# Meta-RL with Bayesian Linear Task Models

**Jingyang You and Hanna Kurniawati**
School of Computing
Australian National University
{jingyang.you, hanna.kurniawati}@anu.edu.au

## Abstract

Deep Bayesian reinforcement learning adapts to unseen tasks by inferring latent transition and reward models, but existing methods typically rely on variational posteriors and evidence lower bounds, introducing approximation error and unstable task representations. We introduce GLiBRL, a deep Bayesian RL framework that combines generalised linear task models with learnable non-linear basis functions. GLiBRL features conjugate Bayesian inference, yielding exact, sequential posterior updates over task parameters and model noise, together with a closed-form marginal likelihood that eliminates variational inference. The update is naturally permutation-invariant, allowing GLiBRL to integrate with both off- and on-policy algorithms. GLiBRL also learns task representation admitting an exact kernel identity, relating distances between task representations to kernel discrepancies over the task contexts. Compared against eight representative or recent meta reinforcement learning methods, GLiBRL achieves the highest aggregate zero-shot test performance on both the MuJoCo locomotion and MetaWorld manipulation benchmarks.

## 1 Introduction

Reinforcement Learning (RL) algorithms possess significant potential to enable autonomous, intelligent robotic control. However, standard RL algorithms typically do not account for variations in transition and reward dynamics, causing them to generalise poorly to unseen tasks where the underlying environmental models deviate from those encountered during training. Bayesian Reinforcement Learning (BRL), a special case of Meta-Reinforcement Learning (Meta-RL), offers an effective framework to overcome this limitation.

Instead of ignoring environmental variations, BRL explicitly accounts for them by assuming distributions over the transition and reward models, continually performing Bayesian inference on these parameters [12]. The resulting parameter distributions implicitly encode varying task dynamics. To solve these BRL problems, many classical methods employ planners [16, 28] to search for Bayes-optimal policies. However, these methods scale poorly and require explicit prior knowledge regarding the structural forms of the transition and reward models, severely restricting their generalisation across diverse tasks.

To address such rigidity, recent deep BRL methods [29, 42] enable model learning by optimising the marginal likelihood of the observed data. Unfortunately, the direct application of neural networks to the joint space of data and task parameters renders exact Bayesian inference intractable, preventing direct optimisation of the exact marginal likelihood. Consequently, these methods rely on variational inference to optimise the Evidence Lower Bound (ELBO). This introduces challenges such as high-variance Monte Carlo estimates, amortisation gaps [5] and posterior collapse [4, 6]. In the context of BRL, these issues often prevent the extraction of meaningful and distinctive task parameters which are crucial for effective BRL.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

To alleviate these issues, we propose **G**eneralised **Li**near Models in Deep **B**ayesian **RL** with Learnable Basis Functions (**GLiBRL**)[1]. Inspired by the deep Bayesian linear regression structure proposed in ALPaCA [19], GLiBRL enforces a generalised linear relation between task parameters and data features computed via learnable basis functions. This architecture circumvents variational inference entirely, permitting exact posterior inference and the computation of a closed-form marginal likelihood. By performing exact inference over the model noise, GLiBRL naturally generalises ALPaCA while reducing the error of predictions in unseen tasks. The permutation-invariance of GLiBRL induced by the exact Bayesian update also allows seamless integration with both on- and off-policy RL algorithms. Moreover, we show an exact kernel identity makes GLiBRL's task-representation geometry analytically traceable, relating distances between task representations to signed-measure kernel discrepancies over the task contexts (Appendix A.10).

We evaluate GLiBRL on the MuJoCo [34] locomotion benchmark and the challenging MetaWorld [41, 24] ML10/45 benchmarks. Compared against a diverse suite of Meta-RL baselines, including PEARL [29], VariBAD [42], SDVT [23], $RL^2$ [37, 8], MAML [10], AMAGO [14], TrMRL [25] and ECET [31], GLiBRL achieves the highest observed aggregate performance among the evaluated methods across both locomotion and manipulation domains.

## 2 Related Work

**Bayesian Last Layers**. GLiBRL is most relevant to ALPaCA [19], which introduces the concept of performing Bayesian inference only on the output layer of neural networks. ALPaCA provides an efficient, online Bayesian linear regression framework that also employs learnable basis functions. However, ALPaCA, alongside its subsequent extensions [20], was not formulated as a BRL method. While related work such as CAMeLiD [18] employs similar controllers for policy computation, they rely on the restrictive assumption of fully known reward functions. GLiBRL generalises ALPaCA and CAMeLiD in two ways: (1) whereas ALPaCA and CAMeLiD assume a *known* noise in the likelihood function, GLiBRL performs exact Bayesian inference over the model noise, (2) GLiBRL operates seamlessly in online BRL settings with unknown rewards, whereas ALPaCA and CAMeLiD focus on offline problems with simple, known rewards. As we will show in Appendix A.16, the assumption of known noise induces predictive errors in both transition and reward models.

**Reinforcement Learning**. RL methods are broadly categorised into model-free and model-based approaches. Here, we use model-free algorithms, specifically Proximal Policy Optimization (PPO) [30] and Soft Actor-Critic (SAC) [17], to learn the BRL policies, aligning with a substantial portion of the Meta-RL literature. Furthermore, as GLiBRL learns transition and reward models, it is naturally compatible with model-based methods.

**Hidden-Parameter MDPs**. Hidden-Parameter MDPs (HiP-MDPs) [7] provide a framework for parametric Bayesian Reinforcement Learning. Initially modelled using Gaussian Processes (GPs) [7], HiP-MDPs were subsequently scaled by replacing GPs with Bayesian Neural Networks (BNNs) [22]. However, updating BNN weights at test-time has proved inefficient [38]. While follow-up work mitigated such inefficiency by fixing the test-time BNN weights and exclusively optimising task parameters [39], doing so strips away the true Bayesian features of the framework. Optimising task parameters while fixing BNN weights is mathematically equivalent to performing Maximum Likelihood Estimation (MLE), diverting the agent from Bayes-optimal exploration strategies (a limitation similarly noted by [42]). In contrast, recent parametric deep BRL methods [29, 27, 42, 23], including GLiBRL, are considered orthogonal to this line of work, as they do not assume known forms of reward functions and perform (approximate) Bayesian inference directly on task parameters rather than the neural network weights.

**Classical Bayesian Reinforcement Learning**. Classical BRL methods assume known forms of transitions and rewards. Early approaches formulated BRL as a Partially Observable MDP (POMDP) [21] utilising sampling-based offline solvers [28], while later methods introduced online, tree-based solvers leveraging posterior sampling [33, 26] for efficiency [16]. These methods plan to find approximately optimal policies, which is orthogonal to the direct policy learning in GLiBRL. GLiBRL shares the idea of using generalised linear models with [35], while it differs fundamentally in that [35] *specifies* the basis functions, instead of learning them as in GLiBRL. Hand-crafting even a simple non-linear basis function, e.g., quadratic function, yields an $O(d^2)$ dimensional feature

[1] https://github.com/RDLLab/GLiBRL

space, where $d$ is the dimension of the raw input. As detailed in Section 4, performing online Bayesian inference scales at least quadratically with respect to the feature space dimension, meaning a prohibitive $O(d^4)$ complexity is required for a single inference step. In contrast, by *learning* basis functions, GLiBRL enables the use of compact low-dimensional feature spaces that capture non-linearities while preserving scalability.

**Meta-Reinforcement Learning**. Meta-Reinforcement Learning (Meta-RL) aims to learn policies from *seen* tasks that are capable of adapting to *unseen* tasks following similar task distributions [3]. According to [3], Meta-RL methods can be categorised as **(1) parameterised policy gradient (PPG)** methods (e.g., [10, 40, 11]) that learn by performing meta-policy gradients on the meta-parameterised policy; **(2) black-box methods** (e.g., [8, 37, 25, 14, 31]) that learn from summaries of histories and **(3) task-inference methods** (e.g., [29, 42, 23]) that learn from the belief of the task parameters (i.e., $\theta_T, \theta_R$ in GLiBRL). Most of the deep BRL methods can be viewed as task inference methods, hence a subset of Meta-RL. BRL differs from other Meta-RL methods in that it performs Bayesian inference on task parameters, affording advantages such as uncertainty quantification. As a result, BRL methods have natural integrations with model-based algorithms, holding significance to research fields such as risk-aware control and planning under uncertainty.

# 3 Background

## 3.1 Markov Decision Processes and Reinforcement Learning

Markov Decision Processes (MDPs) are 5-tuples defined as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, R, T, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $R(s_t, a_t, s_{t+1}, r_{t+1}) = p(r_{t+1}|s_t, a_t, s_{t+1})$ is the *reward* function, $T(s_t, a_t, s_{t+1}) = p(s_{t+1}|s_t, a_t)$ is the *transition* function, and $\gamma \in [0, 1]$ a discount factor. In the above definition, $s_t \in \mathcal{S}, s_{t+1} \in \mathcal{S}, a_t \in \mathcal{A}$ and $r_{t+1} \in \mathbb{R}$. The goal of solving an MDP is to find a policy $\pi(a_t|s_t) : \mathcal{S} \to \mathcal{A}$ that maximises the expected return over a finite horizon $H > 0$, given by $\mathcal{J}(\pi) = \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t r_{t+1}]$. Standard Reinforcement Learning (RL) problems optimise this exact objective; however, they assume the transition dynamics $T$, the reward function $R$, or both are unknown and must be learnt from data.

## 3.2 Bayes-Adaptive MDPs and Bayesian Reinforcement Learning

Bayes-Adaptive MDPs (BAMDPs) [9, 12] parameterise transition and reward functions by unknown variables $\theta_T \in \Theta_T$ and $\theta_R \in \Theta_R$, respectively. Following POMDP terminology [21], a BAMDP maintains their joint distribution, or *belief*, $b_t = p_t(\theta_T, \theta_R)$. After observing $(s_t, a_t, s_{t+1}, r_{t+1})$, Bayesian inference updates it as $b_{t+1} = p_{t+1}(\theta_T, \theta_R)$.

To efficiently use existing MDP frameworks, beliefs can be absorbed into the original state space to form hyper-states $\mathcal{S}^+ = \mathcal{S} \times \mathcal{B}(\Theta_T \times \Theta_R)$. Hence, BAMDPs can be defined as 5-tuple $(\mathcal{S}^+, \mathcal{A}, R^+, T^+, \gamma)$ MDPs. Let $\bar{b}_{t+1} = \tau_s(b_t, s_t, a_t, s_{t+1})$ be the intermediate joint belief after observing the next state. The joint one-step kernel factors as

$$\begin{aligned} &p(s_{t+1}, r_{t+1}, b_{t+1} \mid s_t, b_t, a_t) \\ &= \mathbb{E}_{(\theta_T,\theta_R)\sim b_t}[p(s_{t+1} \mid s_t, a_t, \theta_T)]\, \mathbb{E}_{(\theta_T,\theta_R)\sim \bar{b}_{t+1}}[p(r_{t+1} \mid s_t, a_t, s_{t+1}, \theta_R)] \\ &\quad \cdot \delta\big(b_{t+1} = \tau_r(\bar{b}_{t+1}, s_t, a_t, s_{t+1}, r_{t+1})\big)\,. \end{aligned} \tag{1}$$

The first and second expectations are the predictive factors $T^+$ and $R^+$, respectively. Thus, observing $s_{t+1}$ produces $\bar{b}_{t+1}$, which predicts $r_{t+1}$; observing the reward then produces the final belief $b_{t+1}$ used at the next decision. Accordingly, the expected return to maximise becomes $\mathcal{J}^+(\pi^+) = \mathbb{E}_{T^+,\pi^+,R^+}[\sum_{t=0}^{H^+} \gamma^t r_{t+1}]$, where $H^+ > 0$ is the BAMDP horizon, and $\pi^+ : \mathcal{S}^+ \to \mathcal{A}$ is the policy of BAMDPs [2]. Problems that require solving BAMDPs are named Bayesian Reinforcement Learning (BRL). Beyond improving generalisation, BRL offers a principled mechanism for addressing the exploration-exploitation trade-off [12].

Classical BRL methods [28, 16, 35] typically assume known functional forms for $T^+$ and $R^+$, limiting flexibility. HiP-MDPs [7, 22, 39] instead infer neural-network weights, but scale poorly [38] and retain known reward forms. Recent deep BRL methods [27, 18, 29, 42] use neural networks while

[2] We will use BAMDP policy and BRL policy interchangeably when referring to $\pi^+$.

performing approximate inference over task parameters; GLiBRL follows this scalable paradigm while accommodating unknown rewards.

To contextualise our method, we formalise model learning in the recent deep BRL setting. The agent interacts with a set of MDPs with unknown transition dynamics and/or reward functions. A single sample at timestep $t$ is denoted as $c_t := (s_t, a_t, s_{t+1}, r_{t+1})$. Within a given MDP $\mathcal{M}$, we define the collection of $N$ samples as $\mathcal{C}^{\mathcal{M}} := \{c_t\}_{t=1}^{N}$. During training, assuming access to a batch of $M$ MDPs $\{\mathcal{M}_i\}_{i=1}^{M}$, the resulting joint dataset is given by $\mathcal{C} = \bigcup_{i=1}^{M} \mathcal{C}^{\mathcal{M}_i}$. The objective in most recent deep BRL methods [29, 27, 42] is to maximise the marginal log-likelihood of this joint data

$$\log p_{\zeta,\phi_T,\phi_R}(\mathcal{C}) = \sum_{i=1}^{M} \log p_{\zeta,\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}) = \sum_{i=1}^{M} \log \iint p_\zeta(\theta_T, \theta_R) p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R) d\theta_T d\theta_R \quad (2)$$

where $\zeta$ is an optional neural network parameterising the prior $p(\theta_T, \theta_R)$. The likelihood of an individual task factors as $p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R) \propto \prod_{t=1}^{N} p_{\phi_T}(s_{t+1}|s_t, a_t, \theta_T) p_{\phi_R}(r_{t+1}|s_t, a_t, s_{t+1}, \theta_R)$, where $\phi_T$ and $\phi_R$ denote neural networks to learn the forms of transition and reward functions, respectively.

To facilitate optimisation, $p_{\phi_T}$ and $p_{\phi_R}$ are typically modelled as Gaussians with means and diagonal covariances determined by the output of neural networks $\phi_T, \phi_R$. The prior $p_\zeta(\theta_T, \theta_R)$ is also assumed to be a Gaussian. However, note that even with these distributional simplifications, the exact marginal likelihood in Equation 2 remains intractable. Because the neural network introduces high non-linearity between task parameters $\theta_T, \theta_R$ and the data, the analytical integration is impossible. Consequently, these methods are forced to resort to variational inference, optimising the Evidence Lower Bound (ELBO) as a tractable surrogate objective (proof see Appendix A.5):

$$\log p_{\zeta,\phi_T,\phi_R}(\mathcal{C}) \geq \sum_{i=1}^{M} \mathbb{E}_q \left[\log p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R)\right] - D_{KL}\left(q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i}) \| p_\zeta(\theta_T, \theta_R)\right) \quad (3)$$

where $D_{KL}(\cdot||\cdot)$ is the KL-divergence, and $q(\cdot)$ is an approximate Gaussian posterior of $\theta_T, \theta_R$.

Variational inference enables model learning, but it introduces optimisation challenges such as amortisation gaps [5] and posterior collapse [4, 6]. These issues frequently prevent learning meaningful and distinctive task representations. Different from other tasks where the quality of learnt representations might be less important, BRL policies rely heavily on continually updated task representations to compute actions. Indistinctive representations will substantially harm the performance of BRL policies.

The other challenge is to compute the approximate posterior $q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})$ directly on variable-length $\mathcal{C}^{\mathcal{M}_i}$. The most common approach is to use sequential models like RNNs or Transformers [36, 14, 25, 31] to summarise $\mathcal{C}^{\mathcal{M}_i}$. However, because these models are permutation-variant, directly combining them with sample-efficient off-policy algorithms like Soft Actor-Critic (SAC) [17] causes a severe performance drop, as demonstrated by an ablation study in PEARL [29]. The other approach, used by PEARL, is a factored approximation $q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i}) \approx \prod_{t=1}^{N} \mathcal{N}(\theta_T, \theta_R|g(\mathcal{C}^{\mathcal{M}_i}[t]))$, where $g(\cdot)$ is a neural network that takes the $t$-th sample in $\mathcal{C}^{\mathcal{M}_i}$ as the input and returns the mean and covariance of a Gaussian as the output. However, Bayes' rule reveals that $g(\cdot)$ must essentially predict $p(\theta_T, \theta_R)^{(1/N)-1} q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i}[t])$. Because this target relies heavily on the length $N$, it creates shifting optimisation targets as the dataset grows, resulting in inaccurate posterior approximations. In contrast, GLiBRL completely circumvents the need to compute an approximate posterior in the first place; its exact Bayesian update is naturally permutation-invariant, enabling seamless integration with SAC, as we will demonstrate in Section 5.

