Gibbs Variational Last Layers for Long-Horizon Dynamics Learning

Paul Brunzema **Henrik Hose** **Sebastian Trimpe**
RWTH Aachen University
Aachen, Germany
{firstname.lastname}@dsme.rwth-aachen.de

Abstract

Accurate long-horizon probabilistic predictions of dynamical systems are a prerequisite for model-based decision-making. Variational Bayesian last layers (VBLLs) are an attractive model class for this setting, offering tractable uncertainty at near-deterministic cost, yet they are trained exclusively with single-step likelihood objectives that may be insufficient for reliable multi-step rollouts. We revisit VBLLs through the lens of Gibbs variational inference, which decouples posterior inference from the choice of training loss. Building on this perspective, we introduce a family of multi-step training losses that progressively incorporate long-horizon structure, robustness via continuous ranked probability scores (CRPS), and training on full autoregressive rollouts. Our approach retains the simplicity of VBLLs while directly optimizing for multi-step predictive quality. Across illustrative synthetic environments, a diverse suite of chaotic dynamical systems, and real-world data, we show that multi-step, CRPS-based objectives substantially improve long-horizon accuracy and distributional fit, consistently outperforming single-step likelihood training.

1 Introduction

Learned probabilistic dynamics models underpin a growing range of decision-making systems, from sampling-based control [46, 10] to uncertainty-aware policy search [14, 39, 36, 20]. In all of these settings, the model is rolled out autoregressively over extended horizons, so predictive accuracy and well-calibrated uncertainty are not just desirable but prerequisite: small local errors compound step by step and can render predictions unreliable [43]. Improving the quality of these long-horizon predictions is therefore a foundational challenge, independent of the downstream application. Variational Bayesian last layers (VBLLs) [32], a specific instance of Bayesian last-layer models [58, 54, 63, 19], offer a compelling architecture for this purpose, combining tractable predictive inference with the efficiency of deterministic feature extractors. By restricting uncertainty over parameters to a Gaussian posterior over the final layer, they yield good predictive distributions at computational cost close to that of standard neural networks, and have been successfully deployed [47, 6, 46, 11, 7]. Rather than arguing that VBLLs are the best probabilistic dynamics model in general, we ask how this increasingly used model class should be *trained* when deployed autoregressively.

In practice, VBLLs are trained using single-step likelihood objectives, scoring predictions on isolated transitions under a Gaussian log-likelihood. While this objective is convenient, it implicitly fixes two design choices that are in fact independent of the variational last-layer construction itself. First, the model is evaluated only on isolated transitions, so the learned features are shaped by single-step mini-batches rather than trajectory structure, and the model is never exposed to the compounding errors that arise in autoregressive rollouts at deployment. Second, the data-fit term is restricted to a Gaussian log-likelihood, whose quadratic tails can be sensitive to outliers [57]. Crucially, these are

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

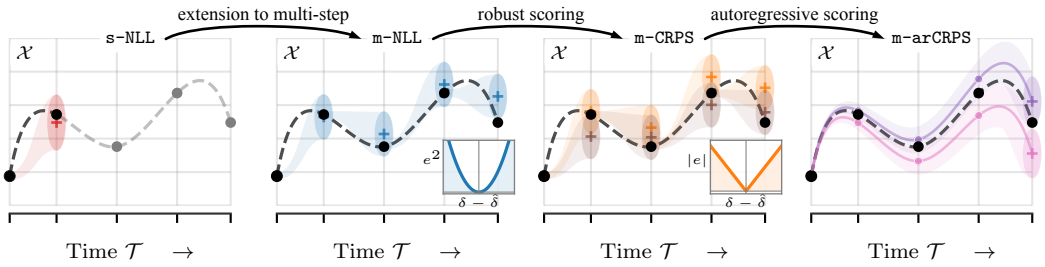


Figure 1: *Overview of the developed loss functions for VBLLs under the Gibbs VI framework.* We extend existing single-step (`s-NLL`) to a multi-step (`m-NLL`), robust (`m-CRPS`), autoregressive loss (`m-arCRPS`).

limitations of the training *objective*, not the *model*: the Gaussian posterior over the last layer and the scoring mechanism are mathematically independent, and the VBLL architecture is compatible with a much broader class of training losses.

We develop this perspective by revisiting VBLLs through the lens of Gibbs variational inference (Gibbs VI) [2, 5, 37], which generalizes standard variational Bayes by replacing the log-likelihood with an arbitrary loss. Applied to last-layer models, this yields what we call *Gibbs variational last layers*: a framework in which the training objective becomes a design choice that can encode multi-step structure, robust scoring rules, and exposure to autoregressive errors (Fig. 1)—all without modifying the model architecture, the posterior family, or the tractability of predictive inference. Within this framework, we develop a family of multi-step training objectives that improve autoregressive performance across diverse metrics, including mean rollout approximation, distributional match, and coverage of true rollouts. Across a wide range of systems, we find that task-aligned training consistently improves long-horizon accuracy and uncertainty estimation, while retaining the simplicity and predictive tractability of VBLLs.

Contributions In summary, our core contributions are as follows:

- **C1** We introduce and formalize Gibbs variational last layers, showing that Gibbs VI strictly generalizes the standard VBLL objective and provides a principled framework for incorporating task-aligned losses into (variational) Bayesian last-layer models.
- **C2** We derive a hierarchy of multi-step training objectives within Gibbs VI that progressively introduces multi-step structure, robustness to outliers, and exposure to autoregressive errors.
- **C3** We confirm the superior performance across 18 dynamical systems and demonstrate on data from a real-world unstable robotic system that a task-aligned objective is essential when idealized assumptions break. Further, in an illustrative example, we isolate a sampling-bias failure mode of single-step likelihood training that multi-step objectives mitigate.

2 Problem Setting

Consider a system with state $\boldsymbol{x}_t \in \mathbb{R}^{d_x}$ evolving according to the unknown stochastic transition $\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t) + \boldsymbol{\nu}_t$, $\boldsymbol{\nu}_t \sim p_\nu$. We denote a trajectory segment of horizon H by $\boldsymbol{\tau}^{(n)} = (\boldsymbol{x}_0^{(n)}, \boldsymbol{x}_1^{(n)}, \dots, \boldsymbol{x}_H^{(n)})$ and assume access to a dataset $\mathcal{D} = \{\boldsymbol{\tau}^{(n)}\}_{n=1}^N$ of N such segments. We approximate the general nonlinear system with a residual model

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_t) + \boldsymbol{\nu}_t, \qquad \boldsymbol{\nu}_t \sim \mathcal{N}(\mathbf{0},\, \boldsymbol{\Sigma}_\nu), \tag{1}$$

where $\phi_{\boldsymbol{\theta}} \colon \mathbb{R}^{d_x} \to \mathbb{R}^p$ is a learned (deep) feature extractor parameterized by $\boldsymbol{\theta}$, $\boldsymbol{W} \in \mathbb{R}^{p \times d_x}$ is the last-layer weight matrix, and $\boldsymbol{\Sigma}_\nu = \operatorname{diag}(\sigma_1^2, \dots, \sigma_{d_x}^2)$ is a diagonal process-noise covariance. The residual structure, also used, *e.g.* in [14], ensures that the model defaults to the identity map, aiding optimization and reflecting the prior belief that consecutive states are similar for sufficiently small Δt. The framework extends naturally to controlled systems by augmenting $\boldsymbol{x}_t$ with a control input $\boldsymbol{u}_t$.

Variational Bayesian Last Layer We place an isotropic Gaussian prior on the columns $\boldsymbol{w}_j \in \mathbb{R}^p$ of $\boldsymbol{W}$ and approximate the posterior with a mean-field Gaussian factorized across columns as in the original VBLL paper [32] as well as in application papers of VBLLs [6, 7]:

$$\underbrace{p(\boldsymbol{W}) = \textstyle\prod_{j=1}^{d_x} \mathcal{N}(\boldsymbol{w}_j;\, \mathbf{0},\, \alpha\, \boldsymbol{I}_p)}_{\text{Prior}}, \qquad \text{and} \qquad \underbrace{q(\boldsymbol{W} \mid \boldsymbol{\eta}) = \textstyle\prod_{j=1}^{d_x} \mathcal{N}(\boldsymbol{w}_j;\, \boldsymbol{\mu}_j,\, \boldsymbol{L}_j \boldsymbol{L}_j^\top)}_{\text{Variational Posterior}}, \tag{2}$$

where $\boldsymbol{\eta} = \{(\boldsymbol{\mu}_j, \boldsymbol{L}_j)\}_{j=1}^{d_x}$ collects the variational parameters and each $\boldsymbol{L}_j \in \mathbb{R}^{p\times p}$ is a lower-triangular Cholesky factor (see Appendix B for the concrete VBLL parameterization). Restricting uncertainty to the last layer keeps predictive inference tractable and has demonstrated strong performance in various decision making under uncertainty settings [31, 47, 46, 13, 11, 6, 7, 54].

3 Gibbs Variational Last Layers

The variational last-layer construction of VBLLs, *i.e.* a Gaussian posterior over $\boldsymbol{W}$, regularized toward a Gaussian prior via a KL penalty, is agnostic to how predictive quality is measured. The standard VBLL objective [32] fills this slot with a single-step Gaussian log-likelihood, but this is a *choice*, not a requirement. This modularity is precisely the premise of Gibbs VI [2, 37, 5], which replaces the log-likelihood in the ELBO with an arbitrary loss function while retaining the KL divergence as a complexity penalty, yielding a variational approximation to the *Gibbs posterior* [5, 9]. Applying this idea to VBLLs yields what we call *Gibbs variational last layers*: the loss becomes a design lever that can be tailored to the deployment setting without touching the model.

3.1 General Framework

Given this decoupling, the natural question is: *What loss should one use?* The Gibbs variational objective [5, 2] provides the formal framework. We seek a variational posterior within the family $\mathcal{Q}$ over the last-layer weights that minimizes expected loss on the training data while remaining close to the prior:

$$q^*(\boldsymbol{W}) \in \arg\min_{q\in\mathcal{Q}} \Big\{ \underbrace{\mathbb{E}_{q(\boldsymbol{W})}\Big[\textstyle\sum_{n=1}^{N} \ell\big(\boldsymbol{W}, \boldsymbol{\tau}^{(n)}\big)\Big]}_{\textit{Data fit term}} + \lambda \underbrace{D_{\mathrm{KL}}(q \,||\, p)}_{\textit{Regularizer}} \Big\} \tag{3}$$

where $\lambda > 0$ controls regularization strength. The unconstrained minimizer of (3) is the *Gibbs posterior* [5, 9] and (3) defines its variational approximation in $\mathcal{Q}$.

The key observation is that ℓ is a *free parameter*: when it is the negative log-likelihood, (3) recovers the standard VBLL objective [32]. For other choices, the optimum retains a PAC-Bayesian interpretation as a trade-off between empirical risk and prior deviation [9, 37]. This flexibility comes at the cost of a likelihood-based Bayesian interpretation as the resulting posterior is no longer an approximation to the conventional Bayesian posterior, and λ becomes a tunable regularization parameter, as in tempered posterior approaches. Nevertheless, the standard ELBO represents only one choice within a broader space of objectives and, as we will show, is not always the most effective.

In practice, we do not fix the feature extractor $\phi_{\boldsymbol{\theta}}$ but optimize it jointly with the variational parameters $\boldsymbol{\eta}$. We therefore seek to solve the following optimization problem:

$$(\boldsymbol{\eta}^*, \boldsymbol{\theta}^*, \boldsymbol{\Sigma}_\nu^*) \in \arg\min_{\boldsymbol{\eta},\boldsymbol{\theta},\boldsymbol{\Sigma}_\nu} \left\{ \mathbb{E}_{q(\boldsymbol{W}|\boldsymbol{\eta})}\left[\frac{1}{N}\sum_{n=1}^{N} \ell\big(\boldsymbol{W}, \boldsymbol{\tau}^{(n)}\big)\right] + \lambda\, D_{\mathrm{KL}}\big(q(\boldsymbol{W} \mid \boldsymbol{\eta}) \,\|\, p(\boldsymbol{W})\big) \right\}. \tag{4}$$

Joint optimization is essential: while (3) determines q^* for a given feature map, optimizing $\boldsymbol{\theta}$ simultaneously allows the representation to co-adapt to the choice of loss. We will show that even for objectives that are equivalent for fixed features (*cf*. Corollary 1), joint optimization can lead to qualitatively different representations.

Since both q and p are Gaussian, the KL term admits a closed-form expression. In the following discussions as well as the experiments, we will always use the same model parameterization with the exact same initialization to isolate the influence of the loss function on the performance. Note that also for all loss functions, the one-step predictive distribution will remain Gaussian (Appendix E.1).

We now instantiate (4) with concrete losses that span two design axes: the *scoring rule* (Gaussian log-likelihood or CRPS) and the *evaluation mode* (teacher-forced or autoregressive). Fig. 1 summarizes the resulting design space that we explore in this paper.

3.2 Scoring Rule Axis: From Log-Likelihood to CRPS

Recovering the standard ELBO (`s-NLL`). The simplest instantiation of (4) uses the Gaussian negative log-likelihood on one-step transitions as the loss ℓ. With $H = 1$ and $\lambda = 1/N$, this directly

recovers the standard ELBO for VBLLs [32] (*cf*. Appendix E.1), to which we refer as `s-NLL`. This baseline anchors the design space: every other objective we develop can be understood as relaxing one or both of the choices that `s-NLL` imposes.

Extending to multi-step scoring (`m-NLL`). The standard `s-NLL` trains on isolated transitions, but at deployment the model is rolled out over many steps. This creates a mismatch: the posterior over $\boldsymbol{W}$ is shaped by features at individual training states, yet never exposed to the features it will encounter along extended trajectories. A natural extension is to change the training horizon from a single step to H steps, scoring the model's predictions along entire trajectory segments.

We evaluate the model in a *teacher-forced* fashion: at each step h, the prediction is conditioned on the true state $\boldsymbol{x}_h^{(n)}$. With this, the one-step conditional distribution is

$$p\big(\boldsymbol{x}_{h+1}^{(n)} \mid \boldsymbol{x}_h^{(n)}, \boldsymbol{W}\big) = \mathcal{N}\big(\boldsymbol{x}_{h+1}^{(n)};\ \boldsymbol{x}_h^{(n)} + \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h^{(n)}),\ \boldsymbol{\Sigma}_\nu\big). \tag{5}$$

Because the conditioning state is always the ground-truth observation, successive transitions are conditionally independent given $\boldsymbol{W}$. The teacher-forced trajectory log-likelihood factorizes as

$$p\big(\boldsymbol{x}_{1:H}^{(n)} \mid \boldsymbol{x}_0^{(n)}, \boldsymbol{W}\big) = \textstyle\prod_{h=0}^{H-1} p\big(\boldsymbol{x}_{h+1}^{(n)} \mid \boldsymbol{x}_h^{(n)}, \boldsymbol{W}\big) = \prod_{h=0}^{H-1} \mathcal{N}\big(\boldsymbol{\delta}_h^{(n)};\ \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h^{(n)}), \boldsymbol{\Sigma}_\nu\big), \tag{6}$$

where $\boldsymbol{\delta}_h^{(n)} := \boldsymbol{x}_{h+1}^{(n)} - \boldsymbol{x}_h^{(n)}$. The `m-NLL` loss directly follows as

$$\ell_{\texttt{m-NLL}}(\boldsymbol{W}, \boldsymbol{\tau}^{(n)}) \ = \ -\textstyle\sum_{h=0}^{H-1} \log \mathcal{N}\big(\boldsymbol{\delta}_h^{(n)};\ \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h^{(n)}),\ \boldsymbol{\Sigma}_\nu\big). \tag{7}$$

Closed-form expected loss. Since each summand in (7) is quadratic in $\boldsymbol{W}$, the closed-form formulation of the expected log-likelihood of standard VBLLs extends as a sum over the H steps. Stacking variational means into $\boldsymbol{\mu}_W = [\boldsymbol{\mu}_{w_1}, \ldots, \boldsymbol{\mu}_{w_{d_x}}] \in \mathbb{R}^{p \times d_x}$, the expectation is tractable as

$$\mathbb{E}_{q(\boldsymbol{W}\mid\boldsymbol{\eta})}\big[\ell_{\texttt{m-NLL}}(\boldsymbol{W}, \boldsymbol{\tau}^{(n)})\big] = \sum_{h=0}^{H-1} \left(-\log \mathcal{N}\big(\boldsymbol{\delta}_h^{(n)}; \boldsymbol{\mu}_W^\top \boldsymbol{\phi}_h^{(n)}, \boldsymbol{\Sigma}_\nu\big) + \frac{1}{2} \sum_{j=1}^{d_x} \frac{\boldsymbol{\phi}_h^{(n)\top} \boldsymbol{\Sigma}_{w_j} \boldsymbol{\phi}_h^{(n)}}{\sigma_j^2} \right) \tag{8}$$

where $\boldsymbol{\phi}_h^{(n)} = \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h^{(n)})$. Therefore, as with `s-NLL`, the `m-NLL` objective can be optimized without Monte Carlo estimation of the expected loss.

Multi-step training reshapes features but not the objective. Because the teacher-forced multi-step loss (7) is simply a sum of H single-step losses, the following equivalence is immediate.

Corollary 1 (Equivalence of `s-NLL` and `m-NLL` for fixed features)**.** *For fixed $\phi_{\boldsymbol{\theta}}$, if $\lambda_{\mathrm{M}} = H\,\lambda_{\mathrm{S}}$ then $\mathcal{L}_{\texttt{m-NLL}}(\boldsymbol{\eta}) = H \cdot \mathcal{L}_{\texttt{s-NLL}}(\boldsymbol{\eta})$ and the two objectives share the same set of minimizers.*

When $\phi_{\boldsymbol{\theta}}$ is learned jointly, `m-NLL` propagates gradients through features evaluated at multiple states along each trajectory segment, whereas `s-NLL` extracts signal from mini-batches of one-step transitions. Although the teacher-forced likelihood still factorizes into single-step terms, this different organization of the training data can lead to different learned features (*cf*. Appendix I.2). Moreover, as demonstrated by Lambert et al. [42], multi-step training can increase the effective amount of supervision. Here, we leverage this by reusing overlapping trajectory segments with stride s, analogous to efficiency gains obtained from overlapping windows in spectral estimation [65, 59].

Still, the corollary also hints that the gains from moving to the multi-step log-likelihood over the single-step version are inherently limited. The Gibbs VI framework offers a second, orthogonal lever: the scoring rule itself. Once the loss is no longer tied to likelihood, the question becomes what properties a desirable scoring rule should have. For long-horizon dynamics from real-world systems, a suitable scoring rule should evaluate full predictive distributions, remain robust to occasional large deviations, and admit tractable computation under Gaussian predictives.

The continuous ranked probability score. The CRPS [51, 27] satisfies these requirements. It is a strictly proper scoring rule that rewards both calibration and sharpness, but penalizes large deviations linearly. Because it is not a log-likelihood, it falls outside classical Bayesian inference; the Gibbs VI framework (3) is precisely what allows us to use it as a principled data-fit term.

For a univariate random variable $X \sim P$ and observation $y \in \mathbb{R}$, the CRPS admits the energy form

$$\mathrm{CRPS}(P, y) = \mathbb{E}_{X\sim P}\left[|X - y|\right] - \tfrac{1}{2}\mathbb{E}_{X,X'\sim P}\left[|X - X'|\right]. \tag{9}$$

Intuitively, the first term rewards accurate predictions, with a linear tail that limits the influence of large errors compared to the log score, while the second term rewards spread between samples and thus counteracts collapse while excessive spread is in turn penalized by the first term. Specifically, two properties make the CRPS attractive for dynamics models: it penalizes errors approximately linearly rather than quadratically in the tails, conferring robustness to outliers; and it admits a closed form for Gaussian predictives [28, 30, 27] (derived in Appendix H):

$$\mathrm{CRPS}\left(\mathcal{N}(\mu,\sigma^2),\, y\right) = \sigma\left[z\big(2\Phi(z)-1\big) + 2\,\varphi(z) - \frac{1}{\sqrt{\pi}}\right], \qquad z = \frac{y-\mu}{\sigma}, \tag{10}$$

where φ and Φ denote the standard normal density and CDF. From these properties, we obtain an additional useful feature: the CRPS has gradients with respect to the location and scale parameters that are uniformly bounded as functions of the standardized residual z. Applying the chain rule with $\varphi'(z) = -z\,\varphi(z)$, we obtain

$$\frac{\partial\,\ell_{\mathrm{NLL}}}{\partial\mu} = -\frac{z}{\sigma} \quad \xrightarrow{|z|\to\infty} \quad \pm\infty, \qquad \frac{\partial\ \mathrm{CRPS}}{\partial\mu} = 1 - 2\Phi(z) \quad \in (-1,1), \tag{11}$$

$$\frac{\partial\,\ell_{\mathrm{NLL}}}{\partial\sigma} = -\frac{z^2}{\sigma} + \frac{1}{\sigma} \quad \xrightarrow{|z|\to\infty} \quad -\infty, \qquad \frac{\partial\ \mathrm{CRPS}}{\partial\sigma} = 2\varphi(z) - \frac{1}{\sqrt{\pi}} \quad \in \left[-\tfrac{1}{\sqrt{\pi}},\, \tfrac{\sqrt{2}-1}{\sqrt{\pi}}\right]. \tag{12}$$

Under the NLL, a single large residual can dominate a gradient step and, via the $-z^2/\sigma$ term, inflate noise estimates to accommodate outliers. Under the CRPS, the same residual produces a bounded gradient on both μ and σ. These bounds hold at the score level and do not bound gradients through the backbone and under autoregressive evaluation, backpropagation through time additionally multiplies rollout Jacobians. Empirically, we show that CRPS attenuates gradient growth, enough to keep autoregressive training finite across all tested systems and horizons, whereas training with the log score can result in gradient norms that are orders of magnitude larger (*cf*., Sec. I.7).

3.3 Evaluation Mode Axis: From Teacher Forcing to Autoregressive Rollouts

Having introduced the scoring-rule axis, we now turn to *where* predictions are evaluated during training. The log-likelihood losses above use teacher forcing, conditioning on ground-truth states. We first combine this evaluation mode with the CRPS, and then move to autoregressive evaluation, where predictions are scored along the model's own rollout.

3.3.1 Teacher-Forced CRPS (`m-CRPS`)

Under teacher forcing, the predictive distribution at each step remains Gaussian given the factorization from (6). For trajectory n, step h, and output j, we define $P_{h,j}^{(n)}(\boldsymbol{W}) = \mathcal{N}\big(\hat{\delta}_{h,j}(\boldsymbol{W}),\ \sigma_j^2\big)$. Applying the Gaussian CRPS (10) to each marginal yields the teacher-forced CRPS loss

$$\ell_{\texttt{m-CRPS}}(\boldsymbol{W}, \boldsymbol{\tau}^{(n)}) = \textstyle\sum_{h=0}^{H-1}\sum_{j=1}^{d_x} \mathrm{CRPS}\left(P_{h,j}^{(n)}(\boldsymbol{W}),\ \delta_{h,j}^{(n)}\right). \tag{13}$$

Thus, `m-CRPS` preserves the teacher-forced structure of `m-NLL` but replaces the tail-sensitive log-score with the CRPS, making the objective less dominated by occasional large residuals while still scoring calibrated and sharp predictive distributions.

Unlike the `m-NLL`, the expected loss $\mathbb{E}_q[\ell_{\texttt{m-CRPS}}]$ does not admit a closed-form expression under the Gaussian posterior. We therefore estimate it via Monte Carlo using reparameterized sampling of $\boldsymbol{W}$ with S samples. Note that even though the training scheme is still teacher-forced, one draw of $\boldsymbol{W}$ is shared across and scored against the whole trajectory, providing informative multi-step gradients to both the feature extractor and the variational posterior.

Summary of teacher-forced losses. The two teacher-forced losses share the same structure, differing only in the scoring rule applied at each step. Following Gneiting and Raftery [27], we write $\mathrm{LogS}(P, y) = -\log p(y)$ for the logarithmic score of a distribution P with density p and have

$$\underbrace{\ell_{\texttt{m-NLL}}^{(\boldsymbol{W},\boldsymbol{\tau}^{(n)})} = \sum_{h=0}^{H-1}\sum_{j=1}^{d_x} \mathrm{LogS}\big(P_{h,j}^{(n)}(\boldsymbol{W}), \delta_{h,j}^{(n)}\big)}_{\text{Teacher-forced NLL (\texttt{m-NLL})}} \qquad \underbrace{\ell_{\texttt{m-CRPS}}^{(\boldsymbol{W},\boldsymbol{\tau}^{(n)})} = \sum_{h=0}^{H-1}\sum_{j=1}^{d_x} \mathrm{CRPS}\big(P_{h,j}^{(n)}(\boldsymbol{W}), \delta_{h,j}^{(n)}\big)}_{\text{Teacher-forced CRPS (\texttt{m-CRPS})}} \tag{14}$$

3.3.2 Autoregressive Evaluation

Teacher-forced objectives, regardless of scoring rule, evaluate predictions at ground-truth states and therefore do not expose the model to the distribution shift induced by its own errors [43]. At deployment, however, predictions are generated autoregressively. To close this gap, we construct losses that evaluate the model under its own rollout dynamics. For a fixed $\boldsymbol{W}$, the generative dynamics remain stochastic due to process noise at every step. Directly evaluating the resulting multi-step predictive distribution is generally intractable for nonlinear feature maps, as the distribution becomes non-Gaussian. We therefore adopt a tractable approximation yielding a simple and effective objective that captures compounding prediction errors while retaining a closed-form scoring rule.

Deterministic rollout. For a fixed draw $\boldsymbol{W} \sim q(\boldsymbol{W})$ and initial condition $\hat{\boldsymbol{x}}_0(\boldsymbol{W}) = \boldsymbol{x}_0^{(n)}$, define the deterministic rollout

$$\hat{\boldsymbol{x}}_{h+1}(\boldsymbol{W}) \;=\; \hat{\boldsymbol{x}}_h(\boldsymbol{W}) + \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}\big(\hat{\boldsymbol{x}}_h(\boldsymbol{W})\big), \qquad h = 0, \dots, H-1. \tag{15}$$

This recursion defines the autoregressive mean trajectory induced by the model. Because $\hat{\boldsymbol{x}}_h(\boldsymbol{W})$ depends on previous predictions, it captures the compounding errors that teacher forcing ignores.

Gaussian predictive approximation. The exact predictive distribution $P_{\boldsymbol{W},h} = p(\boldsymbol{x}_h \mid \boldsymbol{W}, \boldsymbol{x}_0)$ at horizon h is generally non-Gaussian for nonlinear feature maps, due to the repeated transformation of noise. We seek a tractable Gaussian surrogate that is exact in the linear case and well-controlled near identity. We approximate the predictive by decoupling mean propagation from noise accumulation:

$$P_{\boldsymbol{W},h} \;\approx\; \tilde{P}_{\boldsymbol{W},h} \;:=\; \mathcal{N}\big(\hat{\boldsymbol{x}}_h(\boldsymbol{W}),\; h\,\boldsymbol{\Sigma}_\nu\big). \tag{16}$$

The mean is given by the deterministic rollout (15), while the covariance grows linearly with h, reflecting cumulative process noise and neglecting its distortion by the dynamics. A similar linearly expanding Gaussian uncertainty model has been used by Ivanovic et al. [35] for multi-step trajectory prediction under heteroskedastic noise. The following proposition justifies this choice.

