Extended Abstract: AutoRL Hyperparameter Landscapes

Aditya Mohan*
Institute of Artificial Intelligence
Leibniz University of Hannover
Hannover, Germany
a.mohan@ai.uni-hannover.de

Carolin Benjamins*
Institute of Artificial Intelligence
Leibniz University of Hannover
Hannover, Germany
c.benjamins@ai.uni-hannover.de

Konrad Wienecke
Faculty of Electrical Engineering and Computer Science
Leibniz University of Hannover
Hannover, Germany
konrad.wienecke@stud.uni-hannover.de

Alexander Dockhorn
Institute for Information Procedding
Leibniz University of Hannover
Hannover, Germany
dockhorn@tnt.uni-hannover.de

Marius Lindauer
Institute of Artificial Intelligence
Leibniz University of Hannover
Hannover, Germany
m.lindauer@ai.uni-hannover.de

This is an Extended Abstract of a Paper accepted for publication at AutoML-Conf 2023.

1 Motivation

As research in RL soars and the field targets increasingly harder learning-based optimization and control problems [21, 23, 3, 2, 10, 5], extracting good performance out of ever more complicated pipelines becomes the need of the hour. Thus, techniques in Automated Reinforcement Learning (AutoRL; [13]) have started automating RL approach design.

One of the goals of AutoRL is to optimize hyperparameter configurations to help an RL agent achieve the best performance. However, the distribution shift from the RL agent generating the learning data through environmental interaction makes it very different from other forms of Machine Learning. Consequently, RL pipelines can be very sensitive to hyperparameter configurations [8, 13], making it difficult to find an optimal static configuration at the beginning of the training, and thus, potentially necessitating adjusting hyperparameters throughout the training process in RL [13].

We develop a novel method to visualize and analyze the performances of different hyperparameter configurations using landscapes. Landscapes, traditionally a part of the optimization community [14, 12], have previously been the subject of analyses for benign characteristics of hyperparameter configurations in supervised learning [17]. To the best of our knowledge, we are the first to bring this to the realm of hyperparameters in RL by building a novel pipeline for creating hyperparameter

16th European Workshop on Reinforcement Learning (EWRL 2023).

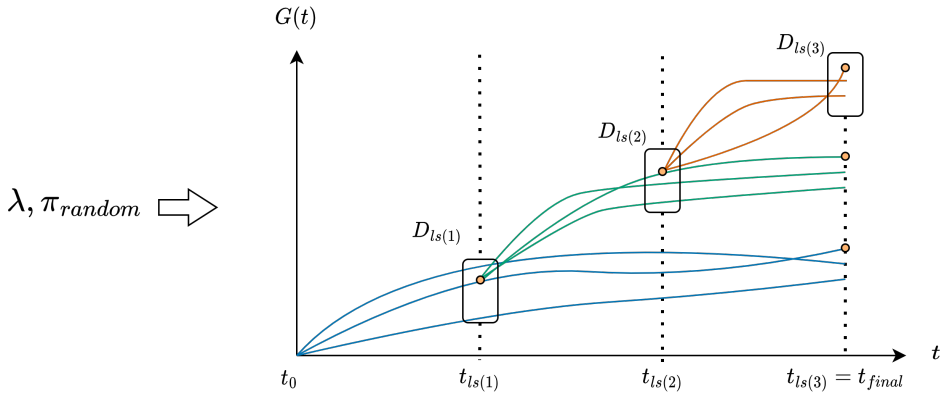


Figure 1: Overview of the data collection process for landscapes with three configurations $\boldsymbol{\lambda}$ and three phases. We initialize the process by training a random RL policy π_{random} on each configuration $\lambda \in \boldsymbol{\lambda}$. The three configurations run till the first landscape point $t_{ls(1)}$, which forms the first landscape dataset $D_{ls(1)}$. The policies are snapshotted at this point, and the policy for the next phase is selected based on the final performance, indicated by the continuation of the blue points. The selected configuration is shown with orange circles. This process is repeated for two more phases to create landscape datasets $D_{ls(1)}$ and $D_{ls(3)}$. The end of the final phase $t_{ls(3)}$ corresponds to the final training point t_{final}

