BAYESIAN EMBEDDING OF PHYSICAL KNOWLEDGE WITHIN CONTINUOUS-TIME LATENT NEURAL MODELS FOR DYNAMIC SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural differential equations provide a flexible framework for learning dynamical systems from data, but can struggle with limited, noisy, and sparse data. Physical relations can provide a valuable inductive bias in these settings, yet existing soft constraints provide no inference-time guarantees, while hard state-space projections can be brittle under noise or misspecified physics. In this paper, we propose a Bayesian framework that embeds physical knowledge as functions of a vector field within a learned latent space, enforced via differentiable weighted projections. Performing joint Bayesian inference over both the model and physics-based parameters propagates uncertainty across the full framework. We show that, at each instant, latent projection approaches the same correction to the observed dynamics as state-space projection in the vanishing-regularization limit, but leaves the null space of the decoder Jacobian unconstrained. Across six forced dynamical systems, latent projection improves accuracy and calibration over unconstrained, soft-constrained and state-space-projected models when the physics is correct, and degrades far less than state-space projection when the embedded physics is misspecified.

1 INTRODUCTION

Dynamical systems across science and engineering are often governed by complex nonlinear phenomena that are only partially understood. When the underlying mechanisms cannot be specified completely, purely mechanistic modelling becomes difficult. Continuous-time neural models, particularly neural differential equations (NDEs) (1; 2; 3), provide a flexible data-driven alternative. By parameterizing the vector field with a neural network and integrating it using a numerical solver, NDEs impose a continuous-time inductive bias while naturally accommodating irregular and asynchronous observations, as commonly encountered in real-world sensor data.

Despite their promise, NDEs can struggle when observations are noisy or cover only a limited region of the state space. In such settings, purely data-driven models may produce predictions that are inconsistent with known physical relations. Existing approaches typically incorporate physics either through soft training penalties (4)or through hard structural constraints, including projections (5; 6; 7; 8), structure-preserving architectures (9; 10), and mechanistic decompositions (11; 12; 13). Soft constraints encourage physical consistency but provide no guarantee that the relation will be satisfied at inference time. Hard constraints provide such guarantees, but can introduce substantial bias when the imposed physics is incomplete or misspecified. This creates a fundamental tension: physical knowledge should constrain the learned dynamics when it is informative, without making the model unnecessarily brittle when that knowledge is imperfect.

Quantifying predictive uncertainty in NDEs presents an additional challenge. Standard approximate Bayesian methods developed for neural networks can perform poorly when applied directly to continuous-time models (14; 15; 16), motivating approaches that instead represent posterior distributions over NDEs through stochastic differential equations (SDEs) (17; 18). These methods provide a principled mechanism for propagating epistemic uncertainty, but their posteriors are typically informed by observations rather than explicit physical relations. Consequently, in data-sparse regions they cannot directly exploit known physics to constrain the posterior predictive distribution. Encoding

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

such knowledge through priors on neural-network parameters is also difficult, because the mapping from parameter space to trajectory space is highly nonlinear and unintuitive (19).

We address these challenges by enforcing physical relations through projection within the learned latent space of a neural ODE, rather than directly in the observed state space. The learned representation introduces flexibility in how the constraint shapes the dynamics, while the projection ensures that the specified relation is satisfied after decoding. We combine this construction with Bayesian inference over the latent dynamics and uncertain physical parameters, enabling both uncertainty quantification and constraint-parameter learning. We further characterize how the projection modifies predictive uncertainty and isolate the reduction in local variability attributable to the embedded physical knowledge.

2 Related Work

Constraining Neural Networks. Approaches for embedding physical knowledge within neural models can broadly be divided into *soft* and hard *constraints*. *Soft constraints*, akin to physics-informed neural networks (PINNs) (4), incorporate physics via penalization, encouraging but not guaranteeing feasibility. *Hard constraints*, by contrast, enforce satisfaction directly by modifying the model or inference procedure, for example through projection operators (20; 21) or structure-preserving architectures (22; 23). Methods such as Hamiltonian and Lagrangian neural networks (24; 9) encode known mechanics directly into the parameterization of the dynamics, allowing quantities such as energy or momentum to be preserved by construction. Universal differential equations (UDEs) (11) similarly combine known mechanistic components with learned residual dynamics or parameterizations, enabling partial incorporation of prior knowledge.

Alternatively, projection-based methods such as neural differential-algebraic equations (DAEs) (8; 25) enforce algebraic constraints by projecting states or vector fields onto feasible manifolds during integration, ensuring invariants are preserved. However, their application is limited as many systems do not exhibit analytic algebraic relations. Manifold-projected neural ODEs (26; 27), and similarly related frameworks such as FINDE (7), impose invariant preservation by explicitly constraining first integrals, either through projection or correction of the learned dynamics, ensuring conservation laws are respected throughout the trajectory. Hard-constrained methods are typically applied directly in the observed state space and can therefore become brittle when observations are noisy or the imposed physics is misspecified. Deep Conservation (23) aids this by embedding *global* conservation laws within a low-dimensional latent manifold, whilst stabilized formulations (6) augment the vector field with corrective terms that drive trajectories toward the constraint manifold; enforcing constraints asymptotically rather than exactly, trading strict feasibility for improved flexibility. Our approach enforces the specified relation at each forward pass within a learned latent representation, allowing the representation itself to adapt when the embedded physics is imperfect.

Approximate Bayesian Inference. Approximate Bayesian inference provides a tractable framework for uncertainty quantification in neural networks by approximating the generally intractable posterior over model parameters to quantify epistemic uncertainty. Typical methods have exploited variational or Laplace approximations (28; 29; 30), ensemble-based and dropout-based approaches (31; 32).

For continuous-time models, these classical approaches often fall short (14; 15). Recent advances have attempted to address this limitation. For instance, Neural ODE Processes augment neural processes with neural ODE-based dynamics to capture uncertainty over functions, while ODE2VAE introduces a VAE-style latent ODE model for probabilistic sequence modelling (33; 34). Other approaches have modeled the evolution of parameters within NDEs using SDEs (17; 18; 35; 36). This SDE-BNN framework shows that imposing a prior continuous-time stochastic process, such as an Ornstein-Uhlenbeck (OU) process, over the network weights yields an infinitely deep Bayesian neural network. Inference is then performed by optimizing an infinite-dimensional ELBO, derived using Girsanov's theorem.

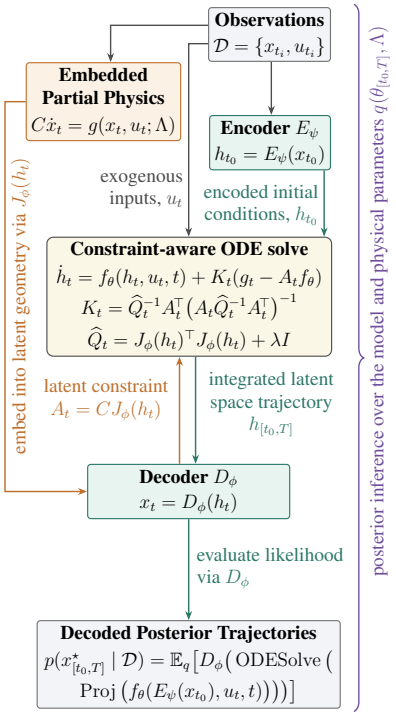
However, existing Bayesian approaches for neural networks and continuous-time models typically define posteriors conditioned solely on observed data, resulting in unconstrained function spaces. Consequently, they lack a principled mechanism for embedding structured physical constraints within the posterior, which can hinder uncertainty reduction in regions with limited data. In contrast, our

framework projects known dynamics within the learned latent space, while jointly inferring a posterior over neural and constraint parameters.

2.1 Our Contributions

Existing approaches either enforce known physical relations as hard constraints, which can introduce bias under noise or misspecification, or impose them softly, sacrificing exact satisfaction. We develop a continuous-time Bayesian framework that enforces partial physical knowledge through projection within a higher-dimensional learned latent representation. Our main contributions are:

- **Latent-space physical projection**. We extend projection-based NDEs to latent dynamics by lifting observation-space constraints through the decoder Jacobian, yielding a closed-form weighted projection in the learned latent space.
- **Bayesian inference over learned and physical dynamics**. We jointly infer the time-varying neural vector-field parameters and uncertain physical parameters using a continuous-time variational formulation with a Girsanov-based path-space KL.
- **Uncertainty characterization**. We characterize the effect of projection on posterior variability and derive a decomposition that isolates the contribution of the embedded physical constraint.
- **Robustness under imperfect information**. Across six controlled dynamical systems, we show that latent projection improves predictive performance relative to unconstrained and state-space alternatives and remains robust under observation noise, partial observability, and physical misspecification.