## 4 Meta-RL with Bayesian Linear Task Models

GLiBRL is an online algorithm whose training alternates between data collection and learning. During data collection, a batch of MDPs $\{\mathcal{M}_i\}_{i=1}^{K}$ is sampled from the task distribution. Samples $\{\mathcal{C}^{\mathcal{M}_i}\}_{i=1}^{K}$ are maintained for each MDP separately. Starting from state $s$, to collect samples $(s, a, s', r)$ to be stored in $\mathcal{C}^{\mathcal{M}_i}$, GLiBRL sequentially performs the following at each timestep: (1) updates the current belief $b$ to the exact posterior distribution $p_{\phi_T,\phi_R}(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})$ (2) samples an action $a$ from the

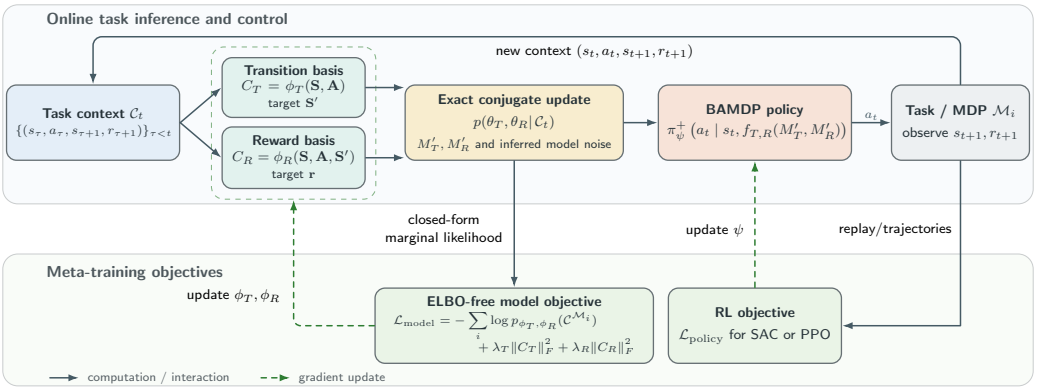



Figure 1: Overview of GLiBRL. A task context is mapped through learned transition and reward basis functions to exact Bayesian posteriors. The posterior means form a task representation supplied to a BAMDP policy. During meta-training, the basis functions are optimised using the closed-form marginal-likelihood objective, while the policy is optimised using the corresponding reinforcement-learning objective. At test time, adaptation consists only of sequential posterior updates.

BAMDP policy $\pi^+_\psi(a|s,b)$ where $\psi$ is a neural network, and (3) executes $a$ to observe the next state $s'$ and reward $r$. Once sufficient samples are collected across tasks, GLiBRL samples a batch of samples $\mathcal{D}$ from the joint buffer $\mathcal{C} = \bigcup_i \mathcal{C}^{\mathcal{M}_i}$ and updates the BAMDP policy network $\psi$ and transition/reward models $\phi_T, \phi_R$ via gradient descent on $\mathcal{L}_{\text{policy}}$ and $\mathcal{L}_{\text{model}}$, respectively. The complete procedure is detailed in Algorithm 1, and we include a diagram of GLiBRL in Figure 1.

The success of GLiBRL relies on solving three primary challenges: (1) formulating a tractable, sequentially updated posterior $p_{\phi_T,\phi_R}(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})$, (2) deriving an ELBO-free objective $\mathcal{L}_{\text{model}}$ for updating $\phi_T, \phi_R$, and (3) constructing theoretically grounded task representations for learning the policy $\pi^+_\psi$. We will detail these solutions in Section 4.1, Section 4.2, and Section 4.3, respectively.

### 4.1 Tractable and Sequentially Updated Posterior Distributions

Inspired by ALPaCA [19], to derive the exact posterior $p_{\phi_T,\phi_R}(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})$, we must explicitly define the prior $p(\theta_T, \theta_R)$ and the likelihood $p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R)$. Let the task data be denoted compactly as $\mathcal{C}^{\mathcal{M}_i} = (\mathbf{S}_i \in \mathbb{R}^{N\times D_S}, \mathbf{A}_i \in \mathbb{R}^{N\times D_A}, \mathbf{S}'_i \in \mathbb{R}^{N\times D_S}, \mathbf{r}_i \in \mathbb{R}^{N\times 1})$, where $N$ is the number of samples and $D_S, D_A$ are the dimensions of the state and action space. Let the task variables be defined as $\theta_T = (T_\mu \in \mathbb{R}^{D_T\times D_S}, T_\sigma \in \mathbb{R}^{D_S\times D_S})$ and $\theta_R = (R_\mu \in \mathbb{R}^{D_R\times 1}, R_\sigma \in \mathbb{R}^{1\times 1})$, where $D_T, D_R$ are task dimensions. We formulate the prior as Normal-Wishart distributions:

$$p(\theta_T, \theta_R) = p(\theta_T) \cdot p(\theta_R) \qquad \text{(Assuming independent } \theta_T, \theta_R) \tag{4}$$

$$p(\theta_T) = \mathcal{MN}(T_\mu|\mathbf{M}_T \in \mathbb{R}^{D_T\times D_S}, \mathbf{\Xi}_T^{-1} \in \mathbb{R}^{D_T\times D_T}, T_\sigma) \cdot \mathcal{W}(T_\sigma^{-1}|\mathbf{\Omega}_T^{-1} \in \mathbb{R}^{D_S\times D_S}, \nu_T) \tag{5}$$

$$p(\theta_R) = \mathcal{MN}(R_\mu|\mathbf{M}_R \in \mathbb{R}^{D_R\times 1}, \mathbf{\Xi}_R^{-1} \in \mathbb{R}^{D_R\times D_R}, R_\sigma) \cdot \mathcal{W}(R_\sigma^{-1}|\mathbf{\Omega}_R^{-1} \in \mathbb{R}^{1\times 1}, \nu_R) \tag{6}$$

Here, $\mathcal{W}(\mathbf{W}|\mathbf{X}, \nu)$ defines a Wishart distribution on positive definite random matrix $\mathbf{W}$ with scale $\mathbf{X}$ and degrees of freedom $\nu$, and $\mathcal{MN}(\mathbf{W}|\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ defines a matrix normal distribution with random matrix $\mathbf{W}$, mean $\mathbf{X}$, row covariance $\mathbf{Y}$ and column covariance $\mathbf{Z}$. Normal-Wishart priors enable fully tractable Bayesian inference over the model noises $T_\sigma, R_\sigma$, generalising previous work [18, 19]. We then define the Matrix-Normal likelihood function:

$$p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R) = p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i, \theta_T) \cdot p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \theta_R) \cdot p(\mathbf{S}_i, \mathbf{A}_i) \tag{7}$$

$$p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i, \theta_T) = \mathcal{MN}(\mathbf{S}'_i|\mathbf{C}_T T_\mu, \mathbf{I}_N, T_\sigma) \tag{8}$$

$$p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \theta_R) = \mathcal{MN}(\mathbf{r}_i|\mathbf{C}_R R_\mu, \mathbf{I}_N, R_\sigma) \tag{9}$$

$$\mathbf{C}_T \in \mathbb{R}^{N\times D_T} = \phi_T(\mathbf{S}_i, \mathbf{A}_i) \qquad \mathbf{C}_R \in \mathbb{R}^{N\times D_R} = \phi_R(\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) \tag{10}$$

The term $p(\mathbf{S}_i, \mathbf{A}_i)$ acts as a constant irrelevant to the optimisation, while $\mathbf{C}_T$ and $\mathbf{C}_R$ [3] represent data features generated by neural networks $\phi_T, \phi_R$ acting as expressive and learnable basis functions.

[3] Dependence on $i$ of $\mathbf{C}_T, \mathbf{C}_R$ is omitted for conciseness. This also applies to $\mathbf{M}'_T, \mathbf{\Xi}'_T, \mathbf{\Omega}'_T, \mathbf{M}'_R, \mathbf{\Xi}'_R, \mathbf{\Omega}'_R$.

By restricting the linearisation exclusively to the mapping between the feature space and the Bayesian parameters, we preserve exact mathematical tractability.

Because the Normal-Wishart prior is conjugate to the Matrix-Normal likelihood, the resulting posteriors also follow Normal-Wishart distributions (proof in Appendix A.6)

$$p_{\phi_T,\phi_R}(\theta_T,\theta_R|\mathcal{C}^{\mathcal{M}_i}) = p_{\phi_T}(\theta_T|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i)\cdot p_{\phi_R}(\theta_R|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i,\mathbf{r}_i) \tag{11}$$

$$p_{\phi_T}(\theta_T|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i) = \mathcal{MN}(T_\mu|\mathbf{M}'_T,\mathbf{\Xi}'^{-1}_T,T_\sigma)\cdot\mathcal{W}(T_\sigma^{-1}|\mathbf{\Omega}'^{-1}_T,\nu'_T) \tag{12}$$

$$p_{\phi_R}(\theta_R|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i,\mathbf{r}_i) = \mathcal{MN}(R_\mu|\mathbf{M}'_R,\mathbf{\Xi}'^{-1}_R,R_\sigma)\cdot\mathcal{W}(R_\sigma^{-1}|\mathbf{\Omega}'^{-1}_R,\nu'_R) \tag{13}$$

where

$$\begin{aligned}
\mathbf{M}'_T &= \mathbf{\Xi}'^{-1}_T\left[\mathbf{C}_T^{\mathrm{T}}\mathbf{S}' + \mathbf{\Xi}_T\mathbf{M}_T\right] & \mathbf{M}'_R &= \mathbf{\Xi}'^{-1}_R\left[\mathbf{C}_R^{\mathrm{T}}\mathbf{r} + \mathbf{\Xi}_R\mathbf{M}_R\right]\\
\mathbf{\Xi}'_T &= \mathbf{C}_T^{\mathrm{T}}\mathbf{C}_T + \mathbf{\Xi}_T & \mathbf{\Xi}'_R &= \mathbf{C}_R^{\mathrm{T}}\mathbf{C}_R + \mathbf{\Xi}_R\\
\mathbf{\Omega}'_T &= \mathbf{\Omega}_T + \mathbf{S}'^{\mathrm{T}}\mathbf{S}' & \mathbf{\Omega}'_R &= \mathbf{\Omega}_R + \mathbf{r}^{\mathrm{T}}\mathbf{r}\\
&\quad + \mathbf{M}_T^{\mathrm{T}}\mathbf{\Xi}_T\mathbf{M}_T - \mathbf{M}'^{\mathrm{T}}_T\mathbf{\Xi}'_T\mathbf{M}'_T & &\quad + \mathbf{M}_R^{\mathrm{T}}\mathbf{\Xi}_R\mathbf{M}_R - \mathbf{M}'^{\mathrm{T}}_R\mathbf{\Xi}'_R\mathbf{M}'_R\\
\nu'_T &= \nu_T + N & \nu'_R &= \nu_R + N
\end{aligned} \tag{14}$$

During online execution, the agent must sequentially update priors to posteriors with each newly observed sample $c_t = \{s_t, a_t, s'_t, r_{t+1}\}$ for arbitrary timestep $t$. Naively computing Equation 14 requires inversion of $\mathbf{\Xi}'_T$ and $\mathbf{\Xi}'_R$, which scale at $\mathcal{O}(D_T^3)$ and $\mathcal{O}(D_R^3)$, respectively. Applying the matrix inversion lemma yields

$$\mathbf{\Xi}'^{-1}_T = \left(\mathbf{C}_T^{\mathrm{T}}\mathbf{C}_T + \mathbf{\Xi}_T\right)^{-1} = \mathbf{\Xi}_T^{-1} - \mathbf{\Xi}_T^{-1}\mathbf{C}_T^{\mathrm{T}}\left(\mathbf{I}_N + \mathbf{C}_T\mathbf{\Xi}_T^{-1}\mathbf{C}_T^{\mathrm{T}}\right)^{-1}\mathbf{C}_T\mathbf{\Xi}_T^{-1} \tag{15}$$

$$\mathbf{\Xi}'^{-1}_R = \left(\mathbf{C}_R^{\mathrm{T}}\mathbf{C}_R + \mathbf{\Xi}_R\right)^{-1} = \mathbf{\Xi}_R^{-1} - \mathbf{\Xi}_R^{-1}\mathbf{C}_R^{\mathrm{T}}\left(\mathbf{I}_N + \mathbf{C}_R\mathbf{\Xi}_R^{-1}\mathbf{C}_R^{\mathrm{T}}\right)^{-1}\mathbf{C}_R\mathbf{\Xi}_R^{-1} \tag{16}$$

With a single sample, $\mathbf{C}_T \in \mathbb{R}^{1\times D_T}$ and $\mathbf{C}_R \in \mathbb{R}^{1\times D_R}$. Hence, $\left(\mathbf{I}_N + \mathbf{C}_T\mathbf{\Xi}_T^{-1}\mathbf{C}_T^{\mathrm{T}}\right)^{-1}$ and $\left(\mathbf{I}_N + \mathbf{C}_R\mathbf{\Xi}_R^{-1}\mathbf{C}_R^{\mathrm{T}}\right)^{-1}$ are reduced to reciprocals of scalars. By caching the inverted matrices, this reduces the entire online sequential update complexity to $\mathcal{O}(\max\left(D_S^2, D_T^2\right))$ for transitions and $\mathcal{O}(D_R^2)$ for rewards, enabling efficient data collection.

### 4.2 ELBO-free Model Updates

The fully tractable posterior update yields a closed-form marginal likelihood, which contributes to the model loss $\mathcal{L}_{\text{model}}$ for optimising model parameters $\phi_T, \phi_R$. Dropping the optional neural network $\zeta$ , Equation 2 can be written analytically as

$$\begin{aligned}
\log p_{\phi_T,\phi_R}(\mathcal{C}) = &\sum_{i=1}^{M}\log\int p(\theta_T)p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i,\theta_T)d\theta_T\\
&+\sum_{i=1}^{M}\log\int p(\theta_R)p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i,\theta_R)d\theta_R + \text{const.}
\end{aligned} \tag{17}$$

Substituting accordingly with equations in Section 4.1, we reduce the integral to a closed-form marginal log-likelihood (proof in Appendix A.7):

$$\log p_{\phi_T,\phi_R}(\mathcal{C}) = -\frac{1}{2}\sum_{i=1}^{M}\left(D_S\log|\mathbf{\Xi}'_T| + \nu'_T\log|\frac{1}{2}\mathbf{\Omega}'_T| + \log|\mathbf{\Xi}'_R| + \nu'_R\log|\frac{1}{2}\mathbf{\Omega}'_R|\right) + \text{const.} \tag{18}$$

To optimise $\phi_T$ and $\phi_R$, Equation 18 is maximised with respect to features $\mathbf{C}_T$ and $\mathbf{C}_R$. To prevent overfitting, we apply squared Frobenius norms $\|\mathbf{C}_T\|_F^2$ and $\|\mathbf{C}_R\|_F^2$ to Equation 18 as regularisations, the effect of which is discussed in Appendix A.16. The regularised loss is defined as

$$\mathcal{L}_{\text{model}} := -\log p_{\phi_T,\phi_R}(\mathcal{C}) + \lambda_T\|\mathbf{C}_T\|_F^2 + \lambda_R\|\mathbf{C}_R\|_F^2 \tag{19}$$

where $\lambda_T > 0$ and $\lambda_R > 0$ are hyperparameters. Note that $\mathcal{L}_{\text{model}}$ can be directly minimised with gradient descent, without any variational inference and the evaluation of the ELBO term.

### 4.3 Task Representation in Policy Networks

The final component in GLiBRL involves integrating beliefs, which define tasks, into the policy network $\pi_\psi^+(a|s,b)$ to drive decision-making and sample collection. We concisely represent this belief using the deterministic means of the task posteriors:

$$\pi_\psi^+(a|s,b) \approx \pi_\psi^+\left(a|s, f\left(\mathbf{M}'_T, \mathbf{M}'_R\right)\right) \tag{20}$$

where $f : \mathbb{R}^{D_T \times D_S} \times \mathbb{R}^{D_R \times 1} \to \mathbb{R}^{D_P \times 1}$ and $D_P$ is the dimension of the belief representation.