Proposition 1. *Suppose $\phi_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{b}$ and define $\boldsymbol{A} := \boldsymbol{I} + \boldsymbol{W}^\top \boldsymbol{\Phi}$. Then for all $h \geq 1$ the predictive distribution is Gaussian as $p(\boldsymbol{x}_h | \boldsymbol{W}, \boldsymbol{x}_0) = \mathcal{N}\big(\hat{\boldsymbol{x}}_h(\boldsymbol{W}),\, \boldsymbol{C}_h\big)$, with $\boldsymbol{C}_h := \sum_{k=0}^{h-1} \boldsymbol{A}^k \boldsymbol{\Sigma}_\nu (\boldsymbol{A}^k)^\top$. We have $\boldsymbol{C}_h = h\,\boldsymbol{\Sigma}_\nu$ whenever $\boldsymbol{A}\boldsymbol{\Sigma}_\nu\boldsymbol{A}^\top = \boldsymbol{\Sigma}_\nu$ (e.g. $\boldsymbol{A}$ orthogonal and $\boldsymbol{\Sigma}_\nu = \sigma^2\boldsymbol{I}$). Writing $\boldsymbol{A} = \boldsymbol{I} + \boldsymbol{E}$, for $h\|\boldsymbol{E}\|$ small, $\|\boldsymbol{C}_h - h\,\boldsymbol{\Sigma}_\nu\| = \mathcal{O}\big(h^2 \|\boldsymbol{E}\|\, \|\boldsymbol{\Sigma}_\nu\|\big)$.* *(Proof in Appendix G)*

In our residual formulation $\boldsymbol{x}_{h+1} = \boldsymbol{x}_h + \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h) + \boldsymbol{\nu}_h$, the transition matrix takes the form $\boldsymbol{A} = \boldsymbol{I} + \mathcal{O}(\Delta t)$. Hence, for sufficiently small time steps, the approximation operates in a near-identity regime where covariance distortion is limited. The surrogate becomes inaccurate when the transition Jacobian strongly amplifies or contracts uncertainty, when state dimensions become strongly correlated, or when the rollout distribution becomes skewed or multimodal. Importantly, the surrogate is only used to construct the training objective and all subsequent predictions still use the full stochastic nonlinear rollouts as in (1). In our results, we will further show that, also with a nonlinear feature extractor, the loss remains effective; particularly on real-world data, where key assumptions such as Gaussian noise and the Markov property are likely violated.

Autoregressive losses. Since we model outputs independently, we can write (16) as marginals $\tilde{P}^{(j)}_{\boldsymbol{W},h} = \mathcal{N}\big(\hat{x}_{h,j}(\boldsymbol{W}),\; h\,\sigma_j^2\big)$. Applying the log score and the CRPS to these marginals defines:

$$\underbrace{\ell^{(\boldsymbol{W},\boldsymbol{\tau}^{(n)})}_{\texttt{m-arNLL}} = \sum_{h=1}^{H}\sum_{j=1}^{d_x} \mathrm{LogS}\left(\tilde{P}^{(j)}_{\boldsymbol{W},h},\; x^{(n)}_{h,j}\right)}_{\text{Autoregressive NLL (}\texttt{m-arNLL}\text{)}} \qquad \underbrace{\ell^{(\boldsymbol{W},\boldsymbol{\tau}^{(n)})}_{\texttt{m-arCRPS}} = \sum_{h=1}^{H}\sum_{j=1}^{d_x} \mathrm{CRPS}\left(\tilde{P}^{(j)}_{\boldsymbol{W},h},\; x^{(n)}_{h,j}\right)}_{\text{Autoregressive CRPS (}\texttt{m-arCRPS}\text{)}} \tag{17}$$

Both losses share the same rollout and covariance surrogate. Under backpropagation through time, the unbounded gradients of the log score compound, which in our following experiments destabilizes `m-arNLL` on all chaotic systems, while `m-arCRPS` trains stably (*cf*. Appendix I.7).

Computational considerations. For a fixed $\boldsymbol{W}$, the loss is deterministic, so stochasticity only arises from sampling $\boldsymbol{W} \sim q(\boldsymbol{W})$. With S samples and horizon H, the forward cost is $\mathcal{O}(SH)$, as in teacher forcing. However, unlike teacher-forced objectives, the rollout is sequential and requires backpropagation through time, making it less parallelizable and more memory-intensive in practice.

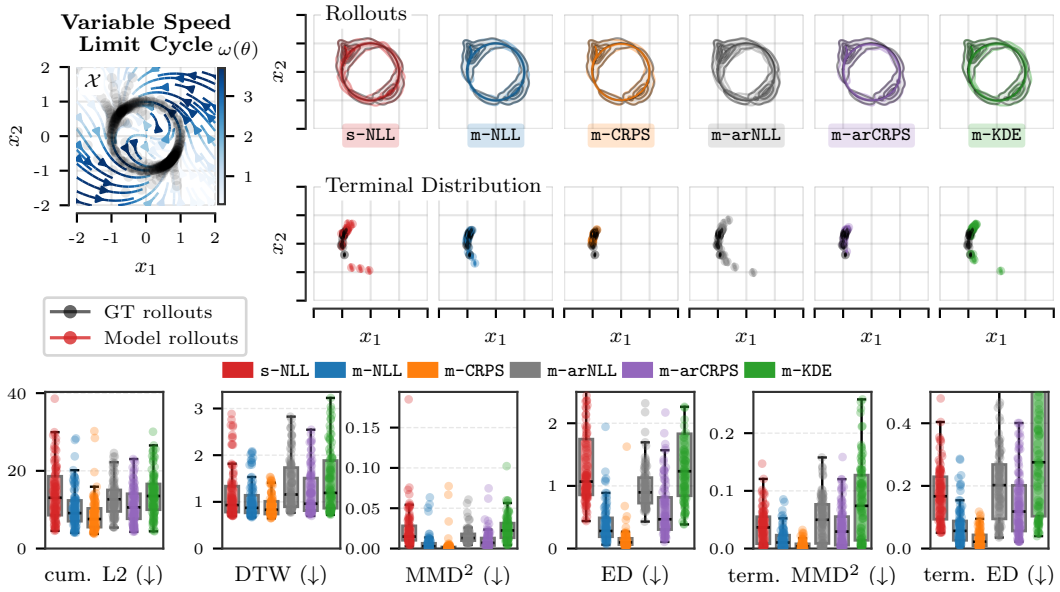


Figure 2: *Results on the VSLC. Top:* Rollouts of all baselines against rollouts (color) from the true dynamics (black). *Bottom row:* Performance across various metrics computed from rollouts from multiple initial conditions.

4 Experimental Results

We organize the evaluation into three parts. First, we use a controlled synthetic setup to isolate a failure mode of single-step training and show how multi-step objectives can address it. Second, we test all methods on a diverse benchmark of chaotic dynamical systems to validate results broadly. Third, we demonstrate real-world performance on an unstable robotic system, where long-horizon prediction is particularly challenging. We compare standard ELBO `s-NLL` [32] and autoregressive `m-KDE` [35] baselines (see Sec. 5) against teacher-forced multi-step objectives `m-NLL` and `m-CRPS`, and autoregressive `m-arNLL` and `m-arCRPS`. A longer discussion on differences between `m-KDE`, `m-arNLL`, and `m-arCRPS` in Appendix E.2. Across experiments, we vary only the loss while keeping the model fixed to control for capacity. For rollouts, we sample $\boldsymbol{W} \sim q(\boldsymbol{W})$ and simulate trajectories from an initial state using (1) with learned noise. We set $\lambda = 1/|\mathcal{D}|$ and for `s-NLL`, we use batch size $B \times H$ to ensure the same number of transitions per gradient step as multi-step methods. Full details are provided in Appendix C and all metrics used in the evaluations are listed in Appendix D.

4.1 Variable-Speed Limit Cycle (VSLC)

The VSLC is a 2D system on a globally attracting limit cycle with state-dependent angular velocity $\omega(\theta) = \omega_0(1 + \varepsilon \sin(n_{\text{modes}}\,\theta))$, where $\theta = \operatorname{atan2}(x_2, x_1)$. We set $\omega_0 = 2.0$, $\varepsilon = 0.85$, $n_{\text{modes}} = 2$ (full dynamics in Appendix C.1; flow field in Fig. 2, top left).

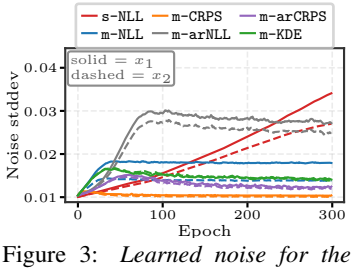


Figure 3: *Learned noise for the VSLC.* GT noise is $\sigma_1, \sigma_2 = 0.01$.

Training data sampled at fixed Δt concentrate in slow zones ($\omega(\theta) < \omega_0$), which contain $\sim$82% of samples ($F_{\text{slow}} \approx 0.823$; see Section C.1). Single-step training thus focuses gradient signal on slow-zone states, and the larger residuals in the under-represented fast regions inflate the noise estimate (Fig. 3). This is a property of the log score rather than of the single-step horizon alone as the log-score-optimal variance matches the mean squared residual, so `m-NLL` and in particular `m-arNLL`, whose autoregressive residuals grow along the rollout, inflate the noise as well. The CRPS objectives `m-CRPS` and `m-arCRPS`, whose penalty grows only linearly in large residuals, recover noise estimates close to the ground truth and sample distributions matching the true distribution (Fig. 2). `m-KDE` occasionally produces divergent samples near sample boundaries (likely due to the LSE loss; *cf*. Appendix E.2), reflected in its larger energy distance. Overall, `m-CRPS` shows the strongest performance.

4.2 Benchmark on Chaotic Dynamical Systems

After evaluating the loss functions on an environment specifically designed to highlight the potential upsides of multi-step loss function also for Markov environments, we next evaluate all methods on a broad set of benchmark of 18 dynamical systems of varying complexity, including limit cycles, an environment to test multimodal rollouts (swimmer, *cf*. Appendix C.2), and 13 chaotic attractors from the `dysts` library [26]. All of these are autonomous systems and satisfy the Markov property. We generate data using rollouts under Gaussian process noise to match the setting in Sec. 2. Figure 4 summarizes the aggregate rank across all systems and metrics. Full per-system results are provided in Section I and Tables 3–6.

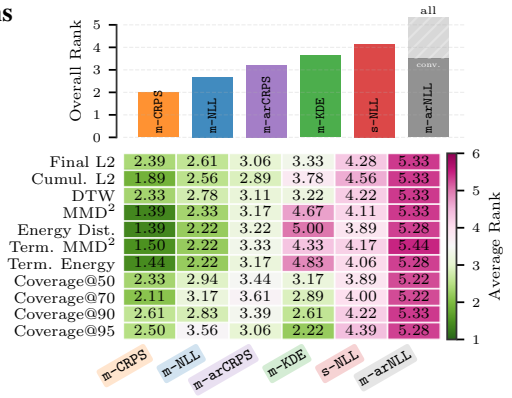


Figure 4: *Rank summary (↓) across benchmark systems.* Full results in Tab. 3 in Appendix I.

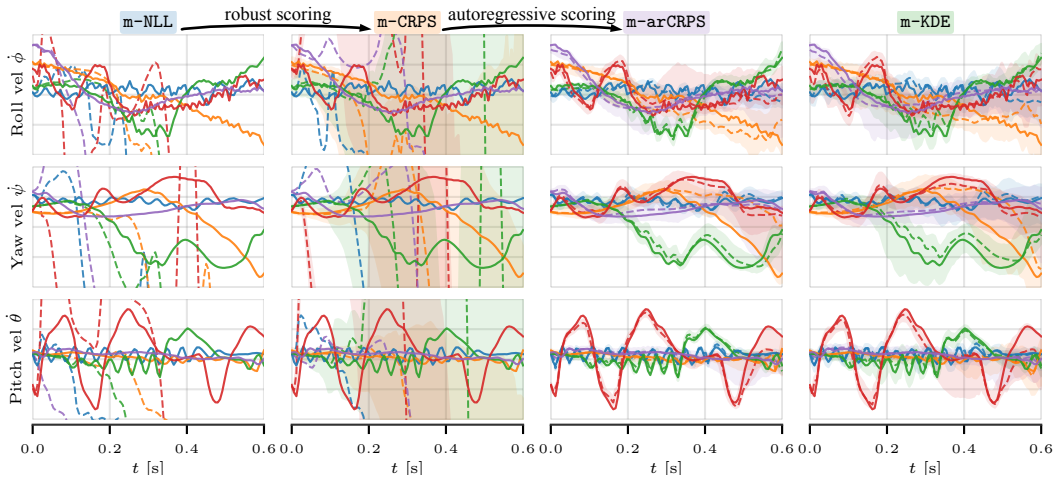
We can observe that over this large set of experiments, all stably trained multi-step loss functions outperform the single-step loss. Similar to the VSLC, the CRPS losses outperform their likelihood-based counter parts. `m-CRPS` demonstrates strong performance across the different metrics. We can further observe that while `m-KDE` shows strong results in terms of coverage, it struggles to match the distributions of the rollout, specifically in terms of energy distance where occasional diverging samples are strongly penalized. This is consistent with the impressions from the VSLC results. The matched `m-arNLL`, which differs from `m-arCRPS` only in the scoring rule, develops non-finite gradients on all chaotic systems. Hence, autoregressive exposure alone is not sufficient: the robustness of the CRPS is what keeps autoregressive training finite (*cf*. Appendix I.7).

4.3 Evaluation on Data From a Real-World Unstable Robotic System

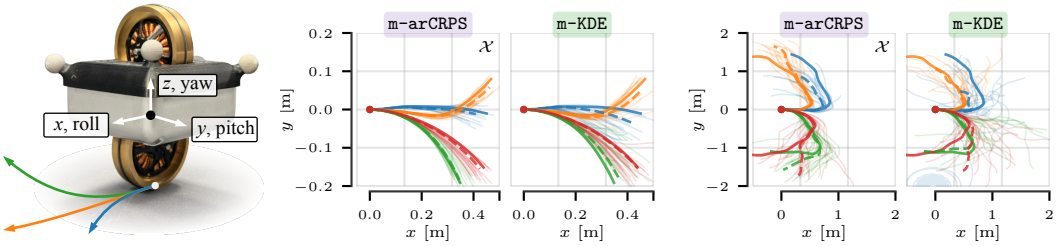
We test on a dataset [34] recorded on the Mini Wheelbot, a balancing unicycle robot [33]. Here, key assumptions (e.g., Gaussian noise, fully Markovian) likely break and the fast, nonlinear dynamics make modeling difficult (Appendix C.3). Table 1 reveals a clear gap: strong one-step accuracy does not imply stable multi-step rollouts. Teacher-forced losses perform best on one-step metrics (best is `m-CRPS`), yet degrade or diverge over long horizons. Figure 5 confirms this trend, *i.e*. autoregressive losses are more robust. In particular, `m-arCRPS` yields stable, accurate trajectories up to 700 steps (trained with $H = 40$), while `m-KDE` shows more chaotic, diverging behavior of some realizations. Model-size ablations (Appendix I.3) show `m-arCRPS` consistently outperforming `m-KDE`. Notably, a small `m-arCRPS` model matches an `m-KDE` model with $70\times$ more parameters. Fig. 11 further shows `m-arNLL` and `m-KDE` diverging beyond $H = 40$ and $H = 60$, respectively, while `m-arCRPS` remains stable up to the maximum horizon tested of $H = 100$.

5 Related Work

Probabilistic dynamics models. Probabilistic dynamics models are central to model-based control [14, 10, 36], and the choice of model class trades off uncertainty quality, computational cost, and ease of training. Gaussian processes provide a principled function-space posterior and strong uncertainty estimates in the low-data regime [14], but generally require approximations as dataset size and dimensionality grow. Probabilistic ensembles [40] remain widely used in model-based RL [10, 36, 20], but they require training and evaluating multiple networks. Fully Bayesian neural networks, *e.g*. via MC-dropout [23, 24], represent uncertainty throughout the network, but they require approximate inference over a high-dimensional parameter space. This makes them considerably more expensive, especially when training on autoregressive rollouts. BLL models [58, 54, 63, 19, 32, 38, 12] offer a practical middle ground. They share a single deterministic feature extractor and place uncertainty only over the final linear weights, which yields inexpensive epistemic uncertainty and closed-form one-step predictive moments. Moreover, holding a weight sample fixed over a rollout corresponds to one consistent dynamics hypothesis along the trajectory. These properties have made BLLs increasingly popular for uncertainty-aware learning and control, including online adaptation via Kalman filtering or recursive least squares [31, 35, 46, 11], safe and uncertainty-aware predictive control [47, 13],



(a) *Autoregressive rollouts (100 steps) for the Mini Wheelbot on different trajectory segments.* The solid line denotes the ground truth, the dashed line the mean rollout, and the shaded region the 5th–95th percentile range of the samples. Only a subset of states is shown; the full set is provided in Fig. 17 in the Appendix.



(b) The Mini Wheelbot (c) Rollouts in xy-plane (100 steps). (d) Rollouts in xy-plane (700 steps).

Figure 5: *Qualitative results on data from the Mini Wheelbot.*

Table 1: *Quantitative results on Wheelbot data.* Multi-step metrics are reported at 100 steps. Each cell reports $\text{median}_{\text{q25}}^{\text{q75}}$. Bold, shaded values are significantly best (paired Wilcoxon signed-rank test, $\alpha = 0.05$).

	One-step metrics			Multi-step metrics				Training[1]
Method	MSE ↓	NLL ↓	CRPS ↓	Final L2 ↓	Cumul. L2 ↓	DTW ↓	Energy dist. ↓	Time [min]
s-NLL [32, 7]	$0.005^{0.019}_{0.002}$	$-8.273^{-6.084}_{-8.932}$	$0.328^{0.509}_{0.273}$	$0.371^{0.759}_{0.223}$	$12.690^{20.954}_{8.285}$	$2.194^{3.919}_{1.161}$	$1.909^{2.314}_{1.561}$	1.366
m-NLL	$0.005^{0.019}_{0.002}$	$-8.304^{-6.071}_{-8.982}$	$0.340^{0.515}_{0.286}$	$0.403^{0.722}_{0.217}$	$13.350^{22.801}_{7.982}$	$2.228^{3.848}_{1.089}$	$5.181^{8.139}_{2.879}$	1.861
m-CRPS	$\mathbf{0.005^{0.017}_{0.001}}$	$\mathbf{-10.676^{-5.489}_{-12.260}}$	$\mathbf{0.248^{0.479}_{0.169}}$	$0.200^{0.259}_{0.131}$	$7.792^{11.838}_{5.540}$	$1.117^{1.582}_{0.767}$	$1.456^{1.931}_{1.148}$	2.048
m-KDE [35]	$0.009^{0.034}_{0.003}$	$-8.249^{1.104}_{-11.956}$	$0.352^{0.707}_{0.197}$	$0.026^{0.044}_{0.017}$	$1.076^{1.776}_{0.678}$	$0.087^{0.123}_{0.043}$	$0.236^{0.285}_{0.190}$	3.725
m-arNLL	$0.010^{0.039}_{0.003}$	$-8.011^{5.626}_{-11.677}$	$0.363^{0.787}_{0.197}$	$0.031^{0.045}_{0.020}$	$1.111^{1.728}_{0.777}$	$0.089^{0.141}_{0.053}$	$0.223^{0.285}_{0.190}$	3.933
m-arCRPS	$0.010^{0.040}_{0.003}$	$-0.663^{43.693}_{-12.089}$	$0.411^{0.875}_{0.194}$	$\mathbf{0.021^{0.031}_{0.015}}$	$\mathbf{0.862^{1.403}_{0.573}}$	$\mathbf{0.065^{0.099}_{0.037}}$	$\mathbf{0.159^{0.224}_{0.126}}$	3.921

Bayesian optimization [6], and autonomous racing [7]. In this paper, we do not claim that BLLs dominate alternative probabilistic models in all settings, nor that they are preferable when only a mean prediction is needed; multi-step training of deterministic models is already well studied (see below). Rather, we ask how this increasingly used model class should be trained when it is deployed autoregressively. To this end, we build on VBLLs [32] but reinterpret them through Gibbs variational inference [2, 5, 37], which decouples the posterior from the training loss and enables task-aligned, multi-step objectives.

Multi-step losses. Training dynamics models with multi-step losses aims to reduce compounding errors at deployment [43], and has been shown to improve long-horizon accuracy and sample efficiency in model-based RL [42]. Doerr et al. [15] argue that multi-step objectives become necessary when the Markov property is violated or when process and observation noise are significant. We further show that one-step losses can fail even in controlled Markovian environments. For Gaussian processes, moment matching and linearization-based uncertainty propagation can enable multi-step training [14, 15]. For overparameterized neural networks, an informative multi-step training signal is all the more important for learning reliable features. For deterministic neural network models,

[1] All baselines were trained on the same HPC cluster, each on a single NVIDIA H100 GPU.

extending the horizon yields mixed results: short horizons can help, but longer ones may degrade downstream performance [50, 8], which motivates careful per-step weighting [3]. For probabilistic ensembles, Levy et al. [45] use mean propagation over short horizons as consistent rollouts are non-trivial given the ensemble structure [21]. For VBLLs, by contrast, a consistent rollout is simply a draw of $\boldsymbol{W}$. Closest to our setting, Ivanovic et al. [35] propose a kernel-density multi-step objective for BLLs, which we adopt as our `m-KDE` baseline. It scores a marginal mixture over sampled rollouts, whereas our objectives score each weight sample inside the Gibbs VI expectation. Our losses build on the CRPS, a strictly proper scoring rule [51, 27] whose linear tail behavior makes it robust to outliers [57]. The CRPS is widely used in weather forecasting [27, 53, 44], but usually to score marginal predictive distributions directly. In this standard form it does not fit the Gibbs VI framework, since draws of $\boldsymbol{W}$ are not scored independently. While these individual ingredients are established, to the best of our knowledge we are the first to combine them into multi-step CRPS-based objectives for VBLLs, grounded in Gibbs VI to facilitate stable training of autoregressive dynamics models.

Objective mismatch and decision-aware model learning. Minimizing prediction error does not necessarily lead to optimal downstream task performance [41], motivating decision-aware model learning approaches [64]. Value-aware approaches incorporate the value function into the model loss [18, 17, 61], policy-aware approaches align it with the policy gradient [1, 16], and goal-aware prediction conditions the model on the goal [52]. These methods typically operate in a deterministic setting and couple the model loss directly to policy optimization. We share the motivation but remain in the probabilistic prediction setting, using Gibbs VI to replace the likelihood with losses aligned with how the model is deployed. Since Gibbs VI accepts any pointwise loss, decision-aware reweightings, *e.g.* scaling per-step CRPS terms by value-gradient magnitudes, can be incorporated directly, a promising direction for future work. A more extensive discussion is provided in Appendix F.

6 Discussion on Limitations and Future Work

While our experiments demonstrate the clear benefits of objectives within the Gibbs VI framework, several limitations remain that also show promising directions for future work.

Calibration. Although the CRPS-based losses consistently improve long-horizon accuracy and distributional match, the empirical coverage of `m-arCRPS` tends to lag behind that of `m-KDE` on several benchmark systems (*cf*. Table 3–6). A key advantage of the Gibbs VI perspective is that the loss is a free design choice and calibration techniques from standard VI or tuning the regularization strength can be combined with our multi-step objectives without modifying the model or posterior family. In particular, increasing the KL weight can improve coverage (see Appendix I.6), and we recommend selecting it on a held-out calibration set when coverage matters. Going one step further, Generalized VI [37] also generalizes the divergence enabling, *e.g.* robust divergences [48, 22]. Since meaningful calibration gains can already be obtained by tuning the existing regularization, we view the KL weight as the first lever, while alternative divergences remain an interesting future direction.

Empirical evaluation on downstream tasks. We deliberately isolated the effect of the training objective by holding the model architecture, posterior family, and optimizer fixed, attributing all observed gains unambiguously to the loss. Consequently, we do not provide a direct comparison against other model classes such as probabilistic ensembles or GPs. We expect downstream tasks that rely on autoregressive rollouts under learned models to benefit the most, *e.g.* sampling-based MPC [66] or uncertainty-aware reinforcement learning [14, 36, 20]. We therefore view systematic evaluation in closed-loop decision-making as an important direction for future work.

7 Conclusion

By reinterpreting VBLLs through Gibbs variational inference, we showed that the training loss is a modular design choice—independent of the model architecture and posterior family—that can be tailored to the deployment regime. Our hierarchy of multi-step objectives reveals two complementary axes of improvement: CRPS-based scoring confers robustness to outliers and yields superior distributional fit, while autoregressive evaluation closes the train–test mismatch that causes teacher-forced models to degrade or diverge on real-world systems. Together, these results establish that how a VBLL is trained matters as much as what it models, opening a direct path toward integrating task-aligned losses with downstream decision-making in control and reinforcement learning.

Acknowledgments and Disclosure of Funding

The authors thank James Harrison, Mikkel Jordahn, John Willes, and Jasper Snoek for various insightful discussions on VBLLs over the last years. Paul Brunzema and Henrik Hose are partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)–RTG 2236/2 (UnRAVeL). This work was additionally supported in part by the DFG under project number 556142469 (ParDyBO). Simulations were performed in part with computing resources granted by RWTH Aachen University under projects rwth2066 and p0027430.

References

[1] Romina Abachi. *Policy-aware model learning for policy gradient methods*. University of Toronto, 2020.

[2] Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of Gibbs posteriors. *Journal of Machine Learning Research*, 17(236):1–41, 2016.

[3] Abdelhakim Benechehab, Albert Thomas, Giuseppe Paolo, Maurizio Filippone, and Balázs Kégl. A multi-step loss function for robust learning of the dynamics in model-based reinforcement learning. *arXiv preprint arXiv:2402.03146*, 2024.

[4] Jonathan Berrisch and Florian Ziel. Crps learning. *Journal of Econometrics*, 237(2):105221, 2023.

[5] Pier Giovanni Bissiri, Chris C Holmes, and Stephen G Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 78(5):1103–1130, 2016.

[6] Paul Brunzema, Mikkel Jordahn, John Willes, Sebastian Trimpe, Jasper Snoek, and James Harrison. Bayesian optimization via continual variational last layer training. In *International Conference on Learning Representations (ICLR)*, 2025.

[7] Paul Brunzema, Thomas Lew, Ray Zhang, Takeru Shirasawa, John Subosits, and Marcus Greiff. Vision-conditioned variational bayesian last layer dynamics models. *arXiv preprint arXiv:2601.09178*, 2026.

[8] Arunkumar Byravan, Leonard Hasenclever, Piotr Trochim, Mehdi Mirza, Alessandro Davide Ialongo, Yuval Tassa, Jost Tobias Springenberg, Abbas Abdolmaleki, Nicolas Heess, Josh Merel, et al. Evaluating model-based planning and planner amortization for continuous control. *arXiv preprint arXiv:2110.03363*, 2021.

[9] Olivier Catoni. PAC-Bayesian supervised classification: the thermodynamics of statistical learning. *arXiv preprint arXiv:0712.0248*, 2007.

[10] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[11] Alexander Davydov, Franck Djeumou, Marcus Greiff, Makoto Suminaka, Michael Thompson, John Subosits, and Thomas Lew. First, learn what you don't know: Active information gathering for driving at the limits of handling. *IEEE Robotics and Automation Letters*, 2025.

[12] Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux – effortless Bayesian deep learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 20089–20103, 2021.

[13] Laura Boca de Giuli, Alessio La Bella, Manish Prajapat, Johannes Köhler, Anna Scampicchio, Riccardo Scattolini, and Melanie Zeilinger. Goal-oriented safe active learning for predictive control using Bayesian recurrent neural networks. *arXiv preprint arXiv:2604.12542*, 2026.

[14] Marc Peter Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, pages 465–472, 2011.