3 Methodology

Problem Setting. We consider the problem of modelling dynamical systems from time-series data that may be noisy, irregularly sampled, and limited in size. We model these dynamics using an NDE defined over a continuous-time latent state $h_t := h(t) \in \mathbb{R}^{n_h}$, with exogenous inputs $u_t := u(t) \in \mathbb{R}^{n_u}$. The latent dynamics are governed by a neural vector field $f_\theta : \mathbb{R}^{n_h} \times \mathbb{R}^{n_u} \times \mathbb{R} \mapsto \mathbb{R}^{n_h}$, which is Lipschitz in all variables:

$$\frac{dh_t}{dt} = f_\theta(h_t, u_t, t), \qquad h_t = h_{t_0} + \int_{t_0}^{t} f_\theta(h_\tau, u_\tau, \tau)\, d\tau. \tag{1}$$

The latent state provides a flexible representation of the underlying system dynamics, and is particularly advantageous in settings where the observed state is only partially known, irregularly sampled or does not fully capture the Markovian structure of the system (37). The mapping between the observed space $x_t \in \mathbb{R}^{n_x}$ and latent spaces is defined via an encoder–decoder pair, $h_{t_0} = E_\psi(x_{t_0})$ and $x_t = D_\phi(h_t)$, which are set as neural networks $E_\psi : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_h}$, $D_\phi : \mathbb{R}^{n_h} \mapsto \mathbb{R}^{n_x}$.

3.1 Embedding Physical Knowledge within the System

Physical knowledge is often most naturally expressed in terms of observable state derivatives, particularly for driven systems where simple algebraic invariants may not exist. We therefore seek to constrain the latent vector field, $f_\theta(h_t, u_t, t)$ so that the corresponding dynamics in observation space are consistent with known physical relations. For example, a set of n_c relations parameterized by physical parameters, $\Lambda \in \mathbb{R}^d$ may encode partial mass or energy balances. We write these

observation-space constraints as:

$$C\frac{dx_t}{dt} = g(x_t, u_t; \Lambda), \tag{2}$$

where $C \in \mathbb{R}^{n_c \times n_x}$ is a full-row-rank matrix operator that picks out the constrained dynamics. One approach for enforcing this constraints are via projections:

Remark 1 *Let $\hat{\dot{x}}_t$ denote the model-predicted time derivative in the observation space. Projection-based methods enforce consistency with the constraint via an orthogonal projection onto the constraint set, which is the solution to the following constrained optimization problem:*

$$\dot{x}_t^C \in \arg\min_{\dot{x}} \ \|\dot{x} - \hat{\dot{x}}_t\|_2^2 \qquad \textit{s.t.} \quad C\dot{x} - g(x_t, u_t; \Lambda) = 0. \tag{3}$$

Since the optimization problem in Eqn. 3 is a strictly convex quadratic program, it admits a unique minimizer corresponding to the orthogonal projection onto the constraint manifold.

Embedding within Latent Neural DEs. Although the underlying physical system may satisfy the specified relation, observed trajectories need not satisfy it exactly because of measurement noise, discretization error, or model mismatch. Rather than projecting the observed-state dynamics directly, we therefore enforce the relation through the learned latent representation. Because the decoder maps latent states to observations, its Jacobian provides the local differential map between latent and observation-space vector fields. Applying the chain rule gives Eqn. 4, which allows the observation-space physical relation to be lifted into latent derivative space as Eqn. 5.

$$\frac{dx_t}{dt} = J_\phi(h_t)\frac{dh_t}{dt}, \qquad J_\phi(h_t) := \frac{\partial D_\phi(h_t)}{\partial h_t}, \tag{4}$$

$$CJ_\phi(h_t)\dot{h}_t = g(D_\phi(h_t), u_t; \Lambda). \tag{5}$$

We can therefore project the vector field directly in latent space while ensuring that the corresponding decoded dynamics satisfy the specified observation-space relation. Trajectories are integrated within the learned representation, with the decoder determining how latent corrections translate into physical-space corrections. The resulting latent projection is

$$f_\theta^C \in \arg\min_{\dot{h}} \|\dot{h} - f_\theta(h_t, u_t, t)\|_{\hat{Q}_t}^2 \qquad \text{s.t.} \qquad CJ_\phi\dot{h} = g(D_\phi(h_t), u_t; \Lambda). \tag{6}$$

In this formulation, the metric $Q_t = J_\phi(h_t)^\top J_\phi(h_t)$ measures latent corrections by their effect in observation space, where the physical knowledge is defined. Since Q_t has rank at most $n_x < n_h$, the objective in Eqn. 6 is not strictly convex. We therefore use the Tikhonov-regularized metric $\hat{Q}_t = Q_t + \lambda I$ with $0 < \lambda \ll 1$, and compare against a pseudoinverse in Appendix B.6. The KKT conditions give the unique closed-form projection

$$f_\theta^C = f_\theta + \hat{Q}_t^{-1}(CJ_\phi)^\top \left(CJ_\phi\hat{Q}_t^{-1}(CJ_\phi)^\top\right)^{-1}(g - CJ_\phi f_\theta) \ =: \ f_\theta + \mathbf{K}_t(g - \tilde{C}_t f_\theta). \tag{7}$$

The projection is applied at every vector-field evaluation, so the specified relation holds for every posterior realization rather than on average, as with penalty-based approaches. The following result characterizes what the projection changes and what it leaves free.

Proposition 1 (Exact enforcement, decoded equivalence and null-space freedom) *Fix h_t, and assume $J_\phi(h_t)$ and C have full row rank. Let $\Phi_t = J_\phi J_\phi^\top(J_\phi J_\phi^\top + \lambda I)^{-1}$ and $\delta_t = f_\theta^C - f_\theta$. Then, for every $\lambda > 0$:*

- *(i) $CJ_\phi f_\theta^C = g$, so the relation holds exactly in observation space;*
- *(ii) $J_\phi\delta_t = \Phi_t C^\top(C\Phi_t C^\top)^{-1}(g - CJ_\phi f_\theta)$, which converges to the state-space projection (Eqn. 3) of the decoded field $J_\phi f_\theta$ as $\lambda \to 0$;*
- *(iii) $\delta_t \in \mathrm{range}(J_\phi^\top)$, so the component of f_θ in the $(n_h - n_x)$-dimensional null space of J_ϕ is unchanged.*

The proof (Appendix A.4) follows from the push-through identity $\hat{Q}_t^{-1}J_\phi^\top = J_\phi^\top(J_\phi J_\phi^\top + \lambda I)^{-1}$.

Proposition 1 locates the difference between latent and state-space projection. Locally, both correct the observed dynamics in the same way. The latent model, carries $n_h - n_x$ directions that the

projection has no effect on. These evolve under the learned dynamics, retain information about past states, and influence future decoded dynamics through the state dependence of $J_\phi(h_t)$. State-space projection has no such directions: the constraint acts on the full state, so an incorrect relation directly restricts every future trajectory. We hypothesize that this is why latent projection remains empirically robust under corrupted observations and imperfect physics.

Remark 2 (Well-posedness) *Assuming f_θ, and g are C^1, D_ϕ is C^2 and $CJ_\phi(h)$ has full row rank on the region visited by the trajectory, then f_θ^C is locally Lipschitz in h, ensuring local existence and uniqueness of solutions.*

Computational Complexity. At each solver step the projection 7 requires forming the lifted constraint operator $\tilde{C}_t \coloneqq CJ_\phi(h_t) \in \mathbb{R}^{n_c \times n_h}$ and evaluating the Kalman-style gain $\mathbf{K}_t$. Naively, this appears to involve a $n_h \times n_h$ matrix inversion at every step, but the structure of $\hat{Q}_t = J_\phi^\top J_\phi + \lambda I$ allows all dominant computations to be reduced to matrices of size at most $n_x \times n_x$ and $n_c \times n_c$ via the Woodbury identity (see A.5). The resulting overall per-step complexity is

$$\mathcal{O}\big(n_x^2 n_h + n_x^3 + n_c^3 + n_x \,\mathrm{cost}(D_\phi)\big)\,, \tag{8}$$

where $\mathrm{cost}(D_\phi)$ is the cost of a single decoder evaluation. The $n_x^2 n_h$ arises from forming $JJ^\top + \lambda I_{n_x}$ and in most cases is dominant as $n_h \gg n_x$; the $n_x \,\mathrm{cost}(D_\phi)$ term accounts for computing the n_x rows of the decoder Jacobian J_ϕ via reverse-mode automatic differentiation. Linear systems are solved throughout via Cholesky factorization, aiding in well-conditioned solves and preserving smoothness of the vector field.