For $\mathbf{x} \in \mathbb{R}^d$ and a fixed $\epsilon > 0$, define the $\epsilon$-normalisation

$$\mathcal{N}_\epsilon(\mathbf{x}) = \mathbf{x} / \left(\|\mathbf{x}\|_2 + \epsilon\right) \tag{21}$$

for normalisation to be well-defined at the empty context. $\mathcal{N}_\epsilon$ agrees with ordinary normalisation up to the factor $\rho(x) = \|x\|_2/(\|x\|_2 + \epsilon) \in [0, 1)$, which is close to one whenever $\|x\|_2 \gg \epsilon$. We focus on the following concatenated representation [4]

$$f_{T,R}\left(\mathbf{M}'_T, \mathbf{M}'_R\right) = \left[\mathcal{N}_\epsilon\Big(\mathrm{tril}\left(\mathbf{M}'_T {\mathbf{M}'_T}^{\mathrm{T}}\right)\Big)^{\mathrm{T}} \quad \mathcal{N}_\epsilon(\mathbf{M}'_R)^{\mathrm{T}}\right]^{\mathrm{T}} \tag{22}$$

Here, $\mathrm{tril}(\cdot)$ retrieves the lower triangle of a matrix and flattens it into a vector. Let samples from two MDPs $\mathcal{M}_1$, $\mathcal{M}_2$ be $\mathcal{C}^{\mathcal{M}_1}$ and $\mathcal{C}^{\mathcal{M}_2}$, where $\mathcal{C}^{\mathcal{M}_1} = (\mathbf{S}_1, \mathbf{A}_1, \mathbf{S}'_1, \mathbf{r}_1)$, $\mathcal{C}^{\mathcal{M}_2} = (\mathbf{S}_2, \mathbf{A}_2, \mathbf{S}'_2, \mathbf{r}_2)$. Given posterior means $\mathbf{M}_T^{1\prime}$, $\mathbf{M}_R^{1\prime}$, $\mathbf{M}_T^{2\prime}$, $\mathbf{M}_R^{2\prime}$ and zero prior means, Equation 14 gives

$$\begin{aligned}
&\left\|f_{T,R}\left(\mathbf{M}_T^{1\prime}, \mathbf{M}_R^{1\prime}\right) - f_{T,R}\left(\mathbf{M}_T^{2\prime}, \mathbf{M}_R^{2\prime}\right)\right\|_2^2 \\
&\qquad = \sum_{k \in \{k_{TT},\, k_R\}} \Big(\rho_k\big(\mathcal{C}^{\mathcal{M}_1}\big)^2 + \rho_k\big(\mathcal{C}^{\mathcal{M}_2}\big)^2 - 2\,\mathrm{sCMS}_k\big(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}\big)\Big)
\end{aligned} \tag{23}$$

where $k_{TT}$ and $k_R$ are kernels induced by the learned basis functions $\phi_T$, $\phi_R$ on pairs of transition samples and on reward samples respectively, $\mathrm{sCMS}_k$ is the signed-measure Cosine Mean Similarity [15] between task datasets under kernel $k$, and $\rho_k(\mathcal{C})^2 = \mathrm{sCMS}_k(\mathcal{C}, \mathcal{C}) \in [0, 1)$.

The unnormalised version $\widetilde{f_{T,R}} = \left[\mathrm{tril}\left(\mathbf{M}'_T {\mathbf{M}'_T}^{\mathrm{T}}\right)^{\mathrm{T}} \quad \mathbf{M}'^{\mathrm{T}}_R\right]^{\mathrm{T}}$ features a cleaner identity:

$$\left\|\widetilde{f_{T,R}}\left(\mathbf{M}_T^{1\prime}, \mathbf{M}_R^{1\prime}\right) - \widetilde{f_{T,R}}\left(\mathbf{M}_T^{2\prime}, \mathbf{M}_R^{2\prime}\right)\right\|_2^2 = \mathrm{sMMD}^2_{k_{TT}}\big(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}\big) + \mathrm{sMMD}^2_{k_R}\big(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}\big) \tag{24}$$

where $\mathrm{sMMD}_k$ is the signed-measure Maximum Mean Discrepancy [13] between the two task datasets under kernel $k$. Detailed definitions and proofs are deferred to Appendix A.10.

### 4.4 Diagonal Noise Inference

The Matrix-Normal-Wishart formulation derived above learns a full covariance and therefore captures cross-dimensions correlations. For cases where only diagonal noise inference suffices, we provide a Matrix-Normal-Gamma formulation. We assign an independent Gamma prior to each output dimension, preserving the exact tractability. Its closed-form update, online execution complexity, ELBO-free marginal log-likelihood and task-representation result are discussed in Appendix A.8.

## 5 Experiments

We evaluate GLiBRL on locomotion tasks from the MuJoCo benchmark [34] and manipulation tasks from the MetaWorld [41, 24] benchmark. For MuJoCo, following standard Meta-RL evaluation protocols, we assess performance on `HalfCheetahDir`, `AntDir`, and `HalfCheetahVel`. In these environments, the agent must adapt its policy from the observed reward to run in the randomly determined forward/backward direction or at a targeted velocity. For MetaWorld, which is considered one of the most challenging meta-RL benchmarks, we focus on its most complex subsets: Meta-Learning 10 (ML10) and Meta-Learning 45 (ML45). ML10 and ML45 consist of 10 and 45 training

[4] We list alternatives and empirical comparisons in Appendix A.9.

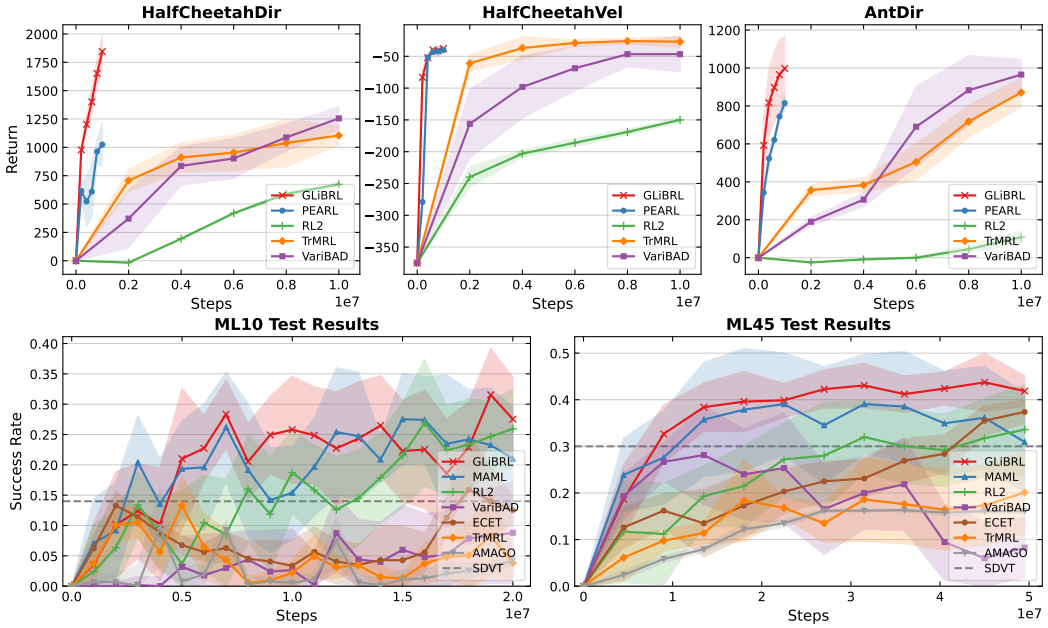



Figure 2: Test-time return and success rate in all benchmarks. Curves report the interquartile mean (IQM) [1], and shaded regions denote $95\%$ confidence intervals . In GLiBRL, $D_T = 16$ and $D_R = 256$ for MetaWorld tasks, and $D_R = 8$ for MuJoCo tasks which do not require inference in transition. In MuJoCo tasks, we run PEARL and GLiBRL with SAC for $1e6$ steps and other baselines with PPO for $1e7$ steps.

tasks, respectively, alongside 5 held-out testing tasks where the agent must adapt to both unseen transition and reward functions. To ensure fair comparison, we use the V2 reward in MetaWorld-V3 [24], matching the reward version of all comparators.

We benchmark GLiBRL against a diverse set of Meta-RL baselines spanning all three main categories: (1) PPG-based methods: **MAML [10]**, (2) black-box methods: **$RL^2$ [8], AMAGO [14], TrMRL [25]** and **ECET [31]**, and (3) task-inference (deep BRL) methods: **PEARL [29], VariBAD [42]** and **SDVT [23]**. Specifically, we compare with $RL^2$, TrMRL, PEARL, VariBAD in the locomotion tasks in MuJoCo, and MAML, $RL^2$, VariBAD, AMAGO, SDVT, TrMRL and ECET in the manipulation tasks in MetaWorld. [5] Full implementation details are provided in Appendix A.11.

Across all experiments, we focus on zero-shot test-time performance, which is critical in Meta-RL settings that require adaptation with minimal data [3]. Here, zero-shot means that the agent receives no task-specific observations and performs no adaptation before evaluation begins. During evaluation, it updates its posterior online using only transitions collected through interaction with the test environment; no gradient updates are performed. For MuJoCo experiments (**Section 5.1**), we follow the setting in PEARL and test GLiBRL with SAC [17] (Algorithm 2) over five random seeds. Both PEARL and GLiBRL are trained with $1e6$ total steps, while others are trained with $1e7$ total steps. For MetaWorld experiments (**Section 5.2**), we follow [24] and run GLiBRL with PPO [30] (Algorithm 3) over ten random seeds. ML10 and ML45 experiments are run for $2e7$ steps and $5e7$ steps, respectively. Beyond quantitative evaluations, we qualitatively inspect the learnt task representations (**Section 5.3**). Finally, using ML10, we conduct ablation studies to empirically validate the necessity of inference on model noises (**Section 5.4**). Each experiment is run on a single L40S GPU.

[5] Methods excluded from these comparisons either lack reported results or demonstrate uncompetitive performance. For example, MAML has return $< 0$ with $1e7$ steps in `HalfCheetahDir` [25]. PEARL has $< 3\%$ success rate in ML10 with $1e8$ steps [41]. We exclude ECET in MuJoCo for it not being open-source and reporting MuJoCo results in meta-steps.

### 5.1 MuJoCo Results

As shown in the top row in Figure 2, GLiBRL combined with SAC demonstrates superior sample efficiency on MuJoCo locomotion tasks. Using only $1e6$ training steps, GLiBRL outperforms or matches all PPO-based baselines ($RL^2$, TrMRL, VariBAD) by up to $1.5\times$, achieving this with merely $10\%$ of their step budget. Crucially, as discussed in Section 3, none of these PPO-based baselines can be trivially integrated with SAC, as they lack permutation invariance critical for off-policy methods [29]. Furthermore, GLiBRL outperforms the strong SAC-based baseline PEARL by up to $1.8\times$, highlighting the advantage of GLiBRL's exact Bayesian task inference.

### 5.2 MetaWorld Results

The bottom row of Figure 2 shows that GLiBRL, when integrated with PPO, achieves the highest observed aggregate success rate among the evaluated methods on both ML10 and ML45. This superior aggregate result, however, does not imply uniform task-level dominance: on ML10, MAML attains a slightly better mean per-task rank over training than GLiBRL ($2.25$ versus $2.50$). On ML45, GLiBRL has the best mean per-task rank ($2.32$). Detailed per-task success rates and ranks are provided in Appendix A.12 and Appendix A.13 [6].

### 5.3 Qualitative Analysis of Task Identifiability

We qualitatively examine the geometry of the learnt task representations on `HalfCheetahVel` using 2D Principal Component Analysis (PCA). As detailed in Appendix A.14, after $5e5$ training steps, GLiBRL produces a visibly ordered representation of the continuous target velocities across five runs. In comparison, TrMRL shows less apparent separation despite a $10\times$ larger training budget. These visualisations qualitatively illustrate that the representation geometry can reflect task similarity, consistent with the exact identity in Section 4.3.

### 5.4 Ablation Studies

We have conducted ablation studies demonstrating (1) different choices of task representations mentioned in Section 4.3 (2) the effect of the regularisation mentioned in Section 4.1, and (3) performing explicit Bayesian inference over model noises mitigates the predictive errors in both transition and reward models. Full experimental results and mathematical derivations are deferred to Appendix A.9 and Appendix A.16.

## 6 Conclusion

We introduced GLiBRL, a novel deep BRL framework that enables fully tractable inference over task parameters and efficient learning of basis functions via exact, ELBO-free marginal likelihood optimisation. By performing Bayesian inference over the model noise, GLiBRL reduces predictive errors. Furthermore, GLiBRL naturally leads to permutation invariance, allowing seamless integration with both on- and off-policy RL algorithms. GLiBRL also establishes an exact kernel identity that relates distances between its learned task representations to signed-measure kernel discrepancies over the task contexts, making the representation geometry analytically traceable. Empirically, GLiBRL achieves the highest observed aggregate performance among the evaluated methods across the MuJoCo locomotion and MetaWorld ML10/45 benchmarks.

Several promising directions for future work arise naturally. Because GLiBRL learns transition and reward models, it is well-suited for integration with model-based RL algorithms, which could further improve sample efficiency. Alternatively, within the model-free paradigm, it is interesting to explore how to incorporate uncertainties within task representations. Empirically, we have observed that directly feeding the covariances in the policy network will introduce high training instability.

[6] SDVT per-task results are excluded as they were not reported at $2e7/5e7$ steps in [23].

## Acknowledgement

We thank anonymous reviewers for their valuable feedback. This work was supported by the ARC Research Hub in Intelligent Robotic Systems for Real-Time Asset Management (IH210100030).

## References

[1] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

[2] Jacob Beck, Matthew Thomas Jackson, Risto Vuorio, and Shimon Whiteson. Hypernetworks in meta-reinforcement learning. In *Conference on Robot Learning*, pages 1478–1487. PMLR, 2023.

[3] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

[4] Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 10–21, 2016.

[5] Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. In *International conference on machine learning*, pages 1078–1086. PMLR, 2018.

[6] Bin Dai, Ziyu Wang, and David Wipf. The usual suspects? reassessing blame for vae posterior collapse. In *International conference on machine learning*, pages 2313–2322. PMLR, 2020.

[7] Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, page 1432, 2016.

[8] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. $Rl^2$: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[9] Michael O'Gordon Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. University of Massachusetts Amherst, 2002.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[11] Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

[12] Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015.

[13] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13(1):723–773, 2012.

[14] Jake Grigsby, Justin Sasek, Samyak Parajuli, Daniel Adebi, Amy Zhang, and Yuke Zhu. Amago-2: Breaking the multi-task barrier in meta-reinforcement learning with transformers. *Advances in Neural Information Processing Systems*, 37:87473–87508, 2024.

[15] Sebastian G Gruber, Pascal Tobias Ziegler, and Florian Buettner. Disentangling mean embeddings for better diagnostics of image generators. *arXiv preprint arXiv:2409.01314*, 2024.

[16] Arthur Guez, David Silver, and Peter Dayan. Scalable and efficient bayes-adaptive reinforcement learning based on monte-carlo tree search. *Journal of Artificial Intelligence Research*, 48:841–883, 2013.

[17] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[18] James Harrison, Apoorva Sharma, Roberto Calandra, and Marco Pavone. Control adaptation via meta-learning dynamics. In *Workshop on Meta-Learning at NeurIPS*, volume 2018, 2018.

[19] James Harrison, Apoorva Sharma, and Marco Pavone. Meta-learning priors for efficient online bayesian regression. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 318–337. Springer, 2018.

[20] James Harrison, John Willes, and Jasper Snoek. Variational bayesian last layers. *arXiv preprint arXiv:2404.11599*, 2024.

[21] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[22] Taylor W Killian, Samuel Daulton, George Konidaris, and Finale Doshi-Velez. Robust and efficient transfer learning with hidden parameter markov decision processes. *Advances in neural information processing systems*, 30, 2017.

[23] Suyoung Lee, Myungsik Cho, and Youngchul Sung. Parameterizing non-parametric meta-reinforcement learning tasks via subtask decomposition. *Advances in Neural Information Processing Systems*, 36:43356–43383, 2023.

[24] Reginald McLean, Evangelos Chatzaroulas, Luc McCutcheon, Frank Röder, Tianhe Yu, Zhanpeng He, KR Zentner, Ryan Julian, JK Terry, Isaac Woungang, et al. Meta-world+: An improved, standardized, rl benchmark. *arXiv preprint arXiv:2505.11289*, 2025.

[25] Luckeciano C Melo. Transformers are meta-reinforcement learners. In *international conference on machine learning*, pages 15340–15359. PMLR, 2022.

[26] Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems*, 26, 2013.

[27] Christian Perez, Felipe Petroski Such, and Theofanis Karaletsos. Generalized hidden parameter mdps: Transferable model-based rl in a handful of trials. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5403–5411, 2020.

[28] Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete bayesian reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 697–704, 2006.

[29] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

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

[31] Gresa Shala, André Biedenkapp, Pierre Krack, Florian Walter, and Josif Grabocka. Efficient cross-episode meta-rl. In *The Thirteenth International Conference on Learning Representations*, 2025.

[32] Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12(7), 2011.

[33] Malcolm Strens. A bayesian framework for reinforcement learning. In *ICML*, volume 2000, pages 943–950, 2000.

[34] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[35] Nikolaos Tziortziotis, Christos Dimitrakakis, and Konstantinos Blekas. Linear bayesian reinforcement learning. In *IJCAI*, pages 1721–1728, 2013.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[37] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[38] Jiachen Yang, Brenden Petersen, Hongyuan Zha, and Daniel Faissol. Single episode policy transfer in reinforcement learning. *arXiv preprint arXiv:1910.07719*, 2019.

[39] Jiayu Yao, Taylor Killian, George Konidaris, and Finale Doshi-Velez. Direct policy transfer via hidden parameter markov decision processes. In *LLARLA Workshop, FAIM*, volume 2018, 2018.

[40] Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

[41] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Avnish Narayan, Hayden Shively, Adithya Bellathur, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning, 2021.

[42] Luisa Zintgraf, Sebastian Schulze, Cong Lu, Leo Feng, Maximilian Igl, Kyriacos Shiarlis, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: Variational bayes-adaptive deep rl via meta-learning. *Journal of Machine Learning Research*, 22(289):1–39, 2021.