[15] Andreas Doerr, Christian Daniel, Duy Nguyen-Tuong, Alonso Marco, Stefan Schaal, Toussaint Marc, and Sebastian Trimpe. Optimizing long-term predictions for model-based policy search. In *Conference on Robot Learning*. PMLR, 2017.

[16] Pierluca D'Oro, Alberto Maria Metelli, Andrea Tirinzoni, Matteo Papini, and Marcello Restelli. Gradient-aware model-based policy search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3801–3808, 2020.

[17] Amir-massoud Farahmand. Iterative value-aware model learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[18] Amir-massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-aware loss function for model-based reinforcement learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1486–1494. PMLR, 2017.

[19] Felix Fiedler and Sergio Lucia. Improved uncertainty quantification for neural networks with Bayesian last layer. *IEEE Access*, 2023.

[20] Bernd Frauenknecht, Artur Eisele, Devdutt Subhasish, Friedrich Solowjow, and Sebastian Trimpe. Trust the model where it trusts itself – model-based actor-critic with uncertainty-aware rollout adaption. In *International Conference on Machine Learning (ICML)*, pages 13973–14005. PMLR, 2024.

[21] Bernd Frauenknecht, Devdutt Subhasish, Friedrich Solowjow, and Sebastian Trimpe. On rollouts in model-based reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2025.

[22] Futoshi Futami, Issei Sato, and Masashi Sugiyama. Variational inference based on robust divergences. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2018.

[23] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1050–1059. PMLR, 2016.

[24] Yarin Gal, Rowan McAllister, and Carl Edward Rasmussen. Improving PILCO with Bayesian neural network dynamics models. In *Data-efficient machine learning workshop, ICML*, volume 4, page 25, 2016.

[25] A René Geist, Jonathan Fiene, Naomi Tashiro, Zheng Jia, and Sebastian Trimpe. The wheelbot: A jumping reaction wheel unicycle. *IEEE Robotics and Automation Letters*, 7(4):9683–9690, 2022.

[26] William Gilpin. Chaos as an interpretable benchmark for forecasting and data-driven modelling. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[27] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[28] Tilmann Gneiting, Adrian E Raftery, Anton H Westveld III, and Tom Goldman. Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation. *Monthly weather review*, 133(5):1098–1118, 2005.

[29] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(1):723–773, 2012.

[30] Eric P Grimit, Tilmann Gneiting, Veronica J Berrocal, and Nicholas A Johnson. The continuous ranked probability score for circular variables and its application to mesoscale forecast ensemble verification. *Quarterly Journal of the Royal Meteorological Society*, 132(621C):2925–2942, 2006.

[31] James Harrison, Apoorva Sharma, and Marco Pavone. Meta-learning priors for efficient online Bayesian regression. In *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2018.

[32] James Harrison, John Willes, and Jasper Snoek. Variational bayesian last layers. In *International Conference on Learning Representations (ICLR)*, 2024.

[33] Henrik Hose, Jan Weisgerber, and Sebastian Trimpe. The mini wheelbot: A testbed for learning-based balancing, flips, and articulated driving. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1339–1346. IEEE, 2025.

[34] Henrik Hose, Paul Brunzema, Devdutt Subhasish, and Sebastian Trimpe. The Mini Wheelbot dataset: High-fidelity data for robot learning. *arXiv preprint arXiv:2601.11394*, 2026.

[35] Boris Ivanovic, James Harrison, and Marco Pavone. Expanding the deployment envelope of behavior prediction via adaptive meta-learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[36] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[37] Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. An optimization-centric view on Bayes' rule: Reviewing and generalizing variational inference. *Journal of Machine Learning Research*, 23(132):1–109, 2022.

[38] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being Bayesian, even just a bit, fixes overconfidence in ReLU networks. In *International Conference on Machine Learning*, pages 5436–5446, 2020.

[39] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[40] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. 2017.

[41] Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In *Learning for Dynamics and Control*. PMLR, 2020.

[42] Nathan Lambert, Albert Wilcox, Howard Zhang, Kristofer SJ Pister, and Roberto Calandra. Learning accurate long-term dynamics for model-based reinforcement learning. In *2021 60th IEEE Conference on decision and control (CDC)*. IEEE, 2021.

[43] Nathan Lambert, Kristofer Pister, and Roberto Calandra. Investigating compounding prediction errors in learned dynamics models. *arXiv preprint arXiv:2203.09637*, 2022.

[44] Simon Lang, Mihai Alexe, Mariana CA Clare, Christopher Roberts, Rilwan Adewoyin, Zied Ben Bouallègue, Matthew Chantry, Jesper Dramsch, Peter D Dueben, Sara Hahner, et al. Aifs-crps: ensemble forecasting using a model trained with a loss function based on the continuous ranked probability score. *npj Artificial Intelligence*, 2(1):18, 2026.

[45] Jacob Levy, Tyler Westenbroek, and David Fridovich-Keil. Learning to walk from three minutes of real-world data with semi-structured dynamics models. In *Conference on Robot Learning*, 2024.

[46] Jacob Levy, Jason Gibson, Bogdan Vlahov, Erica Tevere, Evangelos Theodorou, David Fridovich-Keil, and Patrick Spieler. Meta-learning online dynamics model adaptation in off-road autonomous driving. *Robotics: Science and Systems*, 2025.

[47] Thomas Lew, Apoorva Sharma, James Harrison, Andrew Bylard, and Marco Pavone. Safe active dynamics learning and control: A sequential exploration–exploitation framework. *IEEE Transactions on Robotics*, 38(5):2888–2907, 2022.

[48] Yingzhen Li and Richard E Turner. Rényi divergence variational inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[49] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[50] Michael Lutter, Leonard Hasenclever, Arunkumar Byravan, Gabriel Dulac-Arnold, Piotr Trochim, Nicolas Heess, Josh Merel, and Yuval Tassa. Learning dynamics models for model predictive agents. *arXiv preprint arXiv:2109.14311*, 2021.

[51] James E Matheson and Robert L Winkler. Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096, 1976.

[52] Suraj Nair, Silvio Savarese, and Chelsea Finn. Goal-aware prediction: Learning to model what matters. In *International Conference on Machine Learning*, pages 7207–7219. PMLR, 2020.

[53] Lorenzo Pacchiardi, Sherman Khoo, and Ritabrata Dutta. Generalized bayesian likelihood-free inference. *Electronic Journal of Statistics*, 18(2):3628–3686, 2024.

[54] Carlos Riquelme, George Tucker, and Jasper Snoek. Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for thompson sampling. In *International Conference on Learning Representations (ICLR)*, 2018.

[55] Yusie Rizal, Trihastuti Agustinah, Rudy Dikairono, and Haiping Du. Design, modeling, and control of unicycle robots: A systematic literature review. *IEEE Access*, 14:20570–20588, 2026.

[56] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

[57] Reinhard Selten. Axiomatic characterization of the quadratic scoring rule. *Experimental Economics*, 1(1):43–61, 1998.

[58] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable Bayesian optimization using deep neural networks. In *International Conference on Machine Learning (ICML)*, 2015.

[59] Petre Stoica and Randolph L. Moses. *Spectral Analysis of Signals*. Prentice Hall, Upper Saddle River, NJ, 2005.

[60] Gábor J Székely and Maria L Rizzo. Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249–1272, 2013.

[61] Claas A Voelcker, Victor Liao, Animesh Garg, and Amir-massoud Farahmand. Value gradient weighted model-based reinforcement learning. In *International Conference on Learning Representations*, 2022.

[62] wannesm, khendrickx, Aras Yurtman, Pieter Robberechts, Dany Vohl, Eric Ma, Gust Verbruggen, Marco Rossi, Mazhar Shaikh, Muhammad Yasirroni, Todd, Wojciech Zieliński, Toon Van Craenendonck, and Sai Wu. wannesm/dtaidistance: v2.3.5, January 2022. URL https://doi.org/10.5281/zenodo.5901139.

[63] Joe Watson, Jihao Andreas Lin, Pascal Klink, Joni Pajarinen, and Jan Peters. Latent derivative Bayesian last layer networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

[64] Ran Wei, Nathan Lambert, Anthony McDonald, Alfredo Garcia, and Roberto Calandra. A unified view on solving objective mismatch in model-based reinforcement learning. *arXiv preprint arXiv:2310.06253*, 2023.

[65] Peter D. Welch. The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2):70–73, 1967.

[66] Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M Rehg, Byron Boots, and Evangelos A Theodorou. Information theoretic MPC for model-based reinforcement learning. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

A Broader Impact

This work develops general-purpose training objectives for probabilistic dynamics models. Improved long-horizon predictions and better-calibrated uncertainty estimates are directly relevant to safety-critical applications such as autonomous driving or robotic manipulation. In such settings, wrong long-term prediction by models can lead to unsafe control actions, and our multi-step, CRPS-based objectives help mitigate this risk by improving distributional fit and reducing the train–test mismatch inherent in single-step training. However, no training objective alone can guarantee safe deployment; careful system-level validation, domain-specific safety constraints, and rigorous uncertainty monitoring remain essential when integrating learned dynamics models into closed-loop control. We do not foresee direct negative societal consequences of this work beyond those generally associated with improved automation capabilities.

B On the Parameterization and Implementation of VBLLs

We follow closely the original VBLL codebase of Harrison et al. [32] and re-implement it in JAX using Flax NNX and Optax[2]. This appendix details the concrete parameterization of the variational posterior covariance matrices and the process-noise covariance, both of which require some care to ensure that the resulting matrices remain positive definite throughout optimization.

Cholesky parameterization of the posterior covariance. As stated in (2), the variational posterior over each column $\boldsymbol{w}_j$ of $\boldsymbol{W}$ is $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ with $\boldsymbol{\Sigma}_j = \boldsymbol{L}_j \boldsymbol{L}_j^\top$, where $\boldsymbol{L}_j \in \mathbb{R}^{p\times p}$ is lower triangular with positive diagonal. To enforce this structure during unconstrained gradient-based optimization, we store two sets of free parameters per output dimension j: a vector $\boldsymbol{d}_j \in \mathbb{R}^p$ of unconstrained log-diagonal entries and a matrix $\boldsymbol{O}_j \in \mathbb{R}^{p\times p}$ of unconstrained off-diagonal entries. The Cholesky factor is then assembled as

$$\boldsymbol{L}_j = \mathrm{tril}(\boldsymbol{O}_j, -1) + \mathrm{diag}\big(\exp(\boldsymbol{d}_j)\big), \tag{18}$$

where $\mathrm{tril}(\cdot, -1)$ extracts the strictly lower-triangular part. The exponential map on the diagonal guarantees $(\boldsymbol{L}_j)_{kk} > 0$ for all k, which in turn ensures that $\boldsymbol{\Sigma}_j = \boldsymbol{L}_j \boldsymbol{L}_j^\top$ is positive definite for any value of the free parameters. We initialize $\boldsymbol{d}_j$ near $-\frac{1}{2}\log p$ and both $\boldsymbol{O}_j$ and $\boldsymbol{\mu}_j$ with small random values, so that the posterior starts concentrated near zero with low uncertainty. For all losses, we always use the same initialization.

Process-noise parameterization. The diagonal process-noise covariance $\boldsymbol{\Sigma}_\nu = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_{d_x}^2)$ is parameterized through unconstrained log-standard-deviations $\lambda_j \in \mathbb{R}$, so that $\sigma_j = \exp(\lambda_j)$. This again ensures positivity without constrained optimization. The parameters λ_j are initialized at $\log(\sigma_{\text{init}})$ for a user-specified $\sigma_{\text{init}} > 0$ (we use $\sigma_{\text{init}} = 0.01$ by default), reflecting the prior belief that process noise is initially small.

C Experimental Details

In the following, we will list the experimental details for the empirical studies and introduce the system dynamics where necessary. We first list general settings, *i.e.* the model architecture and the optimizer. All code is publicly available under the MIT license at `https://github.com/brunzema/gibbs-vbll`.

Model architecture As a backbone we use a standard feed forward neural network with three layers and an additional layer that maps to the features. In Tab. 7 and Tab. 10 we list the parameters for the specific set of experiments.

Optimizer and training For all experiments, we employ the AdamW optimizer [49] with a learning rate of 1×10^{-3} (5×10^{-4} for the Mini Wheelbot experiments in Sec. 4.3). We apply a weight decay of 1×10^{-4} exclusively to the backbone, following standard practice for VBLLs, as the final layer is regularized through the KL term in the loss [32]. In addition, we use gradient clipping with a maximum norm of one in all experiments. We train for 300 epochs with a batch size of 64 on the

[2]Flax NNX and Optax are both publicly available under a Apache-2.0 license.

VSLC dataset and the chaotic systems benchmark. For the Mini Wheelbot experiments, we train for 400 epochs with a batch size of 256. We choose these training durations to ensure sufficient convergence of each loss function and select the best model state based on a validation set, evaluating performance every 5 epochs. More details in Tab. 9 and Tab. 11.

C.1 Variable-Speed Limit Cycle

The VSLC state $\boldsymbol{x} = (x_1, x_2)^\top \in \mathbb{R}^2$ evolves according to

$$\dot{x}_1 = \mu(R - r)\tfrac{x_1}{r} - \omega(\theta)\, x_2, \qquad \dot{x}_2 = \mu(R - r)\tfrac{x_2}{r} + \omega(\theta)\, x_1, \tag{19}$$

with $r = \|\boldsymbol{x}\|_2$ and $\theta = \operatorname{atan2}(x_2, x_1)$. The parameter $\mu > 0$ governs radial attraction toward the limit cycle of radius R, while $\omega(\theta)$ controls angular motion. Under the parameterization in Sec. 4.1, the angular velocity ranges over $\omega(\theta) \in [\omega_0(1-\varepsilon),\ \omega_0(1+\varepsilon)]$, which evaluates to $[0.30,\ 3.70]$ rad/s for the chosen parameters. Training data collected at a fixed time step Δt induce a distribution over angles that is proportional to the time spent at each state. Since angular speed is $\omega(\theta)$, this yields a density proportional to $\omega(\theta)^{-1}$.

Defining the slow zone by $\omega(\theta) < \omega_0$, the fraction of samples originating from that region is

$$F_{\text{slow}} = \frac{\displaystyle\int_{\{\theta:\, \omega(\theta) < \omega_0\}} \omega(\theta)^{-1}\, d\theta}{\displaystyle\int_0^{2\pi} \omega(\theta)^{-1}\, d\theta}. \tag{20}$$

Using $\omega(\theta) = \omega_0\big(1 + \varepsilon \sin(n_{\text{modes}}\theta)\big)$ and the substitution $\phi = n_{\text{modes}}\theta$, we obtain

$$F_{\text{slow}} = \frac{\displaystyle\int_{\{\phi:\, \sin\phi < 0\}} \frac{d\phi}{1 + \varepsilon \sin\phi}}{\displaystyle\int_0^{2\pi} \frac{d\phi}{1 + \varepsilon \sin\phi}}. \tag{21}$$

This expression is independent of n_{modes} (for integer n_{modes}). Evaluating the integrals yields the closed-form result

$$F_{\text{slow}} = \frac{1}{2} + \frac{\arcsin(\varepsilon)}{\pi}. \tag{22}$$

For our parameter setting of $\varepsilon = 0.85$, this gives $F_{\text{slow}} \approx 0.823$. Thus, approximately 82.3% of all training samples lie in the slow zone.

Parameters and data generation. We use the parameter values $\omega_0 = 2.0$, $\varepsilon = 0.85$, $n_{\text{modes}} = 2$. Trajectory data are generated by numerically integrating the system with sampling time $\Delta t = 0.1\,\text{s}$ and induce process noise on all the states with $\sigma_1 = \sigma_2 = 0.01$. We generate a total of 300 trajectories with each having a length of 90 time steps from pseudo-random initial conditions within $x, y \in [-2, 2]$ using a Sobol sequence. $20\,\%$ of trajectories are used as validation data for all the methods to determine the best performing model state which is then used for testing on newly random generated trajectories. For the multi-step losses, we use a horizon of $H = 40$ and a stride of one to effectively increase the supervision signal.

C.2 Swimmer

The Swimmer environment is a two-dimensional continuous-time dynamical system that combines a spatially varying tidal-drag field with a localized vortex flow. The state is $\boldsymbol{x} = (x, y)^\top \in \mathbb{R}^2$, and its dynamics are given by

$$\dot{\boldsymbol{x}} = \boldsymbol{v}_{\text{tidal}}(\boldsymbol{x}) + \boldsymbol{v}_{\text{vortex}}(\boldsymbol{x}). \tag{23}$$

Tidal-drag component. The tidal field is defined through an oscillatory base flow

$$\boldsymbol{c}_0(\boldsymbol{x}) = \begin{pmatrix} U_0 \sin(\kappa y) \\ V_0 \sin(\kappa x) \end{pmatrix}, \tag{24}$$

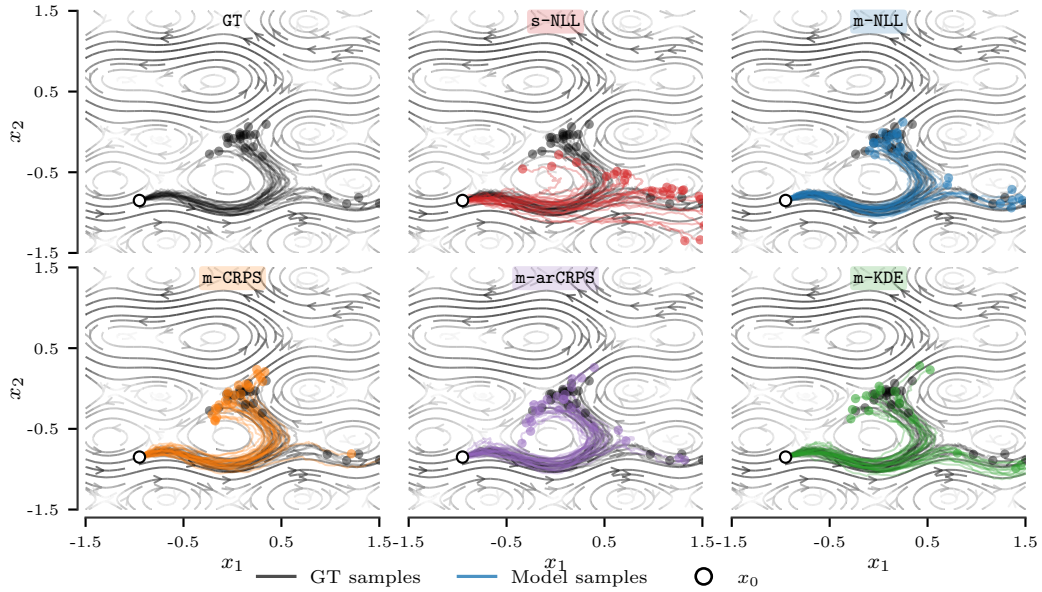


Figure 6: *Model rollouts on Swimmer (flow field in gray).* We test whether the different losses allow the model to capture complex nonlinear dynamics resulting in multi-modal rollouts. We can see that all losses allow the model demonstrate such complex behavior. Still, multi-step losses allow for a better approximation of the distribution.

with fixed amplitudes $U_0 = 0.4$ and $V_0 = 0.2$. To attenuate the flow in regions of high instantaneous magnitude, this base field is scaled by a damping factor

$$\boldsymbol{v}_{\text{tidal}}(\boldsymbol{x}) = \frac{\boldsymbol{c}_0(\boldsymbol{x})}{1 + \eta \|\boldsymbol{c}_0(\boldsymbol{x})\|_2^2}, \tag{25}$$

where $\eta > 0$ controls the strength of the damping.

Vortex component. The vortex is centered at $\boldsymbol{c} = (-0.5,\, 0.5)^\top$ with fixed radius parameter $R_{\text{v}} = 0.5$. Writing $\boldsymbol{d} = \boldsymbol{x} - \boldsymbol{c} = (d_x, d_y)^\top$, the vortex-induced velocity is

$$\boldsymbol{v}_{\text{vortex}}(\boldsymbol{x}) = \frac{\alpha R_{\text{v}}^2}{\|\boldsymbol{d}\|_2^2 + R_{\text{v}}^2} \begin{pmatrix} -d_y \\ d_x \end{pmatrix}, \tag{26}$$

where $\alpha > 0$ sets the vortex strength. This term induces rotational motion around the vortex center, with magnitude that decays smoothly away from the core.

Full dynamics. Combining (25) and (26), the Cartesian dynamics are

$$\dot{x} = \frac{U_0 \sin(\kappa y)}{1 + \eta \|\boldsymbol{c}_0(\boldsymbol{x})\|_2^2} - \frac{\alpha R_{\text{v}}^2}{\|\boldsymbol{x} - \boldsymbol{c}\|_2^2 + R_{\text{v}}^2}(y - c_y), \tag{27}$$

$$\dot{y} = \frac{V_0 \sin(\kappa x)}{1 + \eta \|\boldsymbol{c}_0(\boldsymbol{x})\|_2^2} + \frac{\alpha R_{\text{v}}^2}{\|\boldsymbol{x} - \boldsymbol{c}\|_2^2 + R_{\text{v}}^2}(x - c_x), \tag{28}$$

where $\boldsymbol{c} = (c_x, c_y)^\top = (-0.5, 0.5)^\top$ which are clearly highly non-linear.

Parameters and data generation. We use the parameter values $\alpha = 0.7$, $\eta = 1.0$, and $\kappa = 1.5\pi$. Trajectory data are generated by numerically integrating the system with sampling time $\Delta t = 0.1\,\text{s}$ and induce process noise on all the states with $\sigma_1 = \sigma_2 = 0.01$. We generate a total of 300 trajectories with each having a length of 90 time steps from pseudo-random initial conditions within $x, y \in [-1.5, 1.5]$ using a Sobol sequence. 20 % of trajectories are used as validation data for all the methods to determine the best performing model state which is then used for testing on newly random generated trajectories. For the multi-step losses, we use a horizon of $H = 40$ and a stride of one to effectively increase the supervision signal.

C.3 The Mini Wheelbot

Reaction wheel balancing robots are challenging control benchmarks that have attracted research attention for multiple decades due to fast, unstable dynamics and strong nonlinearities (see [55] for a recent survey). The Mini Wheelbot is such an open-source robot [33] that is based on an earlier prototype [25]. The Mini Wheelbot uses its two wheels for balancing: the driving wheel in the pitch direction like a segway and the reaction wheel in the roll direction like a reaction wheel pendulum. The robot has no third "turntable" actuator for controlling the yaw orientation. Instead, it relies on geometric turning of a single wheel at an angle and the nonlinear gyroscopic effects of the multibody dynamics for reorienting.

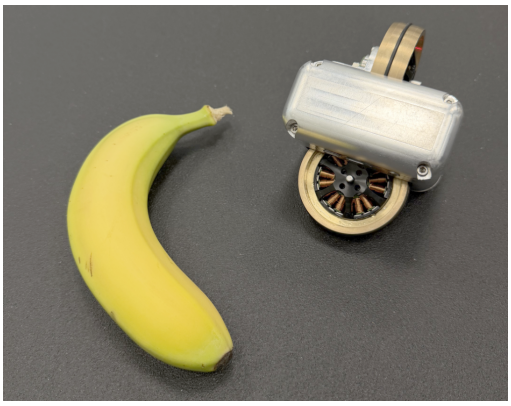


Figure 7: *The Mini Wheelbot [33].* Banana for scale.

Data and training setup To train the models, we use data from the Mini Wheelbot dataset [34][3]. This data is collected at $\Delta t = 0.006\,\mathrm{s}$ and we use the same train, validation, and test split for all loss functions. For all the trajectories, we cut the first $2\,\mathrm{s}$ and the last $2\,\mathrm{s}$. For training and evaluation, we replay the ground-truth actions over the trajectory segment.

Mapping state rollouts onto the xy-plane To visualize rollouts in the plane, we map the predicted dynamics to a planar trajectory on the xy-plane via unicycle-like wheel odometry. Let r_w denote the drive-wheel radius, Δt the sampling period, q_{DR} the drive-wheel angle, and ψ the yaw angle. We first align the yaw to the world frame,

$$\psi_t^{\mathrm{w}} \;=\; \mathrm{unwrap}(\psi_t) \;-\; \psi_0 \;+\; \psi_{\mathrm{init}}^{\mathrm{w}}, \tag{29}$$

where $\mathrm{unwrap}(\cdot)$ removes 2π discontinuities and $\psi_{\mathrm{init}}^{\mathrm{w}}$ is a chosen initial heading (in Fig. 5, we choose $\psi_{\mathrm{init}}^{\mathrm{w}} = 0$ for all segments). The arc length traveled by the drive wheel between two consecutive samples is

$$\Delta s_t \;=\; r_w \left(q_{\mathrm{DR},t} - q_{\mathrm{DR},t-1}\right), \tag{30}$$

and, assuming no wheel slip and motion along the current heading, the planar position is accumulated by projecting each increment onto the heading direction, starting from $(x_0, y_0) = (0, 0)$:

$$x_t \;=\; \sum_{k=1}^{t} \cos\!\left(\psi_k^{\mathrm{w}}\right) \Delta s_k, \quad y_t \;=\; \sum_{k=1}^{t} \sin\!\left(\psi_k^{\mathrm{w}}\right) \Delta s_k. \tag{31}$$

For model rollouts, the learned dynamics output the yaw rate $\dot{\psi}$ and drive-wheel speed $\dot{q}_{\mathrm{DR}}$ rather than ψ and q_{DR} directly, as the MW is invariant to these absolute values. We reconstruct the latter by forward-Euler integration from the measured initial values $\psi_0, q_{\mathrm{DR},0}$,

$$\psi_{t+1} \;=\; \psi_t + \dot{\psi}_t\, \Delta t, \quad q_{\mathrm{DR},t+1} \;=\; q_{\mathrm{DR},t} + \dot{q}_{\mathrm{DR},t}\, \Delta t, \tag{32}$$

and then apply the odometry mapping above to obtain (x_t, y_t).

[3]The dataset is available under the MIT license at https://github.com/wheelbot/dataset.

D Metrics

Throughout, let $\hat{\boldsymbol{x}}_{0:H} = (\hat{\boldsymbol{x}}_0, \ldots, \hat{\boldsymbol{x}}_H)$ denote the model's mean predicted trajectory, $\boldsymbol{x}^*_{0:H}$ the noiseless ground-truth trajectory, $\{\tilde{\boldsymbol{x}}^{(s)}_{0:H}\}_{s=1}^S$ the noisy ground-truth rollouts, and $\{\hat{\boldsymbol{x}}^{(s)}_{0:H}\}_{s=1}^S$ the stochastic model rollout samples, all initialized from the same state $\boldsymbol{x}_0 \in \mathbb{R}^d$. With this we can define the following metrics of interest.

Final Displacement Error. The Final Displacement Error (FDE) measures the Euclidean distance between the mean predicted state and the noiseless ground truth at the terminal timestep H:

$$\mathrm{FDE} = \|\hat{\boldsymbol{x}}_H - \boldsymbol{x}^*_H\|_2. \tag{33}$$

Cumulative L2 Error. The Cumulative L2 Error sums the per-step Euclidean distances between the mean predicted trajectory and the noiseless ground truth over all timesteps:

$$\mathrm{CL2} = \sum_{h=1}^{H} \|\hat{\boldsymbol{x}}_h - \boldsymbol{x}^*_h\|_2. \tag{34}$$

This measures the total accumulated trajectory deviation rather than only the terminal error.

