Universiteit
Leiden
The Netherlands

On the effect of regularization in policy mirror descent

Kleuker, J.F; Plaat, A.; Moerland, T.M.

Citation
Kleuker, J. F., Plaat, A., & Moerland, T. M. (2025). On the effect of regularization in policy mirror descent. *Reinforcement Learning Journal*, *6*, 1931-1950. doi:10.5281/zenodo.13899776

Version:	Publisher's Version
License:	Creative Commons CC BY 4.0 license
Downloaded from:	https://hdl.handle.net/1887/4273671

Note: To cite this publication please use the final published version (if applicable).

On the Effect of Regularization in Policy Mirror Descent

Jan Felix Kleuker, Aske Plaat, Thomas Moerland

Keywords: Policy Mirror Descent, Regularization, Reinforcement Learning

Summary

Policy Mirror Descent (PMD) has emerged as a unifying framework in reinforcement learning (RL) by linking policy gradient methods with a first-order optimization method known as mirror descent. At its core, PMD incorporates two key regularization components: (i) a distance term that enforces a trust region for stable policy updates and (ii) an MDP regularizer that augments the reward function to promote structure and robustness. While PMD has been extensively studied in theory, empirical investigations remain scarce. This work provides a large-scale empirical analysis of the interplay between these two regularization techniques, running over 500k training seeds on small RL environments.

Contribution(s)

(i) We conduct an empirical analysis of the interaction between the regularization components in PMD, running over 500k training seeds and providing insights into the fragility of these algorithms to regularization temperatures.
Context: Empirical studies on PMD-based RL algorithms are limited.

(ii) We examined the impact of temperature scales across different reward scales and identified simple heuristics to aid in tuning these temperatures. Additionally, we evaluated dynamic adaptation strategies for temperature parameters, indicating that maintaining constant values is often more effective than adapting them.
Context: RL algorithms are sensitive to hyperparameter tuning, making the selection of appropriate temperatures challenging.

(iii) We examine the effects of different regularization combinations on robustness to temperature tuning and performance, indicating a notable impact.
Context: To the best of our knowledge, no existing study examines the effects of the interplay between these two components for different choices, leaving a gap in the literature.

On the Effect of Regularization in Policy Mirror Descent

Jan Felix Kleuker[1], **Aske Plaat**[1], **Thomas Moerland**[1]
kleukerjf@vuw.leidenuniv.nl

[1]**Leiden Institute of Advanced Computer Science, Leiden University, the Netherlands**

Abstract

Policy Mirror Descent (PMD) has emerged as a unifying framework in reinforcement learning (RL) by linking policy gradient methods with a first-order optimization method known as mirror descent. At its core, PMD incorporates two key regularization components: (i) a distance term that enforces a trust region for stable policy updates and (ii) an MDP regularizer that augments the reward function to promote structure and robustness. While PMD has been extensively studied in theory, empirical investigations remain scarce. This work provides a large-scale empirical analysis of the interplay between these two regularization techniques, running over 500k training seeds on small RL environments. Our results demonstrate that, although the two regularizers can partially substitute each other, their precise combination is critical for achieving robust performance. These findings highlight the potential for advancing research on more robust algorithms in RL, particularly with respect to hyperparameter sensitivity.

1 Introduction

Recent research has revealed a deep connection between policy gradient methods in RL and mirror descent — a first-order optimization technique (Beck & Teboulle, 2003; Beck, 2017). This insight has led to the development of the PMD framework (Geist et al., 2019; Tomar et al., 2020; Grudzien et al., 2022; Lan, 2023; Xiao, 2022; Vaswani et al., 2022; Alfano et al., 2023; Zhan et al., 2023), which encompasses a broad class of algorithms distinguished by their choice of regularization, such as Trust Region Learning (TRL; Schulman et al. 2015), or Soft Actor-Critic (SAC; Haarnoja et al. 2018a).

In the PMD framework, two regularization components are central. The first is a distance term, referred to as the *Drift regularizer*, that ensures the updated policy remains sufficiently close to its predecessor – an idea that was prominently implemented in trust region policy optimization (TRPO; Schulman et al. 2015), or proximal policy optimization (PPO; Schulman et al. 2017). This trust-region idea has been further generalized through the introduction of a Drift functional (Grudzien et al., 2022), motivating the term Drift regularizer. The second component is a convex *MDP regularizer* that augments the reward function with a structure-promoting term, an approach motivated by the objectives of enhanced exploration and robustness (Ziebart, 2010; Haarnoja et al., 2017; Chow et al., 2018; Lee et al., 2019). A prominent example hereof is the negative Shannon-Entropy, a core component in soft Reinforcement Learning (Haarnoja et al., 2017; 2018a).

While extensive theoretical work has established strong convergence results for PMD, these guarantees largely disappear in approximate settings where the exact value function is inaccessible. On the other hand, only a limited number of numerical experiments have been conducted to validate its practical performance (Vieillard et al., 2020; Tomar et al., 2020; Alfano et al., 2023). Notably, empirical studies have primarily focused on specific cases, such as the (reverse) KL divergence or learning Drift regularizers (Lu et al., 2022; Alfano et al., 2024).

In this work, we complement existing empirical studies by systematically analyzing the impact of different regularization components on algorithmic performance of RL algorithms based on PMD. Our main contributions are:

- We analyze the interaction between two distinct regularization techniques in RL, a study that, to the best of our knowledge, has not been conducted before. To this end, we run over 500k training seeds on small RL environments, demonstrating brittleness of these algorithms to regularization temperatures.
- We investigate the effects of varying temperature parameters both during training and across different reward scales, providing insights to support the design of algorithms that mitigate fragility to regularization temperatures.
- Leveraging recent theoretical advancements, we examine the effects of different regularization combinations, particularly their influence on robustness to temperature tuning. Our findings suggest this aspect may be understudied in the existing literature.

The rest of the paper is organized as follows: In Section 2, we introduce the necessary background on Policy Mirror Descent. Section 3 details the methodology of our numerical experiments, followed by a discussion of the results in Section 4. Finally, we conclude with key findings in Section 6.

2 Background

We assume a standard Markov Decision Process (MDP) defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p_0, p, \gamma)$. Here, $\mathcal{S}$ is a (discrete) state space, $\mathcal{A}$ is a discrete action space, $p_0 \in \Delta(\mathcal{S})$[1] is the initial state distribution, $p(\cdot, \cdot|s, a) \in \Delta(\mathcal{S}\times\mathbb{R})$ is the probabilistic transition and reward function, and $\gamma \in [0, 1)$ is the discount factor. We usually abbreviate the reward by $r(s, a) = \mathbb{E}_{s',r\sim p(s',r|s,a)}[r]$.

The behaviour of an agent interacting with this MDP is defined by a so-called policy, that is, a mapping $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ assigning a distribution over actions to each state. Each policy induces a distribution $p_\pi(\tau)$ on the set of trajectories $\tau = (s_0, a_0, s_1, \dots)$ (Agarwal et al., 2019). The (unregularized) value function is defined as $V^\pi(s) = \mathbb{E}_{\tau\sim p_\pi(\tau)}[\sum_{t=0}^\infty \gamma^t\, r(s_t, a_t)|s_0 = s]$, resulting in the (unregularized & point-wise) MDP objective $\pi^*(\cdot|s) = \arg\max_{\pi\in\Pi} V^\pi(s)$.