3.2 Joint Bayesian Inference in Continuous Time

Having defined the constraint-aware latent dynamics, we now place distributions over both the neural vector-field parameters and, where uncertain, the physical constraint parameters. Joint inference over these quantities allows uncertainty in the learned dynamics and the embedded physical knowledge to be propagated into predictions, while also enabling constraint-parameter learning. The latent state satisfies

$$\frac{dh_t}{dt} = f_\theta^C(h_t, u_t, t; \Lambda), \qquad h_{t_0} = E_\psi(x_{t_0}), \tag{9}$$

$$h_t = h_{t_0} + \int_{t_0}^{t} f_\theta^C(h_\tau, u_\tau, \tau; \Lambda) d\tau, \quad x_t = D_\phi(h_t) \tag{10}$$

where f_θ^C denotes the constraint-aware vector field. The decoder D_ϕ induces the observation model, yielding a likelihood, conditional on the constrained latent trajectory $h(\cdot)$. This depends on both neural parameters across the encoder-decoder pair and the vector field as well as constraint variables Λ from the known dynamics. Inference requires marginalising over the posterior $p(\theta, \Lambda, \psi, \phi \mid \mathcal{D})$, which is intractable. We therefore introduce a variational approximation, $q(\cdot)$, and optimize the corresponding evidence lower bound (ELBO).

Inference over both Embedded Knowledge and Parameters.

Assumption 1 *To obtain a tractable approximation, we adopt a mean-field variational family that factorizes over model and constraint components. Specifically, we define*

$$q(\theta_{[t_0,T]}, \Lambda, \psi, \phi) = q_\zeta(\theta_{[t_0,T]})\, q_\xi(\phi, \psi)\, q_\eta(\Lambda). \tag{11}$$

This factorization decouples the neural and constraint parameters; with this decoupling allowing to treat the parameters of the vector field as stochastically evolving with time. This choice balances computational tractability alongside flexibility in representing uncertainty in both the learned dynamics and the embedded physical structure.

Continuous-Time Inference. The encoder, decoder, and embedded constraint parameters are static and can be approximated using standard variational distributions, $q_\xi(\phi, \psi)$ and $q_\eta(\Lambda)$. However, as the latent dynamics evolves continuously through time, we also model the weights on the latent vector

field as a time-dependent stochastic processes. Following (17), to aid stability in the learning process, we adopt an Ornstein-Uhlenbeck (OU) prior together with a learned posterior drift:

$$\text{Prior:} \quad d\theta_t = -\gamma\theta_t\,dt + \sigma\,dW_t, \tag{12}$$

$$\text{Posterior:} \quad d\theta_t = \Xi_\zeta(\theta_t, t)\,dt + \sigma\,dW_t. \tag{13}$$

Here, γ is a hyperparameter controlling mean reversion, while Ξ_ζ is a small hypernetwork parametrizing the posterior drift. This network size can be altered to allow for arbitrarily expressive posteriors. The prior and posterior share the same diffusion function σ, ensuring that the induced path measures are absolutely continuous and the KL divergence is finite under Girsanov's theorem (35).

The resulting path-space KL divergence over the two path space probability measures μ_q, μ_p induced by the prior-posterior combination in Eqns. 12–13 is given by:

$$D_{\mathrm{KL}}(\mu_q\|\mu_p) = \mathbb{E}_{q_\zeta(\theta)}\left[\frac{1}{2}\int_{t_0}^{T}\left\|u\big(\theta_t, t, \zeta\big)\right\|_2^2 dt\right] \tag{14}$$

$$u(\theta_t, t, \zeta) = \sigma^{-1}\left[\Xi_\zeta(\theta_t, t) - (-\gamma)\theta_t\right] \tag{15}$$

Augmented Infinite-Dimensional ELBO. We train the model using an infinite-dimensional ELBO (Eqn. 16) for joint inference over the neural parameters and the embedded physical constraints, with θ modelled as a time-dependent variable. In practise, we treat the encoder and decoder parameters (ψ, ϕ) as point estimates, following common practice in partially Bayesian NDE models (18). This allows us to focus the variational approximation on the dominant sources of uncertainty arising from the latent dynamics and where appropriate, over the embedded constraints, while avoiding the increased computational overhead associated with fully Bayesian treatment of all network parameters.

$$\begin{aligned}\mathcal{L}(\zeta, \eta) = \mathbb{E}_{q_\zeta(\theta), q_\eta(\Lambda}\left[\log p(\mathcal{D} \mid \theta, \Lambda)\right] - \mathbb{E}_{q_\zeta(\theta)}\left[\int_{t_0}^{T}\frac{1}{2}\left\|u(\theta_t, t, \zeta)\right\|_2^2 dt\right] \\ - D_{\mathrm{KL}}(q_\eta(\Lambda) \parallel p(\Lambda)).\end{aligned} \tag{16}$$

This objective decomposes into a data-fit term and a set of regularization terms arising from the KL divergences between the variational posteriors and their respective priors; whilst the likelihood term is conditional on the latent embedded constraint. Maximizing this objective therefore balances fidelity to the observed data with adherence to prior structure in both the dynamics and the embedded physical knowledge, yielding trajectories that are both data-consistent and physically plausible. Similar to many practical implementations of variational models, we utilize β weighting on the regularizing terms. To reduce variance in the gradients, we utilize the Sticking-the-Landing trick when calculating the stochastic integrals (38).

3.3 Predictive Uncertainty under Latent Projection

For each posterior realization, the encoder defines the initial latent state, the constraint-aware vector field generates a projected latent trajectory h_t^C, and the decoder maps this trajectory to the parameters of the observation likelihood. We define a Gaussian likelihood,

$$x_t \mid \theta_{[t_0,T]}, \Lambda, \phi, \psi \sim \mathcal{N}(\mu_t, \Sigma_t), \qquad \mu_t = D_\phi(h_t^C), \quad \Sigma_t = D_{\phi,\Sigma}(h_t^C).$$

Predictive covariance. Repeated application of the law of total covariance, using the mean-field variational posterior, $q_\zeta(\theta_{[t_0,T]})\,q_\xi(\phi, \psi)\,q_\eta(\Lambda)$ gives

$$\begin{aligned}\mathrm{Cov}_q(x_t) = \underbrace{\mathbb{E}_q[\Sigma_t]}_{\text{Likelihood covariance}} + \underbrace{\mathbb{E}_{q_\xi q_\eta}\left[\mathrm{Cov}_{q_\zeta}(\mu_t \mid \phi, \psi, \Lambda)\right]}_{\text{Dynamics contribution}} \\ + \underbrace{\mathbb{E}_{q_\xi}\left[\mathrm{Cov}_{q_\eta}\left(\mathbb{E}_{q_\zeta}[\mu_t \mid \phi, \psi, \Lambda] \,\middle|\, \psi, \phi\right)\right]}_{\text{Physics-parameter contribution}} + \underbrace{\mathrm{Cov}_{q_\xi}\left(\mathbb{E}_{q_\eta q_\zeta}[\mu_t \mid \phi, \psi]\right)}_{\text{Encoder–decoder contribution}}.\end{aligned} \tag{17}$$

Effect of the projection. For a fixed $(h, u, t, \Lambda, \phi, \psi)$, the projected field is $f^C = f_\theta + \mathbf{K}_t(g - \tilde{C}_t f_\theta) = (I - \mathbf{K}_t\tilde{C})f_\theta + \mathbf{K}_t g$. Considering Σ_f as is the covariance of the unprojected drift across weight samples at the fixed evaluation point,

$$\mathrm{Cov}_{q_\zeta}(f^C \mid h, \Lambda, \phi, \psi) = (I - \mathbf{K}_t\tilde{C})\Sigma_f(I - \mathbf{K}_t\tilde{C})^\top. \tag{18}$$

Projection removes local drift variability normal to the constraint set in the Q-metric. In particular, with $P_t = I - \mathbf{K}_t\tilde{C}_t$, we have $\tilde{C}_t P_t = 0$: at fixed state, decoder, and physical parameters, the projected drift has no variability in the constrained combinations of derivatives. This yields a reduction in Q-weighted drift variance (Appendix A.6). During integration, this propagates through the learned dynamics and decoder, to the output variance. The resulting uncertainty reduction is local; with its effect at longer horizons dependent on dynamical amplification. When the embedded physics is misspecified, this may introduce bias. Our experiments assess when this restriction provides a useful inductive bias through improved predictive accuracy and calibration.