# A Appendix

## A.1 Table of Important Notations

| Symbol | Meaning |
|---|---|
| *MDPs and BAMDPs* | |
| $\mathcal{M} = (S, A, R, T, \gamma)$ | MDP with state space $S$, action space $A$, reward $R$, transition $T$, discount $\gamma$. |
| $S^+ = S \times \mathcal{B}_T \times \mathcal{B}_R$ | BAMDP hyper-state space; $B_T, B_R$ are the belief spaces of $\theta_T, \theta_R$. |
| $b_t = p(\theta_T, \theta_R)$ | Belief at time $t$ over task parameters. |
| $\tau$ | Belief-update function, $b_{t+1} = \tau(b_t, s_t, a_t, s_{t+1}, r_{t+1})$. |
| $\pi^+_\psi(a \mid s, b)$ | BAMDP (BRL) policy parameterised by $\psi$. |
| $H^+, \mathcal{J}^+(\pi^+)$ | BAMDP horizon and expected return. |
| *Datasets and dimensions* | |
| $c_t = (s_t, a_t, s_{t+1}, r_{t+1})$ | Single sample at timestep $t$. |
| $\mathcal{C}^{\mathcal{M}} = \{c_t\}_{t=1}^N$ | $N$ samples collected within MDP $\mathcal{M}$. |
| $\mathcal{C} = \bigcup_{i=1}^M \mathcal{C}^{\mathcal{M}_i}$ | Joint dataset over a batch of $M$ MDPs. |
| $\mathcal{C}^{\mathcal{M}_i} = (\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \mathbf{r}_i)$ | Samples in compact matrix forms. |
| $D_S,\ D_A$ | State and action dimensions. |
| $D_T,\ D_R$ | Transition and reward task dimensions. |
| $D_P$ | Task-representation dimension. |
| *Task parameters* | |
| $\theta_T = (T_\mu, T_\sigma)$ | Transition task parameter: mean $T_\mu$, noise covariance $T_\sigma$. |
| $\theta_R = (R_\mu, R_\sigma)$ | Reward task parameter: mean $R_\mu$, noise covariance $R_\sigma$. |
| *Learnable basis functions* | |
| $\phi_T,\ \phi_R$ | Neural networks acting as basis functions for transition and reward. |
| $\mathbf{C}_T = \phi_T(S_i, A_i)$ | Transition features (design matrix). |
| $\mathbf{C}_R = \phi_R(S_i, A_i, S'_i)$ | Reward features (design matrix). |
| $\zeta$ | Optional network parameterising the prior $p_\zeta(\theta_T, \theta_R)$. |
| *Distributions* | |
| $\mathcal{MN}(\mathbf{W} \mid \mathbf{X}, \mathbf{Y}, \mathbf{Z})$ | Matrix-normal: mean $\mathbf{X}$, row covariance $\mathbf{Y}$, column covariance $\mathbf{Z}$. |
| $\mathcal{W}(\mathbf{W} \mid \mathbf{X}, \nu)$ | Wishart: scale matrix $\mathbf{X}$, degrees of freedom $\nu$. |
| $\mathrm{Gam}(\cdot \mid a, b)$ | Gamma: shape $a$ and rate $b$. |
| *Prior/posterior parameters* | |
| $\mathbf{M}_T / \mathbf{M}'_T$ | Prior/posterior mean of $T_\mu$. |
| $\mathbf{\Xi}_T^{-1} / {\mathbf{\Xi}'_T}^{-1}$ | Prior/posterior row covariance of $T_\mu$ ($\mathbf{\Xi}_T$ is the row precision). |
| $\mathbf{\Omega}_T / \mathbf{\Omega}'_T,\ \nu_T / \nu'_T$ | Prior/posterior Wishart scale and dof on $T_\sigma^{-1}$. |
| $(a_{T,d}, b_{T,d})$ / $(a'_{T,d}, b'_{T,d})$ | Prior/posterior Gamma shape and rate for transition coordinate $d$ under diagonal noise. |
| | Parameters for $\theta_R$ follow analogously. |
| *Task representations and kernel correspondence* | |
| $\mathcal{N}_\epsilon$ | $\epsilon$-normalisation $x/(\|x\|_2 + \epsilon)$ (Equation 21). |
| $k_T,\ k_{TT},\ k_R$ | Kernels induced by the learned basis functions (Appendix A.10). |
| $\mathrm{sCMS}_k(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2})$ | Signed-measure Cosine Mean Similarity under kernel $k$ (Appendix A.10). |
| $\mathrm{sMMD}^2_k(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2})$ | Squared signed-measure Maximum Mean Discrepancy under kernel $k$. |

### A.2 GLiBRL Algorithm

**Algorithm 1:** GLiBRL

For $\mathcal{C} = (\mathbf{S}, \mathbf{A}, \mathbf{S}', \mathbf{r})$, define
$\text{Post}(\mathcal{C}) := p_{\phi_T}(\theta_T \mid \phi_T(\mathbf{S}, \mathbf{A}), \mathbf{S}')\, p_{\phi_R}(\theta_R \mid \phi_R(\mathbf{S}, \mathbf{A}, \mathbf{S}'), \mathbf{r})$;
Initialise: policy $\pi^+_\psi$, horizon $H$, $\phi_T, \phi_R$;
**while** *Training* **do**
  Sample $K$ MDPs $\{\mathcal{M}_i\}_{i=1}^K$; Initialise: Joint data $\mathcal{C} = \{\}$;
  `// collecting samples`
  **for** $i \in \{1, 2, \cdots, K\}$ **do**
    Initialise: samples in $\mathcal{M}_i$, $\mathcal{C}^{\mathcal{M}_i} = \{\}$, state $s$;
    **for** $t < H$ **do**
      $b \leftarrow \text{Post}(\mathcal{C}^{\mathcal{M}_i})$; $a \sim \pi^+_\psi(a|s, b)$;
      Execute $a$ from $s$ in $\mathcal{M}_i$ to get $s', r$;
      $\mathcal{C}^{\mathcal{M}_i} \leftarrow \mathcal{C}^{\mathcal{M}_i} \cup \{s, a, s', r\}$;
      $s \leftarrow s'$;
    $\mathcal{C} = \mathcal{C} \cup \mathcal{C}^{\mathcal{M}_i}$
  `// learning policy, transition and reward models`
  **while** *Learning* **do**
    Sample $\mathcal{D} \subseteq \mathcal{C}$;
    $\psi \leftarrow \psi - \nabla_\psi \mathcal{L}_{\text{policy}}(\mathcal{D})$;
    $\phi_T \leftarrow \phi_T - \nabla_{\phi_T} \mathcal{L}_{\text{model}}(\mathcal{D})$;
    $\phi_R \leftarrow \phi_R - \nabla_{\phi_R} \mathcal{L}_{\text{model}}(\mathcal{D})$;

### A.3 GLiBRL-SAC

**Algorithm 2:** GLiBRL with SAC

**while** *Training* **do**
  `// collecting samples following Algorithm 1`
  **while** *Learning* **do**
    Sample $\mathcal{D} \subseteq \mathcal{C}$ of **variable-length** $N \in [1, \text{Horizon}]$ ;
    $b \leftarrow \text{Post}(\mathcal{D})$;
    $\psi \leftarrow \psi - \nabla_\psi(\mathcal{L}_{\text{actor}}(\mathcal{D}, b) + \mathcal{L}_{\text{critic}}(\mathcal{D}, b) + \mathcal{L}_\alpha)$;
    Polyak averaging for target Q-networks within $\psi$;
    `// Model updates following Algorithm 1`

### A.4 GLiBRL-PPO

**Algorithm 3:** GLiBRL with PPO

**while** *Training* **do**
  `// collecting samples following Algorithm 1`
  Compute advantages $\hat{A}$ and return targets using $\mathcal{C}$;
  **while** *Learning* **do**
    Sample $\mathcal{D} \subseteq \mathcal{C}$ of **fixed length** $N$;
    $\{b_i\}_{i=1}^N \leftarrow \{\text{Post}(\mathcal{D}[1:i])\}_{i=1}^N$;
    $\psi \leftarrow \psi - \nabla_\psi \mathcal{L}_{\text{actor}}\left(\mathcal{D}, \hat{A}, \{b_i\}_{i=1}^N\right)$;
    `// Model updates following Algorithm 1`

### A.5 Evidence Lower Bound

The evidence lower bound in Equation 3 is derived as follows

$$\begin{aligned}\log p_{\phi_T,\phi_R}(\mathcal{C}) &= \sum_{i=1}^{M} \log \iint p(\mathcal{C}^{\mathcal{M}_i}, \theta_T, \theta_R) \frac{q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})}{q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})} d\theta_T d\theta_R \\ &\geq \sum_{i=1}^{M} \mathbb{E}_q \left[\log \frac{p(\theta_T, \theta_R)}{q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i})} + \log p(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R)\right] \\ &= \sum_{i=1}^{M} \mathbb{E}_q \left[\log p(\mathcal{C}^{\mathcal{M}_i}|\theta_T, \theta_R)\right] - D_{KL}\left(q(\theta_T, \theta_R|\mathcal{C}^{\mathcal{M}_i}) \| p(\theta_T, \theta_R)\right)\end{aligned}$$

### A.6 Normal-Wishart-Normal Conjugacy

Given Equation 5

$$p(\theta_T) = \mathcal{MN}(T_\mu|\mathbf{M}_T, \mathbf{\Xi}_T^{-1}, T_\sigma) \cdot \mathcal{W}(T_\sigma^{-1}|\mathbf{\Omega}_T^{-1}, \nu_T)$$

and Equation 8

$$p_{\phi_T}(\mathbf{S}_i'|\mathbf{S}_i, \mathbf{A}_i, \theta_T) = \mathcal{MN}\left(\mathbf{S}_i'|\mathbf{C}_T T_\mu, \mathbf{I}_N, T_\sigma\right) \tag{8}$$

We prove Equation 12

$$p_{\phi_T}(\theta_T|\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}_i') = \mathcal{MN}(T_\mu|\mathbf{M}_T', \mathbf{\Xi}_T'^{-1}, T_\sigma) \cdot \mathcal{W}(T_\sigma^{-1}|\mathbf{\Omega}_T'^{-1}, \nu_T') \tag{12}$$

where the posterior parameters are defined in Equation 14.

**Proof:**

The density of the prior distribution $p(\theta_T)$ is

$$\begin{aligned}p(\theta_T) &= \frac{|\mathbf{\Xi}_T|^{D_S/2}}{\sqrt{(2\pi)^{D_T D_S}}|T_\sigma|^{D_T/2}} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(T_\sigma^{-1}(T_\mu - \mathbf{M}_T)^{\mathrm{T}}\mathbf{\Xi}_T(T_\mu - \mathbf{M}_T)\right)\right] \\ &\cdot \sqrt{\frac{|\mathbf{\Omega}_T|^{\nu_T}}{2^{\nu_T D_S}}} \frac{|T_\sigma|^{(1+D_S-\nu_T)/2}}{\Gamma_{D_S}(\nu_T/2)} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(\mathbf{\Omega}_T T_\sigma^{-1}\right)\right] \qquad (25) \\ &\propto |T_\sigma|^{(1+D_S-\nu_T-D_T)/2} \cdot \exp\left\{-\frac{1}{2}\operatorname{Tr}\Big[T_\sigma^{-1}\Big[(T_\mu - \mathbf{M}_T)^{\mathrm{T}}\mathbf{\Xi}_T(T_\mu - \mathbf{M}_T) + \mathbf{\Omega}_T\Big]\Big]\right\} \qquad (26)\end{aligned}$$

where from Equation 25 to Equation 26 we treat multiplicative parameters irrelevant to $\theta_T$ as constants. The joint density of $p_{\phi_T}(\theta_T, \mathbf{S}_i'|\mathbf{S}_i, \mathbf{A}_i) = p(\theta_T) \cdot p_{\phi_T}(\mathbf{S}_i'|\mathbf{S}_i, \mathbf{A}_i, \theta_T)$ is hence

$$\begin{aligned}&\frac{|\mathbf{\Xi}_T|^{D_S/2}}{\sqrt{(2\pi)^{D_T D_S}}|T_\sigma|^{D_T/2}} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(T_\sigma^{-1}(T_\mu - \mathbf{M}_T)^{\mathrm{T}}\mathbf{\Xi}_T(T_\mu - \mathbf{M}_T)\right)\right] \\ &\cdot \sqrt{\frac{|\mathbf{\Omega}_T|^{\nu_T}}{2^{\nu_T D_S}}} \frac{|T_\sigma|^{(1+D_S-\nu_T)/2}}{\Gamma_{D_S}(\nu_T/2)} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(\mathbf{\Omega}_T T_\sigma^{-1}\right)\right] \qquad (27) \\ &\cdot \frac{1}{\sqrt{(2\pi)^{N D_S}}|T_\sigma|^{N/2}} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(T_\sigma^{-1}(\mathbf{S}_i' - \mathbf{C}_T T_\mu)^{\mathrm{T}}(\mathbf{S}_i' - \mathbf{C}_T T_\mu)\right)\right] \\ &\propto |T_\sigma|^{(1+D_S-\nu_T-N-D_T)/2} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(T_\sigma^{-1}(T_\mu - \mathbf{M}_T)^{\mathrm{T}}\mathbf{\Xi}_T(T_\mu - \mathbf{M}_T)\right)\right] \\ &\qquad \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(T_\sigma^{-1}\left((\mathbf{S}_i' - \mathbf{C}_T T_\mu)^{\mathrm{T}}(\mathbf{S}_i' - \mathbf{C}_T T_\mu) + \mathbf{\Omega}_T\right)\right)\right] \qquad (28)\end{aligned}$$

Matching the second-order then the first-order term with respect to $T_\mu$, we can rewrite

$$28 = |T_\sigma|^{(1+D_S-\nu'_T-D_T)/2} \cdot \exp\left\{-\frac{1}{2}\operatorname{Tr}\Big[T_\sigma^{-1}\Big[(T_\mu - \mathbf{M}'_T)^{\mathrm{T}}\mathbf{\Xi}'_T(T_\mu - \mathbf{M}'_T) + \mathbf{\Omega}'_T\Big]\Big]\right\} \quad (29)$$

We can find Equation 26 and Equation 29 match exactly, indicating the Normal-Wishart-Normal conjugacy. Note, we just use the posterior update of $p(\theta_T)$ as an example. The exact same proof applies to the posterior update of $p(\theta_R)$ as well. Such conjugacy allows exact posterior update and marginal likelihood, enabling efficient learning.

### A.7 Marginal Log-likelihood of Normal-Wishart-Normal

We prove Equation 18 has the following closed form

$$\log p_{\phi_T,\phi_R}(\mathcal{C}) = -\frac{1}{2}\sum_{i=1}^{M}\left(D_S \log|\mathbf{\Xi}'_T| + \nu'_T \log|\frac{1}{2}\mathbf{\Omega}'_T| + \log|\mathbf{\Xi}'_R| + \nu'_R \log|\frac{1}{2}\mathbf{\Omega}'_R|\right) + \text{const.} \quad (18)$$

**Proof:**

Consider

$$p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i) = \frac{p_{\phi_T}(\theta_T,\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i)}{p_{\phi_T}(\theta_T|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i)} \quad (30)$$

From Appendix A.6, we know the numerator is

$$\frac{|\mathbf{\Xi}_T|^{D_S/2}|\mathbf{\Omega}_T|^{\nu_T/2}}{2^{\nu_T D_S/2}\cdot(2\pi)^{D_S(D_T+N)/2}\cdot\Gamma_{D_S}(\nu_T/2)} \cdot \text{Equation 29} \quad (31)$$

The denominator is

$$\frac{|\mathbf{\Xi}'_T|^{D_S/2}|\mathbf{\Omega}'_T|^{\nu'_T/2}}{2^{\nu'_T D_S/2}\cdot(2\pi)^{D_S D_T/2}\cdot\Gamma_{D_S}(\nu'_T/2)} \cdot \text{Equation 29} \quad (32)$$

Hence,

$$p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i) = \frac{1}{(2\pi)^{D_S N/2}}\cdot\frac{|\mathbf{\Xi}_T|^{D_S/2}|\frac{1}{2}\mathbf{\Omega}_T|^{\nu_T/2}\cdot\Gamma_{D_S}(\nu'_T/2)}{|\mathbf{\Xi}'_T|^{D_S/2}|\frac{1}{2}\mathbf{\Omega}'_T|^{\nu'_T/2}\cdot\Gamma_{D_S}(\nu_T/2)} \quad (33)$$

Similarly,

$$p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i) = \frac{1}{(2\pi)^{N/2}}\cdot\frac{|\mathbf{\Xi}_R|^{1/2}|\frac{1}{2}\mathbf{\Omega}_R|^{\nu_R/2}\cdot\Gamma(\nu'_R/2)}{|\mathbf{\Xi}'_R|^{1/2}|\frac{1}{2}\mathbf{\Omega}'_R|^{\nu'_R/2}\cdot\Gamma(\nu_R/2)} \quad (34)$$

Note that

$$p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}) = p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i)p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i)p(\mathbf{S}_i,\mathbf{A}_i) \quad (35)$$

By taking the logarithm, and note the independence of $p(\mathbf{S}_i,\mathbf{A}_i)$ with respect to $\phi_T,\phi_R$,

$$\log p_{\phi_T,\phi_R}(\mathcal{C}^{\mathcal{M}_i}) = \log p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i) + \log p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i) + \text{const.} \quad (36)$$

Sum the above equation on both sides with respect to $i$, then we have Equation 18.