Regularized MDP A natural extension hereof can be obtained by adding a regularization term on the reward level, also referred to as MDP regularization (Ziebart, 2010; Lee et al., 2019). More precisely, given a convex function $h : \Delta(\mathcal{A}) \to \mathbb{R}$ the corresponding regularized Q-value function is defined as $Q^\pi_\alpha(s, a) := \mathbb{E}_{\tau\sim p_\pi(\tau)}\left[\sum_{t=0}^\infty \gamma^t \left\{r(s_t, a_t) - \alpha\, h(\pi(\cdot|s_t))\right\} |s_0 = s, a_0 = a\right]$ (Lan, 2023). Similarly we define state-value function $V^\pi_\alpha(s)$. Note, that this definition of the regularized Q-value function differs slightly from how it is defined in Zhan et al. (2023), or in Haarnoja et al. (2018a). However, both versions of the regularized Q-value functions can be used interchangeably in all schemes presented in this work. The resulting (point-wise) regularized MDP objective can be expressed as

$$\pi^*(\cdot|s) = \arg\max_{\pi\in\Pi} V^\pi_\alpha(s). \tag{1}$$

2.1 Policy Mirror Descent

Mirror Descent A common method to solve optimisation problems like (1) is the so-called *mirror descent* algorithm (Beck & Teboulle, 2003; Beck, 2017). Consider a general differentiable function $f : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}$ and the optimisation problem

$$x^* = \arg\min_{x\in\mathcal{X}} f(x). \tag{2}$$

Mirror descent gives the iterative update scheme starting from some $x_0 \in \mathcal{X}$

$$x_{k+1} = \arg\min_{x\in\mathcal{X}} \left\{\langle x, \nabla f(x)|_{x=x_k}\rangle + \lambda_k\, B_\omega(x, x_k)\right\}. \tag{3}$$

[1] $\Delta(\mathcal{X})$ denotes the set of probability distributions over a set $\mathcal{X}$

Here $\langle \cdot, \cdot \rangle$ denotes the standard inner product and B_ω denotes the (generalized) Bregman divergence (Lan et al., 2011) for a convex potential function $\omega : \mathcal{X} \to \mathbb{R}$

$$B_\omega(x, y) = \omega(x) - \omega(y) - \langle \nabla\omega(y), x - y \rangle, \tag{4}$$

where $\nabla\omega(y)$ can be any vector falling within the subdifferential. This scheme generalizes the (projected) gradient descent, which is recovered by choosing the squared Euclidean distance as the Bregman divergence, i.e. $B_\omega(x, y) = \|x - y\|_2^2$ (Beck & Teboulle, 2003).

Mirror Descent inspired Policy updates By applying the MD scheme (3) to the regularized objective (1) the *Policy Mirror Descent* Update (see (Lan, 2023), Algorithm 1) can be obtained

$$\pi_{k+1}(\cdot|s) = \arg\min_{\pi \in \Pi} \left\{ \langle \pi(\cdot|s), -Q_\alpha^{\pi_k}(s, \cdot) \rangle + \alpha\, h(\pi(\cdot|s)) + \lambda_k\, B_\omega(\pi, \pi_k; s) \right\}, \tag{5}$$

for a sequence of step sizes $\lambda_k > 0$. For notational convenience we write $B_\omega(\pi, \pi_k; s)$ instead of $B_\omega(\pi(\cdot|s), \pi_k(\cdot|s))$. A popular choice for h is the negative Shannon-Entropy $-\mathcal{H}$, while the (reverse) Kullback-Leibler (KL) divergence is a common selection for B_ω. Notably, the KL divergence is induced by the negative entropy as its potential function (Beck & Teboulle, 2003).

2.2 Regularization in Policy Mirror Descent

The policy improvement scheme (5) consists of two components. The first, $\alpha h(\pi(\cdot|s))$, results from regularizing the MDP with a convex potential function h (MDP regularizer). This term modifies the value function V_α^π, reshaping the optimization surface by influencing how V_α^π changes with π and the location of (local) minima. The second component, the Drift regularizer $\lambda_k B_\omega(\pi, \pi_k; s)$, impacts how this landscape is traversed, by penalising large deviations in updating the policy.

Interplay of the regularization terms: Convergence Results Intuitively, one might expect that reshaping the optimisation landscape with an MDP regularizer h would naturally impact the valid choices of the Drift regularizer, or in other words, that these types of regularization have an effect on each other. In fact, in the absence of an MDP regularizer (i.e. $\alpha = 0$), Grudzien et al. (2022) have shown that there exists a broad class of valid Drift regularizers, naturally encompassing the set of Bregman-Divergences. More specifically it was shown that for any Drift functional $D_{\tilde{\pi}}(\pi|s)$ (a distance function with only minimal requirements) the scheme

$$\pi_{k+1}(\cdot|s) = \arg\min_{\pi \in \Pi} \left\{ \langle \pi(\cdot|s), -Q^{\pi_k}(s, \cdot) \rangle \; + \lambda_k\, \mathcal{D}_{\pi_k}(\pi|s) \right\} \tag{6}$$

provably converges to the optimal policy π^* with monotonic improvements of the return. Notably, this scheme encompasses well-known RL algorithms such as PPO (Schulman et al., 2017), and Mirror Descent Policy Optimisation (MDPO; Tomar et al. 2020), highlighting that many RL algorithms naturally emerge from the mirror descent perspective.

In contrast, when an MDP regularizer is present (i.e., $\alpha \neq 0$), similar results could only be shown to hold for the class of Bregman Divergences (Lan, 2023), hence restricting the choice of valid Drift regularizers. Moreover, convergence rates improve from sublinear to linear when the potential function for the Bregman divergence is chosen to be the MDP regularizer, i.e., when $B_\omega = B_h$ (Lan, 2023; Zhan et al., 2023).

3 Methodology

Building on the guaranteed monotonic improvements from policy updates in (5) and (6), these updates can be incorporated into a generalized policy iteration (GPI)-like algorithm that alternates between policy evaluation and improvement (Sutton, 2018; Geist et al., 2019; Vieillard et al., 2020). While theoretical results assume access to exact value functions, practical actor-critic implementations typically rely on neural networks to approximate $Q^{\pi_k} \approx Q^{\phi_k}$, e.g. by minimizing a Bellman

residual loss (Haarnoja et al., 2018a). Similarly, policies are parameterized as neural networks, $\pi_k \approx \pi_{\theta_k}$, rendering point-wise updates as in (5) infeasible. Instead, we optimize an expectation over a state distribution $\mathcal{D} \in \Delta(\mathcal{S})$, leading to a single policy improvement objective:

$$\theta_{k+1} = \arg\min_{\theta \in \Theta} \mathbb{E}_{s\sim\mathcal{D}} \left[\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[-Q^{\phi_k}_{\alpha_k}(s,a)] + \alpha_k \, h((\pi_\theta(\cdot|s)) + \lambda_k \, D(\pi_\theta; \pi_{\theta_k}|s) \right] \tag{7}$$

Objects of study Due to the above assumptions, the theoretical guarantees do not translate into practice. To bridge this gap, we empirically analyze the effects of both commonly used and less conventional regularizers.

- **MDP Regularizers** A common choice for this level of regularization are entropy-like functions, such as the (negative) Shannon-Entropy $-\mathcal{H}$ (Ziebart, 2010; Haarnoja et al., 2017; 2018a;b), or, as a generalization hereof, the (negative) Tsallis Entropy $-\mathcal{H}_m$ (Lee et al., 2019; Chow et al., 2018). Additionally, as less common choices $h = ||\cdot||_2^2$ and the non-smooth $\max$ function will be tested. We refer to Appendix A for details.
- **Drift Regularizers** A common choice as a Drift regularizer is the (reverse) KL-Divergence $D_{\mathrm{KL}}(\pi; \pi_k|s)$, a core component in MDPO (Geist et al., 2019; Tomar et al., 2020), and Uniform TRPO (Shani et al., 2020), but was also studied in (Liu et al., 2019; Vieillard et al., 2020). Moreover, the Bregman-Divergences corresponding to the Tsallis-Entropy, $\max$ function and squared norm, respectively, will be studied in this work. We refer to Appendix A for details.
- **Temperature Parameters** In addition to the choice of regularizers, their weighting coefficients, α_k and λ_k, referred to as temperature parameters (Haarnoja et al., 2018a), significantly influence algorithm performance. In many RL tasks, MDP regularization acts as an auxiliary reward to improve exploration and stability. This makes choosing a fixed $\alpha_k = \alpha$ challenging, as effective regularization should promote exploration early in training and decrease over time, motivating a linear annealing schedule. To simplify the selection of α_k, Haarnoja et al. (2018b) proposed an adaptive adjustment strategy that maintains a desired level of exploration. Specifically, α is updated via gradient descent on the objective $J(\alpha) = \mathbb{E}_{s\sim\mathcal{D}}[\mathbb{E}_{a\sim\pi_k}[-\alpha \log(\pi_k(a|s)) - \alpha\bar{\mathcal{H}}]]$, where $\bar{\mathcal{H}}$ represents the expected minimum entropy of the policy. This approach extends naturally to general MDP regularizers h, leading to

$$J(\alpha) = \mathbb{E}_{s\sim\mathcal{D}}[-\alpha \, h(\pi_k(\cdot|s)) + \alpha \, \bar{h}], \tag{8}$$

