x_2}^2)\right),$$

with $Y = -l_1\cos(x_1) - l_2\cos(x_1 + x_2)$ (height of tip). The variance of uncontrolled dynamics is $\nu = 1$.

Algorithm: batchsize $N = 500$, trust region $\mathcal{E} = 0.1$, and smoothing strenght $\Delta = 0.5$. The Fisher-matrix was inverted for each time step separately using the scipy pseudo-inverse with rcond=10^{-4}.

L.4 Walker Task

For dynamics and the state cost function we used "BipedalWalker-v2" from the OpenAI gym (Brockman et al., 2016). The policy was a Gaussian policy, with static variance $\sigma^2 = 1$. The state dependent mean of the Gaussian policy was a neural network controller with two hidden layers with 32 neurons, each. The activation function is a `tanh`. For the initialization we used Glorot Uniform (Glorot and Bengio, 2010). The inputs to the neural network was the observation space provided by OpenAI gym task "BipedalWalker-v2": State consists of hull angle speed, angular velocity, horizontal speed, vertical speed, position of joints and joints angular speed, legs contact with ground, and 10 lidar rangefinder measurements.

Control-problem:

- $\gamma = 0$
- Time-Horizon: defined by OpenAI gym task "BipedalWalker-v2"
- State-Cost function defined by OpenAI gym task "BipedalWalker-v2": Reward is given for moving forward, total +300 points up to the far end. If the robot falls, it gets −100. Applying motor torque costs a small amount of points, more optimal agent will get better score.

Algorithm: batchsize $N = 100$, trust region $\mathcal{E} = 0.01$, smoothing strength $\Delta = 0.05\log 100$, and 10 conjugate gradient iterations.

Hyperparameters	Value
Number of rollouts (N)	50
Total number of rollouts	50 000
Smoothing strength (Δ)	$\{0.1.0.5\}$
Trust region size ($\mathcal{E}$)	$\{0.025, 0.075\}$
Mini batch size	256
Units per layer	32
Number of hidden layers	1
Learning rate	7e-4
Activation function	tanh
Action distribution	Isotropic Gaussian

Table 1: Hyperparameters for the experiments using Pybullet.

L.5 Pybullet Experiments

For these experiments we use the Pybullet open source engine.[5] In all tasks, we used $N = 50$ rollouts per iteration. The choice of controller as well as the hyperparameters for the conjugate gradient step were optimized as in Schulman et al. (2015). For PG-TR, we also used the same values of the trust region size $\epsilon = 0.01$ and hyperparameters for the conjugate gradient optimizer. For ASPIC, we considered two values of the smoothing strength $\Delta = \{0.1.0.5\}$ and two trust region sizes $\mathcal{E} = \{0.025, 0.075\}$.[6]

References

Oleg Arenz, Gerhard Neumann, and Mingjun Zhong. Efficient gradient-free variational inference using policy search. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 234–243, 2018.

Joris Bierkens and Hilbert J Kappen. Explicit solution of relative entropy weighted control. *Systems & Control Letters*, 72:36–43, 2014.

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

Pratik Chaudhari, Adam Oberman, Stanley Osher, Stefano Soatto, and Guillaume Carlier. Deep relaxation: partial differential equations for optimizing deep neural networks. *Research in the Mathematical Sciences*, 5(3):30, 2018.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

Wendell H Fleming and William M McEneaney. Risk-sensitive control on an infinite time horizon. *SIAM Journal on Control and Optimization*, 33(6):1881–1915, 1995.

5. `https://pybullet.org/wordpress/`
6. For reproducibility, the code will be made available upon acceptance of the final manuscript

WH Fleming and SJ Sheu. Risk-sensitive control and an optimal investment model ii. *Annals of Applied Probability*, pages 730–767, 2002.

Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare, and Joelle Pineau. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, 11(3-4):219–354, 2018. ISSN 1935-8237. doi: 10.1561/2200000071.

Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media, 2013.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

Vicenç Gómez, Hilbert J Kappen, Jan Peters, and Gerhard Neumann. Policy search for path integral control. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 482–497. Springer, 2014.

Nicolas Heess, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, Ali Eslami, Martin Riedmiller, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

Sham Kakade. A natural policy gradient. *Advances in neural information processing systems*, 2:1531–1538, 2002.

H. J. Kappen. Linear theory for control of nonlinear stochastic systems. *Physical Review Letters*, 95(20):200201, 2005. doi: 10.1103/PhysRevLett.95.200201.

H. J. Kappen and H. C. Ruiz. Adaptive importance sampling for control and inference. *Journal of Statistical Physics*, 162(5):1244–1266, 2016. ISSN 1572-9613. doi: 10.1007/s10955-016-1446-7.

H. J. Kappen, V. Gómez, and M. Opper. Optimal control as a graphical model inference problem. *Machine Learning*, 87(2):159–182, 2012.

Huibert Kwakernaak and Raphael Sivan. *Linear optimal control systems*, volume 1. Wiley-Interscience New York, 1972.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(1):1334–1373, January 2016. ISSN 1532-4435.

James Martens. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*, 2014.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 4026–4035. PMLR, 2018.

Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

Jan Peters, Katharina Mülling, and Yasemin Altün. Relative entropy policy search. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1607–1612. AAAI Press, 2010.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black Box Variational Inference. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33, pages 814–822. PMLR, 2014.

Hans-Christian Ruiz and Hilbert J Kappen. Particle smoothing for hidden diffusion processes: Adaptive path integral smoother. *IEEE Transactions on Signal Processing*, 65 (12):3191–3203, 2017.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1889–1897. PMLR, 2015.

Mark W Spong. The swing up control problem for the acrobot. *IEEE control systems*, 15 (1):49–55, 1995.

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

Dominik Thalmeier, Vicenç Gómez, and Hilbert J Kappen. Action selection in growing state spaces: Control of network structure growth. *Journal of Physics A: Mathematical and Theoretical*, 50(3):034006, 2016.

S. Thijssen and H. J. Kappen. Path integral control and state-dependent feedback. *Physical Review E*, 91(3):032104, 2015.

Emanuel Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, 106(28):11478–11483, 2009. ISSN 0027-8424. doi: 10.1073/pnas.0710743106.

Bart van den Broek, Wim Wiegerinck, and Hilbert Kappen. Risk sensitive path integral control. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 615–622. AUAI Press, 2010.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.