### A.8 Diagonal Noise Inference with Matrix-Normal-Gamma

The Matrix-Normal-Wishart formulation in Section 4.1 learns a full covariance. For simplicity, a diagonal formulation can be considered as follows

$$p^{\text{diag}}(\theta_T) = \mathcal{MN}(T_\mu \mid \mathbf{M}_T, \boldsymbol{\Xi}_T^{-1}, T_\sigma) \prod_{d=1}^{D_S} \text{Gam}\left([T_\sigma^{-1}]_{dd} \mid a_{T,d}, b_{T,d}\right) \tag{37}$$

$$p^{\text{diag}}(\theta_R) = \mathcal{MN}(R_\mu \mid \mathbf{M}_R, \boldsymbol{\Xi}_R^{-1}, R_\sigma)\, \text{Gam}\left(R_\sigma^{-1} \mid a_R, b_R\right) \tag{38}$$

where $\text{Gam}(X \mid a, b)$ defines a Gamma distribution with random variable $X$, shape $a$ and rate $b$, and $T_\sigma = \text{diag}\left([T_\sigma]_{11}, \ldots, [T_\sigma]_{D_S,D_S}\right)$ is the diagonal transition covariance.

#### Closed-form posterior update and conjugacy

The posterior distributions remain Matrix-Normal-Gamma:

$$p^{\text{diag}}_{\phi_T}(\theta_T \mid \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \mathcal{MN}(T_\mu \mid \mathbf{M}'_T, {\boldsymbol{\Xi}'_T}^{-1}, T_\sigma) \prod_{d=1}^{D_S} \text{Gam}\left([T_\sigma^{-1}]_{dd} \mid a'_{T,d}, b'_{T,d}\right) \tag{39}$$

$$p^{\text{diag}}_{\phi_R}(\theta_R \mid \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \mathbf{r}_i) = \mathcal{MN}(R_\mu \mid \mathbf{M}'_R, {\boldsymbol{\Xi}'_R}^{-1}, R_\sigma)\, \text{Gam}\left(R_\sigma^{-1} \mid a'_R, b'_R\right) \tag{40}$$

For the transition model, the posterior parameters are

$$\mathbf{M}'_T = {\boldsymbol{\Xi}'_T}^{-1}\left(\mathbf{C}_T^{\mathrm{T}}\mathbf{S}' + \boldsymbol{\Xi}_T\mathbf{M}_T\right) \qquad \boldsymbol{\Xi}'_T = \mathbf{C}_T^{\mathrm{T}}\mathbf{C}_T + \boldsymbol{\Xi}_T \tag{41}$$

$$a'_{T,d} = a_{T,d} + \frac{N}{2} \qquad b'_{T,d} = b_{T,d} + \frac{1}{2}\left[\mathbf{S}'^{\mathrm{T}}\mathbf{S}' + \mathbf{M}_T^{\mathrm{T}}\boldsymbol{\Xi}_T\mathbf{M}_T - {\mathbf{M}'_T}^{\mathrm{T}}\boldsymbol{\Xi}'_T\mathbf{M}'_T\right]_{dd} \tag{42}$$

For the reward model, the posterior parameters are

$$\mathbf{M}'_R = {\boldsymbol{\Xi}'_R}^{-1}\left(\mathbf{C}_R^{\mathrm{T}}\mathbf{r} + \boldsymbol{\Xi}_R\mathbf{M}_R\right) \qquad \boldsymbol{\Xi}'_R = \mathbf{C}_R^{\mathrm{T}}\mathbf{C}_R + \boldsymbol{\Xi}_R \tag{43}$$

$$a'_R = a_R + \frac{N}{2} \qquad b'_R = b_R + \frac{1}{2}\left(\mathbf{r}^{\mathrm{T}}\mathbf{r} + \mathbf{M}_R^{\mathrm{T}}\boldsymbol{\Xi}_R\mathbf{M}_R - {\mathbf{M}'_R}^{\mathrm{T}}\boldsymbol{\Xi}'_R\mathbf{M}'_R\right) \tag{44}$$

The posterior means and row precisions are identical to those in Equation 14. Only the full Wishart update is replaced by independent Gamma updates.

#### Online execution and time complexity

The posterior update of Matrix-Normal-Gamma only differs with that of Matrix-Normal-Wishart in the noise update. Therefore, during online execution, the time complexity of the transition model update is $\mathcal{O}\left(D_T \cdot \max(D_T, D_S)\right)$. The quadratic Wishart update term now disappears. The reward model update remains $\mathcal{O}(D_R^2)$.

#### ELBO-free marginal log-likelihood

Follow the proof in Appendix A.7, the marginal likelihood can be derived as

$$\log p^{\text{diag}}_{\phi_T,\phi_R}(\mathcal{C}) = -\sum_{i=1}^{M}\left[\frac{D_S}{2}\log|\boldsymbol{\Xi}'_T| + \sum_{d=1}^{D_S} a'_{T,d}\log b'_{T,d} + \frac{1}{2}\log|\boldsymbol{\Xi}'_R| + a'_R\log b'_R\right] + \text{const.} \tag{45}$$

#### Task representation

The posterior means $\mathbf{M}'_T$ and $\mathbf{M}'_R$ in Equation 41 and Equation 43 are identical to those under the Wishart model. Consequently, the task representation $f_{T,R}$ in Equation 22 is unchanged. The correspondence identities in Equation 23 and Equation 24, as well as the alternative representations in Appendix A.9, therefore continue to hold without modification under diagonal noise.

### A.9 Choices of Belief Representations

We list several candidate $f$s:

$$
\begin{aligned}
f_1\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[\mathrm{vec}\left(\mathbf{M}'_T\right)^{\mathsf{T}} \ \mathbf{M}'_R{}^{\mathsf{T}}\right]^{\mathsf{T}} \\
f_2\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[\mathrm{tril}\left(\mathbf{M}'_T\mathbf{M}'_T{}^{\mathsf{T}}\right)^{\mathsf{T}} \ \mathbf{M}'_R{}^{\mathsf{T}}\right]^{\mathsf{T}} \\
f_3\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[\mathrm{tril}\left(\mathbf{M}'_T{}^{\mathsf{T}}\mathbf{M}'_T\right)^{\mathsf{T}} \ \mathbf{M}'_R{}^{\mathsf{T}}\right]^{\mathsf{T}} \\
f_4\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[\mathcal{N}_\epsilon(\mathrm{vec}\left(\mathbf{M}'_T\right))^{\mathsf{T}} \ \mathcal{N}_\epsilon(\mathbf{M}'_R)^{\mathsf{T}}\right]^{\mathsf{T}} \\
f_5\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[\mathcal{N}_\epsilon\left(\mathrm{tril}\left(\mathbf{M}'_T\mathbf{M}'_T{}^{\mathsf{T}}\right)\right)^{\mathsf{T}} \ \mathcal{N}_\epsilon(\mathbf{M}'_R)^{\mathsf{T}}\right]^{\mathsf{T}} \\
f_6\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[\mathcal{N}_\epsilon\left(\mathrm{tril}\left(\mathbf{M}'_T{}^{\mathsf{T}}\mathbf{M}'_T\right)\right)^{\mathsf{T}} \ \mathcal{N}_\epsilon(\mathbf{M}'_R)^{\mathsf{T}}\right]^{\mathsf{T}} \\
f_7\left(\mathbf{M}'_T, \mathbf{M}'_R\right) &= \left[g_T\left(\mathbf{M}'_T\right)^{\mathsf{T}} \ g_R\left(\mathbf{M}'_R\right)\right]^{\mathsf{T}}
\end{aligned}
$$

where $\mathrm{tril}(\cdot)$ retrieves the flattened lower triangle of matrices and $g_T(\cdot), g_R(\cdot)$ are neural networks. $f_1$ directly flattens and concatenates the task means, resulting in $D_T \times D_S + D_R$ dimensional representations. $f_2$ and $f_3$ are used for scenarios with large $D_S$ and $D_T$, giving $D_T(D_T+1)/2+D_R$ and $D_S(D_S+1)/2+D_R$ dimensional representations respectively. $f_4$, $f_5$, and $f_6$ are their $\epsilon$-normalised counterparts (Equation 21). $f_7$ suggests learnable belief representations, trained by backpropagating the reinforcement-learning loss. While we focus on $f_5$, i.e. $f_{T,R}$ (Equation 22), we compare these design choices empirically and report the success rates of $f_1$, $f_4$, $f_5$, and $f_7$ on MetaWorld ML10 over 10 seeds in Figure 3.

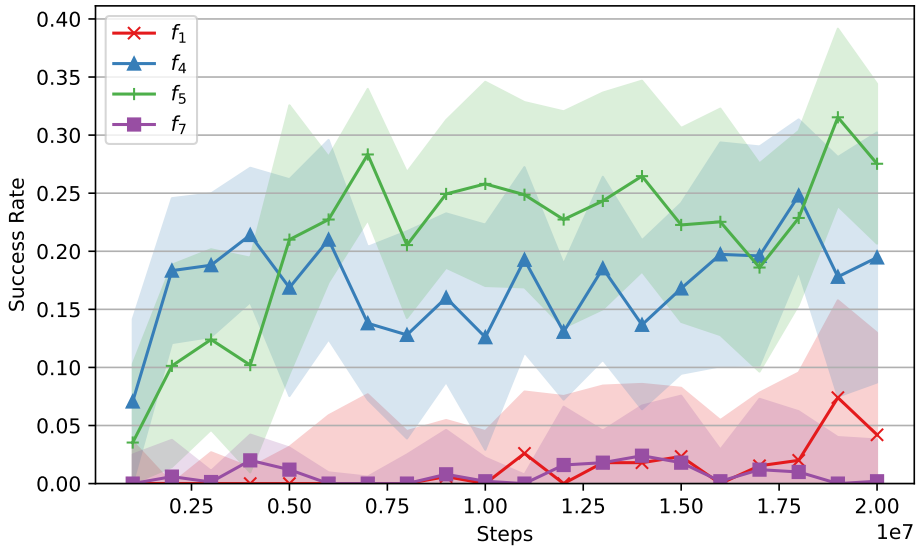



Figure 3: Success rate of different task representation functions in ML10.

As shown in Figure 3, $f_5$, i.e., $f_{T,R}$ (Equation 22), performs best overall. The low success rates of $f_1$ and $f_7$ show that neither raw posterior nor an unconstrained learned mapping suffice in this setting. Comparing $f_1$ and $f_4$ indicates that normalisation provides most of the gain. It substantially stabilises training and, consistently with Equation 23, bounds the squared distance between task representations in $[0, 8)$. Comparing $f_4$ and $f_5$ suggests that the quadratic transition feature further improves success, partly by reducing the input dimension seen by the policy.

### A.10 Kernel Correspondence of Task Representations

Let $\mathcal{C}^{\mathcal{M}_i} = (\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \mathbf{r}_i)$ contain $N_i$ samples $\mathbf{c}_n^i = (s_n^i, a_n^i, {s_n^i}', r_n^i)$. We take the shared prior means to be zero. The shared row precisions can be absorbed by the reparameterisations $\widetilde{\mathbf{C}_T} = \mathbf{C}_T \boldsymbol{\Xi}_T^{-1/2}$, $\widetilde{\mathbf{M}_T} = \boldsymbol{\Xi}_T^{1/2}\mathbf{M}_T$ and $\widetilde{\mathbf{C}_R} = \mathbf{C}_R \boldsymbol{\Xi}_R^{-1/2}$, $\widetilde{\mathbf{M}_R} = \boldsymbol{\Xi}_R^{1/2}\mathbf{M}_R$, which preserve $\mathbf{C}_T\mathbf{M}_T$ and $\mathbf{C}_R\mathbf{M}_R$ and give identity row precisions. We may therefore set $\boldsymbol{\Xi}_T = \boldsymbol{\Xi}_R = \mathbf{I}$ without loss of generality and drop the tildes. We also suppress the task index until comparing tasks. The identities for $f_1, f_2, f_4$, and $f_5$ in Appendix A.9 use three blocks: $\mathbf{M}'_T$, $\mathbf{M}'_R$, and $\mathrm{tril}(\mathbf{M}'_T{\mathbf{M}'_T}^{\mathrm{T}})$. We first express these blocks as kernel embeddings and discuss $f_3$ and $f_6$ separately.

**Linear transition mean.** From Equation 14 and the push-through identity,

$$\mathbf{M}'_T = \mathbf{C}_T^{\mathrm{T}}\mathbf{U}_T = \sum_{n=1}^{N} \boldsymbol{\psi}_T(\mathbf{c}_n)\mathbf{u}_{T,n}^{\mathrm{T}}, \qquad \mathbf{U}_T = (\mathbf{I} + \mathbf{C}_T\mathbf{C}_T^{\mathrm{T}})^{-1}\mathbf{S}', \tag{46}$$

where $\mathbf{u}_{T,n}^{\mathrm{T}}$ is the $n$-th row of $\mathbf{U}_T$ and $\boldsymbol{\psi}_T(\mathbf{c}_n) = \phi_T(s_n, a_n)^{\mathrm{T}}$. Define $k_T(\mathbf{c}, \mathbf{d}) = \boldsymbol{\psi}_T(\mathbf{c})^{\mathrm{T}}\boldsymbol{\psi}_T(\mathbf{d})$ and, for each output dimension $\ell$,

$$\eta_{T,\ell} = \sum_{n=1}^{N} u_{T,n,\ell}\, \delta_{\mathbf{c}_n}. \tag{47}$$

Identifying the RKHS $\mathcal{H}_{k_T}$ with its coefficient space gives $\mu_{k_T}(\eta_{T,\ell}) = \sum_n u_{T,n,\ell}\boldsymbol{\psi}_T(\mathbf{c}_n)$. Thus, in the direct sum $\mathcal{H}_T = \bigoplus_{\ell=1}^{D_S} \mathcal{H}_{k_T}$,

$$\mu_T := \bigoplus_{\ell=1}^{D_S} \mu_{k_T}(\eta_{T,\ell}) \equiv \mathbf{M}'_T, \tag{48}$$

where the direct-sum norm is the Frobenius norm of $\mathbf{M}'_T$.

**Reward mean.** The scalar-output analogue uses $\mathbf{U}_R = (\mathbf{I} + \mathbf{C}_R\mathbf{C}_R^{\mathrm{T}})^{-1}\mathbf{r}$, its $n$-th entry $u_{R,n}$, and $\boldsymbol{\psi}_R(\mathbf{c}_n) = \phi_R(s_n, a_n, s'_n)^{\mathrm{T}}$. With

$$k_R(\mathbf{c}, \mathbf{d}) = \boldsymbol{\psi}_R(\mathbf{c})^{\mathrm{T}}\boldsymbol{\psi}_R(\mathbf{d}), \qquad \eta_R = \sum_{n=1}^{N} u_{R,n}\delta_{\mathbf{c}_n}, \tag{49}$$

the same argument gives $\mu_{k_R}(\eta_R) = \sum_n u_{R,n}\boldsymbol{\psi}_R(\mathbf{c}_n) = \mathbf{M}'_R$.

**Quadratic transition block.** Set $\beta_{mn} = \mathbf{u}_{T,m}^{\mathrm{T}}\mathbf{u}_{T,n}$ and $\mathbf{h}_T(\mathbf{c}_m, \mathbf{c}_n) = \mathrm{tril}(\boldsymbol{\psi}_T(\mathbf{c}_m)\boldsymbol{\psi}_T(\mathbf{c}_n)^{\mathrm{T}})$. The pair kernel and signed measure

$$k_{TT}((\mathbf{c}, \mathbf{d}), (\mathbf{c}', \mathbf{d}')) = \mathbf{h}_T(\mathbf{c}, \mathbf{d})^{\mathrm{T}}\mathbf{h}_T(\mathbf{c}', \mathbf{d}'), \qquad \eta_{TT} = \sum_{m,n=1}^{N} \beta_{mn}\delta_{(\mathbf{c}_m, \mathbf{c}_n)} \tag{50}$$

therefore satisfy

$$\mu_{k_{TT}}(\eta_{TT}) = \sum_{m,n=1}^{N} \beta_{mn}\mathbf{h}_T(\mathbf{c}_m, \mathbf{c}_n) = \mathrm{tril}(\mathbf{M}'_T{\mathbf{M}'_T}^{\mathrm{T}}). \tag{51}$$

Thus $k_T$ and $k_R$ act on individual samples, whereas $k_{TT}$ acts on ordered sample pairs; all three are positive semidefinite by construction.

For task $i$, write $\eta_{T,\ell}^i$, $\eta_R^i$, and $\eta_{TT}^i$ for these scalar measures and define

$$\mu_{k_T}^i := \bigoplus_{\ell=1}^{D_S} \mu_{k_T}(\eta_{T,\ell}^i) \equiv {\mathbf{M}_T^i}', \tag{52}$$

$$\mu_{k_R}^i := \mu_{k_R}(\eta_R^i) = {\mathbf{M}_R^i}', \qquad \mu_{k_{TT}}^i := \mu_{k_{TT}}(\eta_{TT}^i) = \mathrm{tril}({\mathbf{M}_T^i}'{\mathbf{M}_T^i}'^{\mathrm{T}}). \tag{53}$$

For any scalar signed measures, the standard bilinear identity

$$\langle \mu_k(\eta^1), \mu_k(\eta^2) \rangle = \sum_{a,b} w_a^1 w_b^2 \, k(\mathbf{z}_a^1, \mathbf{z}_b^2), \quad \eta^i = \sum_a w_a^i \delta_{\mathbf{z}_a^i}, \tag{54}$$

applies directly to $\eta_R$ and $\eta_{TT}$ and is summed over $\ell$ for the transition direct sum. Hence all inner products and norms below can be evaluated from the corresponding kernel.