4 Numerical Experiments

We evaluate the proposed latent projected neural differential equation across six controlled dynamical systems with externally driven dynamics, each incorporating partial physical knowledge of the form $C\dot{x} = g(x, u; \Lambda)$ (Table 8). This setting tests physical knowledge that remains useful when forcing and dissipation prevent the usual energy or population invariants from being conserved. Synthetic generators provide access to the complete state and true dynamics, allowing us to vary observation corruption and physical misspecification separately. Trajectories are generated from random initial conditions and exogenous input sequences, with 5% additive Gaussian observation noise and an observation dropout probability 0.1 in the base setting. Full system definitions, data-generation procedures, and hyperparameters are provided in Appendix B.2. We assess both forecasting accuracy and probabilistic calibration using mean squared error (MSE), the Continuous Ranked Probability Score (CRPS), and the Weighted Interval Score (WIS), and run experiments across five random seeds.

Baseline Comparisons. We compare against unconstrained, soft constrained (penalized loss function), and hard-projected neural DEs operating in both state and latent spaces, alongside stabilized neural DEs and, where applicable, Hamiltonian and Lagrangian networks (adapted to handle external forcing). All probabilistic neural DE variants use the same SDE-based posterior inference and a Gaussian observation model; the HNN/LNN used a static Gaussian variational posterior, due to numerical stability issues. Under well specified systems, unsurprisingly systems with explicit embedding of physical knowledge shows strong predictive benefits compared to soft/unconstrained methods. In particular latent projection achieves the lowest MSE, CRPS and WIS on five of the six systems and improves on state-space projection approaches on all six, often by an order of magnitude (Table 9). The main exception is the chained pendulum, where our approach places second behind the LNN. State-space projection and stabilization diverged on a subset of seeds for several systems, producing large errors.

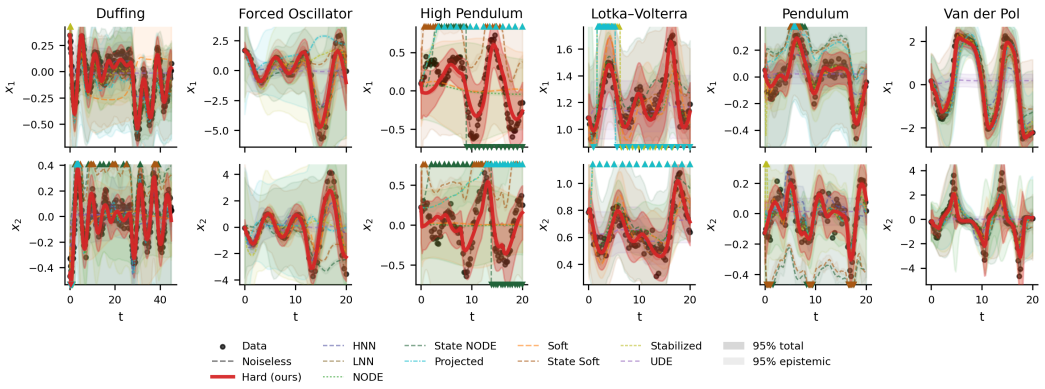


Figure 1: Representative predictions across the six well-specified case studies. Lines denote posterior predictive means, shaded regions show 95% predictive intervals, and markers denote observations.

Robustness to Noise, Dropout and Partial Observability. Data quality is often a limiting factor when modelling real dynamical systems. We vary the observation dropout probability over $\{0.1, 0.3, 0.5\}$, and consider additive Gaussian, state-dependent and temporally ramped heteroscedastic observation noise, as well as SDE-style noise on the dynamics themselves (Appendix B.4). Under

partial observability, only a subset of states is measured: the known physics is still embedded through the full decoded state, the likelihood is evaluated on the measured components only, and the encoder conditions on a short observation history rather than a single initial condition.

Across all corruption settings, latent projection outperforms state-space projection in 55 of 58 system–setting pairs in MSE and 56 of 57 in WIS (Appendix B.5), while retaining exact constraint satisfaction. It also produces better predictive intervals than unconstrained models in 54 of 60 pairs. The largest margins arise under partial observability, where unmeasured components can be represented in the free latent directions rather than forced through the constraint. The interval gains match Section 3.3: projection removes drift variability only in the constrained directions, tightening predictive intervals without restricting the directions that absorb noise and mismatch.

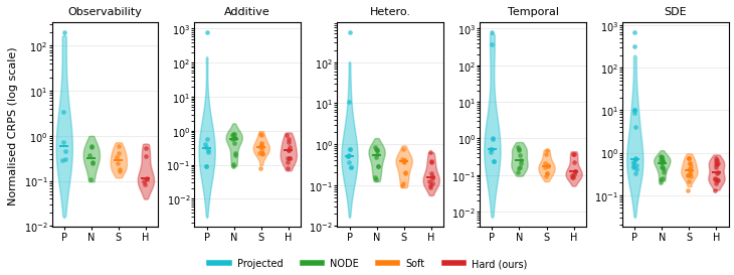


Figure 2: Framework robustness to noise and observability perturbation

Robustness to Physical Misspecification. We next embed misspecified physics, with the constraint parameters Λ either fixed or inferred jointly during training, under four regimes: *informed*, with a correct, narrow prior; *overconfident*, with a narrow prior around an incorrect value; *unsure*, with an incorrect but deliberately wide prior; and *systematic*, where the functional form is wrong so that no parameter recovers the truth. We also consider data-generating processes richer than the embedded physics: an unmodelled forcing term, and a term that enters the dynamics partway through the trajectory.

Latent projection outperforms state-space projection in 44 of 48 misspecified settings in MSE and all 48 in WIS, as state-space projection frequently diverges when the enforced relation is wrong. With informed or unsure priors, it also outperforms unconstrained models in 20 of 24 settings. When Λ is learned, parameter recovery depends on the prior: under an overconfident prior, the posterior remains concentrated near the incorrect value, and performance relative to unconstrained models is mixed (6 of 12). Under systematic misspecification, the latent projection no longer matches, and performance is reduced compared to soft and unconstrained models: we cannot totally remove structural error, but the flexible latent space, allows the trajectory to recover in a manner state-space projection cannot.

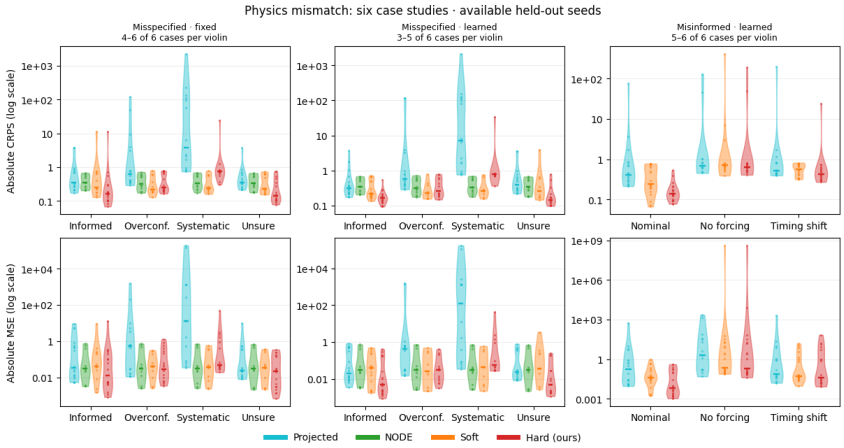


Figure 3: Robustness to physical misspecification

5 DISCUSSION

Effect of Latent Projection. Embedding partial physical knowledge through a projection in the learned latent space improves accuracy and uncertainty calibration relative to state-space projection in our experiments. The model also degrades far less than state-space projection when the embedded physics is misspecified and remains robust when underlying data is noisy. Proposition 1 suggests that robustness arises from the unconstrained null space of J_ϕ rather than from a different instantaneous correction. The decoder shift in Figure 4 is consistent with this: under misspecification, the learned geometry changes so that the free directions absorb the discrepancy.

