Beyond Expected Return: Accounting for Policy Reproducibility When Evaluating Reinforcement Learning Algorithms

Manon Flageat*, Bryan Lim*, Antoine Cully

Adaptive and Intelligent Robotics Lab, Imperial College London, UK
manon.flageat18@imperial.ac.uk, bryan.lim16@imperial.ac.uk, a.cully@imperial.ac.uk

Abstract

Many applications in Reinforcement Learning (RL) usually have noise or stochasticity present in the environment. Beyond their impact on learning, these uncertainties lead the exact same policy to perform differently, i.e. yield different return, from one roll-out to another. Common evaluation procedures in RL summarise the consequent return distributions using solely the expected return, which does not account for the spread of the distribution. Our work defines this spread as the *policy reproducibility*: the ability of a policy to obtain similar performance when rolled out many times, a crucial property in some real-world applications. We highlight that existing procedures that only use the expected return are limited on two fronts: first an infinite number of return distributions with a wide range of performance-reproducibility trade-offs can have the same expected return, limiting its effectiveness when used for comparing policies; second, the expected return metric does not leave any room for practitioners to choose the best trade-off value for considered applications. In this work, we address these limitations by recommending the use of Lower Confidence Bound, a metric taken from Bayesian optimisation that provides the user with a preference parameter to choose a desired performance-reproducibility trade-off. We also formalise and quantify *policy reproducibility*, and demonstrate the benefit of our metrics using extensive experiments of popular RL algorithms on common uncertain RL tasks.

1 Introduction

As Reinforcement Learning (RL) algorithms become increasingly competitive, attention is being shifted from toy and benchmark environments towards realistic everyday problems. However, real-world applications are never fully observable, resulting in various sources of uncertainty: imperfect actuators and sensors, miscalibration, user mental state, system wear and tear, etc. These uncertainties directly impact the policies as they cause stochasticity in the dynamics, rewards, actions as well as observations (Cassandra 1998; Dulac-Arnold et al. 2020). In this work, we highlight that when considering commonly-used uncertain settings, the final policy of a standard RL algorithm may have large variations in its performance, i.e. low reproducibility. We argue that evaluating and accounting for policy reproducibility is a critical and understudied issue in RL.

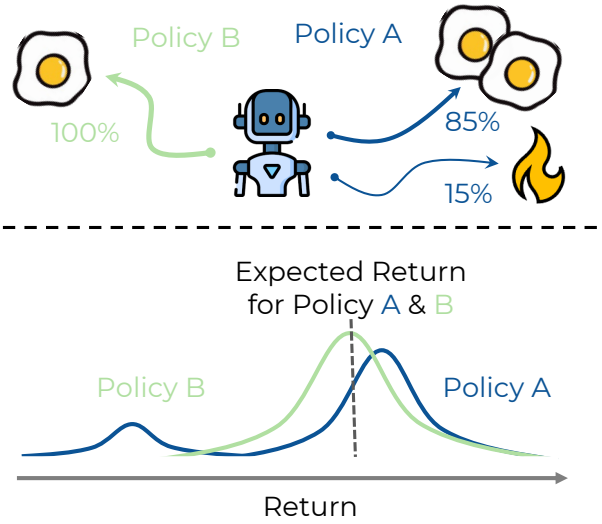

Figure 1: Illustration of the trade-off between policy reproducibility and performance. Policy A (blue) cooks the best-breakfast-ever-made 85% of the time, but burns all the eggs the remaining 15%, while Policy B (green) consistently cooks a lower-quality breakfast. On the distribution of returns (bottom), Policy A and B have the same expected return, highlighting the limitations of this commonly-used metric.

As a motivating example, let's take the hypothetical example of a breakfast-cooking robot, trained using RL to cook the best expected breakfast every morning (see Figure 1). A learnt robot Policy A might get a really high expected return by cooking the best-breakfast-ever-made 85% of the time, but burn all the eggs the remaining 15% (blue Policy in Figure 1). This would result in finding burned eggs in the user's plates approximately one day a week. In this example, it is likely that the user would prefer a slightly lower-quality breakfast but consistently every day of the week, such as proposed by Policy B (green Policy in Figure 1). In other words, the user would likely prefer a more reproducible policy. However, Policy A and Policy B both have the same expected return.

*Equal Contribution

More critically, the expected return metric does not leave any room to take into account this trade-off between quality and reproducibility, in order to choose the desired policy to deploy.

More practically, consider the evaluation of a policy during training or as a result of an RL algorithm. Common practice involves rolling out a policy N (i.e. $N \sim 200$) times and reporting the average return of the policy as the performance (Agarwal et al. 2021). However, in the presence of uncertainty, these N evaluation roll-outs would each lead to different performances, among which could even be failure cases. Thus, these N roll-outs give an approximation of the distribution of returns of the policy (Figure 1), that the average return metric fails to capture fully. In this work, we formalise the spread of this distribution as *policy reproducibility*, which refers to the ability of a policy to obtain similar performance and behaviours when rolled out many times in uncertain domains. Here, we consider the standard *in-distribution* case where the noise and uncertainty during evaluation have the same distribution as during training, but our proposed metrics could also be applied in *out-of-distribution* cases such as sim-to-real transfer.

We argue that for RL to mature and successfully transition toward real-world applications, there is a need to move beyond reporting and evaluating just the expected (i.e. average) return, and start taking into account its trade-off with policy reproducibility. Accordingly, we propose to use the Lower Confidence Bound (LCB), taken from Bayesian Optimisation (Frazier 2018), as an alternative to the expected return. This metric provides practitioners with a preference parameter, which allows them to set the desired trade-off between expected performance and reproducibility. Additionally, the usual expected return can be recovered when this preference is set to zero. We believe this point is crucial for the effective deployment of RL-generated policies in real-world settings.