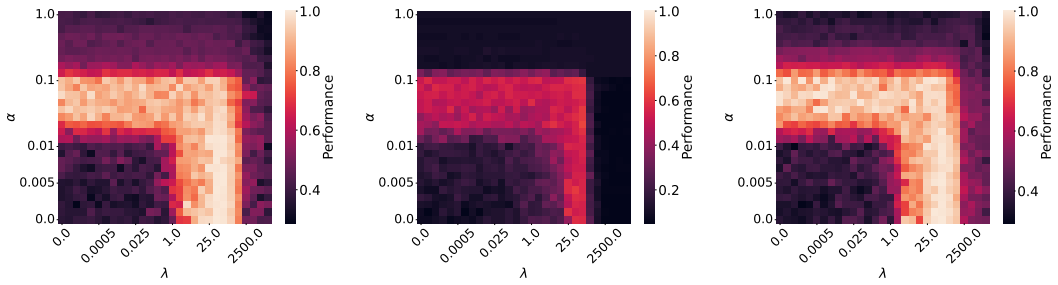
  where $\bar{h}$ can be linearly annealed to reduce the influence of MDP regularization over training. Regarding λ_k, theoretical results (Lan, 2023; Zhan et al., 2023) suggest keeping it constant, i.e., $\lambda_k = \lambda$. However, empirical findings by Tomar et al. (2020) indicate that linearly annealing λ over training may be more effective.

Algorithm Setting $h = -\mathcal{H}$ and $D(p,q) = D_{\mathrm{KL}}(p,q)$ recovers the (soft) MDPO algorithm (Tomar et al., 2020). As noted in their work, MDPO can be implemented in either an off-policy or on-policy manner. In this study, we adopt an off-policy implementation and refer to the resulting algorithm as MDPO(h, D), emphasizing the choice of regularization; see Appendix D for details.

4 Experiments

Regularization is a key component in many deep RL algorithms but is rarely the sole factor. The goal of this work is to study the core effect of RL specific regularization. To hence keep the influence of other regularization techniques (Engstrom et al., 2020; Andrychowicz et al., 2020) as minimal as possible we focus on small environments with a finite action space. This choice also enables more training seeds per experiment, improving statistical reliability (Agarwal et al., 2021). Experiments were conducted on the gymnax implementations (Lange, 2022) of Cartpole, Acrobot (Brockman et al., 2016), Catch, and DeepSea (Osband et al., 2019), representing a diverse set of tasks.

To evaluate a fixed algorithm A, i.e. an instance of MDPO(h, D) with specified temperatures and regularizers, on a metric d, we compute the mean across environments ($e = 1, \ldots, E$), training



(a) On standard environment suite (b) On upscaled versions of Catch (c) With APMD policy updates

Figure 1: Mean normalized return after training of MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) for different temperature levels. Each cell represents the normalized return averaged over environments, seeds and evaluations as described in (9). All heat maps exhibit the same principal "L" structure, highlighting regions where performance mainly depends on either of the regularization terms.

seeds ($n = 1, \ldots, N$), and post-training evaluations ($m = 1, \ldots, M$) (Agarwal et al., 2021)

$$d(A) = \frac{1}{E\,N\,M} \sum_{e=1}^{E} \sum_{n=1}^{N} \sum_{m=1}^{M} d_{e,n,m}(A), \tag{9}$$

where $d_{e,n,m}(A)$ is the metric value for A after training on environment e with seed n. The primary metric is the normalized return after training; although no reward normalization was applied during training, the obtained returns were linearly rescaled to the unit interval $[0, 1]$ to ensure comparability across environments. Further details are provided in Appendix C.

4.1 Interplay of MDP & Drift Regularizers

Baseline As a baseline we select the combination of the two well-studied regularizers $h = -\mathcal{H}$ and $D = D_{\mathrm{KL}}$, using constant temperature parameters during learning and across all environments. We performed minor hyperparameter tuning to ensure learnability with fixed temperature values.

Figure 1a shows results for varying values of α and λ. Despite differences in reward structures, a joint region of well-performing temperature values emerges across environments, highlighting an overlap in optimal temperature values. Moreover, the "L" structure of this region indicates that Entropy and KL regularization may be substitutable: for sufficiently low λ, performance depends primarily on α, and vice versa; a pattern that persists when scaling up the Catch environment (Figure 1b). However, some degree of regularization remains necessary in all cases to successfully solve the given tasks.

Since the goal is to optimize the unregularized value function V^π, MDP regularization may be treated as a means to an end. This raises the question of whether policy updates should use the regularized value function (7). Homotopic PMD (Li et al., 2023), a special case of Approximate PMD (APMD; Lan 2023), provides theoretical justification for replacing the regularized Q-value function Q^π_α with the unregularized one Q^π. Figure 1c shows that, at least for the environments studied, this substitution has no noticeable effect.

Varying MDP regularizer and Drift How does the performance landscape evolve with different MDP regularizers h? Furthermore, is it beneficial to pair the drift term D with its corresponding Bregman divergence $D = B_h$, as suggested in Zhan et al. (2023)? To explore these questions, we replicated the previous experiments using a range of choices for h and D.

The results are summarized in Figure 2, showing the proportion of tested temperature choices, that lead to a performance above a certain threshold, denoted as performance frequency. Notably, for any

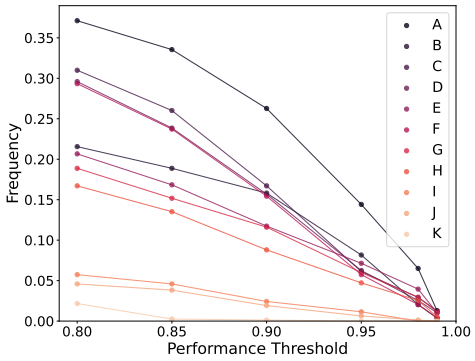


Figure 2: Performance frequency curves for MDPO(h, D) instances, showing the proportion of 784 temperature configurations reaching each performance level. While high-performance regions (≥ 0.98) narrow similarly, regularizers h and D significantly affect the frequency of achieving $\geq 90\%$ performance. Labels correspond to table 1.

Table 1: Robustness Measure of MDPO(h, D) for different pairs (h, D) in descending order, representing the (normalized) value under the curves in Figure 2. Robustness appears to depend on the pair of regularizers rather than on the influence of one alone.

#	h	D	$\mathrm{Rbst}_{0.9}$	$\mathrm{Rbst}_{0.95}$
A	$-\mathcal{H}$	$B_{\max}$	0.1403	0.0728
B	$\max$	$B_{\max}$	0.0801	0.0339
C	$-\mathcal{H}$	D_{KL}	0.0747	0.0327
D	$-\mathcal{H}$	$B_{-\mathcal{H}_{1.5}}$	0.0698	0.0313
E	$\|\|\cdot\|\|_2^2$	D_{KL}	0.0680	0.0390
F	$-\mathcal{H}$	$B_{-\mathcal{H}_{0.5}}$	0.0628	0.0261
G	$\|\|\cdot\|\|_2^2$	$B_{\|\|\cdot\|\|_2^2}$	0.0598	0.0310
H	$\max$	D_{KL}	0.0457	0.0231
I	$-\mathcal{H}_{0.5}$	$B_{-\mathcal{H}_{0.5}}$	0.0101	0.0032
J	$-\mathcal{H}$	$B_{\|\|\cdot\|\|_2^2}$	0.0078	0.0027
K	$-\mathcal{H}_{0.5}$	D_{KL}	0.0001	0.0000

combination, we could still find temperature values for α and λ yielding an average performance ≥ 0.95, confirming the validity of all these theoretically grounded choices.

While the heat maps (obtained similar to the baseline; see supplementary material) exhibit similar structures to the baseline case, with distinct λ- and α-dominant regions, the size of well-performing regions varies, indicating an effect on the robustness w.r.t. temperature selection. To quantify this intuition of robustness, we employ a simple metric $\mathrm{Rbst}_{\mathcal{T}}(\mathcal{A}; \Phi)$, that estimates the probability of a randomly chosen hyperparameter configuration $\phi \in \Phi$ achieving a performance of at least $\mathcal{T}$ across all evaluated environments. Further details can be found in Appendix B.

