# Scalable and Efficient Exploration via Intrinsic Rewards in Continuous-time Dynamical Systems

**Klemens Iten**
ETH Zürich
kiten@ethz.ch

**Andreas Krause**
ETH Zürich
krausea@ethz.ch

## Abstract

Reinforcement learning algorithms are typically designed for discrete-time dynamics, even though the underlying real-world control systems are often continuous in time. In this paper, we study the problem of continuous-time reinforcement learning, where the unknown system dynamics are represented using nonlinear ordinary differential equations (ODEs). We leverage probabilistic models, such as Gaussian processes and Bayesian neural networks, to learn an uncertainty-aware model of the underlying ODE. Our algorithm, COMBRL, greedily maximizes a weighted sum of the extrinsic reward and model epistemic uncertainty. We show that this approach has sublinear regret for the continuous-time setting. Furthermore, in the unsupervised RL setting (i.e., without extrinsic rewards), we provide a sample complexity bound. In our experiments, we evaluate COMBRL in the standard and unsupervised RL settings and show that it outperforms the baselines across several deep RL tasks.

## 1 Introduction

Reinforcement learning (RL) has proven to be a flexible paradigm for learning control policies through interaction, with success stories across robotics [Levine et al., 2016, Hwangbo et al., 2019, Spiridonov et al., 2024], games [Schrittwieser et al., 2020, Hafner et al., 2025], and applications in medicine and energy [Yu et al., 2021, Degrave et al., 2022]. Most RL algorithms assume discrete time, yet many real systems like robots or biological processes are naturally governed by continuous-time dynamics, modelled by ordinary differential equations (ODEs). Discretization can obscure key temporal behaviours and limit control flexibility. In contrast, continuous-time models align better with real-world sensing and actuation. We focus on continuous-time model-based RL, aiming to learn the underlying ODE from interaction.

We address two settings: (*i*) reward-driven RL, where the goal is to solve a specific task; and (*ii*) unsupervised RL, where the objective is to learn the system dynamics globally. The latter requires accurate global modeling, while the former demands task-relevant accuracy.

We propose **COMBRL**, a continuous-time RL algorithm that uses probabilistic models (e.g., Gaussian processes, Bayesian neural nets) to capture epistemic uncertainty. Policies are selected by maximizing a weighted sum of reward and uncertainty, following the optimism-in-the-face-of-uncertainty principle [Curi et al., 2020, Kakade et al., 2020, Treven et al., 2023, Sukhija et al., 2025b]. In the unsupervised case, **COMBRL** reduces to active learning via uncertainty sampling [Taylor et al., 2021, Lewis and Catlett, 1994] and guides the agent toward regions where the model is most uncertain.

We show that **COMBRL** achieves sublinear regret in the reward-driven case and offer a sample complexity bound in the unsupervised setting. Empirical results on several deep continuous-time RL tasks demonstrate improved performance and sample efficiency compared to baselines.

First Exploration in AI Today Workshop at ICML (EXAIT at ICML 2025).

**Contributions**

- We propose **COMBRL**, a scalable continuous-time optimistic model-based RL algorithm. Unlike prior continuous-time methods that are either exploitative or rely on costly co-optimization, **COMBRL** uses a single scalar to balance reward and epistemic uncertainty, supporting both reward-driven and unsupervised learning.
- We provide theoretical guarantees, showing sublinear regret in the reward-driven case and offer sample complexity bounds in the unsupervised setting, with explicit dependence on the measurement selection strategy. These results extend optimism-based analysis to continuous-time RL.
- We demonstrate strong empirical results across continuous-time deep RL benchmarks, showing that **COMBRL** scales better than prior methods and generalizes to unseen downstream tasks.

## 2 Problem setting

Consider an unknown continuous-time dynamical system $\boldsymbol{f}^*(\boldsymbol{x}(t), \boldsymbol{u}(t))$ with initial state $\boldsymbol{x}(0) = \boldsymbol{x}_0 \in \mathcal{X} \subset \mathbb{R}^{d_x}$ and control input $\boldsymbol{u} : [0, \infty) \to \mathcal{U} \subset \mathbb{R}^{d_u}$. Furthermore, consider a policy $\boldsymbol{\pi} : \mathcal{X} \to \mathcal{U}$ so that the control input follows said policy, i.e. $\boldsymbol{u}(t) = \boldsymbol{\pi}(\boldsymbol{x}(t))$. Typically in optimal control (OC, cf. Luenberger [1979]), we consider the associated finite-time OC problem over the policy space $\Pi$ to solve for the optimal policy such that an objective function $J(\boldsymbol{\pi}, \boldsymbol{f}^*)$ is maximized:

$$\begin{aligned} \boldsymbol{\pi}^* &\overset{\text{def}}{=} \operatorname*{argmax}_{\boldsymbol{\pi} \in \Pi} J(\boldsymbol{\pi}, \boldsymbol{f}^*) = \operatorname*{argmax}_{\boldsymbol{\pi} \in \Pi} \int_0^T r\big(\boldsymbol{x}(s), \boldsymbol{\pi}(\boldsymbol{x}(s))\big)\, ds \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}^*\big(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t))\big), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ & \big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \in \mathcal{X} \times \mathcal{U} \subset \mathbb{R}^{d_x + d_u}, \quad t \in [0, T]. \end{aligned} \tag{1}$$

To learn the unknown dynamics, we collect data across episodes $n = 1, \ldots, N$, each deploying a policy $\boldsymbol{\pi}_n$ over horizon $T$. Optimizing solely for the reward function $r(\boldsymbol{x}(t), \boldsymbol{u}(t))$ during learning introduces a directional bias, since $\boldsymbol{f}^*$ is mostly explored in high-reward region, resulting in limited exploration and poor generalization of the learned dynamics. In contrast, system identification or unsupervised RL aims to learn the ODE $\boldsymbol{f}^*$ globally, independent of a reward function. **COMBRL** supports both settings, unlike prior continuous-time RL methods [Yildiz et al., 2021, Treven et al., 2023, 2024], which are only designed for the supervised RL setting.

In each episode $n$, we query $\boldsymbol{f}^*$ by taking a sequence of measurements $S_n$ at $m_n$ different times $t_{n,i}$, $i \in \{1 \ldots, m_n\}$ to form a dataset $\mathcal{D}_n \sim (\boldsymbol{\pi}_n, S_n)$ as follows:

$$\mathcal{D}_n = \{(\boldsymbol{z}_n(t_{n,i}), \dot{\boldsymbol{y}}_n(t_{n,i}))\}_{i=1}^{m_n}, \quad \text{with} \quad \boldsymbol{z}_n = (\boldsymbol{x}_n, \boldsymbol{\pi}_n(\boldsymbol{x}_n)), \quad \dot{\boldsymbol{y}}_n = \dot{\boldsymbol{x}}_n + \boldsymbol{\epsilon}_{n,i}.$$

The learner uses the data up to the current episode $\mathcal{D}_{1:n-1} \overset{\text{def}}{=} \bigcup_{i=1}^{n-1} \mathcal{D}_i$ to update its model of $\boldsymbol{f}^*$ and deploys $\boldsymbol{\pi}_n$ accordingly. While rolling out the policy $\boldsymbol{\pi}_n$, the state derivative $\dot{\boldsymbol{x}}$ is rarely observed directly and is typically estimated via finite differences or filtering. Thus, the derivative measurements $\dot{\boldsymbol{y}}_n$ are subject to measurement noise $\boldsymbol{\epsilon}_{n,i}$.

Note that, since we defined a continuous-time setting, we are not necessarily restricted to a fixed sampling rate. **COMBRL** allows flexible, event-driven sampling taken under a measurement strategy (MSS) $S$, which is a sequence of sets $(S_n)_{n \geq 1}$ specifying the measurement times in each episode [Treven et al., 2023, Definition 1], as well as variable control rates. In this work, we focus on simpler strategies (i.e. fixed sampling and control rates with continuous-time dynamics) to isolate core behavior.

### 2.1 Performance measure

For the supervised RL setting, a natural performance measure is given by the *cumulative regret* that sums the gaps between the performance of the policy $\boldsymbol{\pi}_n$ at episode $n$ and the optimal policy $\boldsymbol{\pi}^*$ over all the episodes:

$$R_N \overset{\text{def}}{=} \sum_{n=1}^{N} r_n \overset{\text{def}}{=} \sum_{n=1}^{N} J(\boldsymbol{\pi}^*, \boldsymbol{f}^*) - J(\boldsymbol{\pi}_n, \boldsymbol{f}^*) \tag{2}$$

If the cumulative regret $R_N$ is sublinear in $N$, then the average reward of the policy converges to the optimal reward, and by extension to the optimal policy $\boldsymbol{\pi}^*$.

## 3 COMBRL: Continuous-time optimistic model-based RL

We now present **COMBRL**, a continuous-time, optimistic model-based reinforcement learning (MBRL) algorithm. **COMBRL** proceeds in a continuous-time, episodic setting, alternating between learning a predictive model of the dynamics from data and selecting policies that trade off extrinsic reward and epistemic uncertainty. The method assumes only access to a simulator or physical system for episodic rollouts and measurements. To enable theoretical analysis, we adopt standard assumptions from RL and Bayesian optimization (see Appendix B.1; cf. Srinivas et al. [2012], Chowdhury and Gopalan [2017], Treven et al. [2023]). Specifically, we assume $L_f$-Lipschitz continuity of the dynamics $\boldsymbol{f}^*$, as well as Lipschitz continuity on the reward $r$ and policies $\boldsymbol{\pi} \in \Pi$, and that observations are corrupted by i.i.d. $\sigma$-sub-Gaussian noise [Rigollet and Hütter, 2023].