Prior work in RL has been concerned with the reproducibility of algorithms across seeds, and showed the importance of reporting statistically-meaningful metrics to ensure fair comparison of RL algorithms given a limited number of runs (Henderson et al. 2018; Agarwal et al. 2021). Our work aims to further complement the evaluation of RL algorithms by focusing on reproducibility at the level of the policy.

Our contributions are as follows: (1) we formalise the concept of policy reproducibility and propose the Mean Absolute Deviation (MAD) metric to quantify it, (2) we highlight a simple and easy-to-use alternative to the expected return that takes into account the trade-off between performance and reproducibility: the Lower Confidence Bound (LCB) of the return, (3) we demonstrate the benefit of the LCB metric via a thorough experimental analysis of common RL algorithms across multiple domains, (4) we extend our evaluation procedures to also consider behavioural reproducibility.

2 Background

2.1 Reinforcement Learning

Problem Statement In this work, we focus on solving Markov Decision Process (MDP) problems (Sutton et al. 2018), in which an agent aims to maximise a reward signal. At each timestep t of an MDP problem, the agent is in a state $s_t \in \mathcal{S}$, takes an action $a_t \in \mathcal{A}$, leading to a new state $s_{t+1} \in \mathcal{S}$. It then receives a reward $r_t = R(s_t, a_t, s_{t+1}) \in \mathbb{R}$ for this transition. The MDP is given as $< \mathcal{S}, \mathcal{A}, R, P >$ where $P : \mathcal{S}, \mathcal{A} \mapsto \mathcal{S}$ characterises the transitions dynamics. In this work, we consider policies represented as a deep neural network π_θ, parameterised by θ. These policies can either be deterministic, i.e. output a fix action, or stochastic, i.e. output a distribution over actions. Algorithms aim to find parameters θ of the the policy $\pi_\theta \in \Pi, s_t \mapsto a_t$ that maximises the cumulative reward over episode of length T: $\sum_{t=0}^{T} r_t$. In this work, we consider and study two different families of algorithms used in RL.

Value-based Methods use the gradient with respect to timestep reward as a learning signal to find the optimal policy. To do so, they usually approximate the state-action value function $Q^\pi : \mathcal{S}, \mathcal{A} \mapsto \mathbb{R}; s_t, a_t \mapsto \mathbb{E}_\pi \left(\sum_{t=0}^{T} \gamma^t r_t | s_t, a_t \right)$, which gives the expected discounted return for being in state s_t, taking action a_t and following policy π. Q is usually approximated using a deep neural network with parameter ϕ, referred to as the critic network Q_ϕ. The policy π_θ learns to maximise Q_ϕ following the policy gradient theorem (Sutton et al. 2018). We consider two popular algorithms: TD3 (Fujimoto et al. 2018), which uses deterministic policies, and SAC (Haarnoja et al. 2018), which uses stochastic policies.

Evolution Strategies (ES) use as a learning signal the (empirical) gradient with respect to the cumulative reward over the episode (Hansen 2006; Wierstra et al. 2014). ES maintain a parameterised search distribution over policy parameters θ. They update this distribution in the direction of higher cumulative reward by following the natural gradient obtained through sampling in the solution space from the search distribution. ES algorithms come in many variants and are classified based on the distribution parameterization and the update procedure. In this paper, we focus on the OpenAI-ES (Salimans et al. 2017) variant, shown to work well for our problem setting, which we refer to as just ES for simplicity.

2.2 Uncertainty

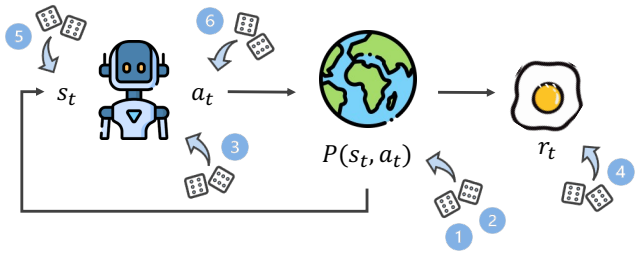

Figure 2: Illustration of the different uncertainties that can be applied within the RL evaluation setting: (1) stochastic dynamics, (2) random initialisation, (3) parameter-space noise, (4) reward noise, (5) observation noise and (6) action noise.

We consider environments with different types of noise and stochasticity, referred to more generally as uncertain. As the

issue of reproducibility only arises in such uncertain settings, this section provides an overview of the types and sources of uncertainty illustrated in Figure 2. For each type, we give the objective maximised by the RL agent, for example, when there is no uncertainty in the environment:

$$J(\theta) = \mathrm{E}_{s_0,P,\pi_\theta}\left(\sum_{t=0}^{T} R(s_t, a_t, s_{t+1})\right)$$

Stochastic Dynamics Stochastic dynamics has been long studied in the field of RL. The transition matrix P defined in Section 2.1 can be stochastic, where the same action from the same state can lead to different next states (Fig. 2.1):

$$J(\theta) = \mathrm{E}_{s_0,\tilde{T},\pi_\theta}\left[\sum_{t=0}^{T} R(s_t, a_t, s_{t+1})\right]$$
$$\text{with } \tilde{T} \text{ such that } \exists (i,j,k), \tilde{T}_{ij} > 0 \text{ and } \tilde{T}_{ik} > 0$$

Prior works in this setting focus on the issue of bias and early commitment (Fox et al. 2015; Hasselt 2010; Azar et al. 2011), where policies encountering lucky states early in learning spent lots of effort later unlearning biased estimates.