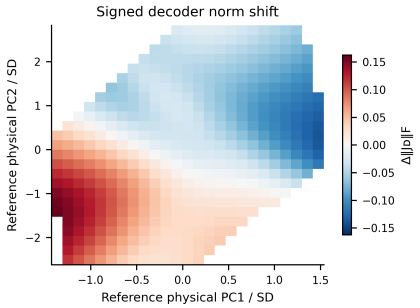


Figure 4: Decoder shift under physical misspecification

Limitations. First, the latent projection weakens parameter identifiability. Although the variational posterior factorizes as $q_\zeta(\theta)q_\eta(\Lambda)$, the projection couples θ and Λ, so different combinations can explain noisy or misspecified data similarly, which can hinder constraint-parameter learning; richer posteriors could retain this dependence. Second, the null-space freedom that confers robustness (Proposition 1) also limits how strongly the physics constrains the dynamics: only the observable constrained directions are fixed, so when $n_h - n_x$ is large, the embedded relation may shape the learned dynamics only weakly. Third, we use point-estimated encoders and decoders, which stabilize training but omit representation uncertainty from the predictive posterior (Appendix B.6). Fourth, the closed-form projection requires relations affine in $\dot{x}$; relations nonlinear in $\dot{x}$ would require iterative or approximate projections, which need not be unique. Finally, the projection is well defined only where $CJ_\phi(h_t)$ has full row rank. Since C has full row rank, at most n_x independent relations can be imposed, and the condition holds whenever J_ϕ has full row rank; it can fail only where the learned decoder is locally degenerate, which we did not observe in our experiments.

Scaling and Scope. The projection costs $\mathcal{O}(n_c^3)$ per step, which is modest for a few relations but may be limiting if using with a discretized PDE system with a constraint per grid cell; matrix-free Jacobian–vector products, sparse or block structure in C and the decoder, subdomain conservation, and rank-revealing compression of redundant constraints are strategies that could be employed to mitigate against this and assist scaling. Our experiments vary corruption, observability and physical fidelity on moderate-dimensional synthetic systems; open directions include testing on real sensor data and larger spatially discretized systems.

6 CONCLUSIONS

We introduced a Bayesian framework that embeds partial physical knowledge into latent neural differential equations through differentiable projections. The framework enforces the specified relation on the latent dynamics, whilst learning the remaining dynamics and inferring uncertain physical parameters. Across six controlled systems, latent projection improves accuracy and predictive intervals over unconstrained, soft-constrained and state-space-projected models, remains robust under observation corruption and partial observation, and degrades far less than state-space projection under misspecified physics. We attempt to explain this robustness via the latent space; the projection corrects the observed dynamics exactly as state-space projection does, while leaving the null space of the decoder free. Together, these results support learned latent representations as a promising approach to making partial physical knowledge useful beyond well-specified, fully observed settings.

AI USE STATEMENT

In this work, we used generative AI tools for assistance with implementing our algorithms and benchmarks as coding agents, assisting with mathematical proofs, editing our paper to improve readability and feedback on research methodology. We have not used generative AI tools for analysing literature, proposing hypotheses or interpreting results. We have reviewed all AI-assisted

work, by manually checking the mathematical proofs and derivations by hand, and auditing the implemented code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility Statement

All derivations and proofs are given in Appendix A (closed-form projections in A.1–A.3, the proof of Proposition 1 in A.4, complexity in A.5 and the uncertainty analysis in A.6–A.7). System definitions, parameters and embedded constraints are in Appendix B; hyperparameters and training settings are in Appendix B.3; corruption and misspecification protocols are in Appendix B.4. An anonymized codebase containing the data generators, all baselines and the experiment scripts is provided as supplementary material. All algorithms were implemented in `Python` and `JAX`, trained on a PC running Ubuntu, with 32 GB DDR5 RAM, an Intel Core i7 CPU and an NVIDIA RTX5090 GPU with 32GB VRAM.

References

[1] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[2] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. In *Advances in Neural Information Processing Systems*, volume 33, pages 6696–6707. Curran Associates, Inc., 2020.

[3] Patrick Kidger. On Neural Differential Equations, February 2022.

[4] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, February 2019.

[5] Muhammad Firmansyah Kasim and Yi Heng Lim. Constants of motion network. *Advances in Neural Information Processing Systems*, 35:25295–25305, December 2022.

[6] Alistair White, Anna Büttner, Maximilian Gelbrecht, Valentin Duruisseaux, Niki Kilbertus, Frank Hellmann, and Niklas Boers. Projected Neural Differential Equations for Learning Constrained Dynamics, October 2024.

[7] Takashi Matsubara and Takaharu Yaguchi. FINDE: Neural Differential Equations for Finding and Preserving Invariant Quantities. In *The Eleventh International Conference on Learning Representations*, September 2022.

[8] Avik Pal, Alan Edelman, and Christopher Rackauckas. Semi-Explicit Neural DAEs: Learning Long-Horizon Dynamical Systems with Algebraic Constraints, 2025.

[9] Michael Lutter, Christian Ritter, and Jan Peters. Deep Lagrangian Networks: Using Physics as Model Prior for Deep Learning. In *International Conference on Learning Representations*, September 2018.

[10] Zhengdao Chen, Jianyu Zhang, Martin Arjovsky, and Léon Bottou. Symplectic Recurrent Neural Networks. In *International Conference on Learning Representations*, September 2019.

[11] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal Differential Equations for Scientific Machine Learning, August 2020.

[12] Yuan Yin, Vincent Le Guen, Jérémie Dona, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari. Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting. *J. Stat. Mech.*, 2021(12):124012, December 2021.

[13] Alistair White, Niki Kilbertus, Maximilian Gelbrecht, and Niklas Boers. Stabilized Neural Differential Equations for Learning Dynamics with Explicit Constraints. *Advances in Neural Information Processing Systems*, 36:12929–12950, December 2023.

[14] Raj Dandekar, Karen Chung, Vaibhav Dixit, Mohamed Tarek, Aslan Garcia-Valadez, Krishna Vishal Vemula, and Chris Rackauckas. Bayesian Neural Ordinary Differential Equations, February 2022.

[15] Katharina Ott, Michael Tiemann, and Philipp Hennig. Uncertainty and Structure in Neural Ordinary Differential Equations, May 2023.

[16] Achille Nazaret and David Blei. Variational Inference for Infinitely Deep Neural Networks. In *Proceedings of the 39th International Conference on Machine Learning*, pages 16447–16461. PMLR, June 2022.

[17] Winnie Xu, Ricky T. Q. Chen, Xuechen Li, and David Duvenaud. Infinitely Deep Bayesian Neural Networks with Stochastic Differential Equations. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 721–738. PMLR, May 2022.

[18] Sergio Calvo Ordoñez, Matthieu Meunier, Francesco Piatti, and Yuantao Shi. Partially Stochastic Infinitely Deep Bayesian Neural Networks. In *Proceedings of the 41st International Conference on Machine Learning*, pages 5436–5452. PMLR, July 2024.

[19] Radford M. Neal. *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer, New York, NY, 1996.

[20] Brandon Amos and J. Zico Kolter. OptNet: Differentiable Optimization as a Layer in Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 136–145. PMLR, July 2017.

[21] Derek Hansen, Danielle C. Maddix, Shima Alizadeh, Gaurav Gupta, and Michael W. Mahoney. Learning Physical Models that Can Respect Conservation Laws. In *Proceedings of the 40th International Conference on Machine Learning*, pages 12469–12510. PMLR, July 2023.

[22] Priya L. Donti, David Rolnick, and J. Zico Kolter. DC3: A learning method for optimization with hard constraints. In *International Conference on Learning Representations*, October 2020.

[23] Kookjin Lee and Kevin T. Carlberg. Deep Conservation: A Latent-Dynamics Model for Exact Satisfaction of Physical Conservation Laws. 35(1):277–285.

[24] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[25] Tue Boesen, Eldad Haber, and Uri Michael Ascher. Neural DAEs: Constrained neural networks, March 2024.

[26] Aaron Lou, Derek Lim, Isay Katsman, Leo Huang, Qingxuan Jiang, Ser Nam Lim, and Christopher M De Sa. Neural Manifold Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, volume 33, pages 17548–17558. Curran Associates, Inc., 2020.

[27] Emma Andersdotter, Daniel Persson, and Fredrik Ohlsson. Equivariant manifold neural odes and differential invariants. *Journal of Machine Learning Research*, 26(290):1–33, 2025.

[28] 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. Curran Associates, Inc., 2021.

[29] Alex Graves. Practical Variational Inference for Neural Networks. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[30] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Network. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1613–1622. PMLR, June 2015.

[31] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[32] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1050–1059. PMLR, June 2016.

[33] Alexander Norcliffe, Cristian Bodnar, Ben Day, Jacob Moss, and Pietro Liò. Neural ODE Processes. In *International Conference on Learning Representations*, October 2020.