**COMBRL** learns a probabilistic model $\boldsymbol{f}_n$ that provides both mean predictions $\boldsymbol{\mu}_n(\boldsymbol{z})$ and uncertainty estimates $\boldsymbol{\sigma}_n(\boldsymbol{z})$. We assume the model is *well-calibrated* [Rothfuss et al., 2023], i.e., the true dynamics $\boldsymbol{f}^*$ lie within a high-probability confidence set $\mathcal{M}_n$ (see Definition 1 in App. B.1) defined by $(\boldsymbol{\mu}_n, \boldsymbol{\sigma}_n)$, and that $\boldsymbol{\sigma}_n$ is Lipschitz. This supports reliable uncertainty quantification and formal regret bounds. The assumption holds for Gaussian processes (GPs) as shown by Rothfuss et al. [2023, Lemma 3.6] and can be approximated for Bayesian neural networks (BNNs) via calibration [Kuleshov et al., 2018]. Thus, **COMBRL** is model-agnostic: the statistical model can be instantiated using GPs [Rasmussen and Williams, 2005, Deisenroth et al., 2015], BNNs [MacKay, 1992], ensembles [Lakshminarayanan et al., 2017], or other estimators that capture epistemic uncertainty.

### 3.1 Optimistic planning objective

In each episode $n$, we select *any* $L_f$-Lipschitz model from the confidence set $\mathcal{M}_{n-1}$ of the previous episode, i.e., $\boldsymbol{f}_n \in \mathcal{M}_{n-1} \cap \mathcal{F}$, where $\mathcal{F}$ is the set of $L_f$-Lipschitz functions. We then choose a policy $\boldsymbol{\pi}_n$ that maximizes a reward-augmented optimistic objective under the current model $\boldsymbol{f}_n$:

$$\begin{aligned} \boldsymbol{\pi}_n = \arg\max_{\boldsymbol{\pi} \in \Pi} & \int_0^T \Big( r\big(\boldsymbol{x}'(s), \boldsymbol{u}(s)\big) + \lambda_n \cdot \|\boldsymbol{\sigma}_{n-1}(\boldsymbol{x}'(s), \boldsymbol{u}(s))\| \Big) ds \qquad (3) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}'(t) = \boldsymbol{f}_n(\boldsymbol{x}'(t), \boldsymbol{u}(t)), \quad \boldsymbol{u}(t) = \boldsymbol{\pi}(\boldsymbol{x}'(t)). \\ & \big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \in \mathcal{Z} = \mathcal{X} \times \mathcal{U} \subset \mathbb{R}^{d_x + d_u}, \quad t \in [0, T]. \end{aligned}$$

The key feature of **COMBRL** is its reward-plus-uncertainty objective in continuous time, inspired by discrete-time optimistic MBRL methods [Sukhija et al., 2023, 2025b], enabling a principled trade-off between exploration and exploitation. The scalar $\lambda_n$ balances reward and model uncertainty, and is treated as a tunable hyperparameter in practice. The epistemic uncertainty term $\boldsymbol{\sigma}_{n-1}(\boldsymbol{z})$ in (3) encourages the agent to visit poorly understood regions of the state-action space. A similar discrete-time objective from Sukhija et al. [2025b] achieves sublinear regret and scales well to high dimensions, unlike Curi et al. [2020]. We extend this guarantee to the continuous-time setting.

Treven et al. [2023] propose an optimistic exploration algorithm for continuous-time RL that requires joint optimization over both policies and dynamics $\boldsymbol{f}_n \in \mathcal{M}_{n-1} \cap \mathcal{F}$. This is intractable and heuristically addressed using a reparametrization trick from Curi et al. [2020], which increases input dimensionality from $d_u$ to $d_u + d_x$, limiting scalability in high-dimensional settings. **COMBRL** avoids this by selecting *any* model from $\mathcal{M}_{n-1} \cap \mathcal{F}$; in practice, using $\boldsymbol{\mu}_n$ works well.[1] Unlike optimistic dynamics or classical control, **COMBRL** balances exploration and exploitation via a single scalar $\lambda_n$ and encourages exploration of uncertain regions to improve model fidelity. **COMBRL** is summarized in Algorithm 1.

**Algorithm 1 COMBRL: Continuous-Time Optimistic MBRL**

1: **Initialize:** statistical model $\mathcal{M}_0$, simulator $\textsc{Sim}(\cdot, \cdot)$, empty dataset $\mathcal{D}_0 = \{\}$, measurement selection strategy $S = (S_n)_{n \geq 1}$, int. reward weight $\lambda_n$ for $i = 1 \dots N$
2: **for** episode $n = 1, \dots, N$ **do**
3: **Select policy** $\boldsymbol{\pi}_n$ by solving (3) subject to $L_f$-Lipschitz dynamics $\boldsymbol{f}_n \in \mathcal{F}$
4: **Roll out policy:** $\mathcal{D}_n \leftarrow \textsc{Sim}(\boldsymbol{\pi}_n, S_n)$ using measurement selection strategy $S_n$
5: **Update model:** $\mathcal{M}_n \leftarrow (\boldsymbol{\mu}_n, \boldsymbol{\sigma}_n) \leftarrow \mathcal{D}_{0:n}$

[1] Even though the mean model might not lie in $\mathcal{M}_{n-1} \cap \mathcal{F}$. For GP dynamics, we show how to pick a model from $\mathcal{M}_{n-1} \cap \mathcal{F}$ in Appendix B.2

### 3.2 The internal reward weight $\lambda_n$

The scalar value $\lambda_n$ in Equation (3) determines the trade-off between maximizing extrinsic reward and exploring regions of high model uncertainty. We differentiate between three key scenarios that affect the agent's behaviour:

- **Greedy ($\lambda_n = 0$):** Pure exploitation with respect to the given reward function. The model is only updated passively through whatever data results from reward-seeking behaviour, as in prior continuous-time MBRL approaches [Yildiz et al., 2021].
- **Balanced ($0 < \lambda_n < \infty$):** Task-driven but exploratory behaviour; most practical and improves model quality over time. We offer some strategies to select $\lambda_n$ subsequently.
- **Unsupervised ($\lambda_n \to \infty$):** Ignores reward and acts solely to reduce uncertainty, akin to active learning methods in discrete time [Pathak et al., 2019, Sekar et al., 2020, Sukhija et al., 2023].

**How to choose $\lambda_n$ in the balanced case?** For the case $0 < \lambda_n < \infty$, we study several practical strategies for setting or adapting $\lambda_n$:

- **Static (hyperparameter):** Set $\lambda_n = \lambda$ to a fixed value tuned via cross-validation or grid search. This is simple and often effective.
- **Scheduled (annealing):** Decrease $\lambda_n$ over time, for example $\lambda_n = \lambda_0 \cdot (1 - n/N)$. This encourages more exploration early in training and more exploitation as the model improves.
- **Auto-tuned:** Use an information-theoretic approach such as the auto-tuning procedure described by Sukhija et al. [2025a], which selects $\lambda_n$ adaptively based on maximizing mutual information gain or related criteria.

In this work, we primarily treat $\lambda_n$ as a tunable hyperparameter with scheduling. In Section 4, we also study the auto-tuning approach and show its effectiveness, as well as the unsupervised RL case.

### 3.3 Theoretical results

The regret of any continuous-time model-based RL algorithm depends on the hardness of learning the true dynamics $\boldsymbol{f}^*$ and on the measurement selection strategy (MSS) $S$, which defines the discrete time points at which the system is observed. Following Treven et al. [2023, Definition 1], an MSS is a sequence of sets $(S_n)_{n \geq 1}$, where each $S_n \subset [0, T]$ contains $m_n$ measurement times.[2] The MSS governs how data is collected – e.g., using equidistant sampling – and thus affects learning efficiency. The model complexity for the unknown dynamics $\boldsymbol{f}^*$ under MSS $S$ is defined as:

$$\mathcal{I}_N(\boldsymbol{f}^*, S) \overset{\text{def}}{=} \max_{\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_N \in \Pi} \sum_{n=1}^{N} \int_0^T \left\| \boldsymbol{\sigma}_{n-1}\big(\boldsymbol{z}_n(t)\big) \right\|^2 \, dt.$$

Intuitively, for a given $N$, the more complicated the dynamics $\boldsymbol{f}^*$, the larger the epistemic uncertainty and thereby the model complexity. Curi et al. [2020] study the model complexity for the discrete-time setting, where the integral is replaced by the sum over the uncertainties. In continuous-time, the MSS $S$ proposes when we measure the system and influences how we collect data to update our model.

**Theorem 1.** *Under regularity assumptions (Assumptions 1 to 4 in Appendix B.1), we have with probability at least* $1 - \delta$*:*

$$R_N \leq \mathcal{O}\left(\sqrt{\mathcal{I}_N^3(\boldsymbol{f}^*, S)\, N}\right).$$

Theorem 1 bounds the regret of **COMBRL** w.r.t. the model complexity. For GP dynamics, where the well-calibration assumption is true and the monotonicity of the variance holds, Treven et al. [2023] show that the model complexity $\mathcal{I}_N(\boldsymbol{f}^*, S)$ is sublinear for many common kernels and MSSs, e.g., grows with $\operatorname{poly}\log(N)$ for the RBF kernel and equidistant MSS. Therefore, for common kernels and MSSs, **COMBRL** enjoys sublinear regret in the GP setting, and the policy converges to the optimal policy $\boldsymbol{\pi}^*$. We provide the proofs for all our theoretical results in Appendix B.2.

[2] Here, the set $S_n$ may depend on observations prior to episode $n$ or is even constructed while we execute the trajectory. For ease of notation, we do not make this dependence explicit.

**Theorem 2.** *Consider the unsupervised setting ($\lambda_n \to \infty$) and let Assumptions 1 to 4 hold. If $\sigma_{n-1,j}(\boldsymbol{z}) \geq \sigma_{n,j}(\boldsymbol{z}) \; \forall \, \boldsymbol{z} \in \mathcal{Z}$, $j \leq d_x$, and $n > 0$, then*

$$\max_{\boldsymbol{\pi} \in \Pi} \int_0^T \left\| \boldsymbol{\sigma}_{n-1}\big(\boldsymbol{x}(s), \boldsymbol{\pi}(\boldsymbol{x}(s))\big) \right\| \, ds \leq \mathcal{O}\left( \sqrt{\frac{\mathcal{I}_N^3(\boldsymbol{f}^*, S)}{N}} \right).$$

Theorem 2 provides a sample-complexity bound for the unsupervised case. Effectively, this shows that pure intrinsic exploration with $\lambda_n \to \infty$ reduces our model epistemic uncertainty with a rate of $\sqrt{\mathcal{I}_N^3 / N}$. To the best of our knowledge, we are the first to show this for continuous-time RL.