Random Initialisation Most environments in RL literature have random initialisation (Tassa et al. 2018) (Fig. 2.2):

$$J(\theta) = \mathrm{E}_{s_0+\epsilon,T,\pi_\theta}\left[\sum_{t=0}^{T} R(s_t, a_t, s_{t+1})\right], \quad \epsilon \sim \mathcal{N}(0,\sigma)$$

Unlike the previous one, this noise does not impact the learning capabilities of RL algorithms. On the contrary, it often proves useful by enhancing exploration.

Parameter-space Noise Literature in ES and, more generally, in Evolutionary Algorithms (EA) also commonly consider noise applied directly on the policy parameters θ (Jin et al. 2005; Lehman et al. 2018) (Fig. 2.3):

$$J(\theta) = \mathrm{E}_{s_0,T,\pi_{\theta+\epsilon}}\left[\sum_{t=0}^{T} R(s_t, a_t, s_{t+1})\right], \quad \epsilon \sim \mathcal{N}(0,\sigma)$$

ES algorithms which optimise based on perturbations in the parameter space have demonstrated greater performance in this setting than common RL methods (Lehman et al. 2018).

Reward Noise Another type of uncertainty considers additive noise directly on the reward value (Everitt et al. 2017; Wang et al. 2020; Romoff et al. 2018), coming for example from imperfect sensors (Fig. 2.4). Such noise directly impacts the optimisation and might lead to learning sub-optimal policies. Interestingly, this issue is also widely studied in EA and ES (Jin et al. 2005; Rakshit et al. 2017).

$$J(\theta) = \mathrm{E}_{s_0,T,\pi_\theta}\left[\sum_{t=0}^{T} R(s_t, a_t, s_{t+1}) + \epsilon_t\right], \quad \epsilon_t \sim \mathcal{N}(0,\sigma_t)$$

Reward noise is a particular case of the reward corruption problem defined in Everitt et al. (2017). However, while the noisy reward problem can be alleviated, Everitt et al. (2017) derived a no-free lunch theorem for reward corruption.

Observation Noise One can also consider noise on the state of the environment (Dulac-Arnold et al. 2020), coming for example from imperfect sensors (Fig. 2.5):

$$J(\theta) = \mathrm{E}_{s_0+\epsilon_0,T,\tilde{\pi_\theta}}\left[\sum_{t=0}^{T} R(s_t+\epsilon_t, a_t, s_{t+1}+\epsilon_{t+1})\right]$$
$$\epsilon_0 \sim \mathcal{N}(0,\sigma_0), \quad \epsilon_t \sim \mathcal{N}(0,\sigma_t), \quad \epsilon_{t+1} \sim \mathcal{N}(0,\sigma_{t+1})$$

Such noise is often considered in work interested in generalisation where it is integrated by the user as a way to enforce robust policies, for example in domain randomisation (Akkaya et al. 2019). While this case is slightly different from the one considered here, it still provides interesting insights into the convergence of policies in such domains.

Action Noise One can also consider action noise, coming for example, from imperfect actuators (Fig. 2.6) and acts similar to observation noise (Dulac-Arnold et al. 2020):

$$J(\theta) = \mathrm{E}_{s_0,T,\tilde{\pi_\theta}}\left[\sum_{t=0}^{T} R(s_t, a_t+\epsilon, s_{t+1})\right], \quad \epsilon \sim \mathcal{N}(0,\sigma)$$

3 Accounting for Policy Reproducibility

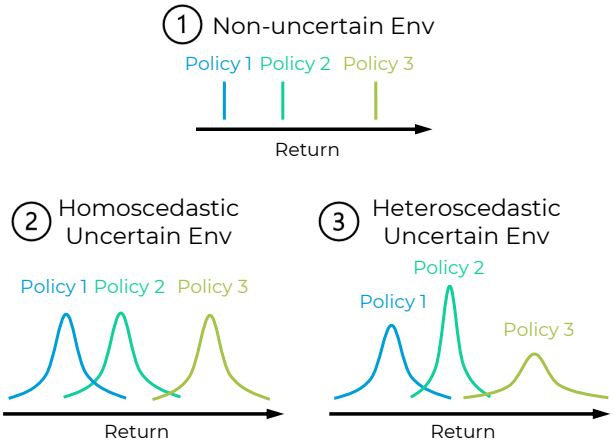

Figure 3: In non-uncertain environments (1), each policy has a fixed return; while in uncertain environments (2, 3), policies have distributions over possible returns. In homoscedastic uncertain environments (2) this distribution is the same for every policy, while in heteroscedastic uncertain environments (3), each policy can take different distribution parameters. The trade-off between policy reproducibility and performance only arises in heteroscedastic uncertain environments (3).

3.1 Uncertain Environments

Section 2.2 provided an overview of the different types of uncertainty that can apply in an environment, but they might not all have the same impact. *Non-uncertain environments* refer to environments where each policy is attributed a single return value in a deterministic manner: every roll-out of a policy leads to the exact same return value (case (1) in Figure 3). This occurs in the absence of uncertainties, or in the presence of uncertainties that do not propagate to the return, for example, uncertainties that impact only unreachable states. On the contrary, *uncertain environments* refer to environments where policies have distributions over return values: the same policy evaluated multiple times gets different return values from one roll-out to another (case (2) and (3) in Figure 3).