Dynamic Time Warping Distance. Dynamic Time Warping (DTW) [56] is a similarity measure between temporal sequences that is robust to local timing differences. For multi-dimensional trajectories we compute the DTW distance per state dimension and sum:

$$\mathrm{DTW} = \sum_{j=1}^{d} \mathrm{DTW}_j\big(\{\hat{x}_{h,j}\}_{h=0}^H,\, \{x^*_{h,j}\}_{h=0}^H\big)\,, \tag{35}$$

where DTW_j denotes the scalar DTW distance along dimension j. Unlike CL2, DTW allows for non-linear temporal alignment, making it less sensitive to phase shifts in oscillatory dynamics. To evaluate this metric, we use the Python package `dtaidistance`[4] [62] for time-series distances.

Maximum Mean Discrepancy. To compare the full predictive distribution against the ground-truth distribution over trajectories, we use the unbiased estimator of the squared Maximum Mean Discrepancy (MMD^2) [29] with an RBF kernel. Each trajectory is flattened into a vector $\boldsymbol{z} = \mathrm{vec}(\boldsymbol{x}_{0:H}) \in \mathbb{R}^{d(H+1)}$, giving sample sets $\{\hat{\boldsymbol{z}}^{(s)}\}_{s=1}^S$ and $\{\tilde{\boldsymbol{z}}^{(s)}\}_{s=1}^S$ for the model and noisy ground truth respectively. The kernel is

$$k(\boldsymbol{z}, \boldsymbol{z}') = \exp\!\left(-\frac{\|\boldsymbol{z} - \boldsymbol{z}'\|_2^2}{2\sigma^2}\right), \tag{36}$$

where σ is set via the median heuristic [29, Section 8] applied to the ground-truth samples and held fixed across all models. The unbiased MMD^2 estimator is

$$\widehat{\mathrm{MMD}}^2 = \frac{1}{S(S-1)} \sum_{s \neq s'} k(\hat{\boldsymbol{z}}^{(s)}, \hat{\boldsymbol{z}}^{(s')}) + \frac{1}{S(S-1)} \sum_{s \neq s'} k(\tilde{\boldsymbol{z}}^{(s)}, \tilde{\boldsymbol{z}}^{(s')}) - \frac{2}{S^2} \sum_{s,s'} k(\hat{\boldsymbol{z}}^{(s)}, \tilde{\boldsymbol{z}}^{(s')}). \tag{37}$$

A value of $\widehat{\mathrm{MMD}}^2 = 0$ indicates that the model and ground-truth trajectory distributions are indistinguishable under the kernel.

Energy Distance. The Energy Distance [60] is a strictly proper divergence between distributions that does not require a kernel bandwidth. Using the same flattened trajectory representations as above, the empirical Energy Distance is

$$\widehat{D}_E = 2\,\mathbb{E}[\|\hat{\boldsymbol{z}} - \tilde{\boldsymbol{z}}\|_2] - \mathbb{E}[\|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}'\|_2] - \mathbb{E}[\|\tilde{\boldsymbol{z}} - \tilde{\boldsymbol{z}}'\|_2]\,, \tag{38}$$

where expectations are estimated by averaging over all pairs of samples. The Energy Distance is equivalent to MMD^2 with the negative-definite kernel $k(\boldsymbol{z}, \boldsymbol{z}') = -\|\boldsymbol{z} - \boldsymbol{z}'\|_2$, and therefore operates directly in the original Euclidean trajectory space rather than in a kernel-induced feature space.

[4] `dtaidistance` is publicly available under the Apache Licence, Version 2.0 at https://github.com/wannesm/dtaidistance.

As a consequence, $\widehat{D}_E$ penalizes distributional discrepancies proportionally to Euclidean distances between trajectories: two distributions are close in Energy Distance if and only if their samples are close on average in ℓ_2. This makes it interpretable and scale-equivariant, but unlike the RBF kernel used in MMD^2, it does not discount large deviations exponentially and thus is sensitive to outlying trajectories. Like MMD^2, $\widehat{D}_E = 0$ if and only if the two distributions are equal.

Empirical Marginal Coverage. Let $\{\hat{\boldsymbol{x}}_{0:H}^{(s)}\}_{s=1}^{S}$ denote S stochastic rollout samples from the model and $\{\tilde{\boldsymbol{x}}_{0:H}^{(s)}\}_{s=1}^{S}$ denote S noisy ground-truth rollouts. For a nominal level $1-\alpha \in (0,1)$, define the per-dimension marginal quantile bounds at each timestep h and dimension j as

$$\hat{q}_{h,j}^{-} = Q_{\alpha/2}\Big(\{\hat{x}_{h,j}^{(s)}\}_{s=1}^{S}\Big), \qquad \hat{q}_{h,j}^{+} = Q_{1-\alpha/2}\Big(\{\hat{x}_{h,j}^{(s)}\}_{s=1}^{S}\Big), \tag{39}$$

where $Q_\beta(\cdot)$ denotes the β-quantile of a set of scalars. The empirical marginal coverage at timestep h is

$$\widehat{\mathrm{Cov}}(h) = \frac{1}{d}\sum_{j=1}^{d}\frac{1}{S}\sum_{s=1}^{S}\mathbf{1}\Big[\hat{q}_{h,j}^{-} \le \tilde{x}_{h,j}^{(s)} \le \hat{q}_{h,j}^{+}\Big]. \tag{40}$$

A perfectly calibrated model satisfies $\widehat{\mathrm{Cov}}(h) \approx 1-\alpha$ for all h. We report the time-averaged marginal coverage, excluding the trivial $h=0$ step at which all samples coincide with the known initial condition:

$$\overline{\mathrm{Cov}} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{H}\sum_{h=1}^{H}\widehat{\mathrm{Cov}}_i(h). \tag{41}$$

Remark 1. *We compute coverage per dimension and then average, rather than checking whether all dimensions are simultaneously inside their respective intervals (joint coverage). To see why this matters, suppose each per-dimension marginal interval is correctly calibrated at nominal level* $1-\alpha$*. The probability that all* d *dimensions are* simultaneously *inside their intervals is bounded by*

$$\max(0,\, 1-d\alpha) \le P\left(\bigcap_{j=1}^{d}\hat{x}^{(j)} \in [\hat{q}_j^{-},\, \hat{q}_j^{+}]\right) \le 1-\alpha, \tag{42}$$

and equals $(1-\alpha)^d$ *under independence across dimensions. For a* $d=2$ *system with independent dimensions this gives a joint coverage of* $(0.90)^2 = 0.81$ *at the* 90% *nominal level and* $(0.95)^2 = 0.90$ *at* 95%*, even for a perfectly calibrated model. Reporting joint coverage against a nominal level of* $1-\alpha$ *would therefore understate calibration for any* $d>1$ *unless the dimensions are perfectly dependent, with the gap growing exponentially in* d *under independence. Averaging marginal coverage across dimensions as in (40) removes this dependence on* d *and yields a quantity that is directly comparable to the nominal level regardless of state dimension.*

E Details on Baselines

We first show that the one-step predictive distribution is independent of the training loss, then state the baselines not derived in the main text.

Loss-independent one-step predictive distribution. For a test point $\boldsymbol{x}_h$, the one-step predictive is obtained by marginalizing over the posterior:

$$p(\boldsymbol{x}_{h+1} \mid \boldsymbol{x}_h) = \int p(\boldsymbol{x}_{h+1} \mid \boldsymbol{x}_h, \boldsymbol{W})\, q(\boldsymbol{W} \mid \boldsymbol{\eta})\, d\boldsymbol{W}. \tag{43}$$

The likelihood $p(\boldsymbol{x}_{h+1} \mid \boldsymbol{x}_h, \boldsymbol{W}) = \mathcal{N}(\boldsymbol{x}_{h+1};\, \boldsymbol{x}_h + \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h),\, \boldsymbol{\Sigma}_\nu)$ is linear in $\boldsymbol{W}$ for fixed features $\boldsymbol{\phi}_h = \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_h)$, and $q(\boldsymbol{W} \mid \boldsymbol{\eta})$ is Gaussian by construction of the variational family $\mathcal{Q}$. Since a Gaussian integrated against a linear-Gaussian observation model yields a Gaussian, the predictive is available in closed form. Concretely, because the outputs are modeled independently with $q(\boldsymbol{w}_j) = \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{L}_j \boldsymbol{L}_j^\top)$, the predicted residual for dimension j satisfies

$$\mathbb{E}_q[\boldsymbol{w}_j^\top \boldsymbol{\phi}_h] = \boldsymbol{\mu}_j^\top \boldsymbol{\phi}_h, \qquad \mathrm{Var}_q[\boldsymbol{w}_j^\top \boldsymbol{\phi}_h] = \boldsymbol{\phi}_h^\top \boldsymbol{L}_j \boldsymbol{L}_j^\top \boldsymbol{\phi}_h. \tag{44}$$

Adding the aleatoric noise σ_j^2, the marginal predictive for each output dimension is

$$p(\delta_{h,j} \mid \boldsymbol{x}_h) = \mathcal{N}\big(\boldsymbol{\mu}_j^\top \boldsymbol{\phi}_h,\ \sigma_j^2 + \boldsymbol{\phi}_h^\top \boldsymbol{L}_j \boldsymbol{L}_j^\top \boldsymbol{\phi}_h\big). \tag{45}$$

The predictive contains variational parameters $\boldsymbol{\eta}$, the noise σ_j^2, and the features $\boldsymbol{\phi}_h$. The choice of scoring rule and evaluation mode affects the *values* of the optimized parameters $(\boldsymbol{\eta}^*, \boldsymbol{\theta}^*, \boldsymbol{\Sigma}_\nu^*)$, but the *functional form* of the predictive remains an axis-aligned Gaussian.

E.1 Single-Step ELBO (`s-NLL`)

For completeness we state the single-step special case of the `m-NLL` objective (Section 3.2) explicitly. Setting $H = 1$ (and therefore inducing a dataset of one-step transitions, *i.e.* $\mathcal{D} = \{(\boldsymbol{x}_0^{(n)}, \boldsymbol{x}_1^{(n)})\}_{n=1}^N$), the loss reduces to the Gaussian negative log-likelihood of a single transition:

$$\underbrace{\ell_{\texttt{s-NLL}}(\boldsymbol{W}, \boldsymbol{\tau}^{(n)}) \;=\; -\log \mathcal{N}\big(\boldsymbol{\delta}_0^{(n)};\ \boldsymbol{W}^\top \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_0^{(n)}),\ \boldsymbol{\Sigma}_\nu\big)}_{\text{Single-step NLL (s-NLL)}}, \tag{46}$$

where $\boldsymbol{\delta}_0^{(n)} := \boldsymbol{x}_1^{(n)} - \boldsymbol{x}_0^{(n)}$. Writing $\boldsymbol{\phi}_n = \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_0^{(n)})$ and stacking variational means into $\boldsymbol{\mu}_W = [\boldsymbol{\mu}_{w_1}, \ldots, \boldsymbol{\mu}_{w_{d_x}}] \in \mathbb{R}^{p \times d_x}$, the closed-form expected loss under the Gaussian posterior is

$$\mathbb{E}_{q(\boldsymbol{W}|\boldsymbol{\eta})}\big[\ell_{\texttt{s-NLL}}(\boldsymbol{W}, \boldsymbol{\tau}^{(n)})\big] \;=\; -\log \mathcal{N}\big(\boldsymbol{\delta}_0^{(n)};\ \boldsymbol{\mu}_W^\top \boldsymbol{\phi}_n,\ \boldsymbol{\Sigma}_\nu\big) \;+\; \frac{1}{2} \sum_{j=1}^{d_x} \frac{\boldsymbol{\phi}_n^\top \boldsymbol{\Sigma}_{w_j} \boldsymbol{\phi}_n}{\sigma_j^2}, \tag{47}$$

With $\lambda = 1/N$ in (3), this recovers the standard VBLL of Harrison et al. [32].

E.2 Kernel Density Estimation of the Marginal Likelihood (`m-KDE`)

`m-KDE` retains a log-likelihood formulation, which is desirable when the planner benefits from an explicit density over future states. Rather than scoring each member independently, it constructs a Gaussian-mixture predictive density [35] that pools all S samples:

$$\hat{p}_h(\boldsymbol{x}_h^{(n)}) = \frac{1}{S} \sum_{s=1}^{S} \mathcal{N}\left(\boldsymbol{x}_h^{(n)};\ \hat{\boldsymbol{x}}_h^{(s)},\ h\, \boldsymbol{\Sigma}_\nu\right). \tag{48}$$

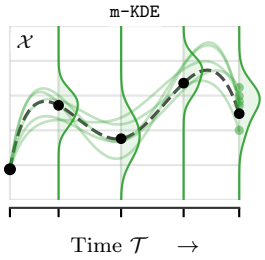


Figure 8: *Viz. of* `m-KDE`.

Each mixture component captures the deterministic prediction error (through its mean $\hat{\boldsymbol{x}}_h^{(s)}$) and the accumulated aleatoric uncertainty (through the bandwidth $h\, \boldsymbol{\Sigma}_\nu$), while the mixture itself represents the epistemic uncertainty over $\boldsymbol{W}$. The loss is the negative marginal log-likelihood of the observed trajectory under the KDE density:

$$\underbrace{\ell_{\mathrm{KDE}}\big(\{\boldsymbol{W}^{(s)}\}_{s=1}^S,\ \boldsymbol{\tau}^{(n)}\big) \;=\; -\sum_{h=1}^{H} \log \hat{p}_h(\boldsymbol{x}_h^{(n)})}_{\text{Multi-step KDE (m-KDE)}}. \tag{49}$$

Differentiating through (49), the log-sum-exp structure assigns each weight sample $\boldsymbol{W}^{(s)}$ a gradient contribution proportional to its *responsibility*—the fraction of the mixture density it explains at the observed state. Samples whose rollouts pass close to $\boldsymbol{x}_h^{(n)}$ receive large gradients; those that diverge are down-weighted. This softmax-like reweighting provides an implicit form of trajectory selection that encourages the posterior to concentrate on weight matrices producing accurate multi-step predictions, while the mixture formulation preserves the ability to represent multi-modal predictive distributions that a single Gaussian rollout would collapse.

The log-sum-exp structure of (49) evaluates $-\log \mathbb{E}_{q(\boldsymbol{W})}[p(\boldsymbol{x}_h^* \mid \boldsymbol{W})]$ rather than $\mathbb{E}_{q(\boldsymbol{W})}[-\log p(\boldsymbol{x}_h^* \mid \boldsymbol{W})]$: the logarithm sits *outside* the expectation, coupling all S weight samples into a single joint evaluation. No pointwise loss $\ell(\boldsymbol{W})$ exists whose expectation reproduces this quantity, so m-KDE is *not* an instantiation of the Gibbs VI objective (4). The correct interpretation is that it applies the log scoring rule directly to the *marginal predictive distribution*

$$\bar{p}(\boldsymbol{x}_h \mid \boldsymbol{x}_0) \;=\; \int p(\boldsymbol{x}_h \mid \boldsymbol{W}, \boldsymbol{x}_0)\, q(\boldsymbol{W})\, d\boldsymbol{W}, \tag{50}$$

estimated via the KDE mixture (48). By Jensen's inequality, the predictive data-fit is always less pessimistic than a corresponding ELBO term: $\log \mathbb{E}_q[p] \geq \mathbb{E}_q[\log p]$; the mixture receives credit for ensemble diversity in a way that individual-sample scoring cannot but comes with its own computational downsides. In our experiment, we show that this scoring of the marginal yields the best coverage across tasks but can also lead to diverging samples from the tails which we did not observe with losses that optimized the computationally "easier" Gibbs VI objective.

Relationship between m-KDE and m-arCRPS.

The m-arCRPS shares its autoregressive rollout and linearly growing noise envelope with the KDE objective of Ivanovic et al. [35]; the two differ in how the resulting ensemble is aggregated and scored. The m-arCRPS evaluates each weight draw independently: $\ell_{\text{m-arCRPS}}(\boldsymbol{W}, \boldsymbol{\tau}^{(n)})$ is a pointwise function of a single $\boldsymbol{W}$, and takes the expectation under $q(\boldsymbol{W})$ outside the scoring rule. This pointwise structure is what makes the m-arCRPS an instantiation of the Gibbs VI objective (4), inheriting the associated PAC-Bayesian interpretation. The KDE objective instead pools all S weight samples into the Gaussian-mixture predictive $\hat{p}_h = \frac{1}{S}\sum_s \mathcal{N}(\hat{\boldsymbol{x}}_h^{(s)},\, h\,\boldsymbol{\Sigma}_\nu)$ and applies the log scoring rule to this marginal density. Because the logarithm sits outside the average over ensemble members, the loss couples all draws into a single joint evaluation that falls outside the GVI framework. The matched m-arNLL disentangles these two differences as it shares the pointwise Gibbs VI structure, rollout, and noise envelope of m-arCRPS, but applies the log score, so comparing m-arNLL with m-arCRPS isolates the scoring rule, while comparing m-arNLL with m-KDE isolates the aggregation structure. This structural difference propagates into the gradient: under the m-arCRPS, every weight sample receives an $\mathcal{O}(1)$ gradient signal thanks to the CRPS linear tail behavior, whereas the KDE's log-sum-exp assigns each sample a gradient proportional to its responsibility in the mixture, exponentially down-weighting draws whose rollouts diverge from the observation. While the CRPS losses under Gibbs VI demonstrated the strongest performance, our results also show very good coverage of m-KDE which indicates that a potential extension of m-KDE with CRPS might be interesting.

F Related Work (extended version)

Bayesian neural networks in dynamics learning. Probabilistic dynamics models have long been central to model-based control, with Gaussian processes providing strong uncertainty estimates in the low-data regime [14]. For neural network dynamics models, probabilistic ensembles (PE) [40] remain popular in model-based reinforcement learning, combining bootstrap ensembles with Gaussian output heads to capture epistemic and aleatoric uncertainty, and underpin a large body of planning and policy-optimization work [10, 36, 20]. Fully Bayesian alternatives such as MC-dropout [23] have also been applied to dynamics learning [24], but typically at a higher computational cost for predictive inference and with weaker calibration in practice. More recently, Bayesian last-layer models [58, 54, 19, 63, 32] have emerged as an attractive middle ground: restricting the Bayesian treatment to the final layer preserves tractable inference while still providing useful predictive uncertainty for downstream tasks [31, 47, 11, 13]. Variational Bayesian last layers [32] instantiate this idea by directly learning a variational posterior of the last layer and have since been applied to Bayesian optimization [6] and autonomous racing [7]. Our work builds directly on VBLLs but reinterprets them through the lens of Gibbs variational inference [2, 5, 37], decoupling the posterior from the training loss and enabling task-aligned objectives. Through this, we extend VBLLs to multi-step losses. BLLs have also been popular in a meta-learning setting–especially for dynamics learning. Here, the goal is usually to learn the prior distribution $p(\boldsymbol{W})$ and then recursively update it via Kalman filtering [35, 46] or recursive least squares [31, 11]. By design, these already are trained via multi-steps at the cost of back-propagating through the conditioning. In this work, we do not consider such online adaptation, though it is noteworthy that recursive updates on the learned posteriors are still possible but following the Gibbs VI framework these would have to be gradient-based as there exist in general no closed-form update for an arbitrary loss ℓ.

Multi-step loss functions. The importance of training dynamics models for long-horizon accuracy is well established and some works explicitly demonstrate the downsides of one-step training losses for compounding errors [43]. Doerr et al. [15] argue that multi-step objectives become necessary when the Markov property is violated or when process and observation noise are significant, and we further demonstrate that a one-step loss can also exhibit failure modes in very controlled Markov environments. Lambert et al. [42] show that multi-step training improves long-horizon accuracy and sample efficiency in model-based RL. Closest to our setting, Ivanovic et al. [35] propose a kernel-density multi-step objective with linearly growing noise for trajectory prediction under heteroskedasticity, which we adopt as our `m-KDE` baseline and which inspired the `m-arCRPS` construction. In contrast to these log-likelihood-based formulations, our CRPS-based losses leverage a strictly proper scoring rule [51, 28, 27, 30] that is robust to outliers [57], and are justified within the Gibbs VI framework rather than through a likelihood interpretation. Such CRPS-based scoring is very popular [4], especially in weather forecasting [27, 44, 53] but often used to directly score marginal distributions [44]. To the best of our knowledge, we are the first to adopt it to the variational setting through Gibbs VI. Note that standard CRPS (or in general losses based on standard formulation of the energy distance or MMD) do not fit into the Gibbs VI framework as draws of $\boldsymbol{W}$ are not scored independently.

In the context of probabilistic ensembles, Levy et al. [45] propose mean propagation for multi-step training over a small horizon ($H = 4$) and show improved performance for a structured dynamics model. Extending multi-step training to probabilistic ensembles in a principled manner would be a natural direction for future work, though the ensemble structure poses additional challenges compared to the Bayesian last-layer setting, as the independence structure across ensemble members differs from the weight posterior in VBLLs making consistent rollouts (for VBLLs by design just a draw of $\boldsymbol{W}$) non-trivial. Here, using approaches as proposed by Frauenknecht et al. [21] also for model learning can be interesting.

For deterministic dynamics models, various works have extended the standard one-step MSE loss to a multi-step setting with mixed results. Lutter et al. [50] demonstrate that while a short multi-step horizon can be beneficial, too long a horizon actually results in worse downstream performance, leading Byravan et al. [8] to directly build on these short-horizon multi-step models. To achieve better performance for training with longer horizons, Benechehab et al. [3] introduce a weighted sum over time-steps to stabilize gradients and down-weight future predictions; however, also with mixed results. We find that for (Gibbs) VBLLs, we did not need to specify additional weighting which would result from tempered likelihoods, as the VBLL setup seems to provide sufficient flexibility, especially in combination with the CRPS losses, accommodate and then also optimize for horizons

of up to 100 time steps (see Appendix I). For Gaussian processes, moment matching and uncertainty propagation via linearization have been used to train with multi-step objectives [14, 15]. In that setting, the set of learnable parameters is significantly smaller; in the regime of overparameterized neural networks, it is even more important to provide informative multi-step training signal to learn good and reliable features.

Objective mismatch and decision-aware model learning. A fundamental tension in model-based control is that minimizing prediction error does not necessarily maximize downstream task performance. Lambert et al. [41] coin the term *objective mismatch* and demonstrate empirically that model accuracy and control performance can be poorly correlated, arguing that the model learning objective should be informed by the decision-making pipeline it serves. This observation has motivated a growing line of work on decision-aware model learning, recently surveyed by Wei et al. [64]. Farahmand et al. [18] introduce value-aware model learning, which replaces the standard maximum-likelihood objective with a loss that incorporates the structure of the value function space, ensuring that the model is accurate where it matters for policy evaluation. This idea was further developed in Farahmand [17], which proposes an iterative scheme that alternates between value estimation and model refinement. D'Oro et al. [16] take a complementary perspective: their gradient-aware model-based policy search algorithm re-weights the model-learning objective according to policy-gradient norms, focusing model accuracy on state-action regions where the policy gradient is large. Similarly, Abachi [1] derives policy-aware model learning, which aligns the model-learning loss with the structure of the policy gradient estimator. More recently, Voelcker et al. [61] propose value-gradient weighted model learning (VaGraM), which rescales the prediction loss with gradient information from the current value function estimate and provides a thorough empirical comparison of decision-aware approaches. In the vision-based setting, Nair et al. [52] propose goal-aware prediction learning, which conditions the dynamics model on the goal state, directing model capacity toward task-relevant aspects of the observation.

These approaches mostly operate in the deterministic setting with point-estimate models and focus on coupling model learning directly with policy optimization. Our work shares the underlying motivation—the training loss should reflect how the model is used—but takes a different and complementary route. Rather than incorporating a value function or policy gradient into the model-learning objective, we remain in the probabilistic prediction setting and use the Gibbs VI framework to replace the likelihood with losses that are structurally aligned with the deployment regime of the model (multi-step rollouts, robust scoring). Notably, the two perspectives are not mutually exclusive: because the Gibbs VI objective (3) accepts any pointwise loss $\ell(\boldsymbol{W}, \tau^{(n)})$, decision-aware reweightings such as VaGraM can be incorporated by simply scaling each per-step CRPS term by the local value-gradient magnitude, yielding a value-aware multi-step scoring rule that retains a PAC-Bayesian interpretation. Exploring such combinations in a closed-loop control setting is a promising direction for future work.

G Proof of Proposition 1

For convenience, we restate the proposition in a slightly expanded form that makes the leading correction term explicit; the bound in the main text follows as an immediate corollary.

Proposition. *Suppose $\phi_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{b}$ and define $\boldsymbol{A} := \boldsymbol{I} + \boldsymbol{W}^\top \boldsymbol{\Phi}$. Then for all $h \geq 1$ the predictive distribution is exactly Gaussian,*

$$p(\boldsymbol{x}_h \mid \boldsymbol{W}, \boldsymbol{x}_0) = \mathcal{N}\big(\hat{\boldsymbol{x}}_h(\boldsymbol{W}),\, \boldsymbol{C}_h\big), \qquad \boldsymbol{C}_h := \sum_{k=0}^{h-1} \boldsymbol{A}^k \boldsymbol{\Sigma}_\nu (\boldsymbol{A}^k)^\top. \tag{51}$$

The covariance collapses to $\boldsymbol{C}_h = h\,\boldsymbol{\Sigma}_\nu$ whenever $\boldsymbol{A}\,\boldsymbol{\Sigma}_\nu\,\boldsymbol{A}^\top = \boldsymbol{\Sigma}_\nu$ (e.g. $\boldsymbol{A}$ orthogonal and $\boldsymbol{\Sigma}_\nu = \sigma^2 \boldsymbol{I}$). More generally, writing $\boldsymbol{A} = \boldsymbol{I} + \boldsymbol{E}$ with $\boldsymbol{E} = \boldsymbol{W}^\top \boldsymbol{\Phi}$,

$$\boldsymbol{C}_h = h\,\boldsymbol{\Sigma}_\nu + \frac{h(h-1)}{2}\big(\boldsymbol{E}\,\boldsymbol{\Sigma}_\nu + \boldsymbol{\Sigma}_\nu\,\boldsymbol{E}^\top\big) + \mathcal{O}\big(h^3 \|\boldsymbol{E}\|^2 \|\boldsymbol{\Sigma}_\nu\|\big), \tag{52}$$

valid whenever $h\,\|\boldsymbol{E}\|$ is small.

Proof. Under affine features, the stochastic dynamics reduce to $\boldsymbol{x}_{h+1} = \boldsymbol{A}\,\boldsymbol{x}_h + \boldsymbol{W}^\top \boldsymbol{b} + \boldsymbol{\nu}_h$, which is a linear time-invariant system driven by i.i.d. Gaussian noise $\boldsymbol{\nu}_h \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\nu)$. Unrolling from $\boldsymbol{x}_0$,

$$\boldsymbol{x}_h = \boldsymbol{A}^h \boldsymbol{x}_0 + \sum_{k=0}^{h-1} \boldsymbol{A}^k\, \boldsymbol{W}^\top \boldsymbol{b} + \sum_{k=0}^{h-1} \boldsymbol{A}^{h-1-k}\, \boldsymbol{\nu}_k. \tag{53}$$

Part (i). The deterministic rollout satisfies $\hat{\boldsymbol{x}}_h(\boldsymbol{W}) = \boldsymbol{A}^h \boldsymbol{x}_0 + \sum_{k=0}^{h-1} \boldsymbol{A}^k\, \boldsymbol{W}^\top \boldsymbol{b}$. Since $\mathbb{E}[\boldsymbol{\nu}_k] = \boldsymbol{0}$, taking expectations in (53) yields $\mathbb{E}[\boldsymbol{x}_h \mid \boldsymbol{W}, \boldsymbol{x}_0] = \hat{\boldsymbol{x}}_h(\boldsymbol{W})$.