[34] Cagatay Yildiz, Markus Heinonen, and Harri Lahdesmaki. ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[35] Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. Scalable Gradients for Stochastic Differential Equations. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 3870–3882. PMLR, June 2020.

[36] Konstantin Hess, Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Bayesian Neural Controlled Differential Equations for Treatment Effect Estimation. *International Conference on Learning Representations*, 2024:52586–52622, May 2024.

[37] Yulia Rubanova, Ricky T. Q. Chen, and David K Duvenaud. Latent Ordinary Differential Equations for Irregularly-Sampled Time Series. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[38] Geoffrey Roeder, Yuhuai Wu, and David K Duvenaud. Sticking the Landing: Simple, Lower-Variance Gradient Estimators for Variational Inference. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[39] Patrick Kidger and Cristian Garcia. Equinox: Neural networks in JAX via callable PyTrees and filtered transformations, October 2021.

A Mathematical Derivations

This appendix provides the derivations underpinning the projection-based constraint enforcement framework introduced in Section 3.

A.1 Solution to the Projection Operator in Closed-Form

We consider the equality-constrained problem defined in Eqn. 3

$$\dot{x}^C \in \arg\min_{\dot{x}} \ \|\dot{x} - f_\theta\|_2^2 \quad \text{s.t.} \quad C\dot{x} = g. \tag{19}$$

This is a strictly convex quadratic program with linear equality constraints, and can therefore be solved via the KKT conditions.

Define the Lagrangian:

$$\mathcal{L}(\dot{x}, \lambda) = \|\dot{x} - f_\theta\|_2^2 + \lambda^\top (C\dot{x} - g). \tag{20}$$

The KKT conditions are

$$\nabla_{\dot{x}} \mathcal{L} = 2(\dot{x} - f_\theta) + C^\top \lambda = 0 \tag{21}$$

$$C\dot{x} - g = 0. \tag{22}$$

From stationarity,

$$\dot{x} = f_\theta - \frac{1}{2} C^\top \lambda. \tag{23}$$

Substituting into primal feasibility gives

$$Cf_\theta - \frac{1}{2} CC^\top \lambda = g. \tag{24}$$

$$\lambda = 2(CC^\top)^{-1}(Cf_\theta - g). \tag{25}$$

Substituting back into the expression for $\dot{x}$ yields

$$\dot{x}^C = f_\theta + C^\top (CC^\top)^{-1}(g - Cf_\theta). \tag{26}$$

A.2 Derivation of Latent Projection from State-Space Knowledge

The physical knowledge is defined in the observable state space:

$$C\dot{x}_t = g(x_t, u_t) \tag{27}$$

Within the latent Neural DE framework, the observable state is reconstructed from the latent state through the decoder

$$x_t = D_\phi(h_t).$$

Applying the chain rule yields

$$\dot{x}_t = \nabla_h D_\phi(h_t)\dot{h}_t = J_\phi(h_t)\dot{h}_t \tag{28}$$

where $J_\phi(h_t) = \nabla_h D_\phi(h_t)$ denotes the decoder Jacobian.

Substituting into Eqn. 27 gives

$$CJ_\phi(h_t)\dot{h}_t = g(D_\phi(h_t), u_t) \tag{29}$$

Thus the physical knowledge induces a linear constraint in latent derivative space:

$$A_t\dot{h}_t = b_t \tag{30}$$

with $A_t = CJ_\phi(h_t), \quad b_t = g(D_\phi(h_t), u_t)$.

Induced Q_t-Norm in Latent Space. The projection should preserve minimum-distance corrections in the *physical space*, where the knowledge is defined.

Define the physical-space correction as

$$\delta x_t = J_\phi(h_t)(\dot{h}_t - f_\theta(h_t, u_t, t)).$$

The natural minimum-distance objective in observable coordinates is therefore

$$\|\delta x_t\|_2^2 = \|J_\phi(h_t)(\dot{h}_t - f_\theta)\|_2^2. \tag{31}$$

$$\|J_\phi(\dot{h}_t - f_\theta)\|_2^2 = (\dot{h}_t - f_\theta)^\top J_\phi^\top J_\phi(\dot{h}_t - f_\theta). \tag{32}$$

Hence the latent-space metric defines the weighted norm

$$Q_t = J_\phi(h_t)^\top J_\phi(h_t). \tag{33}$$

$$\|\dot{h}_t - f_\theta\|_{Q_t}^2 = (\dot{h}_t - f_\theta)^\top Q_t(\dot{h}_t - f_\theta). \tag{34}$$

We ensure this weighting metric is PD and invertible via Tikhonov regularisation. The latent projection problem therefore becomes

$$\dot{h}_t^C = \arg\min_{\dot{h}} \ \|\dot{h} - f_\theta(h_t, u_t, t)\|_{Q_t}^2 \quad \text{s.t.} \quad A_t\dot{h} = b_t \tag{35}$$

The physical knowledge specified in observable coordinates is lifted into latent space through the decoder differential map, while preserving the geometry of the physical state space.

A.3 Closed Form Latent Projection Operator

This problem can be solved in the same way as Eqn 19, using the KKT conditions. Starting with the Lagrangian:

$$\mathcal{L}(\dot{h}, \lambda) = (\dot{h} - f_\theta)^\top Q_t (\dot{h} - f_\theta) + \lambda^\top (A_t \dot{h} - b_t) \tag{36}$$

The KKT conditions give:

$$\nabla_{\dot{h}} \mathcal{L} = 2Q_t(\dot{h} - f_\theta) + A_t^\top \lambda = 0. \tag{37}$$

$$A_t \dot{h} - b_t = 0 \tag{38}$$

It follows that:

$$\boxed{\dot{h}_t^C = f_\theta + Q_t^{-1}(CJ_\phi)^\top \left(CJ_\phi Q_t^{-1}(CJ_\phi)^\top\right)^{-1}(g - CJ_\phi f_\theta)} \tag{39}$$

which is the latent-space projection operator used in the main text.

A.4 Proof of Proposition 1

Write $J := J_\phi(h_t)$, $M := JJ^\top + \lambda I_{n_x}$ and $r := g - CJf_\theta$. Since $J^\top(JJ^\top + \lambda I) = (J^\top J + \lambda I)J^\top$, multiplying on the left by $\hat{Q}_t^{-1}$ and on the right by M^{-1} gives the push-through identity

$$\hat{Q}_t^{-1} J^\top = J^\top M^{-1}. \tag{40}$$

Schur complement. Using Eqn. 40,

$$S := CJ\hat{Q}_t^{-1}J^\top C^\top = C\,JJ^\top M^{-1}\,C^\top = C\Phi_t C^\top, \qquad \Phi_t := JJ^\top M^{-1}.$$

Because $JJ^\top$ and M commute, Φ_t is symmetric, and it is positive definite when J has full row rank. Hence $S \succ 0$ whenever C also has full row rank. Since $\Phi_t = I - \lambda M^{-1}$, this coincides with Eqn. 54 in Appendix A.5.

(iii) Range of the correction. By Eqn. 40, the latent correction is

$$\delta_t = \hat{Q}_t^{-1} J^\top C^\top S^{-1} r = J^\top \left(M^{-1} C^\top S^{-1} r\right) \in \mathrm{range}(J^\top) = \mathrm{null}(J)^\perp.$$

Hence, for the orthogonal projector $\Pi_{\mathcal{N}}$ onto $\mathrm{null}(J)$, $\Pi_{\mathcal{N}} f_\theta^C = \Pi_{\mathcal{N}} f_\theta$. Under full row rank, $\dim \mathrm{null}(J) = n_h - n_x$.

(ii) Decoded correction. Applying J,

$$J\delta_t = JJ^\top M^{-1} C^\top S^{-1} r = \Phi_t C^\top (C\Phi_t C^\top)^{-1} r.$$

If J has full row rank, $JJ^\top \succ 0$ and $\Phi_t \to I$ as $\lambda \to 0$, so $J\delta_t \to C^\top (CC^\top)^{-1}(g - C\,Jf_\theta)$. This is the state-space projection (Eqn. 26) applied to the decoded field $\hat{\dot{x}}_t = Jf_\theta$.

(i) Exact enforcement. Applying C to the decoded correction,

$$CJf_\theta^C = CJf_\theta + C\Phi_t C^\top (C\Phi_t C^\top)^{-1} r = CJf_\theta + r = g,$$

for every $\lambda > 0$.

Remark. If J is rank-deficient, Φ_t converges instead to the orthogonal projector onto $\mathrm{range}(J)$, and the null space has dimension $n_h - \mathrm{rank}(J)$. Part (i) still holds whenever $C\Phi_t C^\top$ is invertible, which is precisely the rank condition on CJ_ϕ stated in the main text.