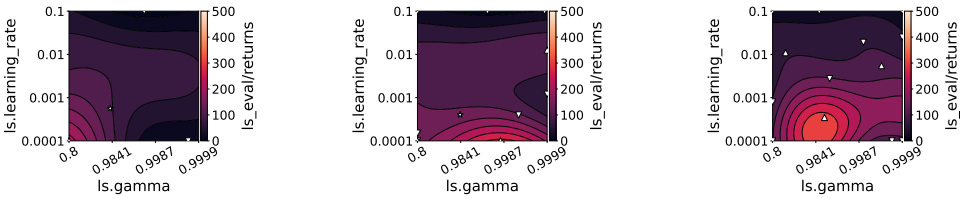
landscapes of dynamic configurations at multiple discrete time steps throughout training. Using the built landscapes, we delineate methods to inspect the landscapes for traits such as their general structure, configuration stability, and hyperparameter importance. Finally, we provide the first empirical evidence supporting the need for dynamic hyperparameter configurations for RL by analyzing the HP landscapes of DQN [21], PPO [19] and SAC [7] on Cartpole, Bipedal Walker and Hopper [4] environments.

2 Building online Hyperparameter Landscapes for Reinforcement Learning

Figure 1 shows an overview of our pipeline for collecting the data to build the landscape. Our approach can be broken down into two steps: (i) Data Collection; and (ii) Landscape Modeling and Analysis.

In the data collection phase, we divide the learning timeframe into different phases $0 < t_{ls(1)} < \cdots < t_{final}$, where $t = t_{ls(i)}$ (ls denoting landscape) denotes the time point for collecting landscape data, and $t = t_{final}$ is at the end of training. Each phase entails: (i) using a checkpoint from the last phase to initialize the different HP configurations and seed combinations that were sampled at the start of training using a scrambled Sobol sampling strategy [22, 9], (ii) running these combinations until the end of the phase, and (iii) determining the fitness of each HP configuration by aggregating the evaluation returns at the end of the phase across the seeds. To determine which configuration should be used as a checkpoint for the next phase, we train the HP configuration until the final timestep t_{final} and use the final performance as an indicator of fitness. To perform the selection, we first choose the configuration set with the highest Interquartile Mean (IQM) in the final phase and initialize this configuration with a seed corresponding to the highest IQM in the selected set. Using IQMs allows us to mitigate the outlier bias prevalent in mean aggregation while incorporating more data in our evaluation than median aggregation [1].

Once the data has been collected, we build the landscapes using Interpolating Landscape Models (ILM), which leverage RBF interpolation with a linear Kernel to construct a continuous surface over the search space from the given samples and Independent Gaussian Process Regressors (IGPRs), that use the mean of the Gaussian Processes to model the surfaces independently. Based on our experiments, IGPRs provide smoother landscapes that show more global patterns. We additionally use Individual Conditional Expectation (ICE) curves [6] to model one-dimensional slices through the hyperparameter landscape, thus creating one curve for each hyperparameter connecting the performance to its range. Finally, we perform a modality analysis using the folding test of unimodality [20],



(a) Phase 1 (b) Phase 2 (c) Phase 3

Figure 2: IGPR plots of the mean surfaces for learning rate and discount factor for DQN.

where we look at the return distributions to find a pivot point around which the distribution can be folded to reduce the variance. Multi-modal return distributions result in a high variance reduction, while uni-modal ones do not. We use this test to classify the returns into percentages of uni-modality and multi-modality and, therefore, provide a comment on the stability of the return distribution. Our code can be found here `https://github.com/automl/AutoRL-Landscape`.

3 Visualizing Landscapes for Inspection

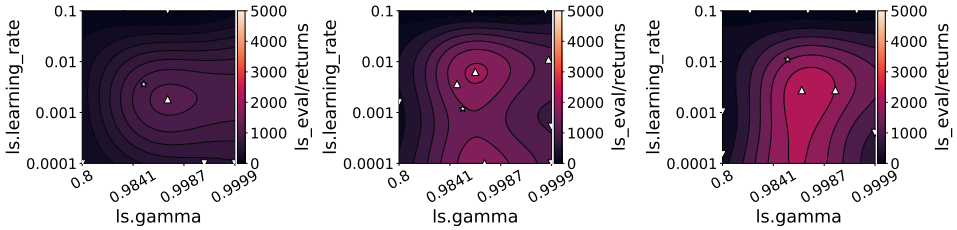
Figure 2, Figure 3, and Figure 4 show the IGPR hyperparameter landscapes of the three agents in our experiments. The local minima are represented by the inverted triangle, and the maxima, by the normal triangle. A star represents the configuration selected for the next stage.