**sMMD and sCMS.** All measures above are scalar finite signed measures, whose RKHS embeddings are standard [32]. For $k \in \{k_T, k_R, k_{TT}\}$, define

$$\mathrm{sMMD}_k^2(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}) = \left\| \mu_k^1 - \mu_k^2 \right\|^2, \tag{55}$$

$$\mathrm{sCMS}_k(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}) = \langle \overline{\mu}_k^1, \overline{\mu}_k^2 \rangle = \frac{\langle \mu_k^1, \mu_k^2 \rangle}{(\|\mu_k^1\| + \epsilon)(\|\mu_k^2\| + \epsilon)}, \tag{56}$$

where $\overline{\mu}_k^i = \mu_k^i / (\|\mu_k^i\| + \epsilon)$ and $\rho_k(\mathcal{C}^{\mathcal{M}_i}) = \|\mu_k^i\| / (\|\mu_k^i\| + \epsilon)$. These extend MMD [13] and Cosine Mean Similarity [15] to finitely supported signed measures; for $k_T$, the norm and inner product are those of the direct sum.

**Representation identities.** Write $f_j(\mathcal{C}^{\mathcal{M}_i})$ for $f_j(\mathbf{M}_T^{i\,\prime}, \mathbf{M}_R^{i\,\prime})$. Because vectorisation is an isometry and concatenated blocks occupy orthogonal coordinates,

$$\left\| f_1(\mathcal{C}^{\mathcal{M}_1}) - f_1(\mathcal{C}^{\mathcal{M}_2}) \right\|_2^2 = \mathrm{sMMD}_{k_T}^2(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}) + \mathrm{sMMD}_{k_R}^2(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}), \tag{57}$$

$$\left\| f_2(\mathcal{C}^{\mathcal{M}_1}) - f_2(\mathcal{C}^{\mathcal{M}_2}) \right\|_2^2 = \mathrm{sMMD}_{k_{TT}}^2(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}) + \mathrm{sMMD}_{k_R}^2(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}), \tag{58}$$

where Equation 58 is Equation 24. For each normalised block,

$$\left\| \overline{\mu}_k^1 - \overline{\mu}_k^2 \right\|^2 = \rho_k(\mathcal{C}^{\mathcal{M}_1})^2 + \rho_k(\mathcal{C}^{\mathcal{M}_2})^2 - 2\,\mathrm{sCMS}_k(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}). \tag{59}$$

Consequently,

$$\left\| f_4(\mathcal{C}^{\mathcal{M}_1}) - f_4(\mathcal{C}^{\mathcal{M}_2}) \right\|_2^2 = \sum_{k \in \{k_T, k_R\}} \left[ \rho_k(\mathcal{C}^{\mathcal{M}_1})^2 + \rho_k(\mathcal{C}^{\mathcal{M}_2})^2 - 2\,\mathrm{sCMS}_k(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}) \right] \tag{60}$$

$$\left\| f_5(\mathcal{C}^{\mathcal{M}_1}) - f_5(\mathcal{C}^{\mathcal{M}_2}) \right\|_2^2 = \sum_{k \in \{k_{TT}, k_R\}} \left[ \rho_k(\mathcal{C}^{\mathcal{M}_1})^2 + \rho_k(\mathcal{C}^{\mathcal{M}_2})^2 \; -2\,\mathrm{sCMS}_k(\mathcal{C}^{\mathcal{M}_1}, \mathcal{C}^{\mathcal{M}_2}) \right] \tag{61}$$

where Equation 61 is Equation 23.

Analogous kernel identities also hold for $f_3$ and $f_6$, using a different pair-kernel construction. The kernel identities derived above do not directly extend to the nonlinear neural representation $f_7$.

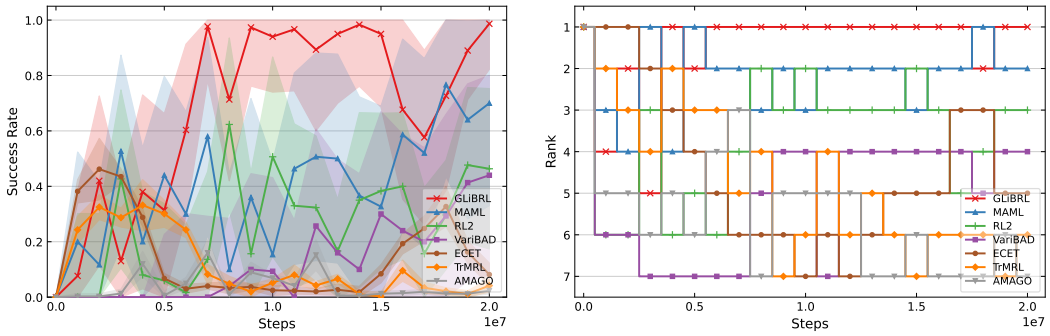
### A.11 Implementation Details

We report the details of our implementations of GLiBRL and other methods. The implementation of GLiBRL is simple with only model networks $\phi_T, \phi_R$ and policy networks $\pi_\psi^+$. The model networks $\phi_T, \phi_R$ are Multi-Layer Perceptrons (MLPs) consisting of *feature* and *mixture* networks. Feature networks convert raw states and actions to features and are shared in $\phi_T$ and $\phi_R$. Mixture networks mix the state and action features, further improving the representativeness. The policy network is actor-only for PPO with linear baseline as suggested by [24] in MetaWorld ML10/45, and actor-critic for SAC in MuJoCo tasks. We adopt the diagonal noise inference in ML10 and full noise inference in ML45.

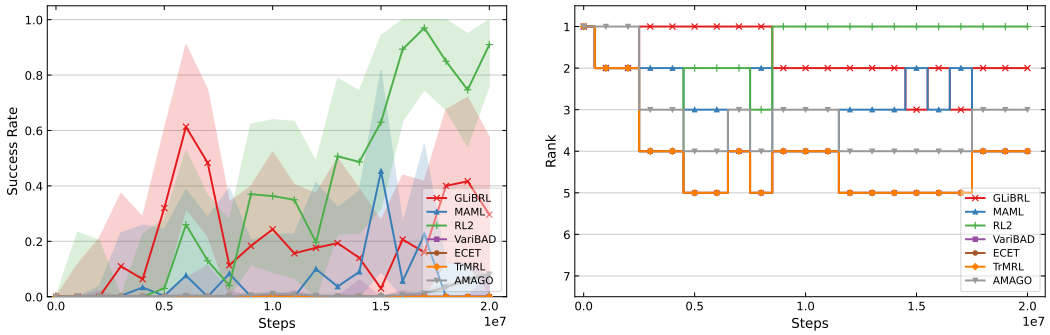
For reproducing all MuJoCo results, we follow the official implementations in TrMRL, PEARL and VariBAD. All results are run with GYMNASIUM==1.2.3 and MUJOCO==3.6.0. For MetaWorld results, we rerun MAML and $RL^2$ using the more performant implementation provided by [2], while re-implement and rerun VariBAD as the official implementation does not contain MetaWorld environments. We use the tuned hyperparameters from official implementations in MAML, $RL^2$ and VariBAD. For remaining baselines, we use the reported results from [31].

The table of all related hyperparameters of GLiBRL is shown in Appendix A.18.

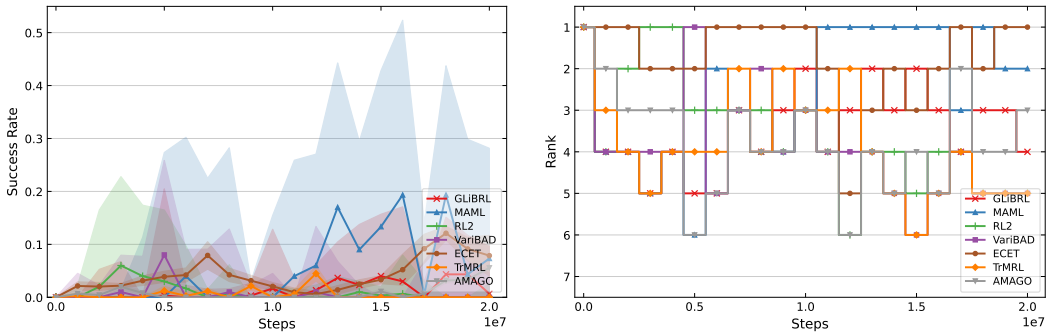
### A.12 ML10 Success Rate Comparisons Per-task



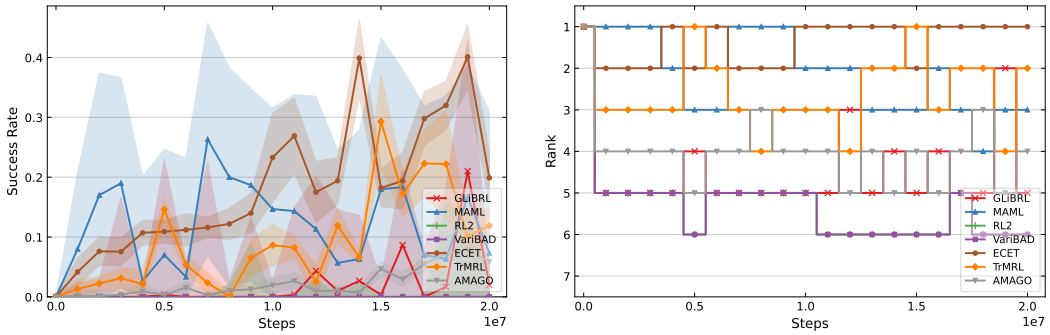

(a) Door Close




(b) Drawer Open

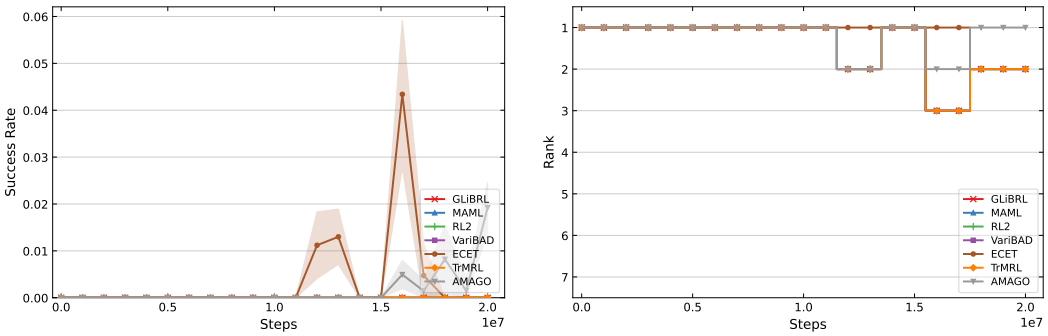



(c) Lever Pull




(d) Sweep Into

Figure 4: Testing success rate (left) and success rate rank (↓) (right) for each scenario in ML10.



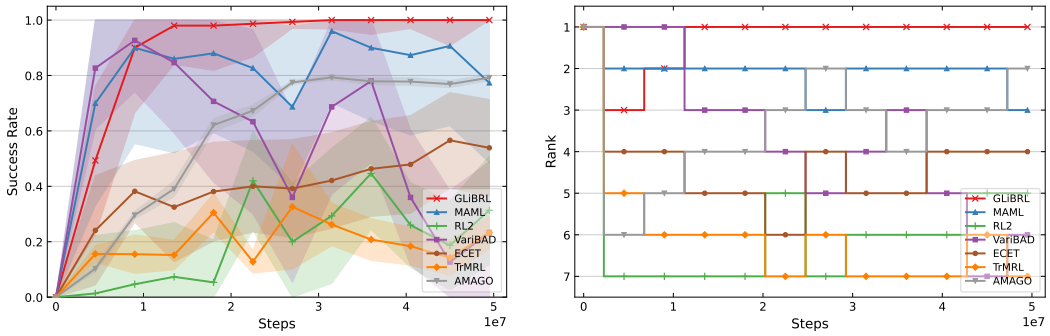

(e) Shelf Place

Figure 4: Testing success rate (left) and success rate rank (↓) (right) for each scenario in ML10.

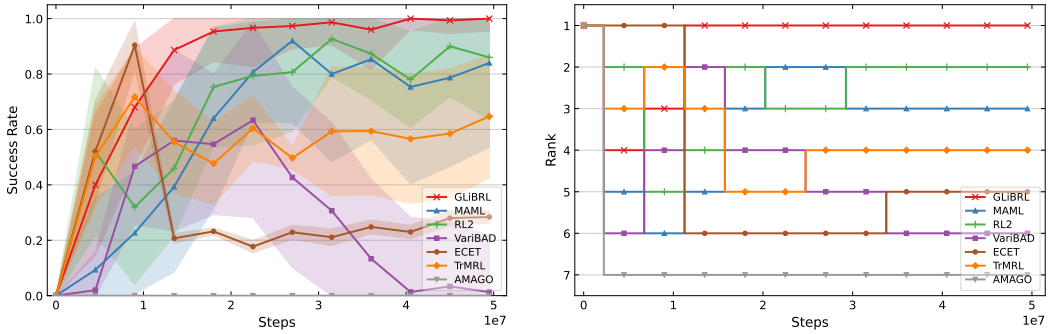
Table 1: Mean rank (↓) of each method over training in ML10.

| Environment | GLiBRL | MAML | RL2 | VariBAD | ECET | TrMRL | AMAGO |
|---|---|---|---|---|---|---|---|
| door-close | **1.48** | 2.19 | 3.62 | 5.00 | 4.38 | 4.71 | 5.52 |
| drawer-open | 1.76 | 2.90 | **1.62** | 4.10 | 4.10 | 4.10 | 3.14 |
| lever-pull | 3.38 | 2.57 | 3.52 | 3.90 | **1.71** | 3.81 | 3.71 |
| shelf-place | 1.43 | 1.43 | 1.43 | 1.43 | **1.14** | 1.43 | 1.19 |
| sweep-into | 4.43 | 2.14 | 5.29 | 5.29 | **1.38** | 2.57 | 4.00 |
| *Average* | 2.50 | **2.25** | 3.10 | 3.94 | 2.54 | 3.32 | 3.51 |

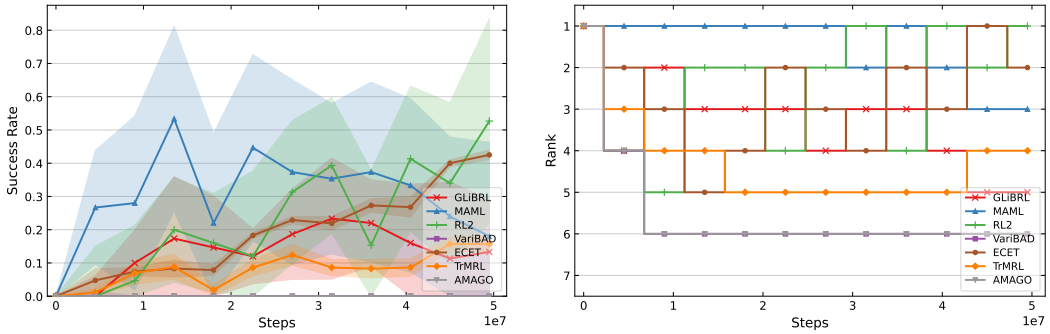
### A.13 ML45 Success Rate Comparisons Per-task



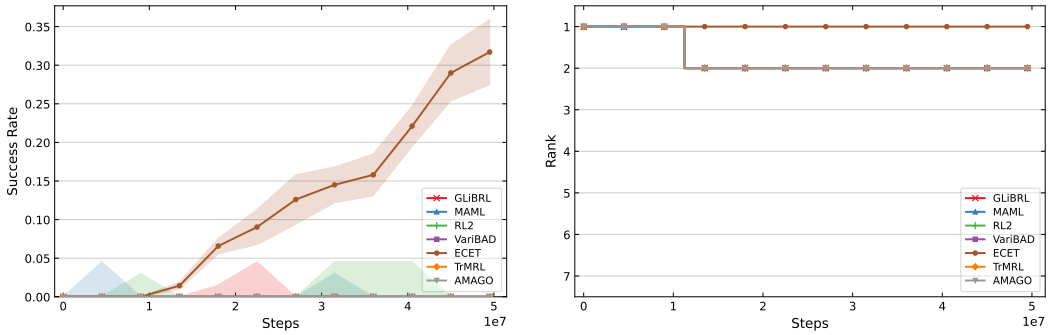

(a) Door Lock




(b) Door Unlock

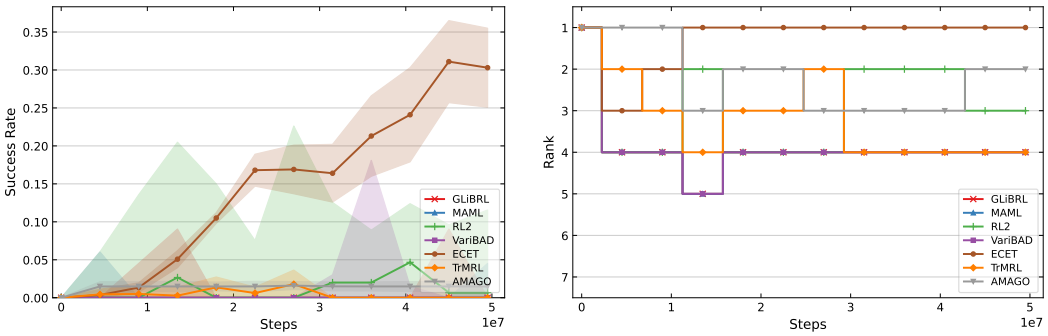



(c) Hand Insert




(d) Bin Picking

Figure 5: Testing success rate (left) and success rate rank (↓) (right) for each scenario in ML45.