A.5 Computational Cost of the Projection Operator

We derive the per-step floating-point cost of the latent projection operator and relate it to the cost of the unconstrained vector-field evaluation.

Notation. At a fixed solver step with latent state h_t define

$$J := J_\phi(h_t) = \tfrac{\partial D_\phi(h_t)}{\partial h_t} \ \in \mathbb{R}^{n_x \times n_h}, \tag{41}$$

$$M := JJ^\top + \lambda I_{n_x} \ \in \mathbb{R}^{n_x \times n_x}, \tag{42}$$

$$A := CJ \ \in \mathbb{R}^{n_c \times n_h}, \tag{43}$$

$$S := A\hat{Q}_t^{-1}A^\top \ \in \mathbb{R}^{n_c \times n_c}, \tag{44}$$

$$r := g(D_\phi(h_t), u_t; \Lambda) - Af_\theta \ \in \mathbb{R}^{n_c}. \tag{45}$$

The projection is $f_\theta^C = f_\theta + \hat{Q}_t^{-1}A^\top S^{-1}r$.

Step 1: Eliminating the $n_h \times n_h$ inverse via Woodbury. Applying the Woodbury matrix identity to $\hat{Q}_t = \lambda I_{n_h} + J^\top J$ gives

$$\hat{Q}_t^{-1} = \frac{1}{\lambda}\big(I_{n_h} - J^\top M^{-1} J\big). \tag{46}$$

Proof 1 *Compute $\hat{Q}_t \cdot \frac{1}{\lambda}(I - J^\top M^{-1}J)$:*

$$= \frac{1}{\lambda}(\lambda I + J^\top J)(I - J^\top M^{-1}J) \tag{47}$$

$$= \frac{1}{\lambda}\big(\lambda I - \lambda J^\top M^{-1}J + J^\top J - J^\top JJ^\top M^{-1}J\big). \tag{48}$$

Note $JJ^\top = M - \lambda I$, so $J^\top JJ^\top = J^\top(M - \lambda I)$ and therefore

$$J^\top J - J^\top JJ^\top M^{-1}J = J^\top J - J^\top(M - \lambda I)M^{-1}J \tag{49}$$

$$= J^\top J - J^\top J + \lambda J^\top M^{-1}J \tag{50}$$

$$= \lambda J^\top M^{-1}J. \tag{51}$$

Substituting back yields $\frac{1}{\lambda}(\lambda I) = I$.

Equation 46 means we never form or factor $\hat{Q}_t \in \mathbb{R}^{n_h \times n_h}$; we only ever need $M^{-1}v$ for vectors v, and M is $n_x \times n_x$.

Step 2: Reducing the inner Schur complement S. Substituting 46 into the definition of S:

$$A\hat{Q}_t^{-1}A^\top = \frac{1}{\lambda}\big(AA^\top - AJ^\top M^{-1}JA^\top\big). \tag{52}$$

Since $A = CJ$, we have $AJ^\top = CJJ^\top = C(M - \lambda I)$, and so

$$AJ^\top M^{-1}JA^\top = C(M - \lambda I)M^{-1}(M - \lambda I)^\top C^\top. \tag{53}$$

Denoting $\Phi := (M - \lambda I)M^{-1} = I - \lambda M^{-1}$,

$$A\hat{Q}_t^{-1}A^\top = \frac{1}{\lambda}C\big(I - \Phi\,\Phi^\top\lambda\big)\,C^\top = C\big(I - \lambda M^{-1}\big)\,C^\top. \tag{54}$$

This is computed from one Cholesky of M and two triangular solves of width n_c, giving an $n_c \times n_c$ result with no $n_h \times n_h$ operations, Cholesky-factoring S costs $\mathcal{O}(n_c^3)$.

Step 3: Applying $\hat{Q}_t^{-1}A^\top$ to a vector. Given the solve $y = S^{-1}r \in \mathbb{R}^{n_c}$ (already computed), the correction is

$$\hat{Q}_t^{-1}A^\top y = \frac{1}{\lambda}\big(A^\top y - J^\top M^{-1}(JA^\top y)\big). \tag{55}$$

This requires two matrix-vector multiplies in $\mathbb{R}^{n_h}$ and one triangular solve of dimension n_x.

Table 1: Per-step floating-point operations for the projection operator. W_D denotes the number of decoder parameters. Dominant terms are highlighted.

Operation	**Cost**	**Notes**
Decoder Jacobian $J = J_\phi(h_t)$	$\mathcal{O}(n_x\, W_D)$	n_x reverse-mode AD passes
Form $M = JJ^\top + \lambda I_{n_x}$	$\mathcal{O}(n_x^2 n_h)$	**dominant** when $n_h \gg n_x$
Cholesky of M	$\mathcal{O}(n_x^3)$	reused for all later solves
Form S via 54 and add R_t	$\mathcal{O}(n_c n_x^2 + n_c^2 n_x)$	$\mathcal{O}(n_c n_x)$ solves of width n_x
Cholesky of $S \in \mathbb{R}^{n_c \times n_c}$	$\mathcal{O}(n_c^3)$	
Residual $r = g - Af_\theta$	$\mathcal{O}(n_h n_c + n_x n_c)$	
Solve $Sy = r$	$\mathcal{O}(n_c^2)$	
Apply $\hat{Q}_t^{-1} A^\top y$	$\mathcal{O}(n_h n_c + n_h n_x + n_x^2)$	via 46
Total projection	$\mathcal{O}(n_x^2 n_h + n_x^3 + n_c^3 + n_x W_D)$	

Step 4: Full per-step FLOP table. For our experimental setting ($n_h = 32$, $n_x \le 4$, $n_c \le 2$, $W_D \approx 1088$ for a one-hidden-layer MLP of width 32):

- Projection overhead: $\sim n_x^2 n_h \approx 512$ FLOPs.
- Single f_θ evaluation (width-32, 1 hidden layer): $\approx 2 \times 32 \times 32 + 2 \times 32 \times n_h \approx 4096$ FLOPs.
- Ratio: $\approx 15\%$ additional cost per step.

Step 5: Full training-step cost. The total cost per ELBO gradient step with S Monte Carlo samples and T_Δ solver steps is

$$\mathcal{C}_{\text{step}} = S \cdot T_\Delta \cdot \Big[\underbrace{C_{\text{SDE}}}_{\text{baseline SDE-BNN}} + \underbrace{\mathcal{O}(n_x^2 n_h + n_x^3 + n_c^3 + n_x W_D)}_{\text{projection}} \Big], \tag{56}$$

where C_{SDE} includes evaluating f_θ, the hypernetwork drift μ_ζ, and the stochastic integrals for the path-space KL. The projection term is a constant multiplicative overhead relative to C_{SDE}; it does not introduce additional ODE calls or increase the effective trajectory length.

A.6 Predictive Uncertainty and Its Propagation under Projection

We derive the predictive covariance decomposition and then examine how the embedded projection affects uncertainty during integration. All expectations are conditional on the training data, observed initialization history $\mathcal{H}_0$, and prescribed controls.

Step 1: Predictive distribution. We marginalize use the variational family defined in Sec 4 $q_\zeta(\theta_{[t_0,T]})\, q_\eta(\Lambda)\, q_\xi(\phi, \psi)$. Each posterior realization defines

$$\begin{aligned} h_{t_0} &= E_\psi(x_0), \quad \dot{h}_t^C = f_{\theta_t}^C(h_t^C, u_t, t; \phi, \Lambda), \\ \mu_t &= D_\phi(h_t^C), \quad \Sigma_t = D_{\phi,\Sigma}(h_t^C), \end{aligned} \tag{57}$$

Which assumes we set the observation likelihood as

$$x_t \mid \Lambda, \psi, \phi, \theta_{[t_0,T]} \sim \mathcal{N}(\mu_t, \Sigma_t).$$

Then, marginalising over the variational posterior, the predictive mean and covariance satisfy

$$\mathbb{E}[x_t] = \mathbb{E}_q[\mu_t], \qquad \text{Cov}(y_t) = \mathbb{E}_q[\Sigma_t] + \text{Cov}_q(\mu_t). \tag{58}$$