As can be seen, the landscapes strongly vary in their structure over the phases and confirm that the effect of hyperparameters and their optimal settings vary throughout training. In this sense, the results promote the use of dynamic configurations, setting a precedent for research on other RL contexts. A deeper look at the plots shows that the performance peaks move strongly for different hyperparameters, indicative of the environment complexity and optimization procedure.

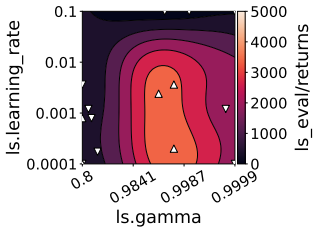
The performance peak for DQN occurs in a narrow region for both learning rate α and discount factor γ, with γ largely influencing the scores on its own in the final phase. On the other hand, we see a more stable region of peak performance for SAC, potentially indicating better exploration capability of SAC for similar HP ranges. Hence, it requires fewer variations in hyperparameter schedules, corroborating the advantage of soft updates and entropy-based exploration inherent in SAC. Consequently, potential HP schedules for α and γ would have a greater impact on the learning of TD-based off-policy algorithms such as DQN, something that could be potentially attributed to the learning dynamics of TD-algorithms themselves [11].

For PPO, we see that there is one region with a high performance whose location also changes over the phases. This implies that PPO is less robust to HP decisions in general.

Table 1 presents the overall sizes of the three categories (unimodal, multimodal, and uncategorized). We regard configurations that produce unimodal return distributions as more stable than those that produce multimodal ones. Generally, more return distributions are categorized as multimodal rather than unimodal. This is especially true for the last phase, where we find 49.22% of configurations to be multimodal for DQN, 60.94% configurations for SAC, and 80.47% for PPO. Although some configurations in this area are classified as unimodal, their return distributions are otherwise not optimal, with their IQMs being dominated by other configurations not classified as unimodal. We additionally see that the configurations with unimodal return distributions are almost double for DQN compared to SAC and PPO, which correlates with the more complicated optimization problem of Hopper and BipedalWalker compared to CartPole. While further analyses are necessary to ablate the various factors that impact modality, these observations contradict previous observations on benign landscapes of static algorithm configuration and AutoML [15, 17, 18].

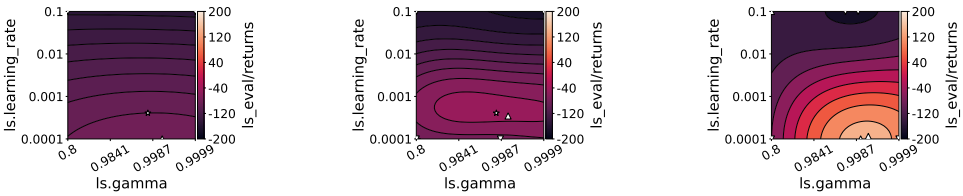


(a) Phase 1 (b) Phase 2 (c) Phase 3



(d) Phase 4

Figure 3: IGPR plots for learning rate and discount factor for SAC across phases



(a) Phase 1 (b) Phase 2 (c) Phase 3

Figure 4: IGPR plots of the mean surfaces for learning rate and discount factor for PPO

Category	**DQN**			**SAC**				PPO		
	Phase 1	Phase 2	Phase 3	Phase 1	Phase 2	Phase 3	Phase 4	Phase 1	Phase 2	Phase 3
% Unimodal	19.53	13.28	15.62	19.53	09.37	06.25	07.81	12.5	10.94	8.59
% Multimodal	40.63	60.94	60.16	49.22	53.90	57.81	60.94	67.18	60.16	80.47
% Uncategorized	39.84	22.66	22.66	28.90	30.47	24.22	27.34	20.31	28.90	10.94

Table 1: Percentages of configurations assigned to each class from the modality analysis.