## 4 Experiments

We evaluate the **COMBRL** algorithm on various environments from the OpenAI/Farama gymnasium benchmark suite (Gym, Brockman et al. [2016], Towers et al. [2024]) and the DeepMind control suite (DMC, Tassa et al. [2018], Tunyasuvunakool et al. [2020]).

For the dynamics model $\boldsymbol{f}_n$, we use Gaussian processes (GPs) and probabilistic ensembles (PEs) to capture uncertainty about well-calibrated statistical models. Since **COMBRL** models continuous-time dynamics directly, it also remains agnostic to the solver or discretization scheme, and can in principle accommodate adaptive strategies [Treven et al., 2024]. In our experiments, we use equidistant MSS and fixed-rate control to isolate core algorithmic behavior. We solve the continuous-time planning problem using standard discrete-time solvers via fine-grained uniform discretization, but unlike discrete-time RL, $\boldsymbol{f}_n$ learns the ODE of the true system $\boldsymbol{f}^*$. We use $\boldsymbol{f}_n$ to simulate trajectories for policy training with SAC [Haarnoja et al., 2018] or real-time planning using the improved cross-entropy method (iCEM) described by Pinneri et al. [2021].

**Baselines** We compare **COMBRL** with two baselines with different planning approaches. The mean planner uses the mean estimate $\boldsymbol{\mu}_n$ of the statistical model and greedily maximizes the extrinsic reward for the task at hand, akin to Yildiz et al. [2021]. Furthermore, we compare our method with the trajectory sampling scheme (TS-1) proposed by Chua et al. [2018], subsequently referred to as PETS. PETS samples trajectories from PEs for state propagation and thus inherently captures the epistemic uncertainty during planning.

Treven et al. [2023] thoroughly study and demonstrate the advantages of continuous-time RL over its discrete-time counter part. To this end, in our experiments we focus on the four questions explored in the following. We provide additional details on our experimental setup in Appendix C.

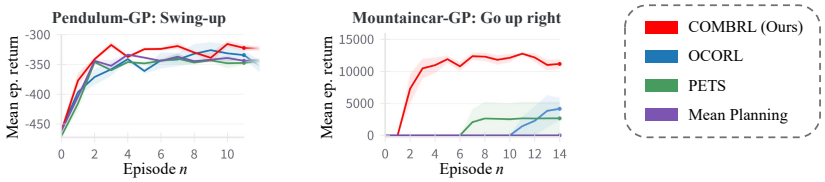


Figure 1: Learning curves for baselines, **COMBRL** and OCORL with fixed internal reward weight $\lambda_n$ using GP dynamics and iCEM planning, averaged over 5 seeds. We report the mean and the standard error bands.

**Does COMBRL scale better than the SOTA?** In Figure 1, we empirically validate our theoretical insights using GPs. We consider two classic continuous control tasks – the pendulum swing-up and the mountaincar problem [Moore, 1990] – and use iCEM for real-time planning. We compare **COMBRL** with the baselines above and with the state-of-the-art continuous-time RL algorithm OCORL from Treven et al. [2023]. In these experiments, $\lambda_n$ is held constant throughout training and treated as a static, hand-tuned hyperparameter.

Results, averaged over five random seeds with standard error bands, show that **COMBRL** consistently achieves higher asymptotic returns than the baselines, confirming the benefits of intrinsic rewards for guiding exploration. For the Pendulum experiments, all algorithms solve the task of swinging up. However, the co-optimization over the reward and the optimistic dynamics as well as the reparametrization trick used by the OCORL algorithm are computationally prohibitive, and require around $3\times$ the compute time compared to **COMBRL** (see Appendix C.2). We believe this shows that our algorithm scales better and is more computationally efficient than the SOTA.

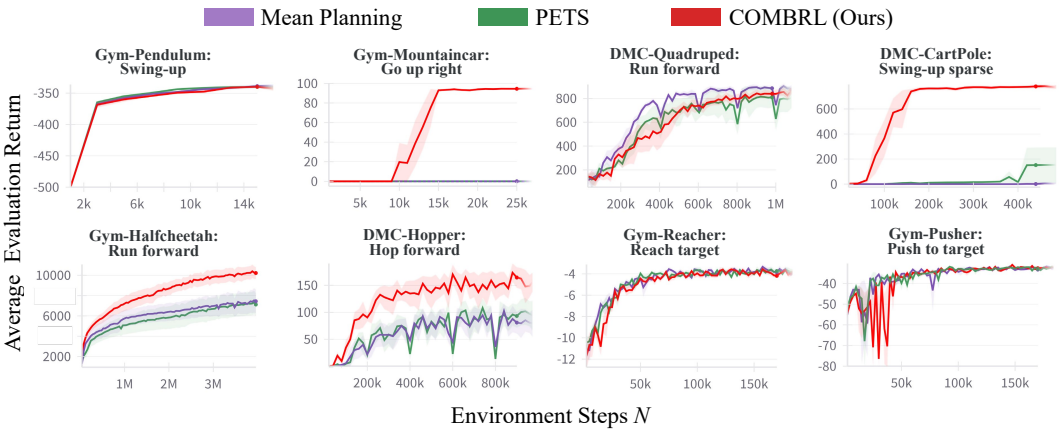


Figure 2: Learning curves for **COMBRL** and baselines with autotuned internal reward weight $\lambda$. We report the mean return when evaluating the learned model on the task at hand, averaged across 10 random seeds along with the standard error.

**How does the intrinsic reward affect learning?** To assess the effect of intrinsic rewards, we compare **COMBRL** with $\lambda_n > 0$ against PETS and the mean planner ($\lambda_n = 0$). Figure 2 shows learning curves on several environments from Gym and DMC. In these experiments, we model dynamics using PEs and use SAC to train the policy. We compare **COMBRL**, with a nonzero intrinsic reward weight $\lambda_n$, against PETS and the mean planner. For **COMBRL**, the internal reward weight $\lambda_n$ is automatically tuned following the strategy proposed by Sukhija et al. [2025a], as outlined in Appendix C.3.

Across a range of environments, **COMBRL** achieves higher asymptotic returns, particularly in sparse-reward or underactuated settings such as MountainCar and CartPole, highlighting that optimism-driven exploration significantly accelerates learning. In higher-dimensional environments such as HalfCheetah and Hopper, COMBRL improves performance by encouraging exploration of uncertain regions, which helps uncover effective behaviors in complex, high-degree-of-freedom systems, even when these behaviors are not directly tied to high immediate reward signals.

**How does COMBRL perform in the unsupervised RL setting?** We evaluate whether exploration driven by model uncertainty improves generalization to unseen tasks. Specifically, we train policies on a primary task (e.g., reaching a target) and assess zero-shot performance on a downstream task not encountered during training (e.g., moving away from the target).

Figure 3 compares standard **COMBRL** as well as its unsupervised variant ($\lambda_n \to \infty$, cf. Section 3.2) to the baselines. To ensure a fair comparison, we let each agent explore the environment for several episodes, and then periodically evaluate the learned model on downstream tasks. While **COMBRL** generally achieves the best performance on the primary task especially in environments which offer sparse rewards, its unsupervised variant performs best on the downstream task across all seven evaluated domains. This suggests that exploration guided by model uncertainty encourages the agent to cover a more diverse state-action space and highlights the trade-off between task focus and exploration breadth: Ignoring the reward signal entirely leads the agent to explore broadly, acquiring a globally accurate model that generalizes better to unseen tasks for $\lambda_n \to \infty$.

**How does the choice of $\lambda_n$ trade-off directed exploration w.r.t. the extrinsic reward and global exploration?** We ablate different values of the internal reward weight $\lambda_n$ for the Gym implementation of the HalfCheetah [Wawrzyński, 2009], Hopper [Erez et al., 2012], as well as the Reacher and Pusher, which are part of the MuJoCo environments [Todorov et al., 2012]. Figure 4 shows that growing nonzero $\lambda_n$ values improve downstream generalization while maintaining strong performance on the primary task. In contrast, large $\lambda_n$ may overly favor exploration and degrade performance on the primary task. This suggests that there exists an intermediate value of $\lambda_n$ that balances goal-directed behaviour with model uncertainty reduction, achieving the best of both **COMBRL** (task-optimal) and its unsupervised variant (exploration-optimal).

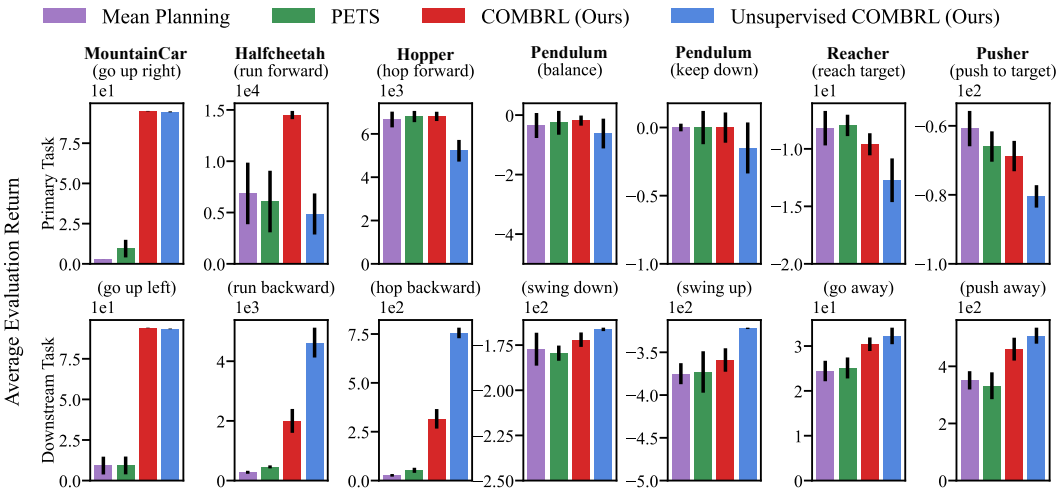


Figure 3: Final performance at convergence on primary and downstream tasks across seven Gym environments. We train the policy using SAC on the primary task with the **COMBRL** objective and compare with the mean and PETS dynamics propagation, then additionally evaluate on a previously unseen secondary task. For **COMBRL**, we differentiate between the balanced case with a static or annealing schedule for $\lambda_n$, or the unsupervised case with $\lambda_n \to \infty$. We report the mean return as well as the standard error for the primary and downstream task.