Step 2: Contributions from uncertain parameter groups. Applying the law of total covariance first conditional on (ψ, ϕ, Λ) and then on ψ, ϕ gives

$$\begin{aligned}\mathrm{Cov}_q(x_t) = \quad & \underbrace{\mathbb{E}_q[\Sigma_t]}_{\text{Likelihood covariance}} + \underbrace{\mathbb{E}_{q_\xi q_\eta}\left[\mathrm{Cov}_{q_\zeta}(\mu_t \mid \phi, \psi, \Lambda)\right]}_{\text{Dynamics contribution}} \\ & + \underbrace{\mathbb{E}_{q_\xi}\left[\mathrm{Cov}_{q_\eta}\left(\mathbb{E}_{q_\zeta}[\mu_t \mid \phi, \psi, \Lambda] \,\middle|\, \psi, \phi\right)\right]}_{\text{Physics-parameter contribution}} + \underbrace{\mathrm{Cov}_{q_\xi}\left(\mathbb{E}_{q_\eta q_\zeta}[\mu_t \mid \phi, \psi]\right)}_{\text{Encoder–decoder contribution}}.\end{aligned} \tag{59}$$

Step 3: Local action of the physics projection. Fix (h, u, t, ϕ, Λ) and assume that $CJ_\phi(h)$ has full row rank. With $\hat{Q}_t = J_\phi(h)^\top J_\phi(h) + \lambda I$, $\lambda > 0$, the projected vector field is

$$f_\theta^C = f_\theta + \hat{Q}_t^{-1}(CJ_\phi)^\top \left(CJ_\phi \hat{Q}_t^{-1}(CJ_\phi)^\top\right)^{-1}(g - CJ_\phi f_\theta). \tag{60}$$

Multiplying by CJ_ϕ gives

$$CJ_\phi f_\theta^C = CJ_\phi f_\theta + (g - CJ_\phi f_\theta) = g. \tag{61}$$

Thus the decoded mean dynamics satisfy the imposed relation for every weight realization.

At this fixed evaluation point, the decoder Jacobian, projection metric, and physical right-hand side are deterministic. Subtracting the posterior mean therefore yields

$$f_\theta^C - \mathbb{E}_{q_\zeta}[f_\theta^C] = \left[I - \hat{Q}_t^{-1}(CJ_\phi)^\top \left(CJ_\phi \hat{Q}_t^{-1}(CJ_\phi)^\top\right)^{-1} CJ_\phi\right]\left(f_\theta - \mathbb{E}_{q_\zeta}[f_\theta]\right). \tag{62}$$

Writing $\Sigma_f = \mathrm{Cov}_{q_\zeta}(f_\theta)$, the covariance of the projected drift is

$$\begin{aligned}\mathrm{Cov}_{q_\zeta}(f_\theta^C) = & \left[I - \hat{Q}_t^{-1}(CJ_\phi)^\top \left(CJ_\phi \hat{Q}_t^{-1}(CJ_\phi)^\top\right)^{-1} CJ_\phi\right]\Sigma_f \\ & \times \left[I - \hat{Q}_t^{-1}(CJ_\phi)^\top \left(CJ_\phi \hat{Q}_t^{-1}(CJ_\phi)^\top\right)^{-1} CJ_\phi\right]^\top.\end{aligned} \tag{63}$$

The corresponding decoded-drift covariance is

$$\mathrm{Cov}_{q_\zeta}(J_\phi f_\theta^C) = J_\phi \,\mathrm{Cov}_{q_\zeta}(f_\theta^C) J_\phi^\top.$$

In particular,

$$CJ_\phi \,\mathrm{Cov}_{q_\zeta}(f_\theta^C)(CJ_\phi)^\top = 0,$$

so variation across sampled drifts is confined to directions that preserve the specified relation.

Expanding Equation 63 and using cyclic invariance of the trace gives

$$\begin{aligned}& \mathrm{tr}(\hat{Q}_t \Sigma_f) - \mathrm{tr}\left(\hat{Q}_t \,\mathrm{Cov}_{q_\zeta}(f_\theta^C)\right) \\ & \quad = \mathrm{tr}\left[\left(CJ_\phi \hat{Q}_t^{-1}(CJ_\phi)^\top\right)^{-1} CJ_\phi \Sigma_f (CJ_\phi)^\top\right] \geq 0.\end{aligned} \tag{64}$$

The inequality follows because the inverse constraint matrix is positive definite and $CJ_\phi \Sigma_f (CJ_\phi)^\top$ is positive semidefinite. Projection therefore reduces centered local drift variance in the $\hat{Q}_t$-metric. Since $\hat{Q}_t = J_\phi^\top J_\phi + \lambda I$, this metric combines decoded-drift variation with regularized latent-drift variation. The guarantee does not imply a reduction in every output direction.

Implications for predictive uncertainty. Equation 64 establishes that projection cannot increase centered local drift variance in the $\hat{Q}_t$-metric. The reduction is strictly positive whenever $CJ_\phi \Sigma_f (CJ_\phi)^\top \neq 0$, meaning that the unprojected drift varies in combinations of derivatives constrained by the physical relation. Projection removes this variability while retaining drift variation compatible with the imposed physics.

During a rollout, these projected drifts are integrated from the encoded initial state and mapped through the decoder. Their accumulated uncertainty is included in the predictive covariance derived above. The local reduction provides a mechanism through which embedded physics can improve predictive confidence; its effect at later times also depends on uncertainty in the initial state, the evolving projection geometry, and dynamical amplification. We therefore evaluate predictive uncertainty using complete posterior rollouts.

A.7 Validity and Interpretation

Conditions and scope. The predictive covariance decomposition is exact under the fitted variational posterior whenever the required second moments exist. The local reduction additionally requires $\hat{Q}_t \succ 0$ and full row rank of CJ_ϕ, and holds at fixed state, input, time, decoder, and physical parameters. Neither result requires decoder linearization or a nominal-trajectory approximation. The reduction concerns the $\hat{Q}_t$-weighted drift variance.

Physical validity and posterior uncertainty. Projection removes locally incompatible drift variation whether or not the imposed relation describes the true dynamics. Consequently, reduced variance can coexist with predictive bias under misspecification. Uncertainty across encoder, decoder, and physics-parameter samples remains represented in the predictive decomposition, while a learned likelihood covariance can also accommodate residual prediction error. These quantities characterize uncertainty under the fitted model; their calibration is assessed empirically.

B Experimental Details

B.1 Computational Setup

All experiments were carried out on a PC, equipped with 32GB DDR5 RAM, with an Intel Core i9 CPU and an NVIDIA RTX5090 GPU with 32GB of DDR6 VRAM. All models were implemented in `jax`, using the `diffrax` (3) and `equinox` (39) packages. A zip file of the codebase is supplied as part of the submission.

B.2 Dataset Parametrization

We provide the full governing equations, parameter values, and control ranges for each of the six case studies below. The constraint forms (C, g, Λ) used at training time are summarised in Table 8.

All datasets share the same data-generation pipeline: trajectories are integrated with `diffrax` from randomly sampled initial conditions under piecewise-constant exogenous controls, then sampled at N evenly spaced time points and corrupted with additive Gaussian observation noise of standard deviation $\sigma_{\text{obs}} = 0.05$. We simulate sparse and irregular measurements by dropping each datapoint with probability 0.1. Each case study uses a fixed seed (per-system, given below) so that all baselines see the same trajectories. Default split: 20 training trajectories and 3 held-out test trajectories, where the model rolls out given the initial condition and current value of exogenous input.

Controlled Lotka–Volterra (LV).

$$\frac{dx_1}{dt} = \alpha x_1 - \beta x_1 x_2 + P(t), \qquad \frac{dx_2}{dt} = \delta x_1 x_2 - \gamma x_2 - I(t)\, x_2.$$

Embedded physics (well-specified): the prey RHS $C\dot{x} = g$ with $C = [1, 0]$ and $g(x, u; \Lambda) = \alpha x_1 - \beta x_1 x_2 + P(t)$, $\Lambda = (\alpha, \beta)$.

Table 2: Lotka–Volterra parameter and dataset settings.

States	x_1 (prey), x_2 (predator)
Controls	$u(t) = [P(t), I(t)]$ (prey injection, predator suppression)
True parameters	$\alpha = \frac{2}{3},\ \beta = \frac{4}{3},\ \delta = 1,\ \gamma = 1$
Control ranges	$P \in [0, 0.5],\ I \in [0, 0.3]$, 4 piecewise-constant switches
Horizon, samples	$t \in [0, 20]$, $N = 100$ ($\sim$3 periods)
Trajectories	20 train, 3 test
$\Lambda_0, \sigma_\Lambda$ (well-spec.)	$\Lambda_0 = [\frac{2}{3}, \frac{4}{3}]$, $\sigma_\Lambda = [0.3, 0.3]$

Forced Damped Oscillator.

$$\frac{dx}{dt} = v, \qquad \