Homo/Hetero-scedastic Uncertain Environments. Uncertain environments can be further divided into two categories. *Homoscedastic uncertain environments* refers to environments where the uncertainty impacts the returns of all

policies in the same way, while in *heteroscedastic uncertain environments*, it depends on the policy considered. In other words, both types of environments are uncertain, but in homoscedastic ones, the return of all policies have the same distribution shape (case (2) in Figure 3) while in heteroscedastic ones, the return distributions of policies can be different (case (3) in Figure 3). An example of a homoscedastic uncertain environment would be an environment where the only source of noise is a noisy sensor that adds a constant white noise to the reward. In this case, all policies would get the same return distribution spread, no matter their performance. An example of a heteroscedastic uncertain environment would be initial state noise as this uncertainty would propagate differently through the episode and can lead to a different return distribution for each policy. We would like to emphasise that homoscedasticity and heteroscedasticity qualify environments and not policies as it is a property of the search space as a whole.

3.2 Policy Reproducibility

Consider a heteroscedastic uncertain environment and a policy π. The performance P_π of π would usually be quantified using its expected return, but this single number does not carry any information on the spread of the distribution. We define the **policy reproducibility** σ_π as the statistical dispersion of the return distribution of π. Both P_π and σ_π can be quantified using multiple metrics, for example, the mean or mode for P_π, and the standard deviation or entropy for σ_π.

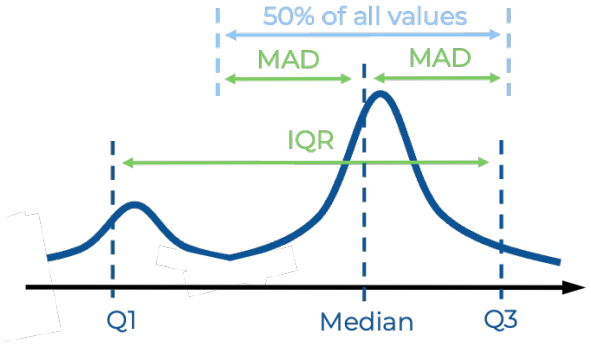

Figure 4: Illustration of the MAD and IQR metrics that quantify policy reproducibility for a given return distribution. IQR corresponds to the distance between first and third quartiles, while MAD corresponds to the median distance to the median of the distribution. As a consequence, half the sampled evaluations are closer to the median than the MAD, and half are further away, as illustrated in the figure.

Robust Quantification of Policy Reproducibility. σ_π can be quantified using any common measure of statistical dispersion, such as the standard deviation. However, common evaluation procedures rely on N evaluations of a policy π to approximate both P_π and σ_π. As N is usually chosen quite low, this procedure is likely to contain outlier evaluations that can greatly impact the approximation of σ_π. Thus, we propose using the Median Absolute Deviation (MAD) estimator, known for its robustness to outliers (Wilcox 2011) to quantify policy reproducibility σ_π. We denote $R^N = (R_i)_{i\in[0,N]}$ as the set of the N evaluations of the return of a policy, also illustrated in Figure 4:

$$MAD_R = median(|R^N - median(R^N)|)$$

While we choose to focus on the MAD metric, an alternative dispersion measure also known for its robustness to outliers would be the Inter-Quartile Range (IQR) (Wilcox 2011) which gives a wider representation of the return distribution and a better indication of worst-case returns. The return IQR (see Figure 4) is computed as $IQR_R = Q_3(R^N) - Q_1(R^N)$, where Q_i is the i^{th} quartile.

Lower Confidence Bound for Policy Comparison. The MAD defined in the previous section, allows the reproducibility of a policy σ_π to be quantified. However, the crucial question remains how to reliably compare RL policies while taking into account the trade-off between performance P_π and reproducibility σ_π. To this end, we propose a simple alternative to the commonly used expected return: the lower-confidence bound (LCB) (Frazier 2018), given as:

$$LCB(\pi) = P_\pi - \alpha\, \sigma_\pi$$

where $\alpha \in \mathrm{R}^+$ is a user-chosen parameter that allows weighting the relative importance of the performance and reproducibility. When the performance and the reproducibility values are normalised within the same interval, α can be interpreted as trading $\alpha\%$ of performance for $\alpha\%$ of reproducibility. As the LCB score is only computed after the runs (during policy evaluation), all the statistics needed for choosing the value of α are available. Additionally, α does not constitute a learning hyper-parameter, as it is only used when evaluating and the learning algorithms are agnostic to it.

In the following, we choose P_π as the expected return over N evaluations and σ_π as the MAD over these N evaluations. This decision allows the conventional expected return to be recovered when $\alpha = 0$, i.e. performance only. For robustness to outlier evaluations when N is low, we recommend using robust estimators such as median for P_π. We also report comparisons using the median for P_π and standard deviation and IQR for σ_π in the Appendix (Flageat et al. 2023b).

4 Experiments

In this section, we evaluate policies resulting from commonly used RL algorithms using the introduced MAD metric, and corresponding LCB measures. We benchmark the reproducibility of resulting policies from SAC, TD3 and ES as detailed in Section 2. For SAC, action sampling is removed during evaluation for fair comparison. We also introduce an ES variant that explicitly optimises for reproducibility which we refer to as Reproducibility ES (R-ES). R-ES uses ES to optimise for a weighted sum of performance and reproducibility; where performance is computed as the average return over 32 re-evaluations, and the reproducibility as the standard deviation of these re-evaluations. Algorithm hyperparameters are provided in Appendix (Flageat et al. 2023b).