4 Concluding Remarks

Our multiphase pipeline for data collection, landscape modeling, and landscape analysis presents a novel way to analyze hyperparameter performances for RL agents across different training points. Through our experiments, we found drastic changes in the hyperparameter landscape over time for three very different RL algorithms on three diverse environments, suggesting that the use of dynamic configurations in RL may be well-motivated. We additionally showed that the stability of configurations is rather unpredictable depending on a context informed jointly by the learning dynamics of the algorithm and the exploration problem. Finally, our modality analysis showed that only a small fraction of hyperparameter configurations with their associated return distributions eventually are unimodal, in contrast to the recent observations of benign landscapes in AutoML and algorithm configuration [16, 17, 18]. This shows that the dynamic configuration of RL agents poses a much harder problem than classical static AutoML addresses so far and calls for new and specialized AutoRL methods. Consequently, our analysis opens up new doors for building more specialized optimizers for RL agents and further research into the interplay between fundamental properties of RL agents, such as exploration, credit assignment, and hyperparameter configurations.

References

[1] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In M. Ranzato, A. Beygelzimer, K. Nguyen, P. Liang, J. Vaughan, and Y. Dauphin, editors, *Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems (NeurIPS'21)*. Curran Associates, 2021.

[2] A. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. Guo, and C. Blundell. Agent57: Outperforming the atari human benchmark. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research. PMLR, 2020.

[3] M. G. Bellemare, S. Candido, P. S. Castro, J. Gong, M. C. Machado, S. Moitra, S. S. Ponda, and Z. Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

[4] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[5] J. Degrave, F. Felici, J. Buchli, M. Neunert, B. Tracey, F. Carpanese, T. Ewalds, R. Hafner, A. Abdolmaleki, D. de las Casas, C. Donner, L. Fritz, C. Galperti, A. Huber, J. Keeling, M. Tsimpoukelli, J. Kay, A. Merle, J.-M. Moret, S. Noury, F. Pesamosca, D. Pfau, O. Sauter, C. Sommariva, S. Coda, B. Duval, A. Fasoli, P. Kohli, K. Kavukcuoglu, D. Hassabis, and M. Riedmiller. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

[6] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 2015.

[7] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR, 2018.

[8] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. In S. McIlraith and K. Weinberger, editors, *Proceedings of the Thirty-Second Conference on Artificial Intelligence (AAAI'18)*. AAAI Press, 2018.

[9] S. Joe and F. Kuo. Constructing sobol sequences with better two-dimensional projections. *SIAM J. Sci. Comput.*, 2008.

[10] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science in Robotics*, 5, 2020.

[11] C. Lyle, M. Rowland, W. Dabney, M. Kwiatkowska, and Y. Gal. Learning dynamics and generalization in deep reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2022.

[12] K. Malan. A survey of advances in landscape analysis for optimisation. *Algorithms*, 2021.

[13] J. Parker-Holder, R. Rajan, X. Song, A. Biedenkapp, Y. Miao, T. Eimer, B. Zhang, V. Nguyen, R. Calandra, A. Faust, F. Hutter, and M. Lindauer. Automated reinforcement learning (AutoRL): A survey and open problems. *Journal of Artificial Intelligence Research (JAIR)*, 74:517–568, 2022.

[14] E. Pitzer and M. Affenzeller. A comprehensive survey on fitness landscape analysis. In *Recent Advances in Intelligent Engineering Systems*, volume 378 of *Studies in Computational Intelligence*, pages 161–191. Springer, 2012.

[15] Y. Pushak and H. Hoos. Algorithm configuration landscapes: - more benign than expected? In A. Auger, C. Fonseca, N. Lourenço, P. Machado, L. Paquete, and L. D. Whitley, editors, *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN'18)*, pages 271–283, 2018.

[16] Y. Pushak and H. Hoos. Golden parameter search: exploiting structure to quickly configure parameters in parallel. In J. Ceberio, editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'20)*, pages 245–253. ACM Press, 2020.

[17] Y. Pushak and H. Hoos. Automl loss landscapes. *ACM Trans. Evol. Learn. Optim.*, 2022.

[18] L. Schneider, L. Schäpermeier, R. Prager, B. Bischl, H. Trautmann, and P. Kerschke. Hpo$\times$ ela: Investigating hyperparameter optimization landscapes by means of exploratory landscape analysis. In *Proc. of (PPSN'22)*, pages 575–589. Springer, 2022.

[19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347 [cs.LG]*, 2017.

[20] A. Siffer, P. Fouque, A. Termier, and C. Largouët. Are your data gathered? In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[21] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[22] I. Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967.

[23] Z. Zhou, X. Li, and R. Zare. Optimizing chemical reactions with deep reinforcement learning. *ACS central science*, pages 1337–1344, 2017.