Part (ii). Subtracting the mean from (53) gives

$$\boldsymbol{x}_h - \mathbb{E}[\boldsymbol{x}_h \mid \boldsymbol{W}, \boldsymbol{x}_0] = \sum_{k=0}^{h-1} \boldsymbol{A}^{h-1-k}\, \boldsymbol{\nu}_k. \tag{54}$$

Since the $\boldsymbol{\nu}_k$ are independent with covariance $\boldsymbol{\Sigma}_\nu$, the conditional covariance is

$$\boldsymbol{C}_h = \sum_{k=0}^{h-1} \boldsymbol{A}^{h-1-k}\, \boldsymbol{\Sigma}_\nu\, (\boldsymbol{A}^{h-1-k})^\top = \sum_{k=0}^{h-1} \boldsymbol{A}^k\, \boldsymbol{\Sigma}_\nu\, (\boldsymbol{A}^k)^\top, \tag{55}$$

where the second equality follows by reindexing $k \mapsto h-1-k$.

Exactness. If $\boldsymbol{A}\,\boldsymbol{\Sigma}_\nu\,\boldsymbol{A}^\top = \boldsymbol{\Sigma}_\nu$, then by induction $\boldsymbol{A}^k\,\boldsymbol{\Sigma}_\nu\,(\boldsymbol{A}^k)^\top = \boldsymbol{\Sigma}_\nu$ for all $k \geq 0$, and therefore $\boldsymbol{C}_h = h\,\boldsymbol{\Sigma}_\nu$ exactly. This holds in particular when $\boldsymbol{A}$ is orthogonal and $\boldsymbol{\Sigma}_\nu = \sigma^2 \boldsymbol{I}$, since then $\boldsymbol{A}^k (\boldsymbol{A}^k)^\top = \boldsymbol{I}$.

Expansion. Write $\boldsymbol{A} = \boldsymbol{I} + \boldsymbol{E}$. The binomial expansion gives

$$\boldsymbol{A}^k = (\boldsymbol{I} + \boldsymbol{E})^k = \boldsymbol{I} + k\,\boldsymbol{E} + \boldsymbol{R}_k, \qquad \|\boldsymbol{R}_k\| = \mathcal{O}\big(k^2 \|\boldsymbol{E}\|^2\big). \tag{56}$$

Substituting into each summand and keeping terms up to first order in $\boldsymbol{E}$,

$$\boldsymbol{A}^k \boldsymbol{\Sigma}_\nu (\boldsymbol{A}^k)^\top = (\boldsymbol{I} + k\boldsymbol{E} + \boldsymbol{R}_k)\, \boldsymbol{\Sigma}_\nu\, (\boldsymbol{I} + k\boldsymbol{E} + \boldsymbol{R}_k)^\top \tag{57}$$

$$= \boldsymbol{\Sigma}_\nu + k\big(\boldsymbol{E}\,\boldsymbol{\Sigma}_\nu + \boldsymbol{\Sigma}_\nu\,\boldsymbol{E}^\top\big) + \mathcal{O}\big(k^2 \|\boldsymbol{E}\|^2 \|\boldsymbol{\Sigma}_\nu\|\big). \tag{58}$$

Summing over $k = 0, \ldots, h-1$ and using $\sum_{k=0}^{h-1} 1 = h$, $\sum_{k=0}^{h-1} k = \frac{h(h-1)}{2}$, and $\sum_{k=0}^{h-1} k^2 = \mathcal{O}(h^3)$,

$$\boldsymbol{C}_h = h\,\boldsymbol{\Sigma}_\nu + \frac{h(h-1)}{2}\big(\boldsymbol{E}\,\boldsymbol{\Sigma}_\nu + \boldsymbol{\Sigma}_\nu\,\boldsymbol{E}^\top\big) + \mathcal{O}\big(h^3 \|\boldsymbol{E}\|^2 \|\boldsymbol{\Sigma}_\nu\|\big), \tag{59}$$

which establishes (52). □

Remark 2. *The bound stated in the main text follows immediately from (52): when $h\,\|\boldsymbol{E}\|$ is small, the $\mathcal{O}(h^3 \|\boldsymbol{E}\|^2 \|\boldsymbol{\Sigma}_\nu\|)$ term is dominated by the leading correction, and therefore*

$$\|\boldsymbol{C}_h - h\,\boldsymbol{\Sigma}_\nu\| = \mathcal{O}\big(h^2\, \|\boldsymbol{E}\|\, \|\boldsymbol{\Sigma}_\nu\|\big). \tag{60}$$

In the residual parameterization $\boldsymbol{A} = \boldsymbol{I} + \boldsymbol{W}^\top \boldsymbol{\Phi}$ with small time step Δt, one has $\|\boldsymbol{E}\| = \mathcal{O}(\Delta t)$, so the condition $h\,\|\boldsymbol{E}\|$ small corresponds to a bounded rollout time $h\,\Delta t$.

H Closed-Form CRPS for Gaussian Predictives

The closed-form expression of the CRPS for Gaussian predictive distributions is well known; see, e.g., [28, 27]. We briefly include the derivation here for completeness.

Starting from the energy form,

$$\mathrm{CRPS}(P, y) \;=\; \mathbb{E}_P|X - y| \;-\; \tfrac{1}{2}\,\mathbb{E}_P|X - X'|, \tag{61}$$

with $X, X' \overset{\text{iid}}{\sim} P$, we evaluate both terms for $P = \mathcal{N}(\mu, \sigma^2)$.

Term 1: $\mathbb{E}|X - y|$.

Let $Z \sim \mathcal{N}(0, 1)$ and write $X = \mu + \sigma Z$, with $z := (y - \mu)/\sigma$. Then

$$\mathbb{E}|X - y| = \sigma\,\mathbb{E}|Z - z|.$$

Using standard manipulations,

$$\begin{aligned}
\mathbb{E}|Z - z| &= \int_{-\infty}^{z} (z - t)\varphi(t)\,dt + \int_{z}^{\infty} (t - z)\varphi(t)\,dt \\
&= z\Phi(z) + \varphi(z) + \varphi(z) - z\big(1 - \Phi(z)\big) \\
&= 2\varphi(z) + z\big(2\Phi(z) - 1\big),
\end{aligned}$$

where we used $\int t\varphi(t)\,dt = -\varphi(t)$. Hence, $\mathbb{E}|X - y| = \sigma\Big[2\varphi(z) + z\big(2\Phi(z) - 1\big)\Big]$.

Term 2: $\mathbb{E}|X - X'|$.

Since $X - X' \sim \mathcal{N}(0, 2\sigma^2)$, we write $X - X' = \sigma\sqrt{2}\,Z$ with $Z \sim \mathcal{N}(0, 1)$, giving

$$\mathbb{E}|X - X'| = \sigma\sqrt{2}\,\mathbb{E}|Z| = \sigma\sqrt{2} \cdot \sqrt{\frac{2}{\pi}} = \frac{2\sigma}{\sqrt{\pi}}.$$

Substituting both derived terms into (61) yields the closed form from (10).

I Additional Results

I.1 Extended Statistical evaluation of the chaotic systems benchmark

To complement the average rank summary in Figure 4, we provide a formal statistical comparison across all 198 (environment, metric) evaluation blocks from Section 4.2. First we conducted a Friedman test that rejects the null hypothesis that all methods perform equally ($\chi^2 = 314.1$, $p < 10^{-65}$) and then followed this with post-hoc comparisons. Table 2 reports the rank distribution and these pairwise post-hoc comparisons.

All pairwise differences are significant at $\alpha = 0.05$ under the Holm-corrected Wilcoxon signed-rank test; under the more conservative Nemenyi test, only the differences between `m-KDE` and `s-NLL` as well as between `m-KDE` and `m-arCRPS` are not significant. In particular, both CRPS-based losses significantly outperform their likelihood-based counterparts (`m-CRPS` vs. `m-NLL` and `m-arCRPS` vs. `m-arNLL`). For the autoregressive pair, this advantage stems from training stability. `m-arNLL` diverged on all 13 chaotic systems and is therefore ranked last in 143 out of 198 cases, whereas `m-arCRPS` trained stably throughout; on the five systems where `m-arNLL` converged, the two are statistically indistinguishable. Notably, among the stably trained losses, `s-NLL` and `m-KDE` have the worst mean ranks, underscoring the limitations of single-step training and of the marginal mixture objective, respectively.

Table 2: *Statistical evaluation across 198 (environment, metric) blocks. Left:* Rank distribution (lower mean rank is better). *Right:* Post-hoc pairwise comparisons (Nemenyi test, critical difference $\Delta r = 0.536$, and Holm-corrected Wilcoxon signed-rank test); pairs below the line are not significant at $\alpha = 0.05$ under the Nemenyi test. Friedman test: $\chi^2 = 314.1$, $p < 10^{-65}$. Diverged runs (143 blocks of `m-arNLL`) are ranked last.

Method	Mean	Med.	Std	1st	2nd	3rd	4th	5th	6th
`m-CRPS`	2.12	2	1.26	85	49	35	16	11	2
`m-NLL`	2.83	2	1.48	46	56	23	33	39	1
`m-arCRPS`	3.24	3	1.51	28	50	27	47	32	14
`s-NLL`	3.78	4	1.32	9	17	66	50	29	27
`m-KDE`	3.83	4	1.43	20	20	31	40	77	10
`m-arNLL`	5.21	6	1.47	10	6	16	12	10	144

Method A	Method B	$\Delta\bar{r}$	p_{Nemenyi}	p_{Holm}
`m-arNLL`	`m-CRPS`	3.10	$< 10^{-10}$	$< 10^{-10}$
`m-arNLL`	`m-NLL`	2.38	$< 10^{-10}$	$< 10^{-10}$
`m-arNLL`	`m-arCRPS`	1.97	$< 10^{-10}$	$< 10^{-10}$
`m-KDE`	`m-CRPS`	1.71	$< 10^{-10}$	$< 10^{-10}$
`s-NLL`	`m-CRPS`	1.66	$< 10^{-10}$	$< 10^{-10}$
`m-arNLL`	`s-NLL`	1.43	$< 10^{-10}$	$< 10^{-10}$
`m-arNLL`	`m-KDE`	1.38	$< 10^{-10}$	$< 10^{-10}$
`m-arCRPS`	`m-CRPS`	1.12	$< 10^{-7}$	$< 10^{-9}$
`m-KDE`	`m-NLL`	1.00	$< 10^{-5}$	$< 10^{-5}$
`s-NLL`	`m-NLL`	0.95	$< 10^{-5}$	$< 10^{-6}$
`m-NLL`	`m-CRPS`	0.71	2×10^{-3}	$< 10^{-5}$
`m-KDE`	`m-arCRPS`	0.59	0.021	8×10^{-4}
`s-NLL`	`m-arCRPS`	0.54	0.047	3×10^{-3}
`m-arCRPS`	`m-NLL`	0.41	0.249	0.073
`m-KDE`	`s-NLL`	0.05	1.000	0.670

I.2 Intuition on the different learned feature representations

Figure 9 visualizes how different training losses shape the learned representation $\phi_\theta(x)$ and its induced sensitivities across the state space of the VSLC. A key property of the VSLC is its state-dependent angular velocity, which induces spatial heterogeneity: the upper-right and lower-left regions correspond to fast zones (also see Fig. 2), while the remaining regions are slow. Due to the inverse relationship between data density and velocity, the training data is heavily biased toward the slow regions.

Feature sensitivity. The top row reveals a strong difference across losses. `s-NLL` shows a low feature sensitivity which slightly larger sensitivity in the fast zones compared to the slow zones. In contrast, multi-step objectives show more diverse feature fields. Especially the CRPS losses and `m-KDE` demonstrate increased sensitivity in the fast zones (upper-right and lower-left), where accurate modeling is most critical for long-horizon predictions. It indicates that the multi-step losses encourage the representation to allocate capacity to dynamically important but under-sampled regions.

Mean prediction sensitivity. The middle row shows that all methods learn a similar structure for the mean dynamics, dominated by a strong horizontal band around $x_2 \approx 0$. This suggests that, despite differences in representation, the mean predictor is largely constrained by the residual model structure as well as the defined prior. There are however still clear observable differences between the different losses.

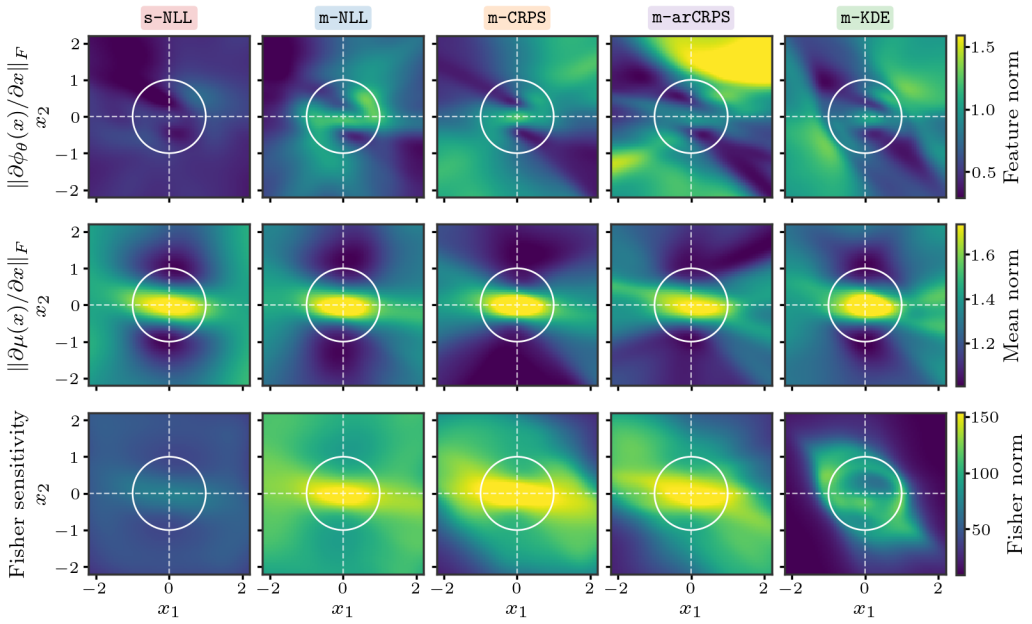


Figure 9: *Feature sensitivity (top), mean prediction sensitivity (middle), and Fisher sensitivity (bottom) for the learned model on the VSLC (Fig. 2).* All models are evaluated on the same data yet there are distinct differences in behavior, most prominently in feature and Fisher sensitivity.

Fisher sensitivity. The bottom row highlights how uncertainty interacts with the learned features. We define the Fisher sensitivity as

$$\mathcal{S}_{\mathrm{F}}(\boldsymbol{x}) := \left(\sum_{j=1}^{d_x} \frac{\left\| \nabla_{\boldsymbol{x}} \mu_j(\boldsymbol{x}) \right\|_2^2}{\sigma_j^2(\boldsymbol{x})} + 2 \sum_{j=1}^{d_x} \left\| \nabla_{\boldsymbol{x}} \log \sigma_j(\boldsymbol{x}) \right\|_2^2 \right)^{1/2}, \tag{62}$$

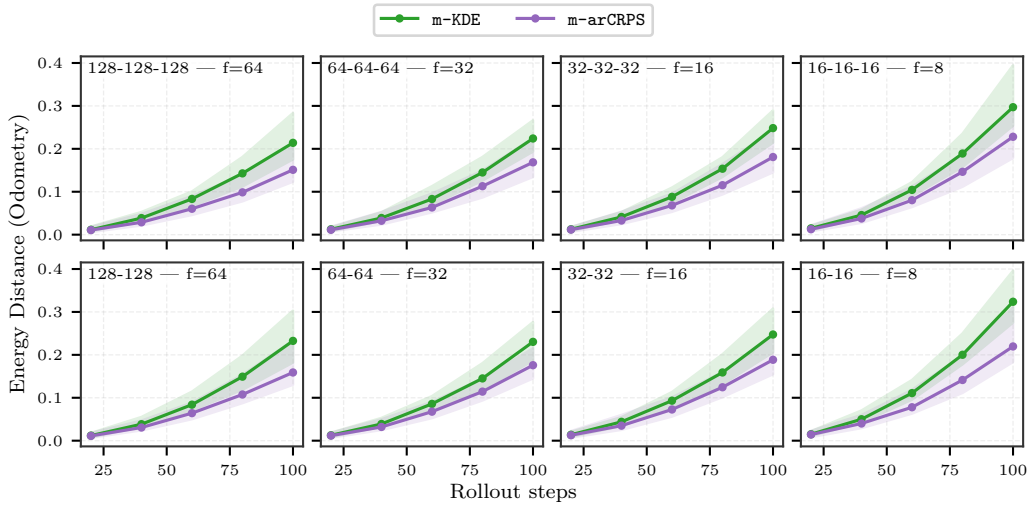
The Fisher sensitivity differs substantially across losses, reflecting how the posterior over the last layer couples with the representation. While `s-NLL` yields relatively uniform and low Fisher sensitivity due to the high noise estimates, multi-step and CRPS-based losses produce structured patterns. In particular, `m-CRPS` exhibits a high Fisher sensitivity at the transitions from slow to fast dynamic sections of the state space while `m-KDE` shows a very pronounced sensitivity around the limit cycle.

In the end, the important evaluation is the one from Figure 2 but still these results show that using the same data, the loss clearly influences both the learned features as well as the learned variational posterior.

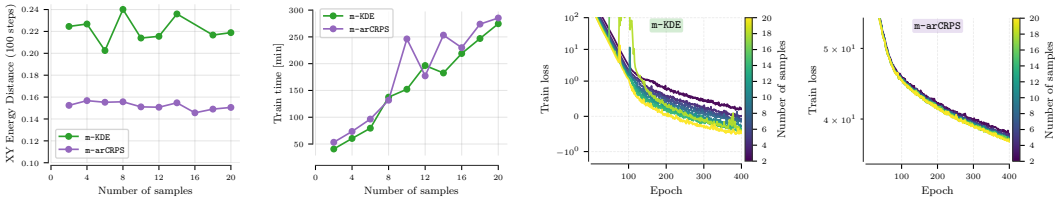
I.3 Sensitivity of `m-KDE` and `m-arCRPS` to Model Capacity

In all experiments, we fixed the model size and capacity and did not perform any dedicated tuning of these aspects. Nevertheless, studying how model size and capacity influence predictive performance is both relevant and insightful. In particular, it is plausible that certain loss functions are better able to exploit increased model capacity. To investigate this, we conduct an ablation study on model capacity for both `m-KDE` and `m-arCRPS` using the Mini Wheelbot. The results are summarized in Fig. 10.

For both three- and two-layer backbone (top and bottom row, respectively), we observe that increasing model capacity consistently improves performance, measured here by the energy distance on the xy-plane across multiple rollouts. Across all model sizes, `m-arCRPS` outperforms `m-KDE`. Interestingly, the smallest model trained with `m-arCRPS` (bottom right) achieves performance comparable to the largest model trained with `m-KDE` (top left). While the absolute model sizes are tiny by modern standards, a reduction of up to $70\times$ in the number of parameters is highly significant, particularly for deployment on embedded hardware.



(a) *Performance of* `m-KDE` *and* `m-arCRPS` *under varying model size.* We evaluate both losses on different sizes of the backbone. `m-arCRPS` consistently outperforms `m-KDE` across the full horizon of 100 time steps.



(b) Performance. (c) Training time. (d) Training losses. (e) Training losses.

Figure 10: *Ablation of model size (top) and the sample size S (bottom) on Mini Wheelbot data.*

I.4 Sensitivity of `m-KDE` and `m-arCRPS` to Sample Size S

In all of our experiments, we use $S = 10$ and did not further tune this for any of the methods. In Fig. 10 (b)–(e), we ablate the sensitivity of the loss functions to different sample sizes S on the Mini Wheelbot data. We can see that both models perform comparable across different S. In Fig. 10 (c) we can further see the $\mathcal{O}(S)$ scaling of the loss functions where small perturbations from the linear trend are due to minor cluster anomalies. Lastly, we can observe that the training with `m-KDE` becomes a bit more unstable for larger S actually resulting in a diverged training for $S = 16$. Training with `m-arCRPS` is very stable and behaves similar for the different S.

I.5 Sensitivity of `m-KDE` and `m-arCRPS` to Horizon H

In Fig. 11, we compare the performance of `m-KDE` and `m-arCRPS` on the Mini Wheelbot data for different horizons H. We find that generally increasing the horizon shows improved performance but it plateaus for $H > 40$. Furthermore, we find that for large horizon, training with `m-KDE` was numerically more unstable resulting in the training runs of $H \in \{70, 90, 100\}$ to diverge. The matched `m-arNLL` trains up to $H = 40$ but diverges for all larger horizons, while `m-arCRPS` remains stable across all tested horizons. In general, we observed that training with a too long horizon can be challenging but having a robust scoring and with this bounded gradients on the noise helped in such long-horizon training scenarios.

I.6 Effect of the regularization factor for `m-CRPS` and `m-arCRPS`

All the results in the main paper were generated with $\lambda = 1/N$. However, we observe that the autoregressive `m-arCRPS` can be overconfident in terms of coverage (also see Tab. 3–6). One way to increase coverage that directly aligns with the Gibbs VI framework is to increase the regularization by a factor K_N resulting in $\lambda = K_N/N$ for the different methods. Note that also [7] use $K_N = 10$

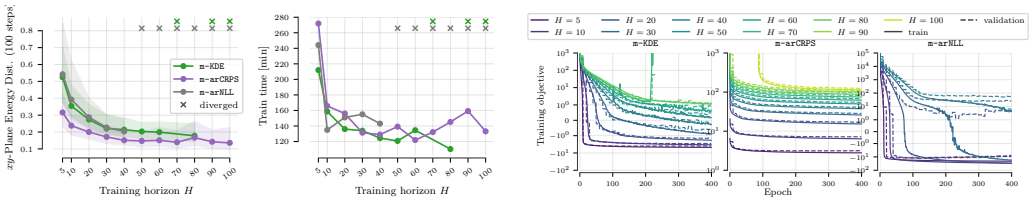


Figure 11: *Sensitivity to different horizon H.* A longer horizon during training increases performance. Training time for small H increases as the effective data set size increase.

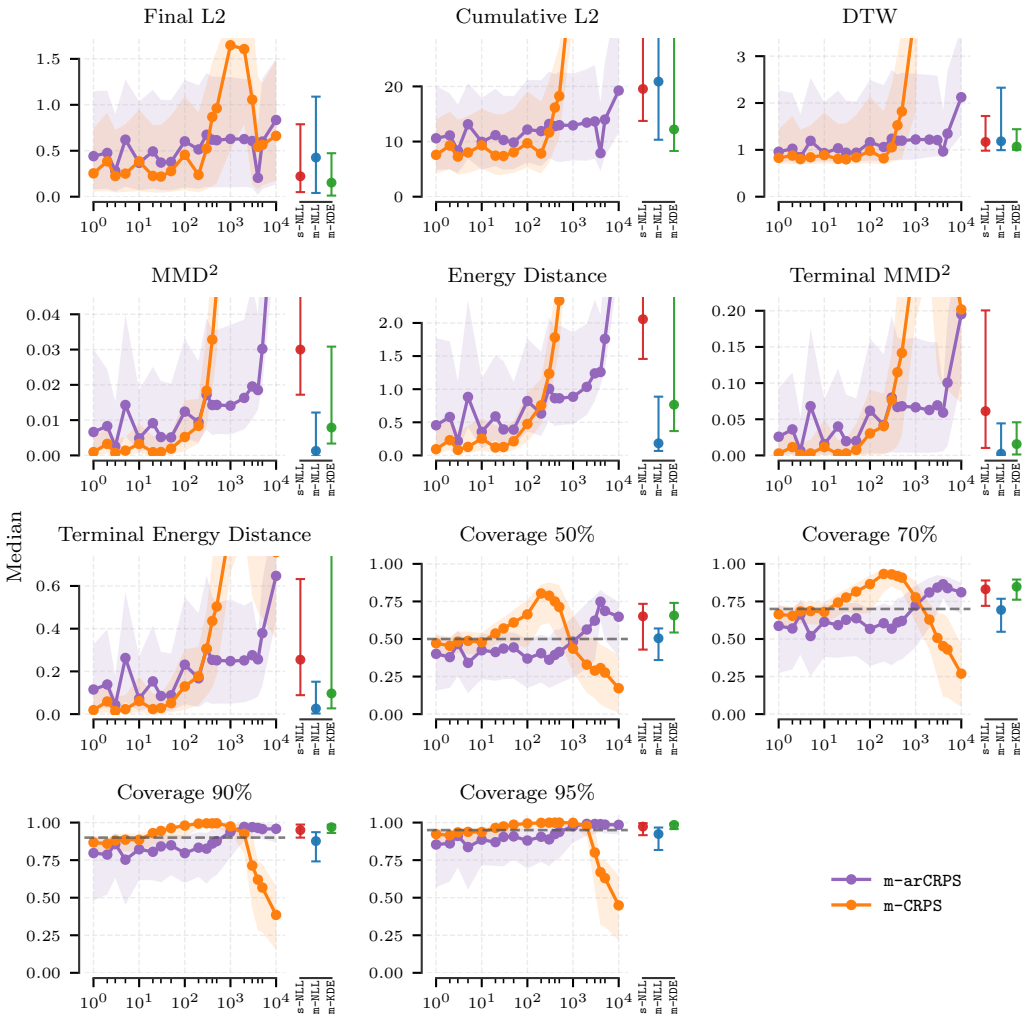


Figure 12: *Influence of changing the KL weight λ for the CRPS baselines.*

for the one-step ELBO. In Fig. 12, we show how increasing K_N influences coverage as well as performance on the VSLC environment, where `m-arCRPS` tended to be overconfident. As the scale of the losses differ between `m-CRPS` and `m-arCRPS`, the magnitudes in which K_N needs to be adapted is different for the two methods. We find that for `m-CRPS`, setting $K_N = 10$ gives good coverage and still strong performance while for `m-arCRPS` $K_N = 10^3$ demonstrates strong coverage at the cost however of slightly reduced performance.

I.7 Gradient norms during optimization

The bounded derivatives of the CRPS in Sec. 3.2 hold at the score level, *i.e.*, with respect to the predictive location and scale. They do not bound gradients through the backbone and, under autoregressive evaluation, backpropagation through time additionally multiplies rollout Jacobians. To assess how the scoring rule affects optimization in practice, we log the pre-clipping gradient norm of the backbone parameters at every training step for all objectives on all benchmark environments and report its 95th percentile per epoch in Fig. 13. All objectives use the same global gradient clipping threshold, which we adopt from the original VBLL implementation [32] and did not tune for any objective (*cf*. Appendix C for all the implementation details).

Teacher forcing. Under teacher forcing, the scoring rule alone already has a strong effect on the gradient magnitude: `m-CRPS` has by far the smallest backbone gradients, whereas `m-NLL` yields gradients that are orders of magnitude larger. Since the per-step contributions only add up without being propagated through a rollout, these large gradients remain benign and both objectives train successfully on all environments.

Autoregressive training. Once the predictions are fed back along the rollout, gradients compound through time. Consequently, `m-arCRPS` exhibits substantially larger backbone gradients than its teacher-forced counterpart `m-CRPS`. Here CRPS attenuates gradient growth under backpropagation through time. Nevertheless, the gradients of `m-arCRPS` remain of the same order as, and typically below, those of the teacher-forced `m-NLL`, and training remains finite on all environments. In contrast, the matched `m-arNLL`, which differs from `m-arCRPS` only in the scoring rule, exhibits the largest gradients on all non-chaotic systems and develops non-finite gradients on all chaotic systems. Since all objectives share the same clipping threshold, clipping alone is insufficient to prevent the instability of the autoregressive log score. We observe the same pattern on the Mini Wheelbot horizon sweep (Fig. 11), where `m-arNLL` trains at shorter horizons but diverges at longer ones, while `m-arCRPS` remains stable across all tested horizons.