Table 1 presents robustness measures for different (h, D) pairs in descending order; the definitions of h and D are provided in Appendix A. Interestingly, the Bregman Divergence derived from the $\max$ function seems to exhibit the highest robustness, despite being neither smooth nor commonly used. However, robustness appears to depend more on the combination of regularizers than on any single component. For instance, pairing entropy with different Drift regularizers results in both very high and very low robustness values, respectively.

4.2 Temperature handling during Training

We now extend our analysis beyond fixed temperature parameters and test whether the adaptive strategies introduced in Section 3 lead to improved performance. For the temperature λ of the Drift regularizer we additionally test linear annealing, paired with either linear annealing or a constant MDP regularization temperature α. Additionally, we allow α to be learned via the loss in (8), using either a linearly decaying $\bar{h}$ ("learned lin. anneal") or a constant $\bar{h}$ ("learned constant"). For linear annealing, the initial temperature was gradually reduced to zero, while for both versions of learned α, the target $\bar{h}$ was varied.

To evaluate these strategies, we repeated these experiments for two pairs of h and D. The right part of Figure 3 shows the average performance of the top 1% of temperature configurations (out of 784) on regions where both regularizations apply ($\alpha, \lambda > 0$). The left and right of each pair of columns shows the results for MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) and MDPO($-\mathcal{H}_{0.5}, D_{-\mathcal{H}_{0.5}}$), respectively. Similarly, the right block shows the same results for the top 10% temperature configurations. These specific percentiles were selected to reflect the typical effort researchers might invest in hyperparameter tuning.

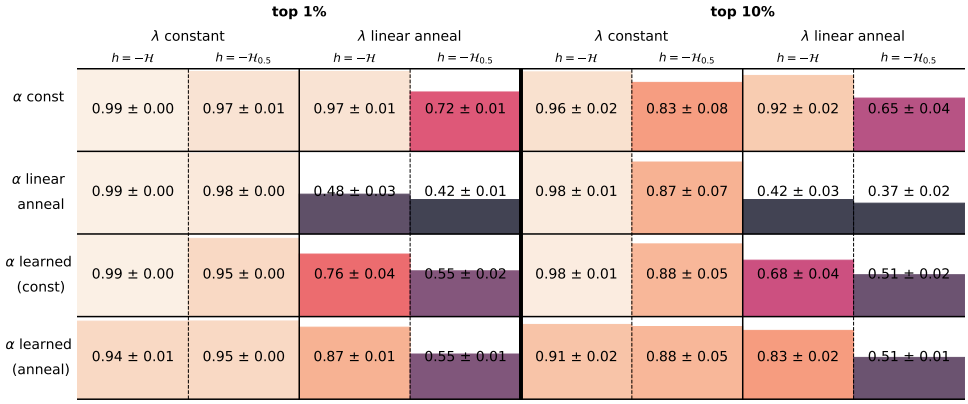


Figure 3: Mean & standard deviations of Top 1% and Top 10% performing hyperparameter configurations (out of 841) for MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) and MDPO($-\mathcal{H}_{0.5}, D_{-\mathcal{H}_{0.5}}$) for different temperature adaption schemes. For both algorithms keeping λ constant outperforms the linear annealing variant.

The results support theoretical predictions that a constant λ performs well, while linearly annealing λ significantly degrades performance at both the 1% and 10% levels. This contrasts with Tomar et al. (2020), where a decaying λ was found beneficial; however, their results were obtained in the absence of MDP regularization. In contrast, α handling has little impact on performance, possibly due to its smaller scale, making it more robust to tuning. Notably, these trends preserve even when changing both the MDP and Drift regularizer.

4.3 Temperature handling between different environments

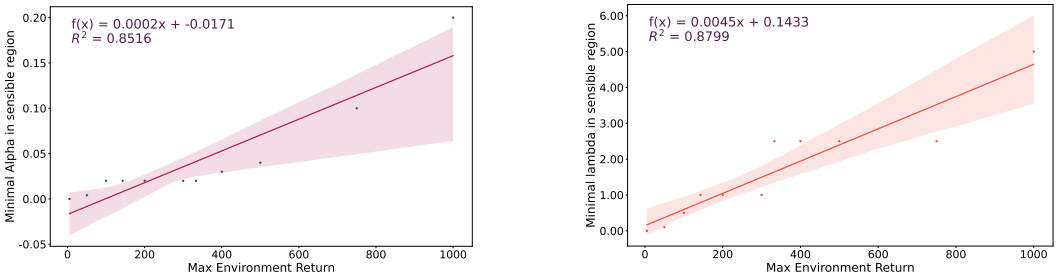
In practice, researchers also need to find an appropriate range to tune their parameters in, which can be challenging and require much intuition. However, from (7), we hypothesize that the preferred range of α is related to the absolute range of returns, rendering this choice heavily task dependent. To study this effect, we reran the experiments for MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) for 841 different constant temperature pairs each on a set of multiplicatively rescaled versions of CartPole, spanning a maximum return range from 5 to 1000.

Figure 4 illustrates the minimum temperature required for successful learning (normalized return ≥ 0.85) as a function of the environment's maximum return. Extracting this value was restricted to the region, where this choice was meaningful, that is, the choice for α was done in regions for sufficiently low λ and vice versa. Full heat maps are provided in the supplementary material.

These experiments indeed confirm empirically that both the temperature for the MDP regularizer α as well as the temperature for the Drift regularizer λ grow linearly with the maximum return obtainable. This can help RL researchers with judging the empirical range they need to test in for setting optimal temperature values, aiding in speeding up the hyperparameter optimisation.

5 Related work

Unified View on Reinforcement Learning Algorithms Studying regularization is partially inspired by attempts to provide a unified view on RL algorithms. Key milestones include the introduction of MDPO in (Geist et al., 2019), later explored in (Tomar et al., 2020; Vieillard et al., 2020), and the Mirror Learning framework in (Grudzien et al., 2022), which extends regularization beyond Mirror Descent. Recent work on Policy Mirror Descent (Lan, 2023; Xiao, 2022; Vaswani et al., 2022; Alfano et al., 2023; Zhan et al., 2023) has further detailed the connection between model-free algorithms and mirror descent.



(a) MDP Regularizer temperature α

(b) Drift Regularizer temperature λ

Figure 4: Minimal required temperature for successful learning (normalized return ≥ 0.75) as a function of maximum return in a rescaled CartPole environment. Points represent minimum temperature values; lines depict linear regressions, and shaded areas indicate 95% confidence intervals. The minimal temperatures exhibit a monotonic trend in the maximum environment return

Regularization in Policy Mirror Descent like Algorithms While RL algorithms can be regularized using techniques from supervised learning, e.g., weight decay, layer normalization (Nauman et al., 2024; Lee et al., 2024; Nauman et al., 2025), this work focuses on RL-specific regularization. From a Generalized Policy Iteration (GPI) perspective, regularization can be applied to three principal components: (i) the MDP itself, including entropy-based and KL-regularized RL (Ziebart, 2010; Haarnoja et al., 2017; Lee et al., 2019; Shani et al., 2020; Rudner et al., 2021; Tiapkin et al., 2023), (ii) policy evaluation (critic regularization) (Fujimoto et al., 2018; Nachum et al., 2019; Eysenbach et al., 2023; Cetin & Celiktutan, 2023), and (iii) policy improvement (Grudzien et al., 2022; Lan, 2023; Xiao, 2022; Vaswani et al., 2022; Alfano et al., 2023; Zhan et al., 2023). Unlike prior work on learning optimal Drift regularizers (Lu et al., 2022; Alfano et al., 2024), this study examines fixed regularizers for policy improvement.

6 Discussion & Conclusion

In this work, we empirically analyzed the interplay between the two regularization components in PMD: Drift and MDP regularization. Across 500k training seeds, we systematically examined a broad range of regularizers and their temperature parameters.

To ensure comprehensive coverage of algorithm configurations, our study focused on a small set of environments with a limited number of environment steps, meaning our findings reflect performance within this training horizon. Future research could extend this analysis to longer training horizons and larger-scale environments to assess whether the observed trends persist.