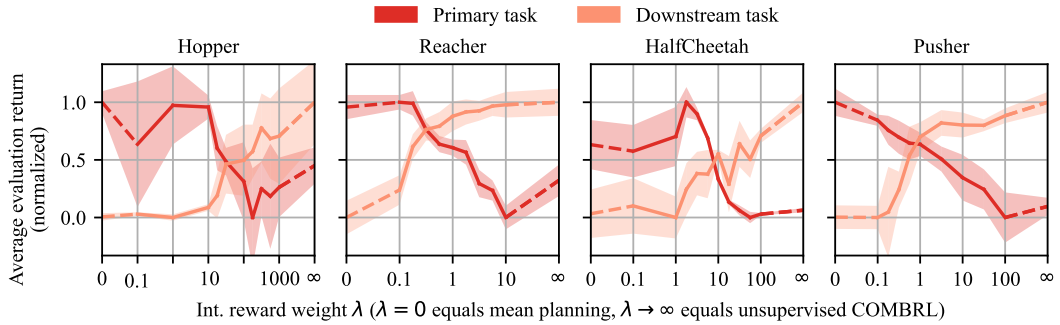


Figure 4: Final performance at convergence for different environments and tasks with varying internal reward weight $\lambda$. We ablate over different choices of $\lambda$ and report the mean return and standard error on a primary task the proposed algorithm was trained on as well as a previously unseen downstream task.

## 5 Conclusion

In this work, we introduced **COMBRL**, a continuous-time model-based reinforcement learning algorithm that uses epistemic uncertainty to guide exploration through an intrinsic reward. Our approach provides a principled and flexible mechanism to balance exploration and exploitation via the internal reward weight $\lambda_n$, generalizing the classical optimism-in-the-face-of-uncertainty paradigm to continuous-time domains that is scalable and easy-to-implement.

Our experiments demonstrate that **COMBRL** excels at goal-directed learning on the task at hand, while its unsupervised RL variant (i.e., $\lambda_n \to \infty$) is particularly effective at generalizing to unseen downstream tasks. This highlights a core trade-off: Reward directed exploration for appropriate $\lambda_n$ values and exploration across the entire reachable domain for $\lambda_n \to \infty$. We differentiate between three cases for the internal reward weight that affect the agent's behaviour – greedy, balanced, and unsupervised. Empirical ablations confirm that there exists a range for $\lambda_n$ that enables generalizable yet sample-efficient learning. Furthermore, the unsupervised version of **COMBRL** acts as an unsupervised system identification strategy, enabling strong zero-shot adaptation to new objectives.

## Acknowledgements and disclosure of funding

We would like to thank Bhavya Sukhija and Lenart Treven for the insightful discussions and feedback on this work. This project has received funding from the Swiss National Science Foundation under NCCR Automation, grant agreement 51NF40 180545. Numerical simulations were performed on the ETH Zürich Euler cluster.

## References

Marc Abeille and Alessandro Lazaric. Efficient optimistic exploration in linear-quadratic regulators via lagrangian relaxation. *International Conference on Machine Learning (ICML)*, 2020. URL https://proceedings.mlr.press/v119/abeille20a.html.

Karl Johan Åström and Pieter Eykhoff. System identification – A survey. *Automatica*, 7(2):123–162, 1971. URL https://doi.org/10.1016/0005-1098(71)90059-8.

Arthur Aubret, Laetitia Matignon, and Salima Hassas. A survey on intrinsic motivation in reinforcement learning, 2019. URL https://arxiv.org/abs/1908.06976.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, 2002. URL https://doi.org/10.1023/A:1013689704352.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Conference on Neural Information Processing Systems (NIPS)*, 2016. URL https://arxiv.org/abs/1606.01868.

Felix Berkenkamp. *Safe Exploration in Reinforcement Learning: Theory and Applications in Robotics*. Doctoral thesis, ETH Zurich, 2019. URL https://research-collection.ethz.ch/handle/20.500.11850/370833.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016. URL https://arxiv.org/abs/1606.01540.

Nicolò Cesa-Bianchi, Claudio Gentile, Gábor Lugosi, and Gergely Neu. Boltzmann exploration done right. *Conference on Neural Information Processing Systems (NIPS)*, 2017. URL https://arxiv.org/abs/1705.10257.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. URL https://arxiv.org/abs/1806.07366.

Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. *International Conference on Machine Learning (ICML)*, 2017. URL https://proceedings.mlr.press/v70/chowdhury17a.html.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. URL http://arxiv.org/abs/1805.12114.

Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks, 2020. URL https://arxiv.org/abs/2003.04630.

Sebastian Curi, Felix Berkenkamp, and Andreas Krause. Efficient model-based reinforcement learning through optimistic policy search and planning. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. URL http://arxiv.org/abs/2006.08684.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de las Casas, Craig Donner, Leslie Fritz, Cristian Galperti, Andrea Huber, James Keeling, Maria Tsimpoukelli, Jackie Kay, Antoine Merle, Jean-Marc Moret, Seb Noury, Federico Pesamosca, David Pfau, Olivier Sauter, Cristian Sommariva, Stefano Coda, Basil Duval, Ambrogio Fasoli, Pushmeet Kohli, Koray Kavukcuoglu, Demis Hassabis, and Martin Riedmiller. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022. URL https://doi.org/10.1038/s41586-021-04301-9.

Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, 2015. URL https://ieeexplore.ieee.org/document/6654139.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=OpC-9aBBVJe.

Kenji Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1):219–245, 2000. URL https://ieeexplore.ieee.org/document/6789455.

Tom Erez, Yuval Tassa, and Emanuel Todorov. Infinite-horizon model predictive control for periodic tasks with contacts. *Robotics: Science and systems (RSS)*, 2012. URL https://roboticsproceedings.org/rss07/p10.html.

Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS Computational Biology*, 9(4):e1003024, 2013. URL https://doi.org/10.1371/journal.pcbi.1003024.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. URL https://arxiv.org/abs/1906.01563.

Riccardo A. Grimaldi, Giacomo Baggio, Ruggero Carli, and Gianluigi Pillonetto. The bayesian separation principle for data-driven control, 2024. URL https://arxiv.org/abs/2409.16717.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International Conference on Machine Learning (ICML)*, 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. URL https://doi.org/10.1038/s41586-025-08744-2.

Nicklas Hansen, Yixin Lin, Hao Su, Xiaolong Wang, Vikash Kumar, and Aravind Rajeswaran. MoDem: Accelerating visual model-based reinforcement learning with demonstrations. *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=JdTnc9gjVfJ.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019. URL https://www.science.org/doi/abs/10.1126/scirobotics.aau5872.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. URL https://arxiv.org/abs/1906.08253.

Sham Kakade, Akshay Krishnamurthy, Kendall Lowrey, Motoya Ohnishi, and Wen Sun. Information theoretic regret bounds for online nonlinear control. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. URL http://arxiv.org/abs/2006.12466.

Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K. Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences, 2018. URL https://arxiv.org/abs/1807.02582.

Hassan Khalil. *Nonlinear Control*. Pearson New York, 2014. ISBN 9781292060507. URL https://elibrary.pearson.de/book/99.150005/9781292060699.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. *International Conference on Machine Learning (ICML)*, 2018. URL https://proceedings.mlr.press/v80/kuleshov18a.html.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Conference on Neural Information Processing Systems (NIPS)*, 2017. URL https://arxiv.org/abs/1612.01474.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016. URL http://jmlr.org/papers/v17/15-522.html.

David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. *Machine Learning Proceedings 1994*, pages 148–156, 1994. URL https://doi.org/10.1016/B978-1-55860-335-6.50026-X.

David G. Luenberger. Optimal control. In *Introduction to Dynamic Systems*, pages 393–435. John Wiley & Sons, New York, 1979. ISBN 0-471-02594-1.

David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3): 448–472, 1992. URL https://doi.org/10.1162/neco.1992.4.3.448.

Andrew William Moore. Efficient memory-based learning for robot control. Technical report, University of Cambridge, 1990. URL https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-209.pdf.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. *International Conference on Machine Learning (ICML)*, 2017. URL https://proceedings.mlr.press/v70/pathak17a.html.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. *International Conference on Machine Learning (ICML)*, 2019. URL https://proceedings.mlr.press/v97/pathak19a.html.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. *Conference on Robot Learning (CoRL)*, 2021. URL https://proceedings.mlr.press/v155/pinneri21a.html.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005. ISBN 026218253X. URL https://GaussianProcess.org/gpml/.

Philippe Rigollet and Jan-Christian Hütter. High-dimensional statistics, 2023. URL http://arxiv.org/abs/2310.19244.

Jonas Rothfuss, Bhavya Sukhija, Tobias Birchler, Parnian Kassraie, and Andreas Krause. Hallucinated adversarial control for conservative offline policy evaluation. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2023. URL https://proceedings.mlr.press/v216/rothfuss23a.html.

Jonas Rothfuss, Bhavya Sukhija, Lenart Treven, Florian Dörfler, Stelian Coros, and Andreas Krause. Bridging the sim-to-real gap with bayesian inference. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024. URL https://ieeexplore.ieee.org/abstract/document/10801505.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment–an introduction. *Guided Self-Organization: Inception*, 2014. URL https://doi.org/10.1007/978-3-642-53734-9_4.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020. URL https://doi.org/10.1038/s41586-020-03051-4.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. *International Conference on Machine Learning (ICML)*, 2020. URL https://proceedings.mlr.press/v119/sekar20a.html.

Alexander Spiridonov, Fabio Buehler, Moriz Berclaz, Valerio Schelbert, Jorit Geurts, Elena Krasnova, Emma Steinke, Jonas Toma, Joschua Wuethrich, Recep Polat, Wim Zimmermann, Philip Arm, Nikita Rudin, Hendrik Kolvenbach, and Marco Hutter. Spacehopper: A small-scale legged robot for exploring low-gravity celestial bodies. *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. URL https://ieeexplore.ieee.org/document/10610057.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 2012. URL https://ieeexplore.ieee.org/document/6138914.