Joint optimization of backbone and posterior. Since the backbone and the variational posterior are trained jointly, the last layer performs inference on a representation that shifts during training. To check whether the posterior keeps up with this representation drift, we track the relative change of the backbone parameters, the posterior mean, and the posterior covariance between consecutive epochs, *i.e.*, the norm of the update divided by the norm of the respective parameters (Figs. 14–16). Across all objectives and environments, the posterior mean and covariance continue to update on a similar relative scale as the backbone throughout training. In particular, we do not observe the posterior stalling while the representation continues to move, *i.e.*, the posterior does not decouple from the evolving features. We note that, since AdamW normalizes the update by a running estimate of the gradient scale and we use a constant learning rate, these quantities are largely insensitive to the gradient magnitudes in Fig. 13 and are not a measure of convergence.

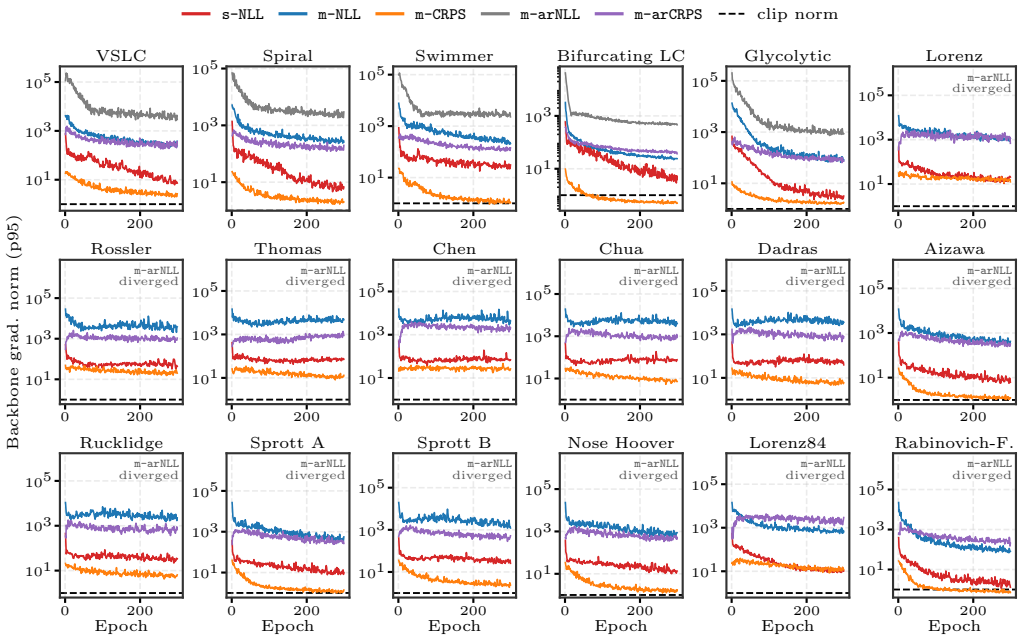


Figure 13: *Backbone gradient norms during training.* 95th percentile of the pre-clipping backbone gradient norm per epoch for all objectives across all benchmark environments; the dashed line marks the clipping threshold. `m-arNLL` develops non-finite gradients on all chaotic systems and is therefore only shown for the non-chaotic environments. CRPS attenuates but does not prevent the gradient growth induced by autoregressive training.

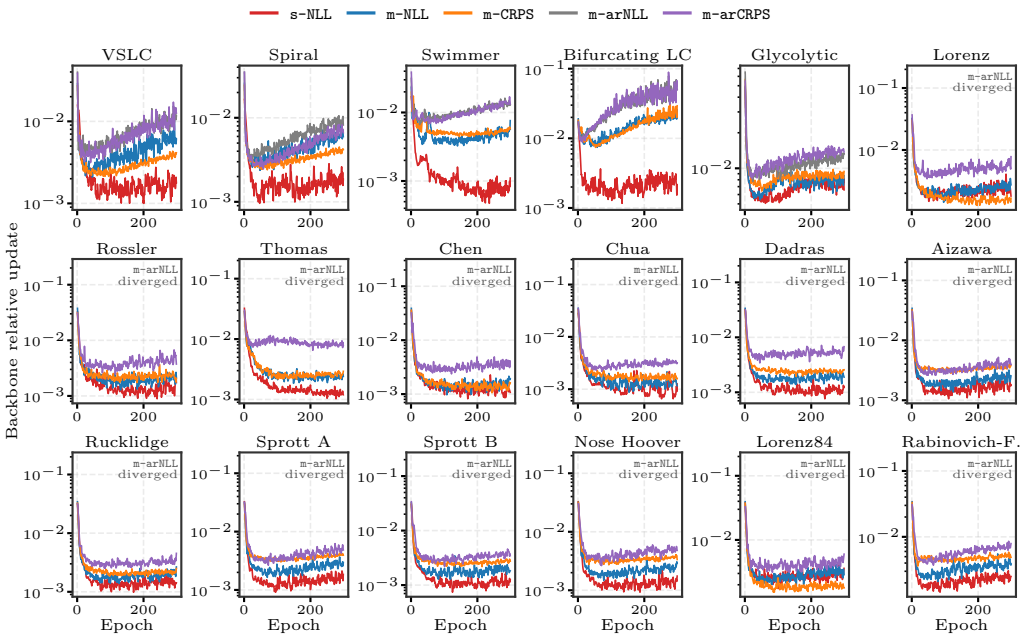


Figure 14: *Relative update of the backbone parameters per epoch.* `m-arNLL` develops non-finite gradients on all chaotic systems and is therefore only shown for the non-chaotic environments.

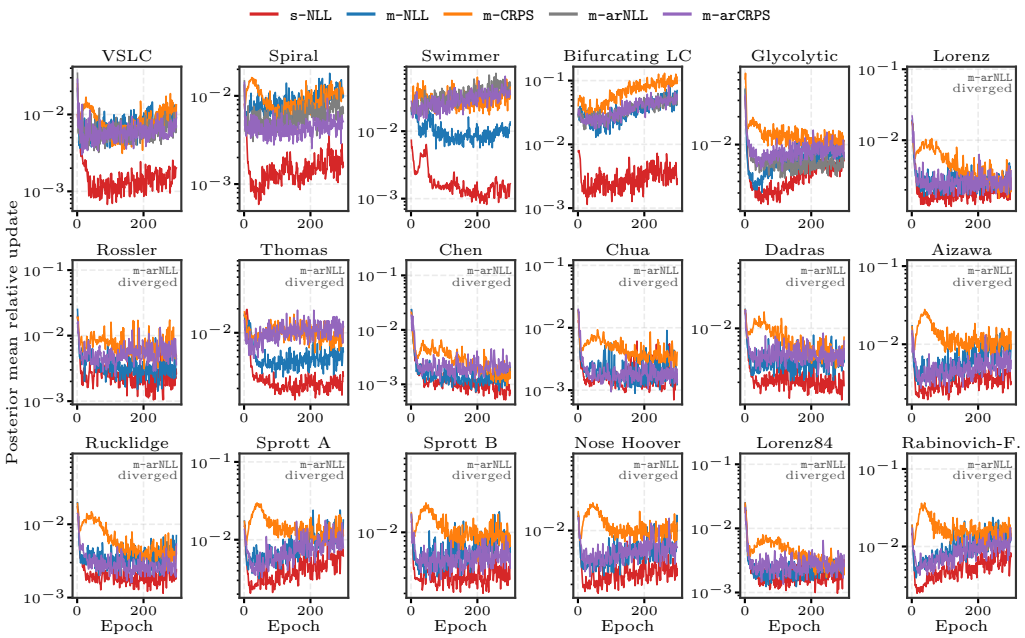


Figure 15: *Relative update of the posterior mean of the last-layer weights per epoch.* The posterior mean keeps updating on a similar relative scale as the backbone (*cf*. Fig. 14).

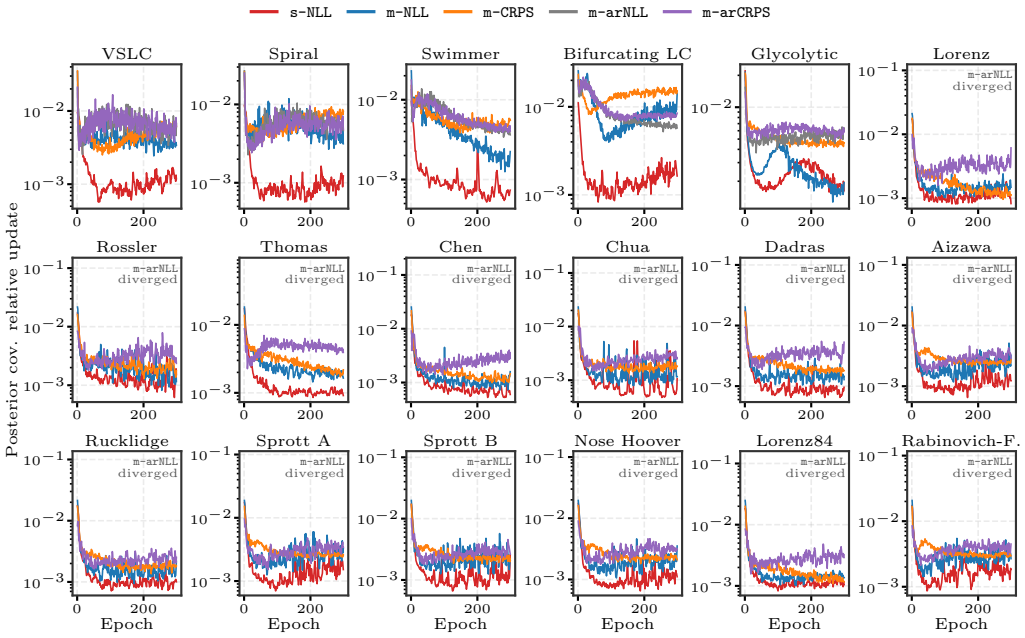


Figure 16: *Relative update of the posterior covariance of the last-layer weights per epoch.* As for the posterior mean, the covariance does not stall while the backbone continues to update (*cf*. Fig. 14).

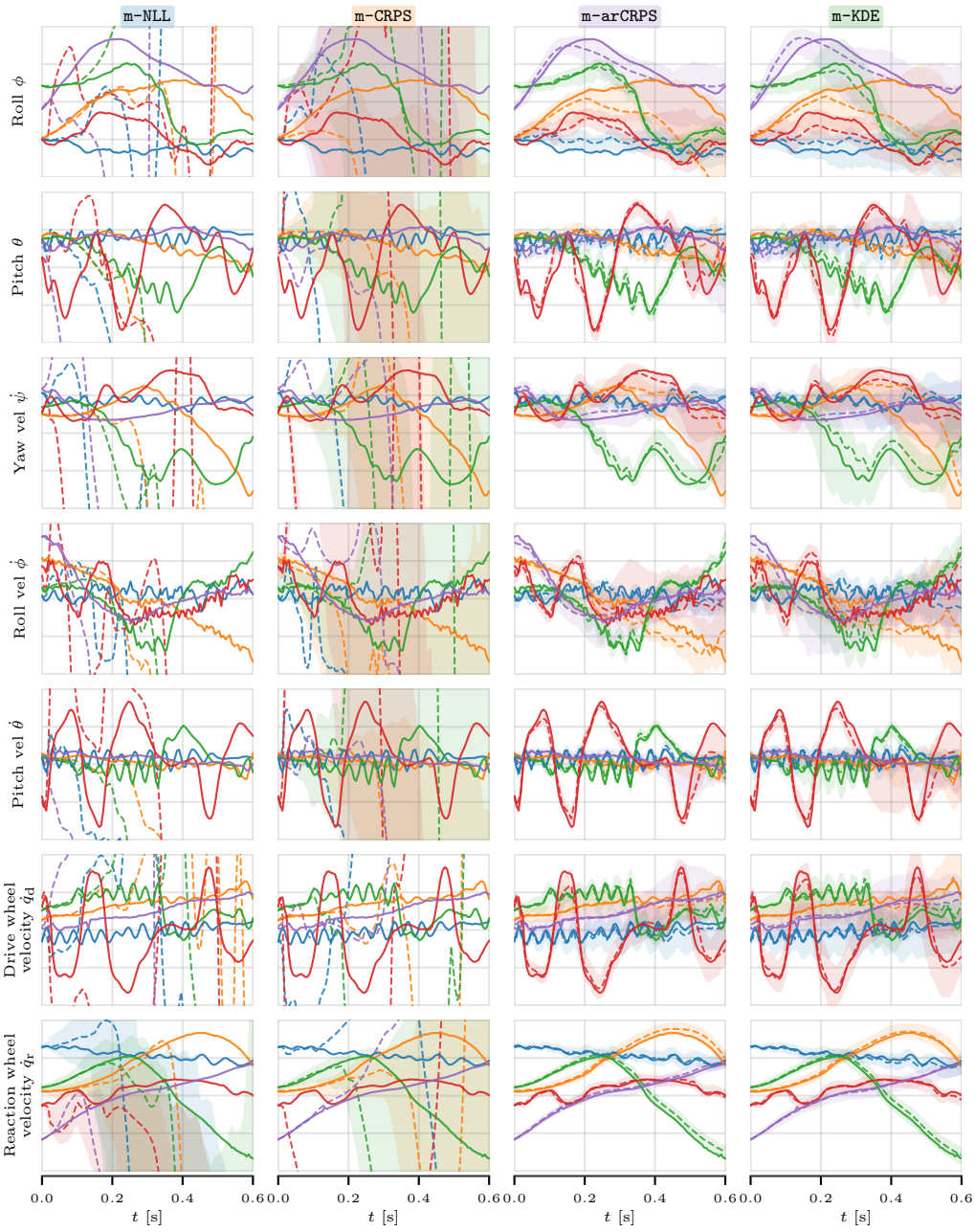


Figure 17: *Rollouts on Mini Wheelbot data.* Figure shows the complete set of states to complement the rollouts shown in Fig. 5 in the main body of the paper.

Table 3: *Results on the benchmarking of Sec. 4.2.* (Part 1/4)

Environment	Metric	s-NLL	m-NLL	m-CRPS	m-arNLL	m-arCRPS	m-KDE
Spiral	Final L2 ↓	$0.325^{+0.236}_{-0.159}$	$0.056^{+0.061}_{-0.031}$	$\mathbf{0.033^{+0.016}_{-0.016}}$	$0.228^{+0.041}_{-0.079}$	$0.122^{+0.020}_{-0.031}$	$0.063^{+0.081}_{-0.047}$
	Cumul. L2 ↓	$28.269^{+23.817}_{-19.836}$	$5.732^{+6.806}_{-2.495}$	$\mathbf{2.605^{+1.451}_{-0.480}}$	$13.834^{+3.687}_{-6.170}$	$7.788^{+1.644}_{-2.293}$	$6.926^{+11.042}_{-4.906}$
	DTW ↓	$0.982^{+0.479}_{-0.273}$	$0.580^{+0.345}_{-0.183}$	$\mathbf{0.351^{+0.191}_{-0.067}}$	$0.732^{+0.166}_{-0.221}$	$0.631^{+0.111}_{-0.076}$	$0.546^{+0.387}_{-0.263}$
	MMD^2 ↓	$0.020^{+0.031}_{-0.018}$	$0.001^{+0.003}_{-0.001}$	$\mathbf{0.000^{+0.000}_{-0.000}}$	$0.005^{+0.003}_{-0.003}$	$0.002^{+0.001}_{-0.001}$	$0.003^{+0.010}_{-0.003}$
	Energy Dist. ↓	$2.867^{+3.429}_{-1.998}$	$0.339^{+0.616}_{-0.199}$	$\mathbf{0.082^{+0.089}_{-0.042}}$	$1.123^{+0.478}_{-0.488}$	$0.440^{+0.240}_{-0.220}$	$(4.519^{+4.034}_{-4.519}) \times 10^{8}$
	Term. MMD^2 ↓	$0.041^{+0.063}_{-0.032}$	$0.001^{+0.003}_{-0.001}$	$\mathbf{0.000^{+0.001}_{-0.000}}$	$0.018^{+0.008}_{-0.010}$	$0.005^{+0.003}_{-0.003}$	$0.003^{+0.009}_{-0.003}$
	Term. Energy ↓	$0.319^{+0.324}_{-0.202}$	$0.030^{+0.037}_{-0.017}$	$\mathbf{0.010^{+0.011}_{-0.006}}$	$0.193^{+0.079}_{-0.093}$	$0.073^{+0.042}_{-0.037}$	$(2.858^{+2.566}_{-2.858}) \times 10^{8}$
	Coverage@50%	$0.433^{+0.384}_{-0.314}$	$0.530^{+0.098}_{-0.210}$	$\mathbf{0.522^{+0.035}_{-0.048}}$	$0.468^{+0.171}_{-0.149}$	$0.432^{+0.067}_{-0.076}$	$0.610^{+0.135}_{-0.194}$
	Coverage@70%	$0.624^{+0.343}_{-0.415}$	$0.727^{+0.094}_{-0.226}$	$\mathbf{0.719^{+0.037}_{-0.045}}$	$0.681^{+0.158}_{-0.151}$	$0.618^{+0.086}_{-0.092}$	$0.822^{+0.094}_{-0.151}$
	Coverage@90%	$0.818^{+0.181}_{-0.389}$	$\mathbf{0.899^{+0.066}_{-0.173}}$	$0.902^{+0.024}_{-0.034}$	$0.898^{+0.077}_{-0.077}$	$0.831^{+0.062}_{-0.072}$	$0.965^{+0.026}_{-0.050}$
	Coverage@95%	$0.871^{+0.129}_{-0.327}$	$0.936^{+0.048}_{-0.149}$	$0.945^{+0.017}_{-0.026}$	$\mathbf{0.946^{+0.046}_{-0.059}}$	$0.891^{+0.054}_{-0.056}$	$0.985^{+0.013}_{-0.030}$
Swimmer	Final L2 ↓	$1.960^{+3.607}_{-1.721}$	$0.779^{+1.961}_{-0.750}$	$\mathbf{0.680^{+1.753}_{-0.624}}$	$1.178^{+2.058}_{-1.035}$	$0.780^{+1.924}_{-0.684}$	$1.397^{+4.511}_{-1.227}$
	Cumul. L2 ↓	$92.705^{+124.538}_{-65.774}$	$34.982^{+86.975}_{-30.736}$	$\mathbf{30.262^{+68.675}_{-24.227}}$	$55.362^{+78.506}_{-43.438}$	$37.375^{+80.045}_{-27.486}$	$68.633^{+158.438}_{-55.948}$
	DTW ↓	$11.353^{+18.860}_{-9.765}$	$3.931^{+12.210}_{-3.721}$	$\mathbf{3.343^{+9.262}_{-3.063}}$	$6.520^{+11.519}_{-5.900}$	$4.214^{+12.508}_{-3.802}$	$7.985^{+24.495}_{-7.113}$
	MMD^2 ↓	$0.139^{+0.236}_{-0.127}$	$0.029^{+0.111}_{-0.029}$	$\mathbf{0.024^{+0.059}_{-0.024}}$	$0.049^{+0.165}_{-0.048}$	$0.024^{+0.075}_{-0.023}$	$0.079^{+0.270}_{-0.076}$
	Energy Dist. ↓	$20.383^{+19.821}_{-18.802}$	$2.026^{+8.273}_{-1.874}$	$\mathbf{1.631^{+3.220}_{-1.427}}$	$3.171^{+8.242}_{-2.686}$	$1.950^{+4.962}_{-1.559}$	$5.201^{+11.835}_{-4.655}$
	Term. MMD^2 ↓	$0.213^{+0.301}_{-0.209}$	$0.068^{+0.213}_{-0.068}$	$\mathbf{0.056^{+0.152}_{-0.057}}$	$0.115^{+0.434}_{-0.113}$	$0.057^{+0.150}_{-0.056}$	$0.107^{+0.389}_{-0.105}$
	Term. Energy ↓	$6.908^{+8.012}_{-6.814}$	$0.377^{+0.966}_{-0.362}$	$\mathbf{0.266^{+0.646}_{-0.242}}$	$0.535^{+1.750}_{-0.499}$	$0.303^{+0.621}_{-0.271}$	$0.952^{+3.545}_{-0.899}$
	Coverage@50%	$0.289^{+0.161}_{-0.136}$	$0.444^{+0.093}_{-0.166}$	$\mathbf{0.496^{+0.125}_{-0.129}}$	$0.506^{+0.142}_{-0.221}$	$0.552^{+0.120}_{-0.141}$	$0.521^{+0.185}_{-0.242}$
	Coverage@70%	$0.427^{+0.213}_{-0.190}$	$0.636^{+0.109}_{-0.207}$	$0.689^{+0.131}_{-0.143}$	$\mathbf{0.692^{+0.152}_{-0.315}}$	$0.750^{+0.120}_{-0.152}$	$0.723^{+0.163}_{-0.261}$
	Coverage@90%	$0.619^{+0.213}_{-0.263}$	$0.838^{+0.067}_{-0.156}$	$0.876^{+0.077}_{-0.111}$	$0.873^{+0.092}_{-0.254}$	$0.913^{+0.068}_{-0.088}$	$\mathbf{0.903^{+0.079}_{-0.167}}$
	Coverage@95%	$0.689^{+0.198}_{-0.270}$	$0.897^{+0.044}_{-0.111}$	$0.923^{+0.054}_{-0.085}$	$0.921^{+0.068}_{-0.181}$	$\mathbf{0.947^{+0.046}_{-0.060}}$	$0.943^{+0.052}_{-0.130}$
Variable Speed Limit Cycle	Final L2 ↓	$0.641^{+0.812}_{-0.562}$	$0.604^{+0.690}_{-0.522}$	$\mathbf{0.506^{+0.527}_{-0.431}}$	$0.947^{+0.698}_{-0.759}$	$0.804^{+0.705}_{-0.659}$	$1.003^{+0.735}_{-0.827}$
	Cumul. L2 ↓	$33.795^{+23.476}_{-18.509}$	$27.046^{+16.965}_{-14.420}$	$\mathbf{22.672^{+10.816}_{-10.660}}$	$39.565^{+16.166}_{-14.162}$	$32.990^{+17.631}_{-16.019}$	$42.588^{+19.399}_{-18.881}$
	DTW ↓	$1.688^{+1.519}_{-0.631}$	$1.573^{+1.186}_{-0.554}$	$\mathbf{1.409^{+0.737}_{-0.384}}$	$2.357^{+1.470}_{-1.136}$	$1.984^{+1.441}_{-0.878}$	$2.449^{+1.759}_{-1.323}$
	MMD^2 ↓	$0.032^{+0.036}_{-0.017}$	$0.010^{+0.016}_{-0.008}$	$\mathbf{0.005^{+0.008}_{-0.005}}$	$0.035^{+0.017}_{-0.014}$	$0.023^{+0.019}_{-0.016}$	$0.053^{+0.033}_{-0.025}$
	Energy Dist. ↓	$2.101^{+1.338}_{-0.806}$	$0.786^{+0.722}_{-0.495}$	$\mathbf{0.405^{+0.561}_{-0.328}}$	$2.111^{+0.723}_{-0.672}$	$1.464^{+1.016}_{-0.894}$	$(5.296^{+7.338}_{-5.296}) \times 10^{13}$
	Term. MMD^2 ↓	$0.057^{+0.072}_{-0.049}$	$0.032^{+0.068}_{-0.032}$	$\mathbf{0.016^{+0.042}_{-0.016}}$	$0.102^{+0.111}_{-0.093}$	$0.081^{+0.123}_{-0.075}$	$0.146^{+0.158}_{-0.135}$
	Term. Energy ↓	$0.225^{+0.208}_{-0.138}$	$0.125^{+0.205}_{-0.109}$	$\mathbf{0.067^{+0.129}_{-0.059}}$	$0.354^{+0.307}_{-0.259}$	$0.285^{+0.355}_{-0.218}$	$(3.770^{+5.097}_{-3.770}) \times 10^{13}$
	Coverage@50%	$0.666^{+0.135}_{-0.259}$	$0.560^{+0.117}_{-0.150}$	$\mathbf{0.465^{+0.059}_{-0.080}}$	$0.544^{+0.124}_{-0.118}$	$0.362^{+0.144}_{-0.137}$	$0.423^{+0.151}_{-0.164}$
	Coverage@70%	$0.836^{+0.091}_{-0.160}$	$0.757^{+0.107}_{-0.147}$	$0.655^{+0.061}_{-0.062}$	$0.761^{+0.085}_{-0.101}$	$0.540^{+0.158}_{-0.182}$	$\mathbf{0.670^{+0.125}_{-0.194}}$
	Coverage@90%	$0.946^{+0.045}_{-0.083}$	$0.920^{+0.058}_{-0.089}$	$0.854^{+0.051}_{-0.054}$	$0.927^{+0.036}_{-0.046}$	$0.766^{+0.126}_{-0.180}$	$\mathbf{0.905^{+0.063}_{-0.089}}$
	Coverage@95%	$0.962^{+0.035}_{-0.070}$	$\mathbf{0.950^{+0.043}_{-0.067}}$	$0.908^{+0.044}_{-0.044}$	$0.958^{+0.026}_{-0.039}$	$0.839^{+0.110}_{-0.163}$	$0.951^{+0.038}_{-0.044}$
Bifurcating Limit Cycle	Final L2 ↓	$0.512^{+0.543}_{-0.190}$	$\mathbf{0.098^{+0.088}_{-0.085}}$	$0.104^{+0.156}_{-0.085}$	$0.152^{+0.208}_{-0.102}$	$0.161^{+0.228}_{-0.118}$	$0.179^{+0.301}_{-0.148}$
	Cumul. L2 ↓	$38.958^{+51.762}_{-17.880}$	$\mathbf{11.930^{+11.627}_{-8.538}}$	$12.839^{+10.576}_{-8.255}$	$14.479^{+6.341}_{-7.294}$	$15.772^{+3.991}_{-8.014}$	$17.685^{+7.511}_{-9.268}$
	DTW ↓	$1.787^{+2.048}_{-0.770}$	$\mathbf{0.687^{+0.709}_{-0.379}}$	$0.786^{+0.663}_{-0.511}$	$1.041^{+0.557}_{-0.681}$	$1.103^{+0.631}_{-0.729}$	$1.173^{+0.553}_{-0.821}$
	MMD^2 ↓	$0.056^{+0.112}_{-0.041}$	$\mathbf{0.003^{+0.006}_{-0.003}}$	$0.003^{+0.004}_{-0.003}$	$0.004^{+0.004}_{-0.003}$	$0.005^{+0.005}_{-0.003}$	$0.005^{+0.010}_{-0.005}$
	Energy Dist. ↓	$2.605^{+4.517}_{-1.642}$	$\mathbf{0.296^{+0.404}_{-0.204}}$	$0.320^{+0.275}_{-0.175}$	$0.352^{+0.242}_{-0.177}$	$0.414^{+0.304}_{-0.213}$	$(4.060^{+3.980}_{-4.060}) \times 10^{8}$
	Term. MMD^2 ↓	$0.115^{+0.136}_{-0.069}$	$0.003^{+0.010}_{-0.005}$	$\mathbf{0.003^{+0.011}_{-0.005}}$	$0.008^{+0.018}_{-0.009}$	$0.009^{+0.014}_{-0.009}$	$0.006^{+0.011}_{-0.007}$
	Term. Energy ↓	$0.375^{+0.206}_{-0.206}$	$\mathbf{0.023^{+0.040}_{-0.019}}$	$0.024^{+0.042}_{-0.019}$	$0.043^{+0.066}_{-0.034}$	$0.046^{+0.059}_{-0.036}$	$(2.769^{+2.723}_{-2.769}) \times 10^{8}$
	Coverage@50%	$0.405^{+0.108}_{-0.196}$	$0.476^{+0.058}_{-0.076}$	$\mathbf{0.482^{+0.051}_{-0.052}}$	$0.449^{+0.051}_{-0.054}$	$0.412^{+0.050}_{-0.062}$	$0.468^{+0.129}_{-0.067}$
	Coverage@70%	$0.586^{+0.128}_{-0.247}$	$0.669^{+0.064}_{-0.093}$	$\mathbf{0.678^{+0.056}_{-0.060}}$	$0.636^{+0.054}_{-0.052}$	$0.592^{+0.053}_{-0.073}$	$0.655^{+0.113}_{-0.071}$
	Coverage@90%	$0.792^{+0.109}_{-0.255}$	$0.865^{+0.049}_{-0.074}$	$\mathbf{0.874^{+0.045}_{-0.048}}$	$0.838^{+0.044}_{-0.038}$	$0.797^{+0.044}_{-0.072}$	$0.851^{+0.056}_{-0.054}$
	Coverage@95%	$0.852^{+0.088}_{-0.240}$	$0.915^{+0.035}_{-0.055}$	$\mathbf{0.923^{+0.029}_{-0.044}}$	$0.894^{+0.032}_{-0.039}$	$0.860^{+0.036}_{-0.064}$	$0.904^{+0.036}_{-0.042}$
Glycolytic Oscillator	Final L2 ↓	$0.096^{+0.080}_{-0.043}$	$0.065^{+0.082}_{-0.038}$	$0.090^{+0.097}_{-0.059}$	$\mathbf{0.038^{+0.064}_{-0.024}}$	$0.042^{+0.095}_{-0.032}$	$0.060^{+0.095}_{-0.042}$
	Cumul. L2 ↓	$8.278^{+6.001}_{-3.756}$	$7.055^{+7.605}_{-3.594}$	$7.715^{+7.113}_{-4.028}$	$\mathbf{5.758^{+5.897}_{-3.397}}$	$6.470^{+5.760}_{-4.258}$	$6.777^{+8.352}_{-4.430}$
	DTW ↓	$0.687^{+0.509}_{-0.252}$	$0.568^{+0.304}_{-0.238}$	$0.727^{+0.568}_{-0.340}$	$\mathbf{0.441^{+0.328}_{-0.228}}$	$0.511^{+0.594}_{-0.298}$	$0.495^{+0.574}_{-0.294}$
	MMD^2 ↓	$0.009^{+0.017}_{-0.008}$	$0.009^{+0.028}_{-0.009}$	$0.010^{+0.019}_{-0.010}$	$\mathbf{0.004^{+0.008}_{-0.004}}$	$0.006^{+0.013}_{-0.006}$	$0.010^{+0.019}_{-0.010}$
	Energy Dist. ↓	$0.308^{+0.479}_{-0.208}$	$0.287^{+0.759}_{-0.224}$	$0.331^{+0.579}_{-0.263}$	$\mathbf{0.201^{+0.270}_{-0.120}}$	$0.268^{+0.355}_{-0.174}$	$0.334^{+0.652}_{-0.255}$
	Term. MMD^2 ↓	$0.030^{+0.060}_{-0.028}$	$0.019^{+0.081}_{-0.022}$	$0.032^{+0.099}_{-0.033}$	$\mathbf{0.008^{+0.024}_{-0.009}}$	$0.016^{+0.071}_{-0.016}$	$0.030^{+0.079}_{-0.030}$
	Term. Energy ↓	$0.033^{+0.058}_{-0.025}$	$0.025^{+0.070}_{-0.020}$	$0.037^{+0.101}_{-0.031}$	$\mathbf{0.015^{+0.023}_{-0.009}}$	$0.022^{+0.064}_{-0.015}$	$0.039^{+0.072}_{-0.032}$
	Coverage@50%	$0.473^{+0.031}_{-0.043}$	$0.482^{+0.040}_{-0.065}$	$\mathbf{0.500^{+0.044}_{-0.052}}$	$0.375^{+0.033}_{-0.039}$	$0.352^{+0.047}_{-0.036}$	$0.463^{+0.064}_{-0.049}$
	Coverage@70%	$0.665^{+0.037}_{-0.047}$	$0.675^{+0.041}_{-0.075}$	$\mathbf{0.696^{+0.049}_{-0.064}}$	$0.544^{+0.049}_{-0.043}$	$0.513^{+0.062}_{-0.048}$	$0.649^{+0.075}_{-0.051}$
	Coverage@90%	$0.862^{+0.033}_{-0.043}$	$0.868^{+0.030}_{-0.057}$	$\mathbf{0.885^{+0.033}_{-0.034}}$	$0.748^{+0.045}_{-0.046}$	$0.714^{+0.061}_{-0.045}$	$0.843^{+0.045}_{-0.039}$
	Coverage@95%	$0.913^{+0.027}_{-0.040}$	$0.917^{+0.026}_{-0.047}$	$\mathbf{0.931^{+0.024}_{-0.029}}$	$0.814^{+0.038}_{-0.043}$	$0.782^{+0.060}_{-0.040}$	$0.896^{+0.033}_{-0.033}$