Our results indicate that, for most combinations, Drift and MDP regularization can act as substitutes, as reflected in the "L"-shaped regions of well-performing temperature values. Selecting well-suited temperature values is vital, as our results demonstrate that even in small environments, poorly chosen parameters can severely degrade performance. To address this brittleness, we examined the impact of temperature scales both during training and across environments, confirming a linear relationship between temperature and reward scale for both regularizers. Additionally, we evaluated dynamic adaptation strategies for temperature parameters and found that maintaining constant values is often more effective, particularly when moving beyond the conventional Entropy-KL regularization pair. This raises the question of whether existing adaptation strategies are overly tailored to well-established regularizers. Another approach to mitigating this brittleness is through improved robustness with respect to temperature selection. Our findings suggest that the choice of regularizers plays an important role in robustness, particularly for less commonly used ones, highlighting an important direction for future research. Overall, our study underscores the need for further investigation into regularization in RL to develop not only more performant but also more robust algorithms[2].

[2]Code is available on Github at https://github.com/JFKleuker/RegEffectsPMD

A Convex Functions and their Bregman Divergences

A.1 Entropy

The Shannon entropy of a (finite) probability distribution $p \in \Delta(\mathcal{X})$ is defined as

$$\mathcal{H}(p) = \sum_{x \in \mathcal{X}} -p(x)\, \log(p(x)) = \mathbb{E}_{x \sim p}[-\log(p(x))]. \tag{10}$$

The Tsallis Entropy for $m \neq 1$ is defined as

$$\mathcal{H}_m(p) = \frac{1}{m-1} \sum_{x \in \mathcal{X}} p(x) - p(x)^m. \tag{11}$$

For $m > 0$ $h(p) = -\mathcal{H}_m(p)$ is a convex function. Moreover, the Tsallis Entropy generalizes the Shannon Entropy in the sense, that $\lim_{m \to 1} \mathcal{H}_m = \mathcal{H}$.

The Bregman Divergence for a convex potential function is defined as

$$B_h(p, q) = h(p) - h(q) - \langle \nabla h(q), p - q \rangle, \tag{12}$$

where $h(q)$ is any vector within the subdifferential and $\langle \cdot, \cdot \rangle$ denotes the standard inner product on $\mathbb{R}^{|\mathcal{X}|}$. A straight forward calculation yields

$$B_{-\mathcal{H}_m}(p, q) = \frac{1}{m-1} \sum_{x \in \mathcal{X}} p(x)^m - m p(x) q(x)^{m-1} - (1-m) q(x)^m. \tag{13}$$

While $|\mathcal{H}(p)| \leq \log(|\mathcal{X}|)$, the Tsallis Entropy for $m \neq 1$ can be bounded on $\Delta(\mathcal{X})$ by

$$|\mathcal{H}_m(p)| \leq \begin{cases} 1/(m-1) & m > 1 \\ |\mathcal{X}|/(1-m) \max_{y \in [0,1]} |y - y^m| & 0 < m < 1. \end{cases} \tag{14}$$

This can be used to provide a reasonable guess for setting $\bar{h}$ in Eq. (8).

A.2 L_p Norm

Let $\mathcal{X}$ be a finite set, $q \in \Delta(\mathcal{X})$. Then

$$|| \cdot ||_p^p : \Delta(\mathcal{X}) \to \mathbb{R}, \tag{15}$$

$$q \mapsto ||q||_p^p := \sum_{x \in \mathcal{X}} |q(x)|^p \tag{16}$$

is a convex function for $p \geq 1$. The corresponding Bregman Divergence for $q, q' \in \Delta(\mathcal{X})$ is given by

$$B_{||\cdot||_p^p}(q, q') = \sum_{x \in \mathcal{X}} q(x)^p - q'(x)^p - p\, q(x)\, q'(x)^{p-1} + p\, q'(x)^p \tag{17}$$

A.3 Max Function

Let $\mathcal{X}$ be a finite set, $q \in \Delta(\mathcal{X})$. Then

$$\max : \Delta(\mathcal{X}) \to \mathbb{R}, \tag{18}$$

$$q \mapsto \max_{x \in \mathcal{X}} |q(x)| = \max_{x \in \mathcal{X}} q(x), \tag{19}$$

is a convex function. This function is not smooth, however, it is convex with subdifferential

$$\partial \max(q) = \left\{ \sum_{j \in J(x)} \alpha_j e_j : \sum_{j \in J(x)} \alpha_j = 1, J(x) = \arg\max_{x \in \mathcal{X}} q(x) \right\}, \tag{20}$$

where e_j denotes the j-th unit vector in $\mathbb{R}^{|\mathcal{X}|}$. The corresponding Bregman Divergence for $q, q' \in \Delta(\mathcal{X})$ can hence be expressed as

$$B_{\max}(q, q') = \max(q) - \max(q') - \langle \nabla \max(q'), q - q' \rangle, \tag{21}$$

where we can canonically select $\nabla \max(q') = e_j$ for some $j \in \arg\max_{x \in \mathcal{X}} q'(x)$.

B Robustness measure of an algorithm

To quantify robustness of an algorithm $\mathcal{A}_\phi$ (in this work an instance of MDPO(h, D) w.r.t. a set of hyperparameters Φ (in this work the temperature levels), we define the performance frequency as the proportion of hyperparameter configurations achieving at least a given performance τ:

$$\text{freq}(\tau; \mathcal{A}, \Phi) = \frac{\#\{\phi \in \Phi | d(\mathcal{A}_\phi) \geq \tau\}}{|\Phi|}. \tag{22}$$

Intuitively, the frequency to a given performance level τ may be seen as the probability of achieving at least this performance level for a random selection of hyperparameters (within Φ). In this sense, the normalized area under this curve would provide a measure of robustness: the likelihood of a random configuration achieving at least a certain performance

$$\text{Rbst}_{\mathcal{T}}(\mathcal{A}; \Phi) := \frac{1}{1 - \mathcal{T}} \int_{\mathcal{T}}^{1} \text{freq}(\tau; \mathcal{A}, \Phi) \, d\tau. \tag{23}$$

C Experiment Details

Unless stated otherwise, all algorithms were trained for an equal number of environment steps across four environments—CartPole, Acrobot, Catch, and DeepSea—using their standard implementations from Gymnax (Lange, 2022). No reward normalization was applied during training. However, to enable averaging of performance across environments, returns were linearly rescaled to the unit interval $[0, 1]$ using

$$\text{Return}_{\text{rescaled}} = \frac{\text{Return} - R_{\min}}{R_{\max} - R_{\min}},$$

where $R_{\max}$ and $R_{\min}$ denote the maximum and minimum attainable returns for each environment, respectively. By design, these were $\{500, 0\}$ for **CartPole**, $\{1, -1\}$ for **Catch**, and $\{1, 0\}$ for **DeepSea**. For **Acrobot**, we considered the task solved at a return of -75, slightly outperforming the PPO baseline in Gymnax, and used $\{R_{\max}, R_{\min}\} = \{-75, -500\}$ for normalization.

For each algorithm configuration, defined by a specific set of temperature parameters for a given MDP regularizer h and Drift regularizer D, we ran $N = 5$ training seeds per environment, followed by $M = 10$ evaluations per trained model. In all experiments, a total of $29^2 = 841$ temperature configurations were tested for each instance of MDPO(h, D). For constant temperature settings, the specified value refers directly to the temperature parameter; for linearly annealed settings, it refers to the initial value.