Bhavya Sukhija, Lenart Treven, Cansu Sancaktar, Sebastian Blaes, Stelian Coros, and Andreas Krause. Optimistic Active Exploration of Dynamical Systems. *Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL http://arxiv.org/abs/2306.12371.

Bhavya Sukhija, Stelian Coros, Andreas Krause, Pieter Abbeel, and Carmelo Sferrazza. MaxInfoRL: Boosting exploration in reinforcement learning through information gain maximization. *International Conference on Learning Representations (ICLR)*, 2025a. URL https://openreview.net/forum?id=R4q3cY3kQf.

Bhavya Sukhija, Lenart Treven, Carmelo Sferrazza, Florian Dorfler, Pieter Abbeel, and Andreas Krause. Optimism via intrinsic rewards: Scalable and principled exploration for model-based reinforcement learning. *7th Robot Learning Workshop at ICLR*, 2025b. URL https://openreview.net/forum?id=VGdqa79ugx.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind Control Suite, 2018. URL https://arxiv.org/abs/1801.00690.

Annalisa T. Taylor, Thomas A. Berrueta, and Todd D. Murphey. Active learning in robotics: A review of control principles. *Mechatronics*, 77:102576, 2021. URL https://doi.org/10.1016/j.mechatronics.2021.102576.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. *International Conference on Intelligent Robots and Systems (IROS)*, 2012. URL https://ieeexplore.ieee.org/document/6386109.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments, 2024. URL https://arxiv.org/abs/2407.17032.

Lenart Treven, Jonas Hübotter, Bhavya Sukhija, Florian Dörfler, and Andreas Krause. Efficient exploration in continuous-time model-based reinforcement learning. *Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL http://arxiv.org/abs/2310.19848.

Lenart Treven, Bhavya Sukhija, Yarden As, Florian Dörfler, and Andreas Krause. When to sense and control? A time-adaptive approach for continuous-time RL. *Conference on Neural Information Processing Systems (NeurIPS)*, 2024. URL https://arxiv.org/abs/2406.01163.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. URL https://www.sciencedirect.com/science/article/pii/S2665963820300099.

Paweł Wawrzyński. A cat-like robot real-time learning to run. *International Conference on Adaptive and Natural Computing Algorithms (ICANNGA)*, 2009. URL https://staff.elka.pw.edu.pl/~pwawrzyn/pub-s/0812_LSCLRR.pdf.

Cagatay Yildiz, Markus Heinonen, and Harri Lähdesmäki. Continuous-time model-based reinforcement learning. *International Conference on Machine Learning (ICML)*, 2021. URL https://proceedings.mlr.press/v139/yildiz21a.html.

Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021. URL https://doi.org/10.1145/3477600.

## Appendices and supplementary material

# A Additional related work

## A.1 Model-based reinforcement learning

Model-based RL (MBRL) has emerged as a sample-efficient alternative to model-free methods, with applications ranging from robotics to online decision-making [Chua et al., 2018, Janner et al., 2019, Hansen et al., 2023, Rothfuss et al., 2024]. Recent deep MBRL methods differ primarily in dynamics modeling and planning strategies, yet often rely on naive exploration heuristics such as Boltzmann exploration [Hansen et al., 2024, Hafner et al., 2025]. However, such heuristics are suboptimal even in simple settings [Cesa-Bianchi et al., 2017].

**COMBRL** addresses this by introducing a principled exploration mechanism that combines epistemic uncertainty with extrinsic reward. Unlike prior methods, it is model- and planner-agnostic, scalable, and comes with sublinear regret guarantees. We show that this intrinsic reward formulation not only improves theoretical performance but also enables meaningful exploration across deep RL benchmarks, where naive methods fail.

## A.2 Unsupervised reinforcement learning and intrinsic exploration

System identification [Åström and Eykhoff, 1971] and unsupervised exploration [Aubret et al., 2019] require efficient exploration, as the agent must learn accurate global dynamics models without access to extrinsic rewards. In such settings, exploration is essential for covering informative regions of the state-action space. To this end, intrinsic motivation techniques have long been used to encourage exploration in RL, making use of objectives like model prediction error [Pathak et al., 2017], novelty [Bellemare et al., 2016], empowerment [Salge et al., 2014], and information gain [Sekar et al., 2020, Sukhija et al., 2023]. However, such techniques are often used in isolation from extrinsic rewards. **COMBRL** instead combines epistemic uncertainty, which poses a principled intrinsic signal, with task rewards, aligning exploration with learning progress. In the unsupervised setting, **COMBRL** reduces to active learning Taylor et al. [2021], specifically uncertainty sampling [Lewis and Catlett, 1994] and guides the agent toward regions where the model is most uncertain. Similar ideas have been explored in bandits [Auer et al., 2002, Srinivas et al., 2012], data-driven control [Grimaldi et al., 2024], and RL [Abeille and Lazaric, 2020, Sukhija et al., 2025a], where joint optimization of reward and model uncertainty is shown to improve learning.

**COMBRL** follows this direction, using model epistemic uncertainty as a reward bonus to guide exploration. Our work is closely related to Sukhija et al. [2025b], who propose a similar reward formulation in the discrete-time setting. However, our focus on continuous-time systems leads to significantly different theoretical analysis and experimental design. In particular, we derive regret and sample complexity bounds tailored to the continuous-time domain and analyze the effect of the intrinsic reward weight $\lambda_n$ in greater depth. Unlike prior work, which is mostly empirical or limited to linear systems, **COMBRL** also provides theoretical guarantees for general nonlinear systems in continuous time and demonstrates scalability to high-dimensional tasks.

## A.3 Continuous-time reinforcement learning

While most model-based RL methods are developed in discrete time, continuous-time formulations have gained increasing interest due to their relevance for real-world control and physical modelling [Doya, 2000, Frémaux et al., 2013, Yildiz et al., 2021]. Recent works explore learning dynamics via neural ODEs [Chen et al., 2018] and physics-informed priors [Greydanus et al., 2019, Cranmer et al., 2020].

Yildiz et al. [2021] propose a continuous-time actor-critic method that plans using the posterior mean of the learned ODE model. Treven et al. [2023, 2024] derive regret bounds for continuous-time MBRL using optimistic dynamics and show that considerably fewer interactions with the environment is needed to achieve the same or better performance compared to discrete time counterpart. **COMBRL** extends this line of work by proposing a flexible framework for both reward-driven and unsupervised exploration in continuous time. In contrast to prior methods, **COMBRL** incorporates optimism directly in the reward function and thus offers a simple, scalable, and theoretically grounded approach that operates directly in the continuous-time domain and supports general-purpose dynamics models.

# B Theory

We provide the assumptions and proofs for Theorems 1 and 2, which formalize the regret and exploration guarantees of **COMBRL**, in this section. The former bounds regret in terms of model complexity, which is sublinear for common GP kernels and MSSs, implying convergence to the optimal policy. The latter shows that intrinsic exploration alone (i.e. $\lambda \to \infty$) reduces epistemic uncertainty at a rate of $\sqrt{\mathcal{I}_N^3/N}$. To the best of our knowledge, we are the first to show this for continuous-time RL.

## B.1 Assumptions

In the following, we make some common assumptions (cf. Curi et al. [2020], Treven et al. [2023]) that allow us to theoretically analyse the regret $R_N$ and prove a regret bound. We first make an assumption on the continuity of the underlying system and the observation noise.

**Assumption 1** (Lipschitz continuity)**.** *The dynamics model $\boldsymbol{f}^*$, reward $r$, and all policies $\boldsymbol{\pi} \in \Pi$ are $L_f$, $L_r$ and $L_{\boldsymbol{\pi}}$ Lipschitz-continuous, respectively.*

**Assumption 2** (Sub-gaussian noise)**.** *We assume that the measurement noise $\epsilon_{n,i}$ is i.i.d. $\sigma$-sub Gaussian.*

The Lipschitz assumption is commonly made for analysing nonlinear systems [Khalil, 2014] and is satisfied for many real-world applications. Furthermore, assuming $\sigma$-sub Gaussian noise [Rigollet and Hütter, 2023] is also fairly general and is common in both RL and Bayesian optimization literature [Srinivas et al., 2012, Chowdhury and Gopalan, 2017, Curi et al., 2020].

In **COMBRL**, we learn an uncertainty-aware model of the underlying dynamics. Therefore, we obtain a mean estimate $\boldsymbol{\mu}_n(\boldsymbol{z})$ and quantify our epistemic uncertainty $\boldsymbol{\sigma}_n(\boldsymbol{z})$ about the function $\boldsymbol{f}^*$.

**Definition 1** (Well-calibrated statistical model of $\boldsymbol{f}^*$, Rothfuss et al. [2023])**.** *Let $\mathcal{Z} \overset{def}{=} \mathcal{X} \times \mathcal{U}$. An all-time well-calibrated statistical model of the function $\boldsymbol{f}^*$ is a sequence $\{\mathcal{M}_n(\delta)\}_{n\geq 0}$, where*

$$\mathcal{M}_n(\delta) \overset{def}{=} \left\{ \boldsymbol{f} : \mathcal{Z} \to \mathbb{R}^{d_x} \mid \forall \boldsymbol{z} \in \mathcal{Z}, \forall j \in \{1, \dots, d_x\} : |\mu_{n,j}(\boldsymbol{z}) - f_j(\boldsymbol{z})| \leq \beta_n(\delta)\sigma_{n,j}(\boldsymbol{z}) \right\},$$

*if, with probability at least $1-\delta$, we have $\boldsymbol{f}^* \in \bigcap_{n\geq 0} \mathcal{M}_n(\delta)$. Here, $\mu_{n,j}$ and $\sigma_{n,j}$ denote the $j$-th element in the vector-valued mean and standard deviation functions $\boldsymbol{\mu}_n$ and $\boldsymbol{\sigma}_n$ respectively, and $\beta_n(\delta) \in \mathbb{R}_{\geq 0}$ is a scalar function that depends on the confidence level $\delta \in (0, 1]$ and which is monotonically increasing in $n$.*

**Assumption 3** (Well-calibration of the model)**.** *The learned model is an all-time well-calibrated statistical model of $\boldsymbol{f}^*$, i.e., with probability at least $1-\delta$, we have $\boldsymbol{f}^* \in \bigcap_{n\geq 0} \mathcal{M}_n(\delta)$ for confidence sets $\mathcal{M}_n(\delta)$ as defined in Definition 1.*

**Assumption 4** (Lipschitz continuity of the uncertainty estimates)**.** *The standard deviation functions $\boldsymbol{\sigma}_n : \mathcal{Z} \to \mathbb{R}^{d_x}$ are $L_{\boldsymbol{\sigma}}$-Lipschitz continuous for all $n \geq 0$.*

Intuitively, Assumption 3 states that we are, with high probability, able to capture the dynamics within a confidence set spanned by our predicted mean and epistemic uncertainty. For Gaussian process (GP) models, the assumption is satisfied [Rothfuss et al., 2023, Lemma 3.6] and for more general classes of models such as Bayesian neural networks (BNNs), re-calibration techniques [Kuleshov et al., 2018] can be used.