Table 4: *Results on the benchmarking of Sec. 4.2.* (Part 2/4)

Environment	Metric	s-NLL	m-NLL	m-CRPS	m-arNLL	m-arCRPS	m-KDE
Lorenz	Final L2 ↓	$11.642^{+20.187}_{-7.280}$	$11.676^{+20.290}_{-7.289}$	$12.378^{+21.322}_{-8.270}$	*div.*	$\mathbf{10.755^{+16.145}_{-6.936}}$	$11.388^{+15.737}_{-7.501}$
	Cumul. L2 ↓	$866.827^{+1,085.481}_{-577.921}$	$865.478^{+1,100.729}_{-592.923}$	$888.891^{+1,070.867}_{-607.088}$	*div.*	$\mathbf{805.358^{+1090.680}_{-605.800}}$	$939.147^{+993.302}_{-664.319}$
	DTW ↓	$101.314^{+176.614}_{-74.273}$	$101.067^{+154.610}_{-75.645}$	$106.475^{+163.943}_{-78.152}$	*div.*	$\mathbf{91.795^{+158.475}_{-70.442}}$	$96.209^{+183.364}_{-76.473}$
	MMD^2 ↓	$0.018^{+0.034}_{-0.017}$	$0.016^{+0.049}_{-0.016}$	$\mathbf{0.002^{+0.001}_{-0.002}}$	*div.*	$0.032^{+0.128}_{-0.031}$	$0.060^{+0.163}_{-0.052}$
	Energy Dist. ↓	$32.190^{+41.168}_{-21.357}$	$27.615^{+43.562}_{-19.143}$	$\mathbf{6.585^{+10.295}_{-4.436}}$	*div.*	$40.672^{+91.551}_{-28.389}$	$66.410^{+87.538}_{-36.004}$
	Term. MMD^2 ↓	$0.052^{+0.181}_{-0.050}$	$0.047^{+0.214}_{-0.046}$	$\mathbf{0.007^{+0.036}_{-0.007}}$	*div.*	$0.089^{+0.164}_{-0.087}$	$0.138^{+0.335}_{-0.118}$
	Term. Energy ↓	$3.365^{+5.308}_{-2.186}$	$2.987^{+6.991}_{-2.040}$	$\mathbf{0.755^{+1.506}_{-0.536}}$	*div.*	$4.847^{+4.751}_{-3.608}$	$6.811^{+9.295}_{-4.165}$
	Coverage@50%	$0.600^{+0.342}_{-0.365}$	$0.657^{+0.331}_{-0.453}$	$0.680^{+0.220}_{-0.300}$	*div.*	$\mathbf{0.453^{+0.387}_{-0.280}}$	$0.587^{+0.250}_{-0.253}$
	Coverage@70%	$0.778^{+0.219}_{-0.334}$	$0.807^{+0.193}_{-0.299}$	$0.851^{+0.124}_{-0.185}$	*div.*	$\mathbf{0.644^{+0.321}_{-0.330}}$	$0.785^{+0.176}_{-0.271}$
	Coverage@90%	$\mathbf{0.899^{+0.101}_{-0.220}}$	$0.903^{+0.097}_{-0.217}$	$0.961^{+0.038}_{-0.085}$	*div.*	$0.865^{+0.135}_{-0.229}$	$0.930^{+0.070}_{-0.175}$
	Coverage@95%	$0.923^{+0.077}_{-0.204}$	$0.922^{+0.078}_{-0.166}$	$0.979^{+0.021}_{-0.062}$	*div.*	$0.919^{+0.081}_{-0.151}$	$\mathbf{0.955^{+0.045}_{-0.108}}$
Rossler	Final L2 ↓	$5.526^{+22.964}_{-5.235}$	$3.043^{+8.113}_{-2.770}$	$2.970^{+5.795}_{-2.639}$	*div.*	$\mathbf{2.055^{+4.283}_{-1.853}}$	$3.122^{+11.468}_{-2.601}$
	Cumul. L2 ↓	$308.492^{+1,162.182}_{-272.137}$	$189.383^{+672.712}_{-170.539}$	$175.177^{+690.083}_{-151.841}$	*div.*	$\mathbf{129.353^{+321.938}_{-93.695}}$	$274.057^{+688.777}_{-227.348}$
	DTW ↓	$31.308^{+101.024}_{-28.241}$	$18.353^{+58.595}_{-16.859}$	$18.073^{+56.194}_{-16.150}$	*div.*	$\mathbf{13.218^{+24.423}_{-10.542}}$	$18.566^{+46.830}_{-14.728}$
	MMD^2 ↓	$0.063^{+0.383}_{-0.063}$	$0.038^{+0.252}_{-0.038}$	$0.031^{+0.087}_{-0.031}$	*div.*	$\mathbf{0.012^{+0.047}_{-0.012}}$	$0.058^{+0.340}_{-0.057}$
	Energy Dist. ↓	$42.987^{+176.541}_{-39.017}$	$34.315^{+148.174}_{-31.185}$	$18.782^{+64.006}_{-17.846}$	*div.*	$\mathbf{18.341^{+30.882}_{-14.410}}$	$57.247^{+134.724}_{-50.274}$
	Term. MMD^2 ↓	$0.112^{+0.851}_{-0.111}$	$0.058^{+0.355}_{-0.058}$	$0.039^{+0.165}_{-0.039}$	*div.*	$\mathbf{0.025^{+0.122}_{-0.025}}$	$0.067^{+0.257}_{-0.066}$
	Term. Energy ↓	$5.333^{+20.409}_{-4.941}$	$4.416^{+15.281}_{-4.169}$	$2.226^{+6.353}_{-2.088}$	*div.*	$\mathbf{1.700^{+3.343}_{-1.247}}$	$9.646^{+5.729}_{-8.883}$
	Coverage@50%	$\mathbf{0.589^{+0.293}_{-0.228}}$	$0.730^{+0.200}_{-0.254}$	$0.610^{+0.179}_{-0.205}$	*div.*	$0.633^{+0.237}_{-0.246}$	$0.681^{+0.257}_{-0.283}$
	Coverage@70%	$0.771^{+0.200}_{-0.246}$	$0.844^{+0.145}_{-0.250}$	$\mathbf{0.764^{+0.142}_{-0.223}}$	*div.*	$0.792^{+0.155}_{-0.205}$	$0.838^{+0.161}_{-0.279}$
	Coverage@90%	$\mathbf{0.915^{+0.084}_{-0.164}}$	$0.915^{+0.084}_{-0.187}$	$0.884^{+0.101}_{-0.179}$	*div.*	$0.919^{+0.079}_{-0.124}$	$0.953^{+0.047}_{-0.137}$
	Coverage@95%	$0.939^{+0.061}_{-0.165}$	$0.931^{+0.069}_{-0.161}$	$0.919^{+0.078}_{-0.158}$	*div.*	$\mathbf{0.943^{+0.057}_{-0.102}}$	$0.976^{+0.024}_{-0.088}$
Thomas	Final L2 ↓	$3.401^{+5.074}_{-3.169}$	$\mathbf{2.563^{+2.897}_{-2.298}}$	$3.055^{+5.636}_{-2.677}$	*div.*	$3.856^{+4.805}_{-3.206}$	$4.203^{+4.664}_{-3.408}$
	Cumul. L2 ↓	$248.753^{+343.640}_{-202.328}$	$\mathbf{209.298^{+266.636}_{-148.305}}$	$234.563^{+314.579}_{-180.084}$	*div.*	$311.617^{+277.511}_{-212.877}$	$345.675^{+326.280}_{-248.642}$
	DTW ↓	$24.646^{+55.097}_{-21.664}$	$\mathbf{17.826^{+30.461}_{-14.815}}$	$22.228^{+55.883}_{-19.148}$	*div.*	$30.894^{+49.974}_{-25.269}$	$35.361^{+58.020}_{-29.046}$
	MMD^2 ↓	$0.114^{+0.197}_{-0.101}$	$0.093^{+0.243}_{-0.086}$	$\mathbf{0.085^{+0.229}_{-0.079}}$	*div.*	$0.145^{+0.147}_{-0.082}$	$0.147^{+0.152}_{-0.089}$
	Energy Dist. ↓	$28.102^{+30.720}_{-20.994}$	$23.674^{+39.580}_{-19.348}$	$\mathbf{21.784^{+34.836}_{-17.308}}$	*div.*	$33.095^{+23.488}_{-16.952}$	$33.354^{+22.130}_{-17.009}$
	Term. MMD^2 ↓	$0.280^{+0.633}_{-0.252}$	$0.234^{+0.578}_{-0.221}$	$\mathbf{0.222^{+0.549}_{-0.211}}$	*div.*	$0.303^{+0.254}_{-0.257}$	$0.269^{+0.292}_{-0.222}$
	Term. Energy ↓	$3.000^{+5.585}_{-2.437}$	$2.540^{+5.243}_{-2.099}$	$\mathbf{2.394^{+5.068}_{-2.115}}$	*div.*	$3.183^{+2.566}_{-2.500}$	$2.896^{+2.781}_{-2.122}$
	Coverage@50%	$0.313^{+0.188}_{-0.222}$	$0.317^{+0.258}_{-0.230}$	$0.371^{+0.259}_{-0.250}$	*div.*	$0.426^{+0.172}_{-0.255}$	$\mathbf{0.517^{+0.199}_{-0.228}}$
	Coverage@70%	$0.460^{+0.275}_{-0.321}$	$0.466^{+0.290}_{-0.338}$	$0.537^{+0.234}_{-0.314}$	*div.*	$0.625^{+0.193}_{-0.307}$	$\mathbf{0.731^{+0.176}_{-0.251}}$
	Coverage@90%	$0.648^{+0.268}_{-0.314}$	$0.648^{+0.309}_{-0.433}$	$0.741^{+0.188}_{-0.346}$	*div.*	$0.845^{+0.113}_{-0.241}$	$\mathbf{0.929^{+0.065}_{-0.114}}$
	Coverage@95%	$0.713^{+0.242}_{-0.288}$	$0.708^{+0.274}_{-0.431}$	$0.807^{+0.155}_{-0.362}$	*div.*	$0.903^{+0.084}_{-0.219}$	$\mathbf{0.968^{+0.032}_{-0.060}}$
Chen	Final L2 ↓	$82.268^{+315.884}_{-75.893}$	$81.588^{+313.748}_{-77.290}$	$80.967^{+313.225}_{-77.519}$	*div.*	$78.405^{+317.573}_{-76.964}$	$\mathbf{78.293^{+322.186}_{-76.454}}$
	Cumul. L2 ↓	$2{,}632.050^{+5{,}696.542}_{-2{,}328.641}$	$2{,}472.056^{+5{,}706.440}_{-2{,}275.327}$	$2{,}390.966^{+5{,}547.503}_{-2{,}118.735}$	*div.*	$2{,}256.695^{+5{,}575.695}_{-2{,}071.567}$	$\mathbf{2225.977^{+5487.926}_{-2043.738}}$
	DTW ↓	$455.176^{+1{,}311.867}_{-427.570}$	$435.915^{+1{,}318.478}_{-411.696}$	$426.888^{+1{,}322.254}_{-409.791}$	*div.*	$398.471^{+1{,}324.148}_{-390.604}$	$\mathbf{390.429^{+1312.737}_{-380.294}}$
	MMD^2 ↓	$0.064^{+0.133}_{-0.056}$	$0.032^{+0.086}_{-0.030}$	$\mathbf{0.006^{+0.013}_{-0.006}}$	*div.*	$0.037^{+0.067}_{-0.027}$	$0.066^{+0.082}_{-0.048}$
	Energy Dist. ↓	$92.980^{+125.503}_{-58.505}$	$61.408^{+75.142}_{-36.156}$	$\mathbf{19.539^{+24.231}_{-11.870}}$	*div.*	$71.982^{+61.931}_{-28.136}$	$99.056^{+69.566}_{-41.005}$
	Term. MMD^2 ↓	$0.247^{+0.482}_{-0.199}$	$0.126^{+0.193}_{-0.107}$	$\mathbf{0.030^{+0.086}_{-0.029}}$	*div.*	$0.203^{+0.251}_{-0.134}$	$0.309^{+0.226}_{-0.153}$
	Term. Energy ↓	$10.460^{+14.919}_{-6.142}$	$6.451^{+5.818}_{-3.530}$	$\mathbf{2.297^{+2.553}_{-1.655}}$	*div.*	$9.101^{+8.244}_{-4.028}$	$12.550^{+8.178}_{-4.623}$
	Coverage@50%	$\mathbf{0.451^{+0.422}_{-0.379}}$	$0.655^{+0.258}_{-0.430}$	$0.559^{+0.307}_{-0.387}$	*div.*	$0.680^{+0.235}_{-0.239}$	$0.639^{+0.177}_{-0.199}$
	Coverage@70%	$0.645^{+0.337}_{-0.534}$	$0.809^{+0.182}_{-0.437}$	$\mathbf{0.727^{+0.243}_{-0.405}}$	*div.*	$0.848^{+0.149}_{-0.181}$	$0.837^{+0.159}_{-0.206}$
	Coverage@90%	$0.825^{+0.175}_{-0.560}$	$\mathbf{0.896^{+0.104}_{-0.331}}$	$0.874^{+0.124}_{-0.271}$	*div.*	$0.944^{+0.056}_{-0.144}$	$0.955^{+0.045}_{-0.100}$
	Coverage@95%	$0.876^{+0.124}_{-0.503}$	$0.914^{+0.086}_{-0.254}$	$0.915^{+0.085}_{-0.174}$	*div.*	$\mathbf{0.957^{+0.043}_{-0.121}}$	$0.969^{+0.031}_{-0.085}$
Chua	Final L2 ↓	$3.435^{+2.315}_{-2.673}$	$2.900^{+2.144}_{-2.235}$	$2.906^{+2.000}_{-2.435}$	*div.*	$3.468^{+2.285}_{-2.938}$	$\mathbf{2.845^{+2.474}_{-2.150}}$
	Cumul. L2 ↓	$205.778^{+146.256}_{-150.998}$	$179.212^{+149.897}_{-127.671}$	$\mathbf{168.414^{+151.822}_{-115.994}}$	*div.*	$204.200^{+168.468}_{-160.485}$	$183.526^{+188.956}_{-131.790}$
	DTW ↓	$26.080^{+18.769}_{-21.161}$	$22.381^{+18.361}_{-19.009}$	$21.718^{+17.437}_{-17.885}$	*div.*	$26.225^{+19.703}_{-22.669}$	$\mathbf{21.549^{+20.888}_{-17.960}}$
	MMD^2 ↓	$0.010^{+0.029}_{-0.010}$	$0.005^{+0.015}_{-0.005}$	$\mathbf{0.003^{+0.004}_{-0.003}}$	*div.*	$0.011^{+0.020}_{-0.010}$	$0.016^{+0.038}_{-0.015}$
	Energy Dist. ↓	$5.019^{+9.946}_{-4.365}$	$2.505^{+7.130}_{-2.189}$	$\mathbf{1.556^{+2.238}_{-1.301}}$	*div.*	$5.411^{+7.285}_{-3.550}$	$6.869^{+10.453}_{-4.943}$
	Term. MMD^2 ↓	$0.037^{+0.115}_{-0.037}$	$\mathbf{0.011^{+0.052}_{-0.011}}$	$0.011^{+0.027}_{-0.012}$	*div.*	$0.037^{+0.069}_{-0.035}$	$0.057^{+0.169}_{-0.053}$
	Term. Energy ↓	$0.701^{+1.881}_{-0.645}$	$0.255^{+0.890}_{-0.225}$	$\mathbf{0.253^{+0.483}_{-0.224}}$	*div.*	$0.688^{+1.027}_{-0.555}$	$0.955^{+2.142}_{-0.742}$
	Coverage@50%	$0.406^{+0.155}_{-0.251}$	$0.465^{+0.135}_{-0.223}$	$\mathbf{0.516^{+0.108}_{-0.105}}$	*div.*	$0.569^{+0.201}_{-0.296}$	$0.548^{+0.197}_{-0.285}$
	Coverage@70%	$0.581^{+0.191}_{-0.342}$	$0.651^{+0.137}_{-0.292}$	$\mathbf{0.711^{+0.120}_{-0.105}}$	*div.*	$0.761^{+0.163}_{-0.299}$	$0.747^{+0.163}_{-0.328}$
	Coverage@90%	$0.780^{+0.173}_{-0.383}$	$0.838^{+0.102}_{-0.284}$	$\mathbf{0.894^{+0.068}_{-0.100}}$	*div.*	$0.917^{+0.076}_{-0.189}$	$0.914^{+0.072}_{-0.193}$
	Coverage@95%	$0.840^{+0.136}_{-0.374}$	$0.889^{+0.080}_{-0.249}$	$0.937^{+0.045}_{-0.088}$	*div.*	$0.949^{+0.048}_{-0.140}$	$\mathbf{0.950^{+0.046}_{-0.156}}$

Table 5: *Results on the benchmarking of Sec. 4.2.* (Part 3/4)