(e) Box Close

Figure 5: Testing success rate (left) and success rate rank (↓) (right) for each scenario in ML45.

Table 2: Mean rank (↓) of each method over training in ML45.

| Environment | GLiBRL | MAML | RL2 | VariBAD | ECET | TrMRL | AMAGO |
|---|---|---|---|---|---|---|---|
| bin-picking | 1.75 | 1.75 | 1.75 | 1.75 | **1.00** | 1.75 | 1.75 |
| box-close | 3.83 | 3.83 | 2.92 | 3.83 | **1.25** | 3.17 | 2.17 |
| door-lock | **1.25** | 2.08 | 5.75 | 3.58 | 4.25 | 5.92 | 3.33 |
| door-unlock | **1.42** | 3.25 | 2.50 | 4.58 | 4.42 | 3.58 | 6.50 |
| hand-insert | 3.33 | **1.50** | 2.42 | 5.42 | 2.67 | 4.17 | 5.42 |
| *Average* | **2.32** | 2.48 | 3.07 | 3.83 | 2.72 | 3.72 | 3.83 |

### A.14 Qualitative Results on Task Identifiability

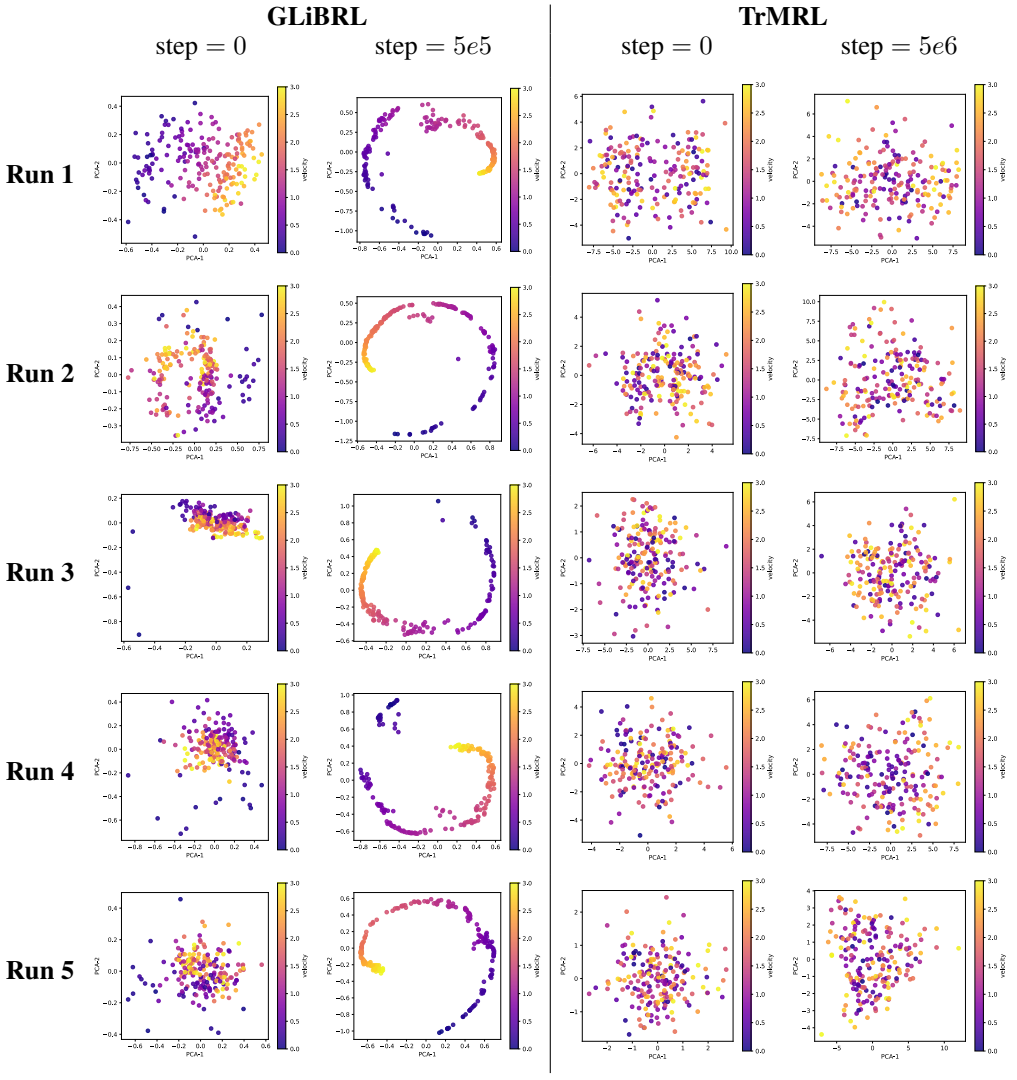
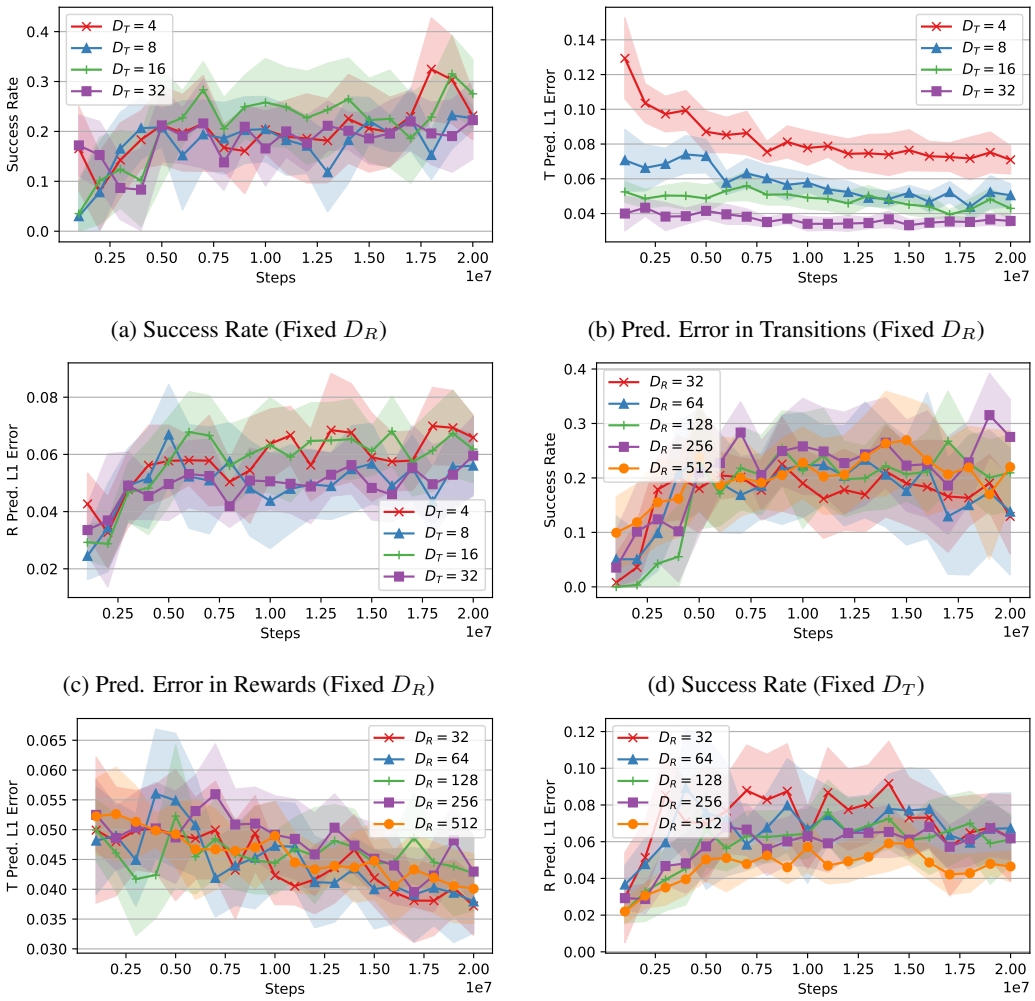



Figure 6: PCA projections of task representations from GLiBRL and TrMRL across five independent runs. These projections provide a qualitative view of task separation.

We visualise the task representations of GLiBRL and TrMRL on the `HalfCheetahVel` benchmark using 2D PCA across five independent runs; TrMRL is allocated a $10\times$ larger training-step budget for fairness. Even before training (step $= 0$), the GLiBRL projections show more apparent task separation than those of TrMRL at $5e6$ steps. After $5e5$ training steps, the GLiBRL projections display a clear ordering of the continuous target velocities and the spherical pattern induced by the normalised representation function $f_{T,R}$. Across these qualitative examples, nearby target velocities generally remain close in the projected space, whereas more distinct velocities tend to separate.

### A.15 Sensitivity of Latent Task Dimensions

To demonstrate the sensitivity of GLiBRL with respect to latent task dimensions $D_T$ and $D_R$, we perform hyperparameter search in ML10 on (1) $D_T = \{4, 8, 16, 32\}$ while fixing $D_R = 256$ and (2) $D_R = \{32, 64, 128, 256, 512\}$ while fixing $D_T = 16$. We compare on (1) the success rate, (2) the predictive error in transitions, and (3) the predictive error in rewards.




(a) Success Rate (Fixed $D_R$)

(b) Pred. Error in Transitions (Fixed $D_R$)

(c) Pred. Error in Rewards (Fixed $D_R$)

(d) Success Rate (Fixed $D_T$)

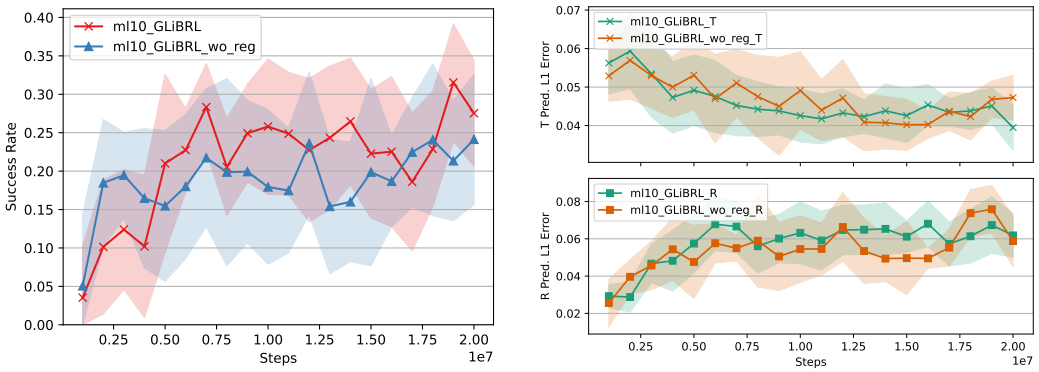
(e) Pred. Error in Transitions (Fixed $D_T$)

(f) Pred. Error in Rewards (Fixed $D_T$)

Figure 7: Sensitivity analysis of latent task dimensions in GLiBRL.

In general, GLiBRL shows robust success rates with respect to latent task dimensions $D_T, D_R$, which can be inferred from Figure (a) and (d). Figure (b) and (f) indicate that as $D_T, D_R$ grows, the corresponding error reduces, offering an efficiency-accuracy trade-off. Figure (c) and (e) are sanity checks that have confirmed transitions do not affect rewards, and vice versa. Overall, $D_R$ are set to be much larger than $D_T$, as reward functions are generally much harder to learn, compared to transition function.

### A.16 Ablation Studies




(a) Success Rate (b) Predictive Error

Figure 8: Success rates and predictive errors of GLiBRL with and without regularisation in ML10.

Running GLiBRL and its variant without regularisation in the ML10 benchmark, we can tell the regularisations do not change the overall trend in both success rate and prediction error. However, as the number of training step increases, GLiBRL begins to show in (a): higher success rates; in (b): lower predictive error in transitions. The higher predictive error in reward is due to the higher magnitude of reward from higher success rates in GLiBRL.

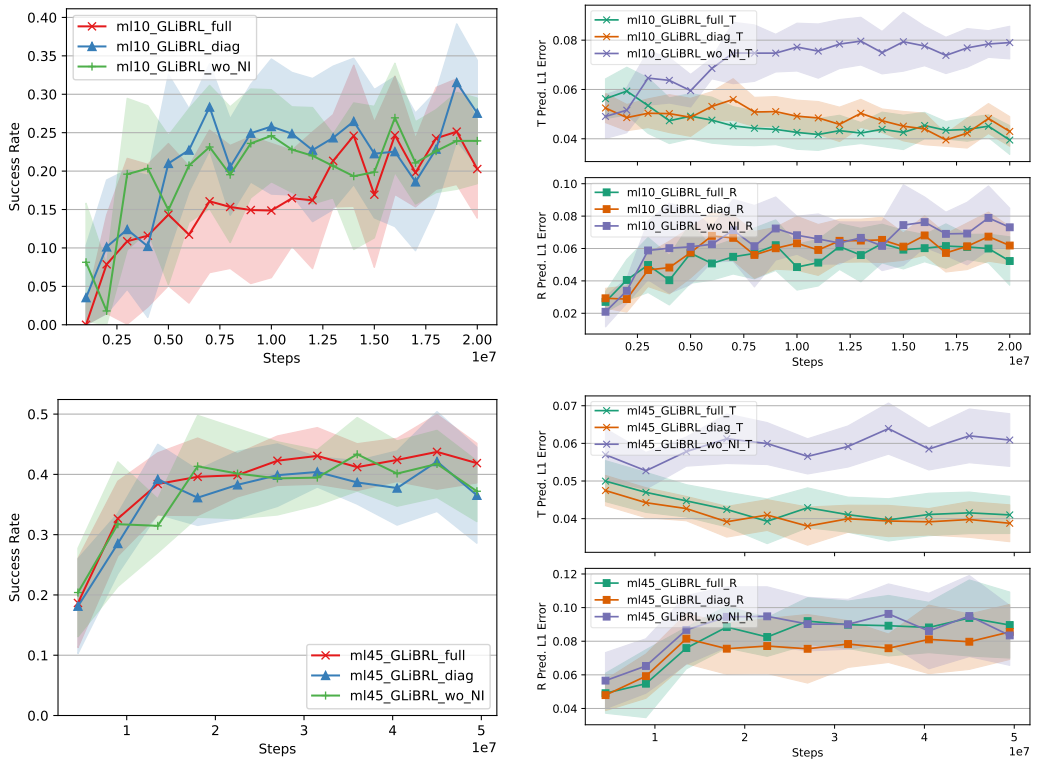



Figure 9: Success rates (Left) and predictive errors in transition and reward (Right) in ML10 and ML45, comparing GLiBRL_full, GLiBRL_diag and GLiBRL_wo_NI.

GLiBRL can be viewed as a generalised, deep BRL version of ALPaCA; whereas ALPaCA assumes the model noises $T_\sigma = \mathbf{\Sigma}_T, R_\sigma = \mathbf{\Sigma}_R$ are fully known a priori, GLiBRL performs full Bayesian

inference over them. Under the assumption of ALPaCA, Equation 18 reduces to (see Appendix A.17)

$$\log_{\phi_T,\phi_R}(\mathcal{C}) = -\frac{1}{2}\sum_{i=1}^{M}\Big(D_S \log|\boldsymbol{\Xi}'_T| - \mathrm{Tr}\left(\boldsymbol{\Sigma}_T^{-1}\mathbf{M}_T'^{\mathrm{T}}\boldsymbol{\Xi}'_T\mathbf{M}'_T\right) + \log|\boldsymbol{\Xi}'_R| - \mathrm{Tr}\left(\boldsymbol{\Sigma}_R^{-1}\mathbf{M}_R'^{\mathrm{T}}\boldsymbol{\Xi}'_R\mathbf{M}'_R\right)\Big) + \text{const.} \tag{62}$$

We studied whether performing Bayesian inference on model noises is necessary for learning accurate transition and reward models. We test GLiBRL with full noise inference (GLiBRL_full), GLiBRL with diagonal noise inference (GLiBRL_diag) and GLiBRL without noise inference (GLiBRL_wo_NI) with identical hyperparameters on ML10 and ML45 for 10 seeds. We have set $\Sigma_T = 0.025$ and $\Sigma_R = 0.5$ to match the initial covariance of GLiBRL_full and GLiBRL_diag. The metrics being evaluated are success rates and $\mathcal{L}_1$ norms of prediction errors in both transitions (defined as $|\mathbf{S}' - \mathbf{C}_T\mathbf{M}'_T|_1$) and rewards (defined as $|\mathbf{r} - \mathbf{C}_R\mathbf{M}'_R|_1$).