We conduct experiments on common continuous control tasks (Ant and HalfCheetah) using the Brax (Freeman et al. 2021) simulator. A policy is evaluated by rolling it out

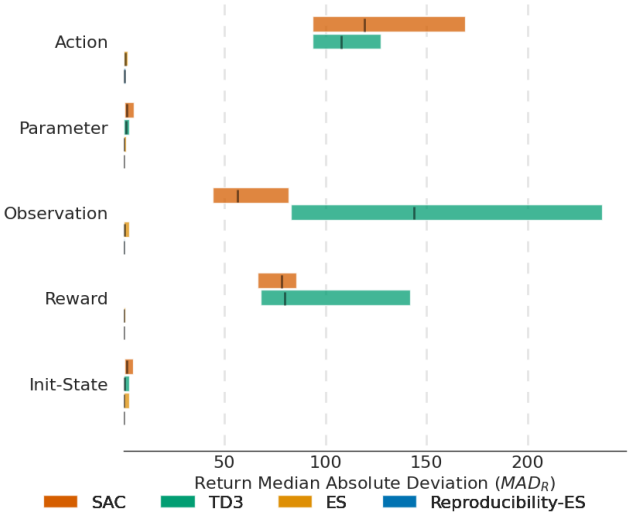

Figure 5: MAD scores of final policies in the Ant environment. y-axis is the type of uncertainty present in the environment. We report the IQM and the CIs across 10 seeds. The lower the MAD score the more reproducible the policy.

Env	Noise	ES	R-ES	SAC	TD3
Ant	Action	0.87	0.55	119.22	107.94
	Init-State	0.31	0.11	1.70	0.74
	Obs	0.72	0.15	56.34	143.77
	Param	0.02	0.00	1.74	1.20
	Reward	0.34	0.10	78.34	79.89
Half Cheetah	Action	182.33	134.01	209.95	139.37
	Init-State	154.06	100.33	150.87	100.32
	Obs	192.72	126.61	279.67	229.85
	Param	200.30	134.96	125.78	174.45
	Reward	184.30	129.36	206.92	141.14

Table 1: MAD results. We report the IQM across 10 seeds. The lower the MAD score, the more reproducible the policy.

$N = 256$ times in the environment, providing a distribution of returns and trajectories. The reproducibility quantification MAD and the LCB metric presented in Section 3.2 are then computed based on these roll-outs. We report results in a consistent manner across metrics and environments, in the form of the inter-quartile mean (IQM) and stratified bootstrap confidence interval (cis) across seeds using the rliable library (Agarwal et al. 2021). We report replications of results for each algorithm and environment with 10 seeds.

4.1 Effect of Types of Noise

We first conduct experiments to study the effect of a variety of noise and uncertainty on reproducibility. It is important to note that the same distribution of noise is applied during training and during evaluation. Figure 5 and Table 2 show the MAD scores, quantifying the reproducibility of the resulting policies. As a base experiment, we consider the standard RL setting with random initial states and stochastic transitions. We then independently consider the noise on the action, observation, reward and parameters, on top of this base setting. From the algorithms considered, policies obtained from SAC are observed to obtain a higher MAD, demonstrating a large spread in return values (low reproducibility). Interestingly, ES-based approaches result in more reproducible policies in all the cases of noise, evident from the lower MAD values. Policies learnt using ES are consistently more reproducible, and explicitly maximising for reproducible policy with the simple R-ES method also shows some additional improvements. This trend is also true for the IQR metric reported in the Appendix (Flageat et al. 2023b).

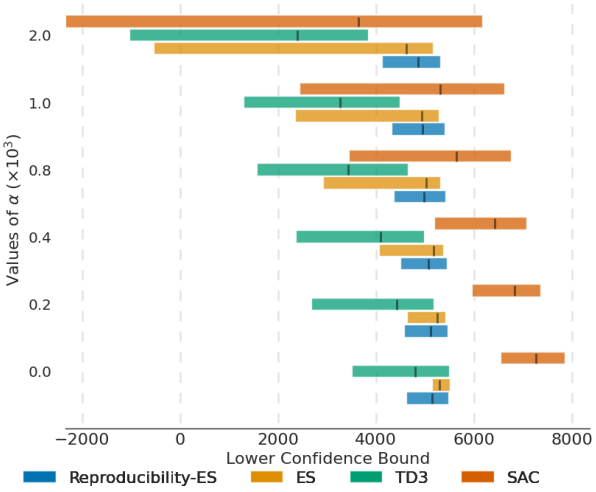

Figure 6: LCB comparison on the Ant with Init-State noise. y-axis corresponds to varying values of α, the trade-off between performance and reproducibility. We report the IQM and the CIs across 10 seeds. The higher the LCB score the better.

4.2 Evaluating Performance and Reproducibility

The increase in reproducibility observed for ES methods seems to come at a performance cost, observed from the lower expected return when $\alpha = 0$ (Fig. 6). To better understand the performance-reproducibility trade-off and make policy comparisons that account for reproducibility, Figure 6 and Table 2 show the use of the proposed LCB metric. Increasing values of α indicates increasing consideration for policy reproducibility. As previously explained, the conventionally used expected return would be recovered when $\alpha = 0$. As α is increased, there is a clear and evident tradeoff between policy reproducibility and performance. We observe that the LCB of a policy that is reproducible (as a result of ES algorithms) is relatively unaffected by the increase in α, while there can be a significant drop in LCB score when a policy is not reproducible due to a large distribution spread (SAC and TD3). In the case of the baseline algorithms considered, we hypothesise that this is due to optimisation properties in ES that come with perturbations in the parameter space, and optimising for the average reward of the population obtained from these perturbations (Lehman et al. 2018). This implicitly leads to more reproducible policies but comes at the cost of not being able to move to less reproducible parts of the