Lastly, we make an assumption on the regularity of the dynamics by placing them in a reproducing kernel Hilbert space (RKHS):

**Assumption 5** (RKHS Prior on Dynamics)**.** *We assume that the functions $f_j^*$, $j \in \{1, \dots, d_{\boldsymbol{x}}\}$ lie in a RKHS with kernel $k$ and have a bounded norm $B$, that is*

$$\boldsymbol{f}^* \in \mathcal{H}_{k,B}^{d_{\boldsymbol{x}}}, \quad \textit{with} \quad \mathcal{H}_{k,B}^{d_{\boldsymbol{x}}} = \{\boldsymbol{f} \mid \|f_j\|_k \leq B, j = 1, \dots, d_{\boldsymbol{x}}\}.$$

*Moreover, we assume that $k(\boldsymbol{z}, \boldsymbol{z}) \leq \sigma_{\max}$ for all $\boldsymbol{x} \in \mathcal{X}$.*

### B.2 Analysis of Gaussian process dynamics

Assumption 5 allows us to model $\boldsymbol{f}^*$ with GPs. The posterior mean $\boldsymbol{\mu}_n(\boldsymbol{z}) = [\mu_{n,j}(\boldsymbol{z})]_{j \leq d_{\boldsymbol{x}}}$ and epistemic uncertainty $\boldsymbol{\sigma}_n(\boldsymbol{z}) = [\sigma_{n,j}(\boldsymbol{z})]_{j \leq d_{\boldsymbol{x}}}$ can then be obtained using the following formula

$$\begin{aligned} \mu_{n,j}(\boldsymbol{z}) &= \boldsymbol{k}_n^\top(\boldsymbol{z})(\boldsymbol{K}_n + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y}_{1:n}^j, \\ \sigma_{n,j}^2(\boldsymbol{z}) &= k(\boldsymbol{z}, \boldsymbol{z}) - \boldsymbol{k}_n^\top(\boldsymbol{z})(\boldsymbol{K}_n + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_n(\boldsymbol{z}), \end{aligned} \tag{4}$$

Here, $\boldsymbol{y}_{1:n}^j$ corresponds to the noisy measurements of $f_j^*$, i.e., the noisy derivative observation from the dataset $\mathcal{D}_{1:n}$, $\boldsymbol{k}_n(\boldsymbol{z}) = [k(\boldsymbol{z}, \boldsymbol{z}_i)]_{\boldsymbol{z}_i \in \mathcal{D}_{1:n}}$, and $\boldsymbol{K}_n = [k(\boldsymbol{z}_i, \boldsymbol{z}_l)]_{\boldsymbol{z}_i, \boldsymbol{z}_l \in \mathcal{D}_{1:n}}$ is the data kernel matrix.

Moreover, since $\boldsymbol{f}^*$ has bounded RKHS norm, i.e., $\|\boldsymbol{f}^*\|_k \leq B$ (Assumption 5), it follows from Srinivas et al. [2012], Chowdhury and Gopalan [2017] that with probability $1 - \delta$ we have for every episode $n$:

$$\|\boldsymbol{f}^* - \boldsymbol{\mu}_n\|_{k_n} \leq \beta_n.$$

Instead of planning with the mean, which in general might not be Lipschitz continuous for all $n$, we select a function $\boldsymbol{f}_n$ that not only approximates the $\boldsymbol{f}^*$ function well, i.e. satisfies $\|\boldsymbol{f}^* - \boldsymbol{f}_n\|_{k_n} \leq \beta_n$, but also has an RKHS norm that does not grow with $n$. To achieve this, we propose solving the following quadratic optimization problem:

$$\begin{aligned} \boldsymbol{f}_n = \underset{\boldsymbol{f} \in \text{span}(k(\boldsymbol{x}_1, \cdot), \ldots, k(\boldsymbol{x}_n, \cdot))}{\operatorname{argmin}} & \|\boldsymbol{f} - \boldsymbol{\mu}_n\|_{k_n} \\ \text{s.t. } & \|\boldsymbol{f}\|_k \leq B \end{aligned} \tag{5}$$

**Theorem 3.** *The optimization problem Equation* (5) *is feasible and we have* $\|\boldsymbol{f}_n - \boldsymbol{\mu}_n\|_{k_n} \leq 2\beta_n$.

*Proof.* Consider the noise-free case, i.e., $\epsilon_{n,i} = 0$, and let $\bar{\boldsymbol{\mu}}_n$ be the posterior mean for this setting. For the function $\bar{\boldsymbol{\mu}}_n$, it holds that $\|\boldsymbol{f}^* - \bar{\boldsymbol{\mu}}_n\|_{k_n} \leq \beta_n$ (Corollary 3.11 of Kanagawa et al. [2018]) and $\|\bar{\boldsymbol{\mu}}_n\|_k \leq B$ (Theorem 3.5 of Kanagawa et al. [2018]). Thus it follows that

$$\|\bar{\boldsymbol{\mu}}_n - \boldsymbol{\mu}_n\|_{k_n} \leq \|\bar{\boldsymbol{\mu}}_n - \boldsymbol{f}^*\|_{k_n} + \|\boldsymbol{f}^* - \boldsymbol{\mu}_n\|_{k_n} \leq 2\beta_n.$$

By representer theorem, it also holds that $\bar{\boldsymbol{\mu}}_n \in \text{span}(k(\boldsymbol{z}_1, \cdot), \ldots, k(\boldsymbol{z}_n, \cdot))$. □

Let $\boldsymbol{\alpha}_n = (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y} \in \mathbb{R}^n$ and reparametrize $\boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^n \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x})$. We have $\|\boldsymbol{f}\|_k^2 = \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha}$. We also have:

$$\|\boldsymbol{f} - \boldsymbol{\mu}_n\|_{k_n}^2 = (\boldsymbol{\alpha} - \boldsymbol{\alpha}_n)^\top \boldsymbol{K} \left( \boldsymbol{I} + \frac{1}{\sigma^2} \boldsymbol{K} \right) (\boldsymbol{\alpha} - \boldsymbol{\alpha}_n)$$

Hence the optimization problem Equation (5) is equivalent to:

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \mathbb{R}^n} & (\boldsymbol{\alpha} - \boldsymbol{\alpha}_n)^\top \boldsymbol{K} \left( \boldsymbol{I} + \frac{1}{\sigma^2} \boldsymbol{K} \right) (\boldsymbol{\alpha} - \boldsymbol{\alpha}_n) \\ \text{s.t. } & \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha} \leq B^2 \end{aligned}$$

This is a quadratic program that can be solved using any QP solver. The program finds the closest function to the posterior mean $\boldsymbol{\mu}_n$ that is Lipschitz continuous. In particular, note that since $\|\boldsymbol{f}_n\|_k \leq B$, for Lipschitz kernels, $\boldsymbol{f}_n$ has a Lipschitz constant $L_B$ which is independent of $n$ [Berkenkamp, 2019]. From hereon, let $L_* = \max\{L_f, L_B\}$.

Next, we plan with the dynamics $\boldsymbol{f}_n$ that are obtained from solving Equation (5).

$$\begin{aligned} \boldsymbol{\pi}_n = & \arg\max_{\boldsymbol{\pi} \in \Pi} \mathbb{E}_{\boldsymbol{\pi}} \left[ \int_0^T \left( r(\boldsymbol{x}'(t), \boldsymbol{u}(t)) + \lambda_n \|\boldsymbol{\sigma}_n(\boldsymbol{x}'(t), \boldsymbol{u}(t))\| \right) dt \right] \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}'(t) = \boldsymbol{f}_n(\boldsymbol{x}'(t), \boldsymbol{u}(t)). \end{aligned} \tag{6}$$

**Lemma 4.** *Let Assumption 2 and Assumption 5 hold. Consider the following definitions:*

$$J(\boldsymbol{\pi}, \boldsymbol{f}^*) = E\left[\int_0^T r(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t)))\, dt.\right]$$
$$\textit{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}^*(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t))), \quad \boldsymbol{x}_0 = \boldsymbol{x}(0),$$

$$J(\boldsymbol{\pi}, \boldsymbol{f}_n) = E\left[\int_0^T r(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t)))\, dt.\right]$$
$$\textit{s.t.} \quad \dot{\boldsymbol{x}'} = \boldsymbol{f}_n(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t))), \quad \boldsymbol{x}'_0 = \boldsymbol{x}(0),$$

$$\Sigma_n(\boldsymbol{\pi}, \boldsymbol{f}^*) = E\left[\int_0^T \|\boldsymbol{\sigma}_n(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t)))\|\ dt.\right]$$
$$\textit{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}^*(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t))) \quad \boldsymbol{x}_0 = \boldsymbol{x}(0),$$

$$\Sigma_n(\boldsymbol{\pi}, \boldsymbol{f}_n) = E\left[\int_0^T \|\boldsymbol{\sigma}_n(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t)))\|\ dt.\right]$$
$$\textit{s.t.} \quad \dot{\boldsymbol{x}'} = \boldsymbol{f}_n(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t))), \quad \boldsymbol{x}'_0 = \boldsymbol{x}(0).$$

*Furthermore, let* $\lambda_n = 2\beta_n L_r(1 + L_{\boldsymbol{\pi}})e^{L_*(1+L_{\boldsymbol{\pi}})T}$.