As demonstrated in Figure 9, both noise-inference variants substantially reduce transition prediction error relative to GLiBRL_wo_NI; the differences in reward error are smaller and partly reflect the increasing reward magnitude as policies become more successful. The increasing transition error of GLiBRL_wo_NI is undesirable because state magnitudes remain comparatively bounded. More accurate predictive models also make GLiBRL_diag and GLiBRL_full) better suited to model-based extensions that rely on imagined samples.

Interestingly, the relative behaviour of GLiBRL_diag and GLiBRL_full reverses between ML10 and ML45. On ML10, GLiBRL_diag achieves a higher success rate despite having a higher predictive error, whereas on ML45, the same pattern is observed for GLiBRL_full. A plausible explanation is a bias–variance trade-off in noise modelling. The greater task diversity of ML45 makes correlations between state dimensions more informative, allowing the full-covariance marginal likelihood to learn more useful task representations. On the less diverse ML10 benchmark, modelling these correlations may instead increase estimation and optimisation variance or fit spurious correlations, while the diagonal formulation acts as an effective regulariser. The predictive-error ordering is likely a consequence of the resulting policies: the more successful variant reaches a broader range of contact-rich states and receives larger rewards, making its on-policy samples harder to predict.

### A.17 Marginal Log-likelihood of Normal-Normal

We prove Equation 62 has the following form

$$\log_{\phi_T,\phi_R}(\mathcal{C}) = -\frac{1}{2}\sum_{i=1}^{M}\Big(D_S \log|\boldsymbol{\Xi}'_T| - \mathrm{Tr}\left(\boldsymbol{\Sigma}_T^{-1}\mathbf{M}_T'^{\mathrm{T}}\boldsymbol{\Xi}'_T\mathbf{M}'_T\right) + \log|\boldsymbol{\Xi}'_R| - \mathrm{Tr}\left(\boldsymbol{\Sigma}_R^{-1}\mathbf{M}_R'^{\mathrm{T}}\boldsymbol{\Xi}'_R\mathbf{M}'_R\right)\Big) + \text{const.} \tag{62}$$

**Proof:**

The distributions without inferring on the noise are listed as follows:

**Likelihood:**

$$p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i,\mathbf{A}_i,\theta_T) = \mathcal{MN}\left(\mathbf{S}'_i|\mathbf{C}_T T_\mu, \mathbf{I}_N, \boldsymbol{\Sigma}_T \in \mathbb{R}^{D_S\times D_S}\right) \tag{63}$$

$$p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i,\theta_R) = \mathcal{MN}\left(\mathbf{r}_i|\mathbf{C}_R R_\mu, \mathbf{I}_N, \boldsymbol{\Sigma}_R \in \mathbb{R}^{1\times 1}\right) \tag{64}$$

**Prior:**

$$p(\theta_T) = p(T_\mu) = \mathcal{MN}(T_\mu|\mathbf{M}_T \in \mathbb{R}^{D_T\times D_S}, \boldsymbol{\Xi}_T^{-1} \in \mathbb{R}^{D_T\times D_T}, \boldsymbol{\Sigma}_T) \tag{65}$$

$$p(\theta_R) = p(R_\mu) = \mathcal{MN}(R_\mu|\mathbf{M}_R \in \mathbb{R}^{D_R\times 1}, \boldsymbol{\Xi}_R^{-1} \in \mathbb{R}^{D_R\times D_R}, \boldsymbol{\Sigma}_R) \tag{66}$$

**Posterior:**

$$p_{\phi_T}(\theta_T|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i) = \mathcal{MN}(T_\mu|\mathbf{M}'_T, \boldsymbol{\Xi}_T'^{-1}, \boldsymbol{\Sigma}_T) \tag{67}$$

$$p_{\phi_R}(\theta_R|\mathbf{S}_i,\mathbf{A}_i,\mathbf{S}'_i,\mathbf{r}) = \mathcal{MN}(R_\mu|\mathbf{M}'_R, \boldsymbol{\Xi}_R'^{-1}, \boldsymbol{\Sigma}_R) \tag{68}$$

where

$$\begin{aligned} \mathbf{M}'_T &= {\mathbf{\Xi}'_T}^{-1}\left[\mathbf{C}_T^{\mathrm{T}}\mathbf{S}' + \mathbf{\Xi}_T\mathbf{M}_T\right] & \mathbf{M}'_R &= {\mathbf{\Xi}'_R}^{-1}\left[\mathbf{C}_R^{\mathrm{T}}\mathbf{r} + \mathbf{\Xi}_R\mathbf{M}_R\right] \\ \mathbf{\Xi}'_T &= \mathbf{C}_T^{\mathrm{T}}\mathbf{C}_T + \mathbf{\Xi}_T & \mathbf{\Xi}'_R &= \mathbf{C}_R^{\mathrm{T}}\mathbf{C}_R + \mathbf{\Xi}_R \end{aligned} \tag{69}$$

Similar to Appendix A.7,

$$p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i) = \frac{p_{\phi_T}(\theta_T, \mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i)}{p_{\phi_T}(\theta_T|\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i)} \tag{70}$$

As

$$\begin{aligned} p_{\phi_T}(\theta_T, \mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i) = \frac{|\mathbf{\Xi}_T|^{D_S/2}|\mathbf{\Sigma}_T|^{-D_T/2}}{\sqrt{(2\pi)^{D_T D_S}}} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(\mathbf{\Sigma}_T^{-1}(T_\mu - \mathbf{M}_T)^{\mathrm{T}}\mathbf{\Xi}_T(T_\mu - \mathbf{M}_T)\right)\right] \\ \cdot \frac{|\mathbf{\Sigma}_T|^{-N/2}}{\sqrt{(2\pi)^{N D_S}}} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(\mathbf{\Sigma}_T^{-1}(\mathbf{S}'_i - \mathbf{C}_T T_\mu)^{\mathrm{T}}(\mathbf{S}'_i - \mathbf{C}_T T_\mu)\right)\right] \end{aligned} \tag{71}$$

And

$$p_{\phi_T}(\theta_T|\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \frac{|\mathbf{\Xi}'_T|^{D_S/2}|\mathbf{\Sigma}_T|^{-D_T/2}}{\sqrt{(2\pi)^{D_T D_S}}} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(\mathbf{\Sigma}_T^{-1}(T_\mu - \mathbf{M}'_T)^{\mathrm{T}}\mathbf{\Xi}'_T(T_\mu - \mathbf{M}'_T)\right)\right] \tag{72}$$

Hence

$$p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i) \propto |\mathbf{\Xi}'_T|^{-D_S/2} \cdot \exp\left[-\frac{1}{2}\operatorname{Tr}\left(-\mathbf{\Sigma}_T^{-1}\mathbf{M}_T'^{\mathrm{T}}\mathbf{\Xi}'_T\mathbf{M}'_T\right)\right] \tag{73}$$

$$\log p_{\phi_T}(\mathbf{S}'_i|\mathbf{S}_i, \mathbf{A}_i) = -\frac{D_S}{2}\log|\mathbf{\Xi}'_T| + \frac{1}{2}\operatorname{Tr}\left(\mathbf{\Sigma}_T^{-1}\mathbf{M}_T'^{\mathrm{T}}\mathbf{\Xi}'_T\mathbf{M}'_T\right) + \text{const.} \tag{74}$$

Similarly,

$$\log p_{\phi_R}(\mathbf{r}_i|\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = -\frac{1}{2}\log|\mathbf{\Xi}'_R| + \frac{1}{2}\operatorname{Tr}\left(\mathbf{\Sigma}_R^{-1}\mathbf{M}_R'^{\mathrm{T}}\mathbf{\Xi}'_R\mathbf{M}'_R\right) + \text{const.} \tag{75}$$

Following Appendix A.7,

$$\begin{aligned} \log_{\phi_T,\phi_R}(\mathcal{C}) = -\frac{1}{2}\sum_{i=1}^{M}\Big(D_S\log|\mathbf{\Xi}'_T| - \operatorname{Tr}\left(\mathbf{\Sigma}_T^{-1}\mathbf{M}_T'^{\mathrm{T}}\mathbf{\Xi}'_T\mathbf{M}'_T\right) \\ + \log|\mathbf{\Xi}'_R| - \operatorname{Tr}\left(\mathbf{\Sigma}_R^{-1}\mathbf{M}_R'^{\mathrm{T}}\mathbf{\Xi}'_R\mathbf{M}'_R\right)\Big) + \text{const.} \end{aligned} \tag{62}$$

### A.18 Hyperparameters and Runtime

We list all hyperparameters of GLiBRL in the following tables.

In `HalfCheetahDir`:

| Name | Value | Name | Value |
|---|---|---|---|
| **policy_learner** | SAC | **feat_out_activation** | True |
| **critic_layer** | [256, 256] | **t_mix_layers** | [64, 32] |
| **actor_layer** | [256, 256, 256] | **t_mix_layernorm** | True |
| **policy_activation** | Tanh | **t_mix_out_activation** | False |
| **policy_optimiser** | Adam | **r_mix_layers** | [128, 64] |
| **actor_lr/critic_lr** | 3e-4/4e-4 | **r_mix_layernorm** | True |
| **policy_max_norm** | 10 | **r_mix_out_activation** | False |
| **policy_weight_init** | Xavier | **t_reg_coef** | 5e-3 |
| **policy_bias_init** | 0 | **r_reg_coef** | 1e-3 |
| **actor_logstd_range** | $[\ln 10^{-6}, \ln 2]$ | **model_activation** | ReLU |
| **tau** | 5e-3 | **model_optimiser** | Adam |
| **actor_grad_steps** | 500 | **model_lr** | 2e-4 |
| **critic_grad_steps** | 500 | **model_opt_max_norm** | None |
| **policy_update_freq** | 200 | **model_grad_epochs** | 1 |
| **init_alpha** | 1 | **model_grad_steps** | 1 |
| **alpha_lr** | 3e-4 | **init_mt** | zeros |
| **policy_bs_range** | [1, 200] | **init_mr** | zeros |
| **s_feat_layers** | [64, 32] | **init_xit** | ones |
| **s_feat_outdim** | 32 | **init_xir** | ones |
| **s_feat_layernorm** | False | **init_omegat** | ones |
| **a_feat_layers** | [32, 16] | **init_omegar** | ones |
| **a_feat_outdim** | 16 | **init_nut** | 22 |
| **a_feat_layernorm** | False | **init_nur** | 2 |

In `HalfCheetahVel`:

| Name | Value | Name | Value |
|---|---|---|---|
| **policy_learner** | SAC | **feat_out_activation** | True |
| **policy_layers** | [256, 256] | **t_mix_layers** | [64, 32] |
| **a_feat_out_activation** | True | **t_mix_layernorm** | True |
| **policy_activation** | Tanh | **t_mix_out_activation** | False |
| **policy_optimiser** | Adam | **r_mix_layers** | [128, 64] |
| **actor_lr/critic_lr** | 3e-4/4e-4 | **r_mix_layernorm** | True |
| **policy_max_norm** | 10 | **r_mix_out_activation** | False |
| **policy_weight_init** | Xavier | **t_reg_coef** | 5e-3 |
| **policy_bias_init** | 0 | **r_reg_coef** | 1e-3 |
| **actor_logstd_range** | $[\ln 10^{-6}, \ln 2]$ | **model_activation** | ReLU |
| **tau** | 1e-2 | **model_optimiser** | Adam |
| **actor_grad_steps** | 2000 | **model_lr** | 2e-4 |
| **critic_grad_steps** | 2000 | **model_opt_max_norm** | None |
| **policy_update_freq** | 200 | **model_grad_epochs** | 1 |
| **init_alpha** | 0.2 | **model_grad_steps** | 50 |
| **alpha_lr** | 3e-4 | **init_mt** | zeros |
| **policy_bs_range** | [1, 200] | **init_mr** | zeros |
| **s_feat_layers** | [64, 32] | **init_xit** | **I** |
| **s_feat_outdim** | 32 | **init_xir** | **I** |
| **s_feat_layernorm** | False | **init_omegat** | **I** |
| **a_feat_layers** | [32, 16] | **init_omegar** | **I** |
| **a_feat_outdim** | 16 | **init_nut** | 22 |
| **a_feat_layernorm** | False | **init_nur** | 2 |

In `AntDir`:

| Name | Value | Name | Value |
|---|---|---|---|
| **policy_learner** | SAC | **feat_out_activation** | True |
| **policy_layers** | [256, 256] | **t_mix_layers** | [64, 32] |
| **a_feat_out_activation** | True | **t_mix_layernorm** | True |
| **policy_activation** | Tanh | **t_mix_out_activation** | False |
| **policy_optimiser** | Adam | **r_mix_layers** | [128, 64] |
| **actor_lr/critic_lr** | 3e-4/4e-4 | **r_mix_layernorm** | True |
| **policy_max_norm** | 1 | **r_mix_out_activation** | False |
| **policy_weight_init** | Xavier | **t_reg_coef** | 5e-3 |
| **policy_bias_init** | 0 | **r_reg_coef** | 1e-3 |
| **actor_logstd_range** | $[\ln 10^{-6}, \ln 2]$ | **model_activation** | ReLU |
| **tau** | 1e-2 | **model_optimiser** | Adam |
| **actor_grad_steps** | 1000 | **model_lr** | 2e-4 |
| **critic_grad_steps** | 2000 | **model_opt_max_norm** | None |
| **policy_update_freq** | 400 | **model_grad_epochs** | 1 |
| **init_alpha** | 0.1 | **model_grad_steps** | 5 |
| **alpha_lr** | 3e-4 | **init_mt** | zeros |
| **policy_bs_range** | [1, 200] | **init_mr** | zeros |
| **s_feat_layers** | [64, 32] | **init_xit** | **I** |
| **s_feat_outdim** | 32 | **init_xir** | **I** |
| **s_feat_layernorm** | False | **init_omegat** | **I** |
| **a_feat_layers** | [32, 16] | **init_omegar** | **I** |
| **a_feat_outdim** | 16 | **init_nut** | 28 |
| **a_feat_layernorm** | False | **init_nur** | 2 |

We use the same hyperparameters for both ML10 and ML45.

| Name | Value | Name | Value |
|---|---|---|---|
| **policy_learner** | PPO | **feat_out_activation** | True |
| **policy_layers** | [256, 256] | **t_mix_layers** | [64, 32] |
| **a_feat_out_activation** | True | **t_mix_layernorm** | True |
| **policy_activation** | Tanh | **t_mix_out_activation** | False |
| **policy_optimiser** | Adam | **r_mix_layers** | [128, 64] |
| **policy_lr** | 5e-4 | **r_mix_layernorm** | True |
| **policy_opt_max_norm** | N/A | **r_mix_out_activation** | False |
| **policy_weight_init** | Xavier | **t_reg_coef** | 5e-3 |
| **policy_bias_init** | 0 | **r_reg_coef** | 1e-3 |
| **policy_log_std_min** | $\ln 10^{-6}$ | **model_activation** | ReLU |
| **policy_log_std_max** | $\ln 2$ | **model_optimiser** | Adam |
| **policy_grad_epochs** | 10 | **model_lr** | 2e-4 |
| **policy_grad_steps** | 20 | **model_opt_max_norm** | None |
| **ppo_clip_eps** | 0.5 | **model_grad_epochs** | 1 |
| **ppo_gamma** | 0.99 | **model_grad_steps** | 20 |
| **ppo_gae_lambda** | 0.95 | **init_mt** | zeros |
| **ppo_entropy_coef** | 5e-3 | **init_mr** | zeros |
| **s_feat_layers** | [64, 32] | **init_xit** | **I** |
| **s_feat_outdim** | 32 | **init_xir** | **I** |
| **s_feat_layernorm** | False | **init_omegat** | **I** |
| **a_feat_layers** | [32, 16] | **init_omegar** | **I** |
| **a_feat_outdim** | 16 | **init_nut** | 40 |
| **a_feat_layernorm** | False | **init_nur** | 2 |

The runtime of GLiBRL does not vary much with changes in $D_T$ and $D_R$. In the following table, we show the average runtime in minutes for each single run with different $D_T$ and $D_R$.

| $D_T$ | $D_R$ | Avg. Runtime (min) |
|---|---|---|
| 16 | 32 | 72 |
| 16 | 512 | 80 |
| 4 | 256 | 75 |
| 32 | 256 | 77 |

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims and the results are consistent.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In the Conclusion section, we mentioned the limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All proofs are in the appendix. We state assumptions whenever needed.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: In the appendix we mentioned the implementation details and hyperparameters.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Experiments are reproducible in our official implementation.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We follow the exact procedure from previous papers, and provided hyperparameters in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run experiments five times for MuJoCo and ten times for MetaWorld. We reported the $95\%$ CI.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In both experiment section and appendix we mentioned the resource.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We checked the Code of Ethics and do not observe any violations.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The goal of this paper is to forward the research of Meta-RL / Bayes-RL. The authors do not see a potential societal impact of this work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks. We use open source datasets without any modifications.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use open source datasets and cited them.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We use existing datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: All experiments are performed in simulation. No human participants involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: All experiments are performed in simulation. No human participants involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLM is only used for improving the writing.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.