Env	Noise	$\alpha = 0.0$				$\alpha = 2.0 \times 10^3$			
		SAC	TD3	ES	R-ES	SAC	TD3	ES	R-ES
Ant	Action	**6751.87**	5002.73	4349.05	4789.71	-231613.46	-211449.83	2578.14	**3663.98**
	Init-State	**7269.30**	4796.49	5306.23	5155.73	3646.34	2393.72	4627.16	**4865.26**
	Obs	-184.10	158.18	4593.58	**4644.19**	-112922.20	-287392.25	2501.99	**4093.65**
	Param	-536.90	-429.24	-17.73	**6.11**	-4035.81	-2901.43	-50.10	**6.11**
	Reward	**8115.54**	5917.15	5234.34	5345.27	-148633.70	-153633.47	4152.09	**5042.58**
		$\alpha = 0.0$				$\alpha = 2.0 \times 10^2$			
HalfCheetah	Action	**4700.75**	3462.12	2212.68	2818.90	-36949.3	-24171.7	-33946.0	**-23400.6**
	Init-State	**5656.20**	3399.78	2456.96	2450.23	-24634.6	**16278.7**	-28458.4	-17322.4
	Obs	2505.37	2858.40	2782.10	**2947.29**	-53471.0	-42928.2	-35405.4	**-21383.4**
	Param	-208.22	315.36	1562.45	**2827.84**	-25366.7	-34592.7	-37925.7	**-24100.1**
	Reward	**5622.17**	3217.24	2348.76	3200.21	-35460.5	-24493.9	-34150.5	**-22274.4**

Table 2: LCB results. We report the IQM across 10 seeds. The higher the LCB score, the better.

search space where higher performance is possible.

Overall, our results demonstrate the benefit of LCB to account for both policy reproducibility and performance. More practically, the LCB allows practitioners to account for this reproducibility using α, given the tolerance in corresponding applications. Interestingly, the performance/reproducibility trade-off can also be studied through the lens of multi-objective optimisation, and a Pareto plot can be used to compare algorithms (see Appendix (Flageat et al. 2023b)).

5 Extension to Behaviour Reproducibility

While the reward can at times capture behaviour, this is not always the case and policy behaviour can differ in uncertain domains despite achieving similar returns (see Figure 7). Using the example of the breakfast-cooking robot from the introduction again, consider the robot gathering ingredients. If the robot always uses the same trajectory to do so, the user might get used to and expect a certain behaviour and adapt its own habit in the kitchen accordingly. If the robot randomly changes its trajectory in the exact same setting despite accomplishing the task, this might become a risk for the user. Hence, behavioural reproducibility is essential for policy interpretability and can become critical in applied settings. In this section, we extend our formalisation and metrics for performance reproducibility to behaviour reproducibility.

Behavioural Representation. To evaluate the behaviour reproducibility, we use two different measures of behavioural representation $\mathbf{b}$. The first measure is derived from novelty search (Lehman et al. 2011) and Quality-Diversity (Pugh et al. 2016; Chatzilygeroudis et al. 2021) literature, where a *behavioural descriptor* is defined for each solution or policy to characterise a policy with respect to another. For example, the average time each foot is in contact with the ground to characterise the gait of a legged robot (Cully et al. 2015), or the final position of an agent in a maze (Lehman et al. 2011). Behaviour descriptors have already been used to quantify behaviour reproducibility for Quality-Diversity (Flageat et al. 2023a). Next, we also consider the *state marginal* distribution of the different roll-outs of the policy as a representation of the behaviour. While we use these two measures, other

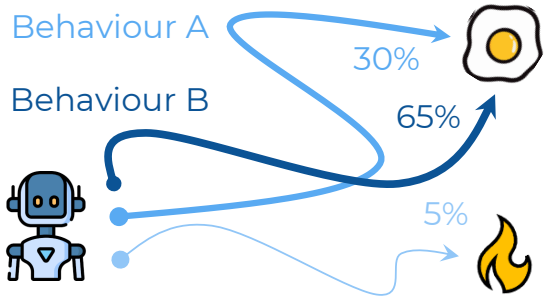

Figure 7: Illustration of the *behaviour reproducibility* in uncertain domains. *Behaviour reproducibility* refers to the different behaviours exhibited by a policy, agnostic to the return obtained. Behaviour A and B can have similar performance.

forms of behavioural representation can also be used such as action embeddings (Parker-Holder et al. 2020) or learnt state-trajectory encodings (Cully 2019; Lynch et al. 2020).

Quantifying Behaviour Reproducibility Unlike the return, behaviour representations are multidimensional and their absolute values carry no effective meaning (such as "larger is better"). Thus, the MAD metric defined in Section 3.2 cannot be applied out-of-the-box. We propose to apply them as second-order metrics on the distance between behaviour representations. In other words, denoting $B^N = (b_i)_{i \in [0,N]}$ as the set of the N evaluations of the behaviour of a policy, $D^N = (distance(B^N, B^N))$ is the set of all the two-by-two distances of B^N, for any distance function. We define the behavioural MAD as: $MAD_B = median(|D - median(D)|)$.