For the Drift regularizer temperature λ, we tested the following values:

$$\lambda \in \{0.0,\ 5{\cdot}10^{-5},\ 7.5{\cdot}10^{-5},\ 10^{-4},\ 2.5{\cdot}10^{-4},\ 5{\cdot}10^{-4},\ 7.5{\cdot}10^{-4},\ 10^{-3},\ 5{\cdot}10^{-3},\ 10^{-2},\ 2.5{\cdot}10^{-2},\ 5{\cdot}10^{-2},\ 10^{-1},\ 2.5{\cdot}10^{-1},\ 5{\cdot}10^{-1},\ 1,\ 2.5,\ 5,\ 7.5,\ 10,\ 25,\ 50,\ 10^{2},\ 5{\cdot}10^{2},\ 10^{3},\ 2.5{\cdot}10^{3},\ 5{\cdot}10^{3},\ 10^{4},\ 5{\cdot}10^{4}\}.$$

For the MDP regularizer temperature α, we used:

$$\alpha \in \{0.0,\ 0.001,\ 0.002,\ 0.003,\ \ldots,\ 0.009,\ 0.01,\ 0.02,\ 0.03,\ \ldots,\ 0.1,\ 0.2,\ 0.3,\ \ldots,\ 1.0\},$$

where values increment in steps of 0.001 from 0.001 to 0.009, then in coarser steps up to 1.0 (29 values in total).

In the case of a learned MDP regularizer temperature α with a constant target, the target was defined as $\bar{h} = w \cdot \bar{h}_0$, where $\bar{h}_0$ is an environment-specific reference value and w was varied over the same 29 values used for α. For learned parameters with linearly annealed targets of the form $\bar{h}_k = w_k \cdot \bar{h}_0$, the initial weight w_0 was similarly varied over the same set.

On each environment, we ran 841 temperature configurations with 5 seeds each, totaling 4205 runs per environment and algorithm. In addition to the baseline setup (4 environments), we evaluated 10 variations of MDPO(h, D) on four environments each (40 additional environments). The baseline algorithm was also run on a scaled-up version of Catch (Figure 1b), where the number of rows and columns was increased by factors of two and three, respectively, compared to the default configuration (2 additional environments). Furthermore, an AMPD-inspired variant was evaluated on a set of 4 environments (Figure 1c; 4 additional environments). To study the influence of temperature scaling, 15 additional algorithm configurations were tested on four environments each (Figure 3; 60 additional environments). Finally, we ran the baseline algorithm on 10 rescaled versions of CartPole (10 additional environments), resulting in a total of $120 \times 4205 = 504{,}600$ training seeds.

D Algorithm Details

Table 2: Fixed hyperparameters for all MDPO(h, D) instances

Parameter	Value
Number of environments	16
Max grad norm	1.0
Gamma	0.99
Replay buffer size	10^5
Environment steps per update	256
Train batch size	512
Critic update epochs	1
Actor update epochs	2
Tau	0.95
Learning rates	0.0025
Total Environment steps	10^6

Algorithm 1: Off-policy MDPO(h, D) Algorithm

Init networks $(Q_{\phi_0,i})_{i=1,2}, \pi_{\theta_0}$, data buffer $D = \emptyset$, target networks $\phi_{\text{target, i}} = \phi_{0,i}$
for *budget* **do**
  for *environment steps per update* **do**
    /* sample data */
    sample $(s, a, r, s', d) \sim \pi_{\theta_k}, p$,
    $D \leftarrow D \cup \{(s, a, r, s', d)\}$
  end
  Sample train batch $\mathcal{D} \sim D$

  /* Policy Improvement: update Actor network, cf. (7) */
  $L(\theta, \theta_k) = \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[-Q^{\phi_k}_{\alpha_k}(s, a)] + \alpha_k\, h((\pi_\theta(\cdot|s)) + \lambda_k\, D(\pi_\theta; \pi_{\theta_k}|s)\right]$
  for $i = 0, \ldots,$ actor epochs $- 1$ **do**
    $\theta_k^{(i+1)} \leftarrow \theta_k^{(i)} - \eta_\theta \nabla_\theta L(\theta, \theta_k)|_{\theta=\theta_k^{(i)}}$
  end
  $\theta_{k+1} = \theta_k^{\text{actor epochs}}$

  /* Policy Evaluation */
  $L_Q(\phi) =$
   $\mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}[(Q_\phi(s, a) - \{r + \gamma\, \mathbb{E}_{a'\sim\pi_{\theta_{k+1}}}[\min Q_{\phi_{\text{target,i}}}(s', a')] - \alpha_k\, h(\pi_{\theta_{k+1}}(\cdot|s')\})^2]$
   for $j = 0, \ldots,$ critic epochs $- 1$ **do**
    $\phi_k^{(j+1)} \leftarrow \phi_k^{(j)} - \eta_\phi \nabla_\phi L_Q(\phi)|_{\phi=\phi_k^{(i)}}$
  end
  $\phi_{k+1} = \phi_k^{\text{critic epochs}}$

  $\phi_{\text{target, k+1}} = \tau\, \phi_{\text{target, k}} + (1 - \tau)\, \phi_{k+1}$
end

Acknowledgments

The authors thank Koen Ponse and Jakob Schuhmacher for their invaluable feedback during the process of this work. This work was supported by Shell Information Technology International Limited and the Netherlands Enterprise Agency under the grant PPS23-3-03529461.

References

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32:96, 2019.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29304–29320. Curran Associates, Inc., 2021.

Carlo Alfano, Rui Yuan, and Patrick Rebeschini. A novel framework for policy mirror descent with general parameterization and linear convergence. *Advances in Neural Information Processing Systems*, 36:30681–30725, 2023.

Carlo Alfano, Sebastian Rene Towers, Silvia Sapora, Chris Lu, and Patrick Rebeschini. Learning mirror maps in policy mirror descent. In *Seventeenth European Workshop on Reinforcement Learning*, 2024.

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphael Marinier, Léonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. What

matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*, 2020.

Amir Beck. *First-order methods in optimization*. SIAM, 2017.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, May 2003. ISSN 01676377. DOI: 10.1016/S0167-6377(02)00231-6.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL http://arxiv.org/abs/1606.01540.

Edoardo Cetin and Oya Celiktutan. Learning Pessimism for Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6):6971–6979, June 2023. ISSN 2374-3468, 2159-5399. DOI: 10.1609/aaai.v37i6.25852.

Yinlam Chow, Ofir Nachum, and Mohammad Ghavamzadeh. Path consistency learning in tsallis entropy regularized mdps. In *International conference on machine learning*, pp. 979–988. PMLR, 2018.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*, 2020.

Benjamin Eysenbach, Matthieu Geist, Sergey Levine, and Ruslan Salakhutdinov. A Connection between One-Step RL and Critic Regularization in Reinforcement Learning. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 9485–9507. PMLR, July 2023.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A Theory of Regularized Markov Decision Processes. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2160–2169. PMLR, May 2019.

Jakub Grudzien, Christian A Schroeder De Witt, and Jakob Foerster. Mirror learning: A unifying framework of policy optimisation. In *International Conference on Machine Learning*, pp. 7825–7844. PMLR, 2022.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement Learning with Deep Energy-Based Policies. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1352–1361. PMLR, July 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1861–1870. PMLR, July 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Guanghui Lan. Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes. *Mathematical Programming*, 198(1):1059–1106, March 2023. ISSN 1436-4646. DOI: 10.1007/s10107-022-01816-5.

Guanghui Lan, Zhaosong Lu, and Renato DC Monteiro. Primal-dual first-order methods with iteration-complexity for cone programming. *Mathematical Programming*, 126(1):1–29, 2011.

Robert Tjarko Lange. gymnax: A JAX-based reinforcement learning environment library, 2022. URL http://github.com/RobertTLange/gymnax.

Hojoon Lee, Dongyoon Hwang, Donghu Kim, Hyunseung Kim, Jun Jet Tai, Kaushik Subramanian, Peter R Wurman, Jaegul Choo, Peter Stone, and Takuma Seno. Simba: Simplicity bias for scaling up parameters in deep reinforcement learning. *arXiv preprint arXiv:2410.09754*, 2024.

Kyungjae Lee, Sungyub Kim, Sungbin Lim, Sungjoon Choi, and Songhwai Oh. Tsallis reinforcement learning: A unified framework for maximum entropy reinforcement learning. *arXiv preprint arXiv:1902.00137*, 2019.

Yan Li, Guanghui Lan, and Tuo Zhao. Homotopic policy mirror descent: policy convergence, algorithmic regularization, and improved sample complexity. *Mathematical Programming*, pp. 1–57, 2023.

Boyi Liu, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural trust region/proximal policy optimization attains globally optimal policy. *Advances in neural information processing systems*, 32, 2019.

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 35:16455–16468, 2022.

Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

Michal Nauman, Michał Bortkiewicz, Piotr Miłoś, Tomasz Trzciński, Mateusz Ostaszewski, and Marek Cygan. Overestimation, overfitting, and plasticity in actor-critic: the bitter lesson of reinforcement learning. *arXiv preprint arXiv:2403.00514*, 2024.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. *Advances in Neural Information Processing Systems*, 37:113038–113071, 2025.

Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, et al. Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568*, 2019.

Tim GJ Rudner, Cong Lu, Michael A Osborne, Yarin Gal, and Yee Teh. On pathologies in kl-regularized reinforcement learning from expert demonstrations. *Advances in Neural Information Processing Systems*, 34:28376–28389, 2021.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

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

Lior Shani, Yonathan Efroni, and Shie Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5668–5675, 2020.

Richard S Sutton. Reinforcement learning: an introduction. *A Bradford Book*, 2018.

Daniil Tiapkin, Denis Belomestny, Daniele Calandriello, Eric Moulines, Alexey Naumov, Pierre Perrault, Michal Valko, and Pierre Menard. Regularized rl. *arXiv preprint arXiv:2310.17303*, 2023.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. *arXiv preprint arXiv:2005.09814*, 2020.

Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Müller, Shivam Garg, Matthieu Geist, Marlos C Machado, Pablo Samuel Castro, and Nicolas Le Roux. A general class of surrogate functions for stable and efficient reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 8619–8649. PMLR, 2022.

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, and Matthieu Geist. Leverage the average: an analysis of kl regularization in rl. *arXiv preprint arXiv:2003.14089*, 2020.

Lin Xiao. On the convergence rates of policy gradient methods. *Journal of Machine Learning Research*, 23(282):1–36, 2022.

Wenhao Zhan, Shicong Cen, Baihe Huang, Yuxin Chen, Jason D Lee, and Yuejie Chi. Policy mirror descent for regularized reinforcement learning: A generalized framework with linear convergence. *SIAM Journal on Optimization*, 33(2):1061–1091, 2023.

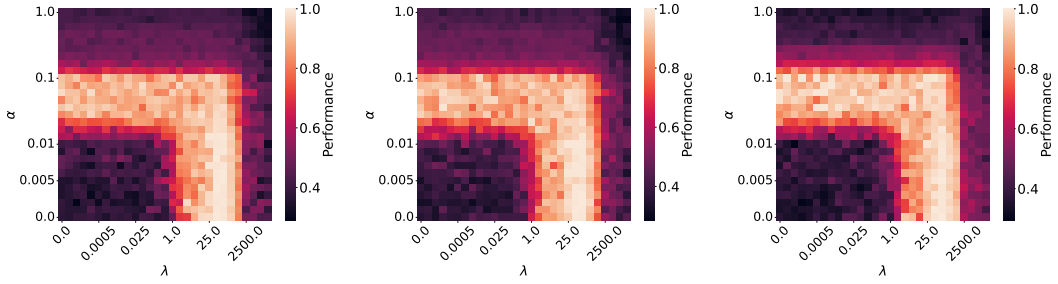
Brian D Ziebart. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

Supplementary Materials

The following content was not necessarily subject to peer review.

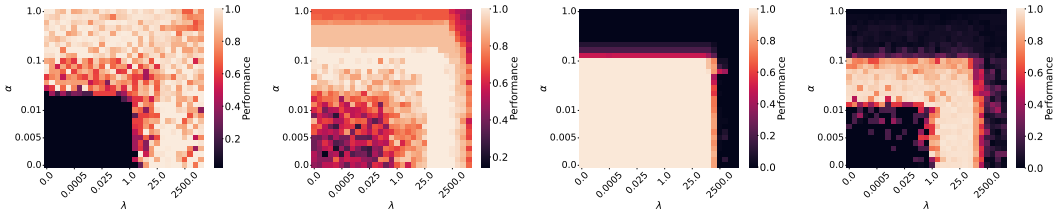
E Additional Plots

Additional Heat maps



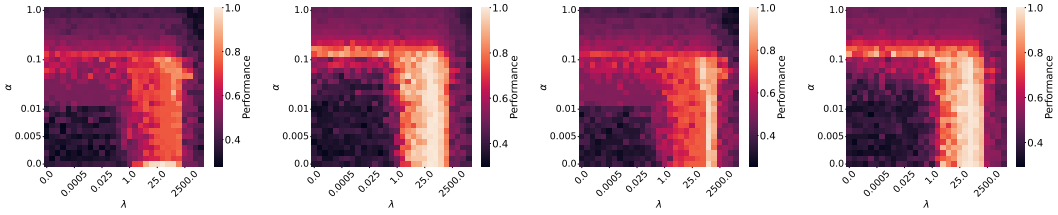
(a) $h = -\mathcal{H}, D = D_{\mathrm{KL}}$ (b) $h = -\mathcal{H}, D = D_{\mathrm{fKL}}$ (c) $h = -\mathcal{H}, D = D_{\mathrm{KL}}$; AMPD

Figure 5: Different MDPO(h, D) configurations with constant temperatures



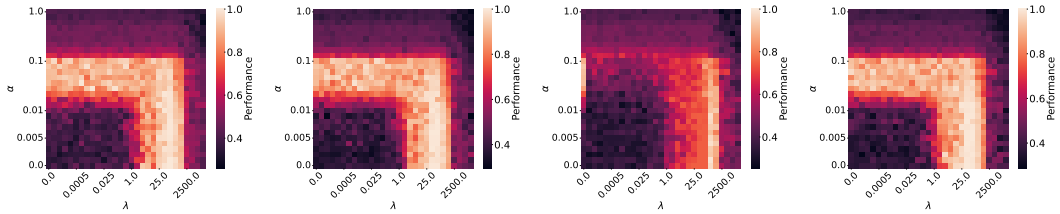
(a) CartPole (b) Acrobot (c) Catch (d) DeepSea

Figure 6: MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) with constant temperatures on different environments



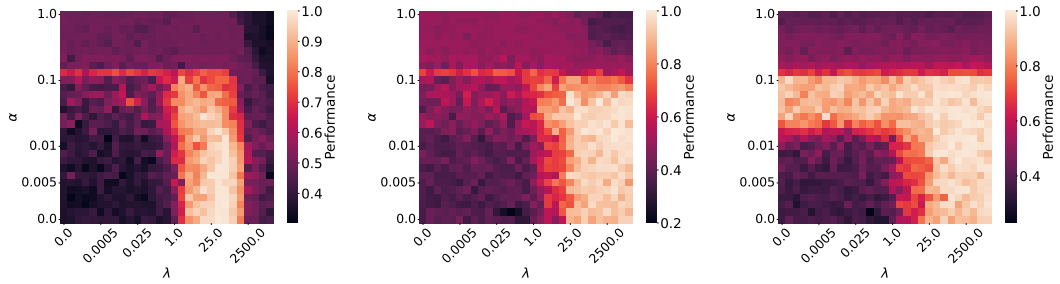
(a) $h = -\mathcal{H}_{0.5}$ & $D = D_{\mathrm{KL}}$ (b) $h = ||\cdot||_2^2$ & $D = D_{\mathrm{KL}}$ (c) $h = -\mathcal{H}_{0.5}$& $D = B_{-\mathcal{H}_{0.5}}$ (d) $h = ||\cdot||_2^2$ & $D = B_{L2}$

Figure 7: MDPO(h, D) for different h, D pairs



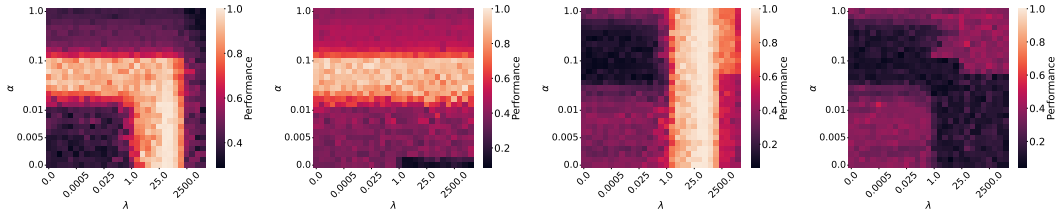
(a) $h = -\mathcal{H}$ & $D = B_{-\mathcal{H}_{0.5}}$ (b) $h = -\mathcal{H}$ & $D = B_{-\mathcal{H}_{1.5}}$ (c) $h = -\mathcal{H}$ & $D = B_{L2}$ (d) $h = -\mathcal{H}$ & $D = D_{\mathrm{KL}}$