Environment	Metric	s-NLL	m-NLL	m-CRPS	m-arNLL	m-arCRPS	m-KDE
Dadras	Final L2 ↓	$\mathbf{0.625^{+1.436}_{-0.504}}$	$0.971^{+1.933}_{-0.808}$	$0.747^{+1.564}_{-0.628}$	*div.*	$1.658^{+2.602}_{-1.316}$	$1.623^{+2.702}_{-1.391}$
	Cumul. L2 ↓	$64.177^{+142.690}_{-48.394}$	$86.223^{+163.618}_{-72.316}$	$\mathbf{63.810^{+143.738}_{-53.438}}$	*div.*	$165.292^{+242.559}_{-123.316}$	$155.975^{+262.996}_{-132.129}$
	DTW ↓	$\mathbf{5.327^{+13.106}_{-4.150}}$	$7.868^{+16.345}_{-6.798}$	$6.245^{+15.722}_{-5.639}$	*div.*	$14.775^{+25.840}_{-12.719}$	$14.076^{+27.889}_{-12.537}$
	MMD^2 ↓	$0.003^{+0.006}_{-0.003}$	$0.004^{+0.006}_{-0.004}$	$\mathbf{0.001^{+0.001}_{-0.001}}$	*div.*	$0.034^{+0.013}_{-0.032}$	$0.021^{+0.128}_{-0.021}$
	Energy Dist. ↓	$5.273^{+8.197}_{-3.674}$	$4.221^{+7.851}_{-3.194}$	$\mathbf{1.905^{+2.392}_{-1.287}}$	*div.*	$19.186^{+14.491}_{-15.053}$	$14.222^{+40.953}_{-10.325}$
	Term. MMD^2 ↓	$0.012^{+0.030}_{-0.011}$	$0.010^{+0.022}_{-0.009}$	$\mathbf{0.003^{+0.007}_{-0.003}}$	*div.*	$0.080^{+0.154}_{-0.077}$	$0.050^{+0.181}_{-0.048}$
	Term. Energy ↓	$0.535^{+0.554}_{-0.393}$	$0.333^{+0.588}_{-0.232}$	$\mathbf{0.198^{+0.203}_{-0.148}}$	*div.*	$1.770^{+2.161}_{-1.460}$	$1.216^{+2.295}_{-0.769}$
	Coverage@50%	$0.414^{+0.262}_{-0.246}$	$0.424^{+0.221}_{-0.235}$	$\mathbf{0.556^{+0.189}_{-0.231}}$	*div.*	$0.316^{+0.367}_{-0.163}$	$0.687^{+0.310}_{-0.343}$
	Coverage@70%	$0.590^{+0.242}_{-0.324}$	$0.613^{+0.218}_{-0.313}$	$\mathbf{0.766^{+0.147}_{-0.215}}$	*div.*	$0.459^{+0.469}_{-0.206}$	$0.875^{+0.125}_{-0.317}$
	Coverage@90%	$0.791^{+0.175}_{-0.360}$	$0.821^{+0.147}_{-0.303}$	$\mathbf{0.937^{+0.052}_{-0.103}}$	*div.*	$0.644^{+0.354}_{-0.257}$	$0.981^{+0.019}_{-0.101}$
	Coverage@95%	$0.852^{+0.139}_{-0.358}$	$0.879^{+0.110}_{-0.275}$	$\mathbf{0.968^{+0.028}_{-0.067}}$	*div.*	$0.713^{+0.287}_{-0.263}$	$0.992^{+0.008}_{-0.039}$
Aizawa	Final L2 ↓	$0.143^{+0.248}_{-0.116}$	$0.174^{+0.237}_{-0.126}$	$0.155^{+0.184}_{-0.112}$	*div.*	$\mathbf{0.106^{+0.085}_{-0.049}}$	$0.136^{+0.149}_{-0.096}$
	Cumul. L2 ↓	$13.840^{+14.133}_{-8.795}$	$16.972^{+21.204}_{-12.383}$	$13.237^{+9.535}_{-9.093}$	*div.*	$\mathbf{10.353^{+13.076}_{-6.823}}$	$13.989^{+16.426}_{-9.214}$
	DTW ↓	$1.032^{+1.071}_{-0.683}$	$1.074^{+1.102}_{-0.630}$	$0.985^{+0.855}_{-0.644}$	*div.*	$\mathbf{0.640^{+0.530}_{-0.336}}$	$0.966^{+0.846}_{-0.601}$
	MMD^2 ↓	$0.003^{+0.007}_{-0.002}$	$0.004^{+0.014}_{-0.004}$	$\mathbf{0.001^{+0.004}_{-0.001}}$	*div.*	$0.002^{+0.004}_{-0.001}$	$0.002^{+0.006}_{-0.002}$
	Energy Dist. ↓	$0.731^{+1.351}_{-0.562}$	$0.891^{+2.336}_{-0.778}$	$\mathbf{0.406^{+0.674}_{-0.318}}$	*div.*	$0.480^{+1.020}_{-0.380}$	$0.630^{+1.445}_{-0.530}$
	Term. MMD^2 ↓	$0.012^{+0.043}_{-0.012}$	$0.015^{+0.042}_{-0.014}$	$0.007^{+0.028}_{-0.007}$	*div.*	$\mathbf{0.006^{+0.017}_{-0.006}}$	$0.009^{+0.012}_{-0.009}$
	Term. Energy ↓	$0.062^{+0.163}_{-0.054}$	$0.072^{+0.174}_{-0.064}$	$0.041^{+0.108}_{-0.032}$	*div.*	$\mathbf{0.037^{+0.055}_{-0.028}}$	$0.046^{+0.054}_{-0.039}$
	Coverage@50%	$0.436^{+0.087}_{-0.160}$	$0.411^{+0.102}_{-0.206}$	$0.536^{+0.116}_{-0.082}$	*div.*	$0.474^{+0.077}_{-0.161}$	$\mathbf{0.505^{+0.091}_{-0.112}}$
	Coverage@70%	$0.622^{+0.105}_{-0.204}$	$0.586^{+0.128}_{-0.277}$	$0.735^{+0.100}_{-0.089}$	*div.*	$0.664^{+0.086}_{-0.210}$	$\mathbf{0.703^{+0.083}_{-0.111}}$
	Coverage@90%	$0.827^{+0.080}_{-0.192}$	$0.785^{+0.115}_{-0.298}$	$0.913^{+0.054}_{-0.043}$	*div.*	$0.860^{+0.070}_{-0.200}$	$\mathbf{0.897^{+0.048}_{-0.066}}$
	Coverage@95%	$0.886^{+0.061}_{-0.168}$	$0.846^{+0.095}_{-0.285}$	$\mathbf{0.951^{+0.034}_{-0.027}}$	*div.*	$0.910^{+0.055}_{-0.176}$	$0.939^{+0.032}_{-0.044}$
Rucklidge	Final L2 ↓	$2.182^{+3.195}_{-1.635}$	$1.433^{+2.534}_{-1.245}$	$1.437^{+2.852}_{-1.293}$	*div.*	$1.934^{+2.242}_{-1.598}$	$\mathbf{1.157^{+1.507}_{-0.946}}$
	Cumul. L2 ↓	$215.658^{+352.235}_{-182.725}$	$140.230^{+265.235}_{-122.588}$	$140.198^{+285.035}_{-130.461}$	*div.*	$160.935^{+249.114}_{-130.320}$	$\mathbf{116.752^{+184.853}_{-85.862}}$
	DTW ↓	$19.645^{+30.644}_{-17.305}$	$10.978^{+22.607}_{-9.856}$	$12.302^{+24.060}_{-11.481}$	*div.*	$15.212^{+23.054}_{-12.925}$	$\mathbf{9.534^{+15.602}_{-7.638}}$
	MMD^2 ↓	$0.007^{+0.019}_{-0.007}$	$0.002^{+0.005}_{-0.002}$	$\mathbf{0.001^{+0.001}_{-0.000}}$	*div.*	$0.004^{+0.007}_{-0.004}$	$0.014^{+0.044}_{-0.013}$
	Energy Dist. ↓	$11.691^{+21.468}_{-9.447}$	$5.409^{+8.360}_{-4.354}$	$\mathbf{2.153^{+3.170}_{-1.734}}$	*div.*	$8.058^{+10.914}_{-5.205}$	$16.142^{+26.409}_{-12.120}$
	Term. MMD^2 ↓	$0.047^{+0.146}_{-0.042}$	$0.016^{+0.053}_{-0.015}$	$\mathbf{0.004^{+0.007}_{-0.003}}$	*div.*	$0.053^{+0.245}_{-0.052}$	$0.094^{+0.171}_{-0.091}$
	Term. Energy ↓	$0.972^{+1.752}_{-0.674}$	$0.403^{+0.674}_{-0.300}$	$\mathbf{0.174^{+0.234}_{-0.132}}$	*div.*	$0.895^{+2.348}_{-0.698}$	$1.586^{+1.390}_{-1.171}$
	Coverage@50%	$0.367^{+0.315}_{-0.242}$	$\mathbf{0.528^{+0.269}_{-0.235}}$	$0.645^{+0.175}_{-0.161}$	*div.*	$0.460^{+0.248}_{-0.223}$	$0.547^{+0.372}_{-0.376}$
	Coverage@70%	$0.529^{+0.353}_{-0.344}$	$0.714^{+0.236}_{-0.275}$	$0.846^{+0.115}_{-0.128}$	*div.*	$0.657^{+0.233}_{-0.277}$	$\mathbf{0.696^{+0.294}_{-0.435}}$
	Coverage@90%	$0.717^{+0.275}_{-0.399}$	$\mathbf{0.878^{+0.118}_{-0.280}}$	$0.968^{+0.031}_{-0.050}$	*div.*	$0.866^{+0.126}_{-0.262}$	$0.857^{+0.143}_{-0.356}$
	Coverage@95%	$0.778^{+0.220}_{-0.401}$	$0.914^{+0.085}_{-0.264}$	$0.986^{+0.014}_{-0.024}$	*div.*	$\mathbf{0.917^{+0.083}_{-0.223}}$	$0.916^{+0.084}_{-0.242}$
Sprott A	Final L2 ↓	$0.213^{+0.182}_{-0.150}$	$0.124^{+0.157}_{-0.093}$	$\mathbf{0.114^{+0.199}_{-0.086}}$	*div.*	$0.192^{+0.211}_{-0.142}$	$0.197^{+0.528}_{-0.141}$
	Cumul. L2 ↓	$15.822^{+16.399}_{-9.695}$	$8.906^{+9.702}_{-6.804}$	$\mathbf{8.506^{+14.522}_{-6.073}}$	*div.*	$14.211^{+20.560}_{-9.903}$	$15.700^{+22.311}_{-11.504}$
	DTW ↓	$1.230^{+1.455}_{-0.918}$	$0.648^{+0.833}_{-0.470}$	$\mathbf{0.600^{+1.028}_{-0.462}}$	*div.*	$1.060^{+1.423}_{-0.848}$	$1.067^{+2.414}_{-0.803}$
	MMD^2 ↓	$0.002^{+0.003}_{-0.002}$	$0.001^{+0.003}_{-0.001}$	$\mathbf{0.001^{+0.002}_{-0.001}}$	*div.*	$0.001^{+0.004}_{-0.001}$	$0.004^{+0.013}_{-0.004}$
	Energy Dist. ↓	$0.929^{+1.124}_{-0.747}$	$0.395^{+0.826}_{-0.332}$	$\mathbf{0.366^{+0.524}_{-0.304}}$	*div.*	$0.568^{+1.019}_{-0.445}$	$1.223^{+2.974}_{-1.020}$
	Term. MMD^2 ↓	$0.009^{+0.011}_{-0.009}$	$0.004^{+0.011}_{-0.004}$	$\mathbf{0.003^{+0.006}_{-0.003}}$	*div.*	$0.005^{+0.016}_{-0.005}$	$0.015^{+0.073}_{-0.015}$
	Term. Energy ↓	$0.127^{+0.164}_{-0.110}$	$0.056^{+0.105}_{-0.051}$	$\mathbf{0.048^{+0.082}_{-0.041}}$	*div.*	$0.070^{+0.203}_{-0.054}$	$0.154^{+0.484}_{-0.129}$
	Coverage@50%	$0.426^{+0.110}_{-0.142}$	$\mathbf{0.471^{+0.067}_{-0.096}}$	$0.574^{+0.094}_{-0.068}$	*div.*	$0.533^{+0.071}_{-0.114}$	$0.543^{+0.112}_{-0.127}$
	Coverage@70%	$0.607^{+0.119}_{-0.175}$	$0.664^{+0.076}_{-0.102}$	$0.776^{+0.090}_{-0.065}$	*div.*	$\mathbf{0.731^{+0.080}_{-0.119}}$	$0.750^{+0.094}_{-0.115}$
	Coverage@90%	$0.811^{+0.101}_{-0.166}$	$0.865^{+0.052}_{-0.094}$	$0.937^{+0.041}_{-0.039}$	*div.*	$\mathbf{0.910^{+0.046}_{-0.083}}$	$0.929^{+0.039}_{-0.061}$
	Coverage@95%	$0.872^{+0.080}_{-0.126}$	$0.917^{+0.040}_{-0.081}$	$0.968^{+0.024}_{-0.023}$	*div.*	$\mathbf{0.949^{+0.030}_{-0.059}}$	$0.964^{+0.022}_{-0.038}$
Sprott B	Final L2 ↓	$0.841^{+1.833}_{-0.722}$	$0.360^{+0.494}_{-0.278}$	$\mathbf{0.297^{+0.534}_{-0.239}}$	*div.*	$0.548^{+0.921}_{-0.441}$	$0.658^{+0.673}_{-0.431}$
	Cumul. L2 ↓	$49.921^{+78.618}_{-41.964}$	$25.833^{+40.166}_{-16.695}$	$\mathbf{20.661^{+38.655}_{-14.484}}$	*div.*	$32.873^{+51.218}_{-24.128}$	$46.435^{+63.514}_{-28.021}$
	DTW ↓	$4.144^{+7.032}_{-3.664}$	$1.812^{+3.254}_{-1.337}$	$\mathbf{1.693^{+3.849}_{-1.306}}$	*div.*	$2.649^{+4.091}_{-2.094}$	$3.613^{+4.297}_{-2.181}$
	MMD^2 ↓	$0.009^{+0.028}_{-0.009}$	$0.002^{+0.005}_{-0.002}$	$\mathbf{0.000^{+0.001}_{-0.000}}$	*div.*	$0.004^{+0.018}_{-0.003}$	$0.006^{+0.014}_{-0.006}$
	Energy Dist. ↓	$4.415^{+11.009}_{-4.112}$	$1.563^{+2.580}_{-1.259}$	$\mathbf{0.842^{+1.226}_{-0.584}}$	*div.*	$2.287^{+5.525}_{-1.666}$	$4.505^{+5.810}_{-3.001}$
	Term. MMD^2 ↓	$0.043^{+0.188}_{-0.043}$	$0.010^{+0.016}_{-0.010}$	$\mathbf{0.002^{+0.007}_{-0.002}}$	*div.*	$0.019^{+0.068}_{-0.018}$	$0.027^{+0.065}_{-0.024}$
	Term. Energy ↓	$0.645^{+1.974}_{-0.610}$	$0.219^{+0.288}_{-0.187}$	$\mathbf{0.103^{+0.144}_{-0.076}}$	*div.*	$0.333^{+0.688}_{-0.261}$	$0.611^{+0.604}_{-0.395}$
	Coverage@50%	$0.390^{+0.215}_{-0.250}$	$\mathbf{0.463^{+0.115}_{-0.161}}$	$0.599^{+0.120}_{-0.120}$	*div.*	$0.567^{+0.161}_{-0.176}$	$0.551^{+0.211}_{-0.279}$
	Coverage@70%	$0.555^{+0.239}_{-0.326}$	$0.653^{+0.128}_{-0.182}$	$0.796^{+0.100}_{-0.115}$	*div.*	$0.762^{+0.127}_{-0.183}$	$\mathbf{0.747^{+0.196}_{-0.295}}$
	Coverage@90%	$0.748^{+0.202}_{-0.392}$	$0.852^{+0.091}_{-0.165}$	$0.946^{+0.039}_{-0.064}$	*div.*	$0.924^{+0.054}_{-0.117}$	$\mathbf{0.904^{+0.094}_{-0.190}}$
	Coverage@95%	$0.810^{+0.166}_{-0.400}$	$0.904^{+0.070}_{-0.139}$	$0.973^{+0.021}_{-0.034}$	*div.*	$\mathbf{0.958^{+0.034}_{-0.091}}$	$0.937^{+0.062}_{-0.161}$

Table 6: *Results on the benchmarking of Sec. 4.2.* (Part 4/4)

Environment	Metric	s-NLL	m-NLL	m-CRPS	m-arNLL	m-arCRPS	m-KDE
Nose Hoover	Final L2 ↓	$0.402^{+0.207}_{-0.321}$	$0.192^{+0.211}_{-0.158}$	$\mathbf{0.191^{+0.148}_{-0.144}}$	*div.*	$0.193^{+0.149}_{-0.148}$	$0.318^{+0.245}_{-0.284}$
	Cumul. L2 ↓	$21.467^{+17.987}_{-16.129}$	$14.180^{+17.436}_{-11.045}$	$\mathbf{13.527^{+13.745}_{-9.536}}$	*div.*	$14.617^{+18.193}_{-8.045}$	$21.230^{+38.851}_{-15.677}$
	DTW ↓	$2.316^{+1.011}_{-1.992}$	$\mathbf{1.063^{+1.892}_{-0.877}}$	$1.172^{+0.818}_{-0.907}$	*div.*	$1.205^{+1.479}_{-0.898}$	$1.903^{+2.600}_{-1.626}$
	MMD^2 ↓	$0.005^{+0.004}_{-0.005}$	$0.002^{+0.002}_{-0.002}$	$\mathbf{0.001^{+0.002}_{-0.001}}$	*div.*	$0.004^{+0.006}_{-0.004}$	$0.007^{+0.016}_{-0.007}$
	Energy Dist. ↓	$1.090^{+1.581}_{-0.890}$	$\mathbf{0.465^{+0.555}_{-0.383}}$	$0.488^{+0.702}_{-0.390}$	*div.*	$1.148^{+1.264}_{-0.883}$	$1.484^{+3.402}_{-1.184}$
	Term. MMD^2 ↓	$0.017^{+0.025}_{-0.016}$	$\mathbf{0.004^{+0.002}_{-0.004}}$	$0.004^{+0.008}_{-0.004}$	*div.*	$0.011^{+0.013}_{-0.011}$	$0.017^{+0.044}_{-0.017}$
	Term. Energy ↓	$0.175^{+0.183}_{-0.149}$	$\mathbf{0.056^{+0.077}_{-0.051}}$	$0.069^{+0.108}_{-0.057}$	*div.*	$0.137^{+0.153}_{-0.110}$	$0.189^{+0.515}_{-0.164}$
	Coverage@50%	$0.429^{+0.099}_{-0.119}$	$0.472^{+0.059}_{-0.070}$	$0.545^{+0.086}_{-0.097}$	*div.*	$\mathbf{0.514^{+0.131}_{-0.142}}$	$0.599^{+0.123}_{-0.212}$
	Coverage@70%	$0.614^{+0.106}_{-0.140}$	$0.663^{+0.066}_{-0.085}$	$0.744^{+0.078}_{-0.102}$	*div.*	$\mathbf{0.709^{+0.124}_{-0.155}}$	$0.804^{+0.104}_{-0.197}$
	Coverage@90%	$0.825^{+0.078}_{-0.104}$	$0.862^{+0.048}_{-0.078}$	$0.918^{+0.043}_{-0.060}$	*div.*	$\mathbf{0.894^{+0.068}_{-0.114}}$	$0.954^{+0.036}_{-0.060}$
	Coverage@95%	$0.884^{+0.062}_{-0.098}$	$0.915^{+0.036}_{-0.057}$	$\mathbf{0.954^{+0.030}_{-0.042}}$	*div.*	$0.938^{+0.045}_{-0.084}$	$0.978^{+0.019}_{-0.033}$
Lorenz84	Final L2 ↓	$3.295^{+2.256}_{-2.279}$	$3.416^{+2.305}_{-2.378}$	$3.564^{+1.675}_{-2.275}$	*div.*	$3.229^{+1.867}_{-2.115}$	$\mathbf{2.947^{+2.012}_{-2.067}}$
	Cumul. L2 ↓	$292.399^{+103.244}_{-160.067}$	$277.622^{+105.910}_{-186.037}$	$277.188^{+108.958}_{-177.776}$	*div.*	$\mathbf{266.729^{+113.328}_{-149.018}}$	$268.412^{+120.224}_{-173.575}$
	DTW ↓	$26.705^{+14.609}_{-20.112}$	$27.702^{+15.586}_{-18.169}$	$26.974^{+14.066}_{-19.040}$	*div.*	$26.787^{+14.864}_{-16.988}$	$\mathbf{25.833^{+15.500}_{-17.385}}$
	MMD^2 ↓	$0.108^{+0.149}_{-0.075}$	$0.082^{+0.151}_{-0.059}$	$\mathbf{0.024^{+0.055}_{-0.022}}$	*div.*	$0.111^{+0.102}_{-0.063}$	$0.110^{+0.101}_{-0.065}$
	Energy Dist. ↓	$12.432^{+13.804}_{-6.361}$	$9.965^{+15.637}_{-5.314}$	$\mathbf{3.509^{+7.024}_{-2.844}}$	*div.*	$12.784^{+7.537}_{-5.237}$	$12.667^{+8.176}_{-6.214}$
	Term. MMD^2 ↓	$0.155^{+0.144}_{-0.112}$	$0.135^{+0.125}_{-0.106}$	$\mathbf{0.065^{+0.143}_{-0.065}}$	*div.*	$0.173^{+0.152}_{-0.125}$	$0.161^{+0.150}_{-0.128}$
	Term. Energy ↓	$1.129^{+0.848}_{-0.637}$	$0.985^{+0.742}_{-0.638}$	$\mathbf{0.501^{+0.931}_{-0.421}}$	*div.*	$1.249^{+1.017}_{-0.809}$	$1.172^{+0.975}_{-0.839}$
	Coverage@50%	$0.531^{+0.245}_{-0.287}$	$0.579^{+0.220}_{-0.285}$	$\mathbf{0.518^{+0.138}_{-0.217}}$	*div.*	$0.420^{+0.254}_{-0.203}$	$0.579^{+0.144}_{-0.168}$
	Coverage@70%	$0.726^{+0.199}_{-0.327}$	$0.765^{+0.169}_{-0.296}$	$\mathbf{0.717^{+0.126}_{-0.248}}$	*div.*	$0.606^{+0.237}_{-0.257}$	$0.765^{+0.086}_{-0.108}$
	Coverage@90%	$0.895^{+0.100}_{-0.213}$	$0.908^{+0.087}_{-0.201}$	$\mathbf{0.905^{+0.069}_{-0.178}}$	*div.*	$0.823^{+0.141}_{-0.247}$	$0.924^{+0.047}_{-0.059}$
	Coverage@95%	$0.927^{+0.072}_{-0.146}$	$0.935^{+0.065}_{-0.174}$	$\mathbf{0.946^{+0.041}_{-0.136}}$	*div.*	$0.885^{+0.098}_{-0.205}$	$0.957^{+0.038}_{-0.047}$
Rabinovich Fabrikant	Final L2 ↓	$0.123^{+0.266}_{-0.101}$	$0.115^{+0.120}_{-0.082}$	$\mathbf{0.096^{+0.093}_{-0.060}}$	*div.*	$0.134^{+0.201}_{-0.106}$	$0.185^{+0.276}_{-0.126}$
	Cumul. L2 ↓	$6.946^{+15.980}_{-5.217}$	$6.783^{+6.836}_{-4.116}$	$\mathbf{5.673^{+5.278}_{-3.373}}$	*div.*	$7.487^{+11.353}_{-5.561}$	$10.493^{+13.605}_{-6.617}$
	DTW ↓	$0.488^{+0.992}_{-0.369}$	$0.615^{+0.937}_{-0.479}$	$\mathbf{0.458^{+0.501}_{-0.326}}$	*div.*	$0.586^{+0.927}_{-0.492}$	$0.827^{+0.876}_{-0.607}$
	MMD^2 ↓	$0.008^{+0.042}_{-0.008}$	$0.004^{+0.008}_{-0.004}$	$\mathbf{0.003^{+0.012}_{-0.003}}$	*div.*	$0.007^{+0.026}_{-0.008}$	$0.013^{+0.043}_{-0.012}$
	Energy Dist. ↓	$0.367^{+1.497}_{-0.309}$	$0.229^{+0.348}_{-0.168}$	$\mathbf{0.196^{+0.459}_{-0.143}}$	*div.*	$0.350^{+0.931}_{-0.284}$	$0.558^{+1.346}_{-0.433}$
	Term. MMD^2 ↓	$0.026^{+0.133}_{-0.028}$	$0.010^{+0.019}_{-0.012}$	$\mathbf{0.010^{+0.043}_{-0.011}}$	*div.*	$0.024^{+0.101}_{-0.026}$	$0.041^{+0.117}_{-0.041}$
	Term. Energy ↓	$0.063^{+0.251}_{-0.058}$	$0.034^{+0.044}_{-0.026}$	$\mathbf{0.031^{+0.084}_{-0.026}}$	*div.*	$0.061^{+0.208}_{-0.054}$	$0.096^{+0.219}_{-0.086}$
	Coverage@50%	$0.465^{+0.057}_{-0.111}$	$0.464^{+0.059}_{-0.051}$	$0.501^{+0.049}_{-0.050}$	*div.*	$0.516^{+0.076}_{-0.083}$	$\mathbf{0.500^{+0.063}_{-0.077}}$
	Coverage@70%	$0.657^{+0.068}_{-0.135}$	$0.655^{+0.052}_{-0.062}$	$\mathbf{0.700^{+0.057}_{-0.059}}$	*div.*	$0.711^{+0.072}_{-0.074}$	$0.699^{+0.060}_{-0.075}$
	Coverage@90%	$0.858^{+0.047}_{-0.077}$	$0.856^{+0.032}_{-0.048}$	$0.891^{+0.040}_{-0.041}$	*div.*	$0.893^{+0.043}_{-0.062}$	$\mathbf{0.894^{+0.034}_{-0.047}}$
	Coverage@95%	$0.910^{+0.035}_{-0.056}$	$0.909^{+0.027}_{-0.038}$	$0.936^{+0.029}_{-0.028}$	*div.*	$0.936^{+0.029}_{-0.044}$	$\mathbf{0.939^{+0.023}_{-0.028}}$

Table 7: Training hyperparameters for the cluster sweep: model configuration.

Hyperparameter	**Value**
Hidden width	64 (128 for Glycolytic oscillator)
Hidden layers	3
Feature dimension	32 (64 for Glycolytic oscillator)
Activation	`mish`
Prior scale	1

Table 8: Training hyperparameters for the cluster sweep: loss-specific training settings.

Loss	Epochs	LR	Batch	Weight decay	Grad. clip	λ	S
`s-NLL`	300	0.001	$64 \times$ H	1.0×10^{-4}	1	$1/\lvert\mathcal{D}_{\text{single}}\rvert$	–
`m-NLL`	300	0.001	64	1.0×10^{-4}	1	$1/\lvert\mathcal{D}_{\text{traj}}\rvert$	–
`m-CRPS`	300	0.001	64	1.0×10^{-4}	1	$1/\lvert\mathcal{D}_{\text{traj}}\rvert$	10
`m-KDE`	300	0.001	64	1.0×10^{-4}	1	$1/\lvert\mathcal{D}_{\text{traj}}\rvert$	10
`m-arNLL`	300	0.001	64	1.0×10^{-4}	1	$1/\lvert\mathcal{D}_{\text{traj}}\rvert$	10
`m-arCRPS`	300	0.001	64	1.0×10^{-4}	1	$1/\lvert\mathcal{D}_{\text{traj}}\rvert$	10

Table 9: Training hyperparameters for the cluster sweep: environment, data, and post-evaluation settings.

Name	Dim.	Norm.	Δt	σ	Traj.	Len.	Stride s	H	Eval steps	States	Samples S for eval
Spiral	2	no	0.1	0.01	300	90	1	40	120	80	100
Swimmer	2	no	0.1	0.01	300	90	1	40	120	80	100
Variable-speed limit cycle	2	no	0.1	0.01	300	90	1	40	120	80	100
Bifurcating limit cycle	2	no	0.05	0.03	300	90	1	40	120	80	100
Glycolytic oscillator	7	yes	0.01	0.01	300	90	1	40	120	80	100
Lorenz	3	yes	0.01	0.01	400	90	20	40	120	80	100
Rossler	3	yes	0.02	0.01	400	90	20	40	120	80	100
Thomas	3	yes	0.02	0.01	400	90	20	40	120	80	100
Chen	3	yes	0.005	0.01	400	90	20	40	120	80	100
Chua	3	yes	0.02	0.01	400	90	20	40	120	80	100
Dadras	3	yes	0.005	0.01	400	90	20	40	120	80	100
Aizawa	3	yes	0.01	0.01	400	90	20	40	120	80	100
Rucklidge	3	yes	0.02	0.01	400	90	20	40	120	80	100
Sprott A	3	yes	0.015	0.01	400	90	20	40	120	80	100
Sprott B	3	yes	0.015	0.01	400	90	20	40	120	80	100
Nose–Hoover	3	yes	0.02	0.01	400	90	20	40	120	80	100
Lorenz 84	3	yes	0.015	0.01	400	90	20	40	120	80	100
Rabinovich–Fabrikant	3	yes	0.01	0.01	400	90	20	40	120	80	100

Table 10: Training hyperparameters for the cluster sweep: Wheelbot model configuration.

Hyperparameter	**Value**
Hidden width	128
Hidden layers	3
Feature dimension	64
Activation	`mish`
Prior scale	1

Table 11: Training hyperparameters for the cluster sweep: Wheelbot loss-specific training settings.

Loss	Epochs	LR	Batch	Weight decay	Grad. clip	H	Stride s	S	S_{val}
`s-NLL`	400	5.0×10^{-4}	256	1.0×10^{-4}	1	–	–	–	–
`m-NLL`	400	5.0×10^{-4}	256	1.0×10^{-4}	1	40	20	10	100
`m-CRPS`	400	5.0×10^{-4}	256	1.0×10^{-4}	1	40	20	10	100
`m-KDE`	400	5.0×10^{-4}	256	1.0×10^{-4}	1	40	20	10	100
`m-arNLL`	400	5.0×10^{-4}	256	1.0×10^{-4}	1	40	20	10	100
`m-arCRPS`	400	5.0×10^{-4}	256	1.0×10^{-4}	1	40	20	10	100

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: All the claims from the introduction are backed by empirical evidence mainly provided in Section 4 as well as in ablations and additional results in Appendix I.

   Guidelines:

   - The answer NA 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 NA 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: We explicitly include Section 6 in the main paper to thoroughly discuss the limitations of this paper and the presented approaches.

   Guidelines:

   - The answer NA 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: For Proposition 1, we include the full proof in Appendix G and list the assumptions for the statement in the Proposition itself. Since Corollary 1 is based on a simple sum over equivalent terms, we do not provide a full proof here.

Guidelines:

- The answer NA 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: We list the most relevant information in the main body of the text in the introduction to Section 4. Furthermore, we give more details in Appendix C and list all the necessary hyperparameters in Tables 7–11.

Guidelines:

- The answer NA 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: All benchmarks including the real-world data are publicly available and we provide instructions to reproduce all experiments. The code is publicly available under the MIT license under https://github.com/brunzema/gibbs-vbll.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: As stated above, we list the most relevant information in the main body of the text in the introduction to Section 4. Furthermore, we give more details, *e.g.* on the type of optimizer, in Appendix C and list all the necessary hyperparameters in Tables 7–11.

   Guidelines:

   - The answer NA 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: All performance plots include quantiles. Furthermore, the statistical significance of the results in Table 1 is provided by a Wilcoxon signed-rank test and the results in Figure 2 (bottom) shows the full distribution over different initial conditions via box plots.

   Guidelines:

   - The answer NA 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: We report runtime in Table 1 for the Mini Wheelbot and include a footnote on the type of compute used to produce the results. Still, the methods and models in this paper are rather small for todays standard and all models can also be trained on a standard laptop CPU.

   Guidelines:

   - The answer NA 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: The paper does not violate any of the guidelines in the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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: [Yes]

Justification: As the paper develops general-purpose loss functions primarily in the context of dynamics learning, we believe the direct societal impact to be limited in scope, though we acknowledge relevance to safety-critical control applications.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper proposed general loss functions for regression. As such, the paper poses no such risks.

Guidelines:

- The answer NA 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: All models were build in Flax NNX which is available under the Apache-2.0 license. Furthermore, we use the chaotic systems from the `dysts` library [26] which is also available under the Apache-2.0 license. We calculate the DTW using `dtaidistance` available under the Apache-2.0 license. The Mini Wheelbot dataset is available under the MIT license.

Guidelines:

- The answer NA 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: The paper does not provide additional assets such as new datasets but builds on existing publicly available simulations and real-world data.

    Guidelines:

    - The answer NA 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: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA 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: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA 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 rigorousness, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The paper does not include LLMs in any part of the method.

Guidelines:

- The answer NA 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 (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.