Experiments Figure 8 shows the behavioural MAD when considering the behavioural descriptor. The descriptor in this case corresponds to the average foot contact of the robot across the episode (Cully et al. 2015). We observe similar trends when comparing the algorithms in the case of behavioural reproducibility, where R-ES consistently performs well despite optimising only for return reproducibility. However, the behavioural reproducibility of ES is less strong

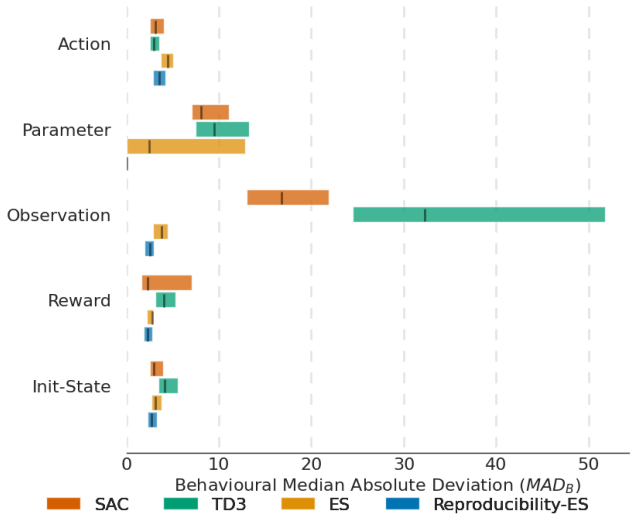

Figure 8: Behavioural MAD scores across types of uncertainty present in the Ant environment. We report the IQM and the CIs across 10 seeds. The lower the MAD score the more reproducible the behaviour.

across other tasks (see Appendix (Flageat et al. 2023b)). We hypothesise that the computation of behavioural representations are more sensitive than return values, causing uncertainty and errors to have a larger effect on behaviours. Similar results are observed when using state-marginal representation (see Appendix (Flageat et al. 2023b)).

6 Related Work

6.1 Safe and Robust RL

Safe RL. Many real-world applications such as robotics or autonomous driving require guarantees that policies will not lead to unsafe situations such as damages or collisions. Safe RL (Garcıa et al. 2015; Brunke et al. 2022; Gu et al. 2022) focuses on such safety-critical applications and aims to solve them using RL. To do so, it expresses the unsafe regions using a set of constraints that need to be satisfied by the RL agent to ensure safe operation. In many applications, these constrains do not necessarily have to be hard constraints, and only need to be satisfied with a high probability. Our work improves evaluation procedures in the general RL setting and not necessarily in constrained setups. While behaviour reproducibility shares some similarities with stability, which refers to the boundedness of the system output or state in Safe RL (Brunke et al. 2022), reproducibility only quantifies statistical dispersion and does not enforce constraints. Our proposed metrics are complementary and can also be applied in the Safe RL setup as policies can have low reproducibility while satisfying required safety constraints.

Robust RL. Robust RL (Morimoto et al. 2005; Moos et al. 2022; Chen et al. 2020) focuses on learning policies that, while trained on a given uncertainty distribution, are able to maintain performance when applied to a distinct test uncertainty distribution. This property makes them better at handling sim-to-real gaps or facing new and unplanned situations. A common approach in robust RL is to re-formulate the RL problem as a two-player game, where an adversary aims to deteriorate the policy performance by manipulating the environment (Morimoto et al. 2005; Moos et al. 2022; Chen et al. 2020). Moos et al. (2022) proposed a classification of robust RL approach based on the part of environment targeted by the adversary, that draws an interesting parallel with the classification of uncertainties from Section 2.2. Similar to Robust RL, our work is concerned with the evaluation and the maintenance of the performance of policies under uncertainty. Closest to our work, Xu et al. (2006) explores the trade-off between performance and robustness. However, our work defines reproducibility more generally for any policy without the need for robustness algorithms and the robust RL setting. We are interested in improving existing evaluation procedures and metrics more generally across RL.

6.2 Accounting for Value Distribution in RL

Distributional RL. While many RL works learn the expectation of the value function, Distributional RL (Bellemare et al. 2023, 2017) proposes to learn the full value distribution. Learning such a distribution improves the performance of standard RL approaches as it provides a more stable learning target. In comparison, our work considers return distributions of a policy, meaning episode-based and not timestep-based distributions. Additionally, these distributions are used to improve evaluation procedures and comparisons between policies and do not affect the learning algorithms in any way.

Using return variance for risk-sensitive RL. Risk-sensitive RL approaches are aware that optimal policies may perform poorly in some cases, causing a risk of large negative outcomes. Thus, this body of work accounts for the variance of the expected return from the current state (value function), in order to minimise this risk (Heger 1994; Coraluppi et al. 1999). Similar to Distributional RL, Risk-sensitive RL uses information on the distributions of time-step quantities (value functions) to improve learning while our work focuses on evaluation procedures and episodic return distributions.

7 Conclusion

In summary, our work highlights policy reproducibility, an important but commonly overlooked issue in evaluation of RL policies in uncertain environments. We define policy reproducibility as the dispersion of the distribution of returns of a policy and propose two robust statistical measures to quantify this: Mean Absolute Deviation (MAD) and the Interquartile Range (IQR). To account for reproducibility when comparing and evaluating policies in RL, we advocate for the use of the Lower Confidence Bound (LCB) of the return distribution as an alternative to the commonly used mean return. The LCB allows practitioners to set their preference for reproducibility with a single parameter α and is more general, as the mean return can also be recovered if $\alpha = 0$. Finally, by extensively evaluating popular RL algorithms, we experimentally show that there exists a performance-reproducibility trade-off which can be covered and effectively evaluated when using the proposed LCB metric.

References

Agarwal, R.; Schwarzer, M.; Castro, P. S.; Courville, A. C.; and Bellemare, M. 2021. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34: 29304–29320.

Akkaya, I.; Andrychowicz, M.; Chociej, M.; Litwin, M.; McGrew, B.; Petron, A.; Paino, A.; Plappert, M.; Powell, G.; Ribas, R.; et al. 2019. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*.

Azar, M. G.; Munos, R.; Ghavamzadeh, M.; and Kappen, H. 2011. Speedy Q-learning. In *Advances in neural information processing systems*.

Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In *International conference on machine learning*, 449–458. PMLR.

Bellemare, M. G.; Dabney, W.; and Rowland, M. 2023. *Distributional reinforcement learning*. MIT Press.