*Then, we have for all* $n \geq 0$, $\boldsymbol{\pi} \in \Pi$ *with probability at least* $1 - \delta$*:*

$$|J(\boldsymbol{\pi}, \boldsymbol{f}^*) - J(\boldsymbol{\pi}, \boldsymbol{f}_n)| \leq \lambda_n \Sigma_n(\boldsymbol{\pi}, \boldsymbol{f}_n)$$
$$|J(\boldsymbol{\pi}, \boldsymbol{f}^*) - J(\boldsymbol{\pi}, \boldsymbol{f}_n)| \leq \lambda_n \Sigma_n(\boldsymbol{\pi}, \boldsymbol{f}^*).$$

*Proof.*

$$\begin{aligned}
|J(\boldsymbol{\pi}, \boldsymbol{f}^*) - J(\boldsymbol{\pi}, \boldsymbol{f}_n)| &= \mathbb{E}\left[\int_0^T r(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t))) - r(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t)))\, dt.\right] \\
&\leq L_r(1 + L_{\boldsymbol{\pi}})\mathbb{E}\left[\int_0^T \|\boldsymbol{x}(t) - \boldsymbol{x}'(t)\|\ dt.\right] \\
&\leq 2\beta_n L_r(1 + L_{\boldsymbol{\pi}})Te^{L_{\boldsymbol{f}}(1+L_{\boldsymbol{\pi}})T}\int_0^T \|\boldsymbol{\sigma}_{n-1}(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t)))\|\ dt.
\end{aligned}$$

(Treven et al. [2023] Lemma 4.)

Similarly, we can also use Lemma 4 from Treven et al. [2023] and bound the regret with $\Sigma_n(\boldsymbol{\pi}, \boldsymbol{f}_n)$.

$$\begin{aligned}
|J(\boldsymbol{\pi}, \boldsymbol{f}^*) - J(\boldsymbol{\pi}, \boldsymbol{f}_n)| &= \mathbb{E}\left[\int_0^T r(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t))) - r(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t)))\, dt.\right] \\
&\leq L_r(1 + L_{\boldsymbol{\pi}})\mathbb{E}\left[\int_0^T \|\boldsymbol{x}(t) - \boldsymbol{x}'(t)\|\ dt.\right] \\
&\leq 2\beta_n L_r(1 + L_{\boldsymbol{\pi}})Te^{L_B(1+L_{\boldsymbol{\pi}})T}\int_0^T \|\boldsymbol{\sigma}_{n-1}(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t)))\|\ dt.
\end{aligned}$$

□

**Lemma 5.** *Let Assumption 2 and Assumption 5 hold and consider the simple regret at episode $n$:*

$$r_n = J(\boldsymbol{\pi}^*, \boldsymbol{f}^*) - J(\boldsymbol{\pi}_n, \boldsymbol{f}^*).$$

*The following holds for all $n > 0$ with probability at least $1 - \delta$:*

$$r_n \leq (2\lambda_n + \lambda_n^2)\Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*).$$

*Proof.*

$$\begin{aligned}
r_n &= J(\boldsymbol{\pi}^*, \boldsymbol{f}^*) - J(\boldsymbol{\pi}_n, \boldsymbol{f}^*) \\
&\leq J(\boldsymbol{\pi}^*, \boldsymbol{f}_n) + \lambda_n \Sigma_n(\boldsymbol{\pi}^*, \boldsymbol{f}_n) - J(\boldsymbol{\pi}_n, \boldsymbol{f}^*) && \text{(Lemma 4)} \\
&\leq J(\boldsymbol{\pi}_n, \boldsymbol{f}_n) + \lambda_n \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}_n) - J(\boldsymbol{\pi}_n, \boldsymbol{f}^*) && \text{(Equation (6))} \\
&= J(\boldsymbol{\pi}_n, \boldsymbol{f}_n) - J(\boldsymbol{\pi}_n, \boldsymbol{f}^*) + \lambda_n \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}_n) \\
&\leq \lambda_n \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*) + \lambda_n \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}_n) && \text{(Lemma 4)} \\
&= 2\lambda_n \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*) + \lambda_n (\Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}_n) - \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*)) \\
&\leq (\lambda_n^2 + 2\lambda_n)\Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*).
\end{aligned}$$

□

**Theorem 1** (Regret bound in the sub-Gaussian noise case)**.** *Let Assumption 2 and Assumption 5 hold. Then we have for all $N > 0$ with probability at least $1 - \delta$:*

$$R_N \leq \mathcal{O}\left(\mathcal{I}_N^{3/2}\sqrt{N}\right).$$

*Proof of Theorem 1.*

$$\begin{aligned}
R_N &= \sum_{n=1}^{N} r_n && \text{(Equation (2))} \\
&\leq \sum_{n=1}^{N} (\lambda_n^2 + 2\lambda_n)\Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*) && \text{(Lemma 5)} \\
&\leq (\lambda_N^2 + \lambda_N) \sum_{n=1}^{N} \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*) \\
&= (\lambda_N^2 + 2\lambda_N) \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{f}^*}\left[\int_0^T \|\boldsymbol{\sigma}_n(\boldsymbol{x}(t), \boldsymbol{\pi}_n(\boldsymbol{x}(t)))\| \, dt\right] \\
&\leq (\lambda_N^2 + 2\lambda_N)\sqrt{NT} \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{f}^*}\left[\int_0^T \left\|\boldsymbol{\sigma}_n^2(\boldsymbol{x}(t), \boldsymbol{\pi}_n(\boldsymbol{x}(t)))\right\| \, dt\right] \\
&\leq (\lambda_N^2 + 2\lambda_N)\sqrt{NT\mathcal{I}_N(\boldsymbol{f}^*, S)} && \text{(Treven et al. [2023], Proposition 1)}
\end{aligned}$$

□

**Theorem 2** (Sample complexity bound in the unsupervised case)**.** *Let Assumption 2 and Assumption 5 hold. Consider Algorithm 1 with extrinsic reward $r = 0$, i.e.,*

$$\begin{aligned}
&\boldsymbol{\pi}_n = \arg\max_{\boldsymbol{\pi} \in \Pi} \mathbb{E}_{\boldsymbol{\pi}}\left[\int_0^{T-1} \|\boldsymbol{\sigma}_n(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}'(t)))\| \, dt\right], \\
&\textit{s.t.} \quad \dot{\boldsymbol{x}}'(t) = \boldsymbol{f}_n(\boldsymbol{x}'(t), \boldsymbol{\pi}(\boldsymbol{x}(t))).
\end{aligned}$$

*Then we have for all $N > 0$, with probability at least $1 - \delta$:*

$$\max_{\boldsymbol{\pi} \in \Pi} \mathbb{E}_{\boldsymbol{f}^*}\left[\int_0^{T-1} \|\boldsymbol{\sigma}_n(\boldsymbol{x}(t), \boldsymbol{\pi}(\boldsymbol{x}(t)))\| \, dt\right] \leq \mathcal{O}\left(\sqrt{\frac{\mathcal{I}_N^3}{N}}\right).$$

*Proof of Theorem 2.* Let $\Sigma_N^* = \max_{\boldsymbol{\pi}} \Sigma_N(\boldsymbol{\pi}, \boldsymbol{f}^*)$ and $\boldsymbol{\pi}_N^*$ the corresponding policy.

$$
\begin{aligned}
\Sigma_N^* &\leq \frac{1}{N}\sum_{n=1}^{N} \Sigma_n^* && \text{(monotonicity of the variance)} \\
&\leq \frac{1}{N}\sum_{n=1}^{N} (1+\lambda_n)\Sigma_n(\boldsymbol{\pi}_n^*, \boldsymbol{f}_n) && \text{(Lemma 4)} \\
&\leq \frac{1}{N}\sum_{n=1}^{N} (1+\lambda_n)\Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}_n) && (\boldsymbol{\pi}_n \text{ is the maximizer for dynamics } \boldsymbol{f}_n) \\
&\leq \sum_{n=1}^{N} (1+\lambda_n)^2\Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*) && \text{(Lemma 4)} \\
&\leq (1+\lambda_N)^2 \frac{1}{N}\sum_{n=1}^{N} \Sigma_n(\boldsymbol{\pi}_n, \boldsymbol{f}^*) \\
&\leq (1+\lambda_N)^2 \frac{1}{\sqrt{N}}\sum_{n=1}^{N} \Sigma_n^2(\boldsymbol{\pi}_n, \boldsymbol{f}^*) \\
&\leq \mathcal{O}\left(\sqrt{\frac{\mathcal{I}_N^3}{N}}\right)
\end{aligned}
$$

□

# C Experimental setup

We provide additional details for our experiments in this section.

## C.1 GP experiments

We evaluate our method on two low-dimensional continuous control tasks: Pendulum-GP and MountainCar-GP [Moore, 1990]. Unlike in the other, following experiments, these environments are implemented directly by us as continuous-time systems with known physical dynamics given by nonlinear ODEs, rather than relying on Gym or DMC implementations. We emulate a continuous-time setting by using a fine time discretization for state propagation. As for measurements, we assume that we have direct access to the state derivatives.

In **Pendulum-GP**, the agent must swing up and stabilize a pendulum in the upright position. The state vector is defined as

$$
\boldsymbol{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \dot{\theta} \end{bmatrix}, \quad \boldsymbol{u} = u \in [-2, 2],
$$

where $\theta \in [-\pi, \pi]$ is the pendulum angle and $\dot{\theta}$ is the angular velocity. The underlying nonlinear ODE is:

$$
\frac{d\theta}{dt} = \dot{\theta}, \quad \frac{d\dot{\theta}}{dt} = \frac{3g}{2\ell}\sin\theta + \frac{3}{m\ell^2}u,
$$

with constants $g = 9.81\,\text{m/s}^2$, $m = 1$, and $\ell = 1$. We use a *Gym-style* reward, which penalizes deviations from the target angle $\theta = 0$, angular velocity $\dot{\theta}$, and control input $u$:

$$
r(\boldsymbol{x}, \boldsymbol{u}) = -\theta^2 - 0.1\,\dot{\theta}^2 - 0.02\,u^2,
$$

where $\theta = \arctan 2(x_1, x_0)$ and $\dot{\theta} = x_2$.