Figure 8: MDPO(h, D) for different h, D pairs



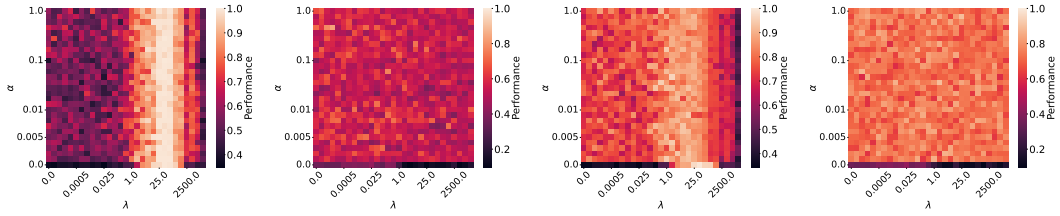
(a) $h = \max$ & $D = D_{\mathrm{KL}}$ (b) $h = \max$ & $D = B_{\max}$ (c) $h = -\mathcal{H}$ & $D = B_{\max}$

Figure 9: MDPO(h, D) for different h, D pairs



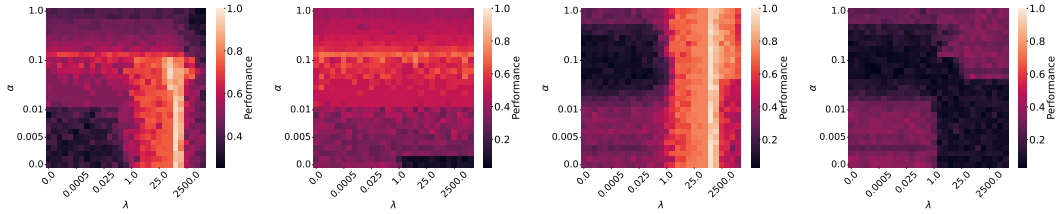
(a) α = const., λ = const. (b) α = const., λ = lin. anneal (c) α = lin. anneal, λ = const. (d) α = lin. anneal, λ = lin. anneal

Figure 10: MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) for different temperature scheduling schemes



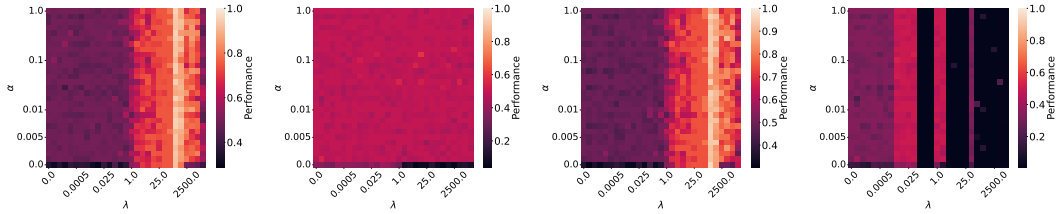
(a) α = learned const., λ = const. (b) α = learned const., λ = lin. anneal (c) α = learned lin. anneal, λ = const. (d) α = learned lin. anneal, λ = lin. anneal

Figure 11: MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) for different temperature scheduling schemes



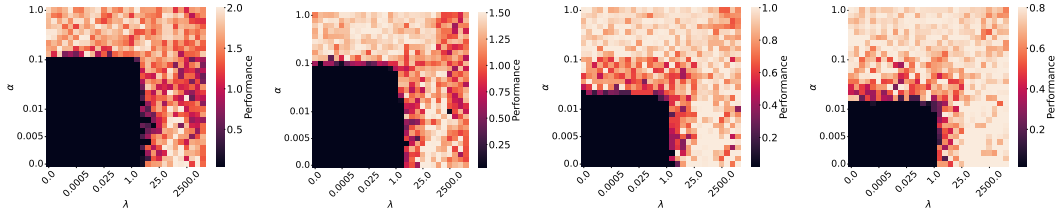
(a) α = const., λ = const. (b) α = const., λ = lin. anneal (c) α = lin. anneal, λ = const. (d) α = lin. anneal, λ = lin. anneal

Figure 12: MDPO($-\mathcal{H}_{0.5}, B_{-\mathcal{H}_{0.5}}$) for different temperature scheduling schemes



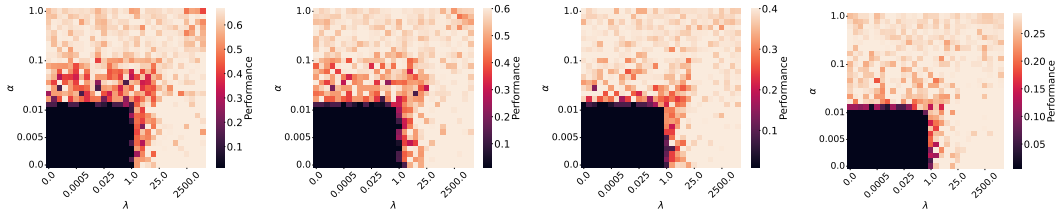
(a) α = learned const., λ = const. (b) α = learned const., λ = lin. anneal (c) α = learned lin. anneal, λ = const. (d) α = learned lin. anneal, λ = lin. anneal

Figure 13: MDPO($-\mathcal{H}_{0.5}, B_{-\mathcal{H}_{0.5}}$) for different temperature scheduling schemes



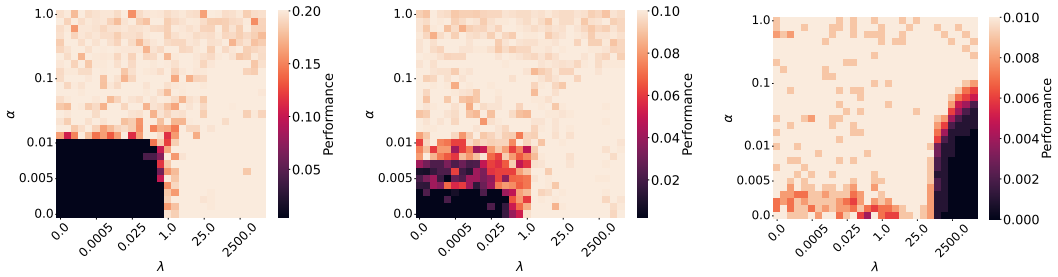
(a) Maximum Return = 1000 (b) Maximum Return = 750 (c) Maximum Return = 500 (d) Maximum Return = 400

Figure 14: MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) with constant temperatures on CartPole with different maximum returns



(a) Maximum Return = 333 (b) Maximum Return = 300 (c) Maximum Return = 200 (d) Maximum Return = 143

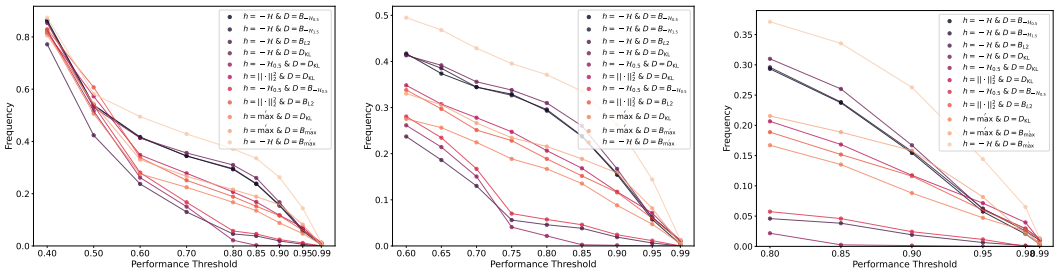
Figure 15: MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) with constant temperatures on CartPole with different maximum returns



(a) Maximum Return = 100 (b) Maximum Return = 50 (c) Maximum Return = 5

Figure 16: MDPO($-\mathcal{H}, D_{\mathrm{KL}}$) with constant temperatures on CartPole with different maximum returns

Robustness



(a) Starting at $\tau = 0.4$ (b) Starting at $\tau = 0.6$ (c) Starting at $\tau = 0.8$

Figure 17: Performance frequency curves starting at different performance levels