Brunke, L.; Greeff, M.; Hall, A. W.; Yuan, Z.; Zhou, S.; Panerati, J.; and Schoellig, A. P. 2022. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5: 411–444.

Cassandra, A. R. 1998. A survey of POMDP applications. In *Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes*, volume 1724.

Chatzilygeroudis, K.; Cully, A.; Vassiliades, V.; and Mouret, J.-B. 2021. Quality-Diversity Optimization: a novel branch of stochastic optimization. In *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems*, 109–135. Springer.

Chen, S.; and Li, Y. 2020. An overview of robust reinforcement learning. In *2020 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 1–6. IEEE.

Coraluppi, S. P.; and Marcus, S. I. 1999. Risk-sensitive and minimax control of discrete-time, finite-state Markov decision processes. *Automatica*, 35(2): 301–309.

Cully, A. 2019. Autonomous skill discovery with quality-diversity and unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 81–89.

Cully, A.; Clune, J.; Tarapore, D.; and Mouret, J.-B. 2015. Robots that can adapt like animals. *Nature*, 521(7553): 503–507.

Dulac-Arnold, G.; Levine, N.; Mankowitz, D. J.; Li, J.; Paduraru, C.; Gowal, S.; and Hester, T. 2020. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*.

Everitt, T.; Krakovna, V.; Orseau, L.; Hutter, M.; and Legg, S. 2017. Reinforcement learning with a corrupted reward channel. *arXiv preprint arXiv:1705.08417*.

Flageat, M.; and Cully, A. 2023a. Uncertain Quality-Diversity: Evaluation methodology and new methods for Quality-Diversity in Uncertain Domains. *IEEE Transactions on Evolutionary Computation*.

Flageat, M.; Lim, B.; and Cully, A. 2023b. Beyond Expected Return: Accounting for Policy Reproducibility when Evaluating Reinforcement Learning Algorithms. arXiv:2312.07178.

Fox, R.; Pakman, A.; and Tishby, N. 2015. Taming the noise in reinforcement learning via soft updates. *arXiv preprint arXiv:1512.08562*.

Frazier, P. I. 2018. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Freeman, C. D.; Frey, E.; Raichuk, A.; Girgin, S.; Mordatch, I.; and Bachem, O. 2021. Brax–A Differentiable Physics Engine for Large Scale Rigid Body Simulation. *arXiv preprint arXiv:2106.13281*.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, 1587–1596. PMLR.

Garcıa, J.; and Fernández, F. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1): 1437–1480.

Gu, S.; Yang, L.; Du, Y.; Chen, G.; Walter, F.; Wang, J.; Yang, Y.; and Knoll, A. 2022. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. PMLR.

Hansen, N. 2006. The CMA evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, 75–102.

Hasselt, H. 2010. Double Q-learning. *Advances in neural information processing systems*, 23.

Heger, M. 1994. Consideration of risk in reinforcement learning. In *Machine Learning Proceedings 1994*, 105–111. Elsevier.

Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; and Meger, D. 2018. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Jin, Y.; and Branke, J. 2005. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on evolutionary computation*, 9(3): 303–317.

Lehman, J.; Chen, J.; Clune, J.; and Stanley, K. O. 2018. ES is more than just a traditional finite-difference approximator. In *Proceedings of the genetic and evolutionary computation conference*, 450–457.

Lehman, J.; and Stanley, K. O. 2011. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2): 189–223.

Lynch, C.; Khansari, M.; Xiao, T.; Kumar, V.; Tompson, J.; Levine, S.; and Sermanet, P. 2020. Learning latent plans from play. In *Conference on robot learning*, 1113–1132. PMLR.

Moos, J.; Hansel, K.; Abdulsamad, H.; Stark, S.; Clever, D.; and Peters, J. 2022. Robust reinforcement learning: A review of foundations and recent advances. *Machine Learning and Knowledge Extraction*, 4(1): 276–315.

Morimoto, J.; and Doya, K. 2005. Robust reinforcement learning. *Neural computation*, 17(2): 335–359.

Parker-Holder, J.; Pacchiano, A.; Choromanski, K. M.; and Roberts, S. J. 2020. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 18050–18062.

Pugh, J. K.; Soros, L. B.; and Stanley, K. O. 2016. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3: 40.

Rakshit, P.; Konar, A.; and Das, S. 2017. Noisy evolutionary optimization algorithms–a comprehensive survey. *Swarm and Evolutionary Computation*, 33: 18–45.

Romoff, J.; Henderson, P.; Piché, A.; Francois-Lavet, V.; and Pineau, J. 2018. Reward estimation for variance reduction in deep reinforcement learning. *arXiv preprint arXiv:1805.03359*.

Salimans, T.; Ho, J.; Chen, X.; Sidor, S.; and Sutskever, I. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tassa, Y.; Doron, Y.; Muldal, A.; Erez, T.; Li, Y.; Casas, D. d. L.; Budden, D.; Abdolmaleki, A.; Merel, J.; Lefrancq, A.; et al. 2018. Deepmind control suite. *arXiv preprint arXiv:1801.00690*.

Wang, J.; Liu, Y.; and Li, B. 2020. Reinforcement learning with perturbed rewards. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 6202–6209.

Wierstra, D.; Schaul, T.; Glasmachers, T.; Sun, Y.; Peters, J.; and Schmidhuber, J. 2014. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1): 949–980.

Wilcox, R. R. 2011. *Introduction to robust estimation and hypothesis testing*. Academic press.

Xu, H.; and Mannor, S. 2006. The robustness-performance tradeoff in Markov decision processes. *Advances in Neural Information Processing Systems*, 19.