In **MountainCar-GP**, the agent must build momentum to propel a car up a steep hill. The state vector is defined as

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \boldsymbol{u} = u \in [-1, 1],$$

where $x_1 \in [-1.2, 0.6]$ is the position and $x_2 \in [-0.07, 0.07]$ is the velocity. The underlying nonlinear ODE is given by:

$$\frac{dx_1}{dt} = x_2, \quad \frac{dx_2}{dt} = 0.0015 \cdot u - 0.0025 \cos(3 \cdot x_1).$$

Position and velocity are clipped to their bounds, and backward motion is blocked at $x_1 = -1.2$ if $x_2 < 0$.The reward includes a terminal bonus of +100 for reaching the goal and penalizes control effort:

$$r(\boldsymbol{x}, \boldsymbol{u}) = -0.1\, u^2 + 100 \cdot \mathbf{1}_{\text{goal reached}},$$

where the goal is reached if the car's position exceeds $0.45$ and its velocity is non-negative.

For our GP experiments in Figure 1, we use the RBF kernel. The kernel parameters are updated online using maximum likelihood estimation [Rasmussen and Williams, 2005]. We use a hand-tuned static regime for the internal reward weight, i.e. $\lambda_n = \lambda$. For the OCORL baseline, we provide the confidence level function $\beta_n(\delta) = \beta$. The hyperparameters for the statistical model as well as for the environments and algorithms are given in Table 1.

Table 1: Model training hyperparameters and experimental setup for the GP-based experiments in Figure 1.

| Environment | $T$ [s] | $N$ | $\nu$ [$s^{-1}$] | Algorithm | $\lambda$ | $\beta$ | Learning Rate |
|---|---|---|---|---|---|---|---|
| Pendulum-GP | 2.5 | 12 | 20 | **COMBRL** | 1.0 | – | 0.01 |
| | | | | OCORL | 0 | 7.5 | |
| | | | | PETS | 0 | – | |
| | | | | Mean | 0 | – | |
| MountainCar-GP | 200 | 15 | 1 | **COMBRL** | $10^6$ | – | 0.01 |
| | | | | OCORL | 0 | 30 | |
| | | | | PETS | 0 | – | |
| | | | | Mean | 0 | – | |

Episode horizon $T$, number of episodes $N$, and measurement/control frequency $\nu$ are shared across algorithms for each environment.

For planning, we use the iCEM optimizer [Pinneri et al., 2021] even though it is a discrete-time algorithm. We emulate the continuous-time setting by using a fine time discretization (see measurement/control frequency $\nu$ in Table 1) and using the equidistant MSS. The hyperparameters for the planning are given in Table 2.

Table 2: iCEM hyperparameters used for planning in the GP-based experiments.

| Environment | Horizon | # Particles | # Samples | # Elites | Steps | $\alpha$ | Exponent |
|---|---|---|---|---|---|---|---|
| Pendulum-GP | 30 | 10 | 500 | 50 | 10 | 0.2 | 2 |
| MountainCar-GP | 100 | 10 | 500 | 50 | 5 | 0.2 | 2 |

*Horizon* refers to the iCEM planning horizon (in time steps). *Steps* indicates how many CEM optimization iterations are performed per control decision to refine the action distribution using elite samples. The number of time steps and control decisions are given by the measurement/control frequency $\nu$.

### C.2 Computational costs

We give the computational costs for our GP experiments in Table 3. This shows that the co-optimization over the reward and the optimistic dynamics as well as the reparametrization trick used by the OCORL algorithm are computationally prohibitive, and require around $3\times$ the compute time compared to **COMBRL**.

Table 3: Computation cost comparison (training time) for each algorithm across environments.

| Environment | **COMBRL** | OCORL | Mean | PETS |
|---|---|---|---|---|
| Pendulum-GP [a] | 10.8 ± 0.13 min | 30.6 ± 0.4 min | 10.4 ± 0.04 min | 30.78 ± 0.27 min |
| MountainCar-GP [b] | 1.29 ± 0.01 h | 4.55 ± 0.1 h | 1.28 ± 0.03 h | 4.67 ± 0.16 h |

[a] Mean total training time. GPU: NVIDIA GeForce RTX 2080 Ti.
[b] Mean training time per episode. GPU: NVIDIA GeForce RTX 2080 Ti.

### C.3 Auto-tuning experiments

In Figure 2, we auto-tune the intrinsic reward weight $\lambda_n$ following the method proposed by Sukhija et al. [2025a], who demonstrate that this approach is effective across a range of model-free off-policy RL methods in discrete time. Specifically, we adjust $\lambda$ by minimizing the loss:

$$L(\lambda) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}_{1:n},\boldsymbol{u}\sim\boldsymbol{\pi}_n,\bar{\boldsymbol{u}}\sim\bar{\boldsymbol{\pi}}_n} \log(\lambda)(\boldsymbol{\sigma}_n(\boldsymbol{x},\boldsymbol{u}) - \boldsymbol{\sigma}_n(\boldsymbol{x},\bar{\boldsymbol{u}})). \tag{7}$$

Here, $\bar{\boldsymbol{\pi}}_n$ denotes a slowly updated target policy obtained via Polyak averaging of $\boldsymbol{\pi}_n$. The objective promotes larger $\lambda$ values when the current policy explores less than the target policy.

### C.4 BNN experiments

In our experiments that do not explicitly use Gaussian Processes, we train an ensemble of 5 neural networks to model forward dynamics. Model epistemic uncertainty is estimated via the disagreement among the ensemble members [Pathak et al., 2019]. To further leverage the model, we augment the data by including synthetic transitions. For each policy update, we sample real transitions $(\boldsymbol{x},\boldsymbol{u},\dot{\boldsymbol{y}})$ from the replay buffer $\mathcal{D}_{1:n}$ and add corresponding model-predicted transitions $(\boldsymbol{x},\boldsymbol{u},\dot{\boldsymbol{y}}')$, where $\dot{\boldsymbol{y}}'$ is generated by the mean model $\boldsymbol{\mu}_n$. This lets us blend real and synthetic rollouts, similar to the strategy used by Janner et al. [2019], thereby increasing the update-to-data (UTD) internal ratio.

We adopt the same hyperparameters as Sukhija et al. [2025a] for optimizing the loss via stochastic gradient descent in Equation (7) and for configuring the UTD. We also periodically perform soft resets for the policy for training stability [D'Oro et al., 2023]. The hyperparameters for the statistical model and SAC are given in Table 4. For the ensemble-based experiments, we use several environments from the Gym and DMC benchmark suites [Brockman et al., 2016, Tunyasuvunakool et al., 2020]. We adapt them to the continuous-time setting by approximating the derivatives using a finite difference filter. The measurement/control frequency is given by the duration of an environment step `dt`.

Table 4: Hyperparameters for ensemble-based experiments with SAC in Section 4, grouped by environment family.

| Environments | Action Repeat | Policy / Critic Architecture | Model Architecture | Learning Rate | Batch Size |
|---|---|---|---|---|---|
| Gym – Pendulum / MountainCar | 1 | (256,256) | $5\times$(256,256) | $3\times10^{-4}$ | 256 |
| Gym – Reacher / DMC – Quadruped | 2 | (256,256) | $5\times$(512,512) | $3\times10^{-4}$ | 256 |
| Gym – other environments[a] | 2 | (256,256) | $5\times$(256,256) | $3\times10^{-4}$ | 256 |
| DMC – all except Quadruped | 2 | (256,256) | $5\times$(256,256) | $3\times10^{-4}$ | 256 |

[a] HalfCheetah, Hopper, Pusher.

### C.5 Downstream tasks

To evaluate zero-shot generalization in Figures 3 and 4, we introduce custom downstream tasks that differ semantically from the primary training objective. While the primary task corresponds to the default reward in each Gym or DMC environment, the downstream task uses a custom reward function that incentivizes behaviour that contrasts with the original task (e.g., moving away instead of toward a goal).

We implement each downstream task by overriding the reward computation in the Gym or DMC environment using environment wrappers. Table 5 summarizes the evaluated primary and downstream tasks. It also gives the internal reward parameter $\lambda_n$ for the experiments shown in Figure 3. For said experiments, the downstream rewards are defined as follows:

- **MountainCar** – go up left: Encourages the car to reach the leftmost side of the hill, in contrast to the standard goal on the right.
- **HalfCheetah** – run backwards: Reverses the locomotion objective by rewarding backward velocity.
- **Hopper** – hop backwards: Rewards hopping backwards while maintaining a healthy posture.[3]
- **Pendulum**
  - Balance upright: Starts upright and rewards maintaining the upright position.
  - Swing up: Starts with the pendulum pointing downward and rewards swinging it up.
  - Swing down: Starts upright and rewards swinging the pendulum downward.
  - Keep down: Starts downward and rewards staying down.
- **Reacher** – go away: Penalizes proximity to the goal, inverting the standard reaching task.
- **Pusher** – push away from target: Encourages the agent to push the object away from the goal location.

Table 5: Primary and downstream tasks used in our evaluation. Each downstream task is defined via a custom reward that encourages behaviour contrasting the primary task. We also provide the algorithm hyperparameters used in Figure 3, namely the strategy used and the optimized internal reward weight $\lambda_n$.

| Environment | Primary Task | Downstream Task | Strategy[a] | $\lambda$ |
|---|---|---|---|---|
| MountainCar | Go up right | Go up left | Annealing | 50 |
| HalfCheetah | Run forward | Run backward | Annealing | 2 |
| Hopper | Hop forward | Hop backward | Static | 10 |
| Pendulum | Balance upright | Swing down | Annealing | 10 |
| Pendulum | Keep down | Swing up | Annealing | 50 |
| Reacher | Reach target | Keep away | Static | 0.17 |
| Pusher | Push to target | Push away | Static | 0.56 |

[a] Static denotes a fixed internal reward weight $\lambda_n = \lambda$, annealing a decreasing reward weight $\lambda_n \propto \lambda \cdot (1 - n/N)$.

[3] The Hopper environment in Gym introduces a `healthy_reward`, which is preserved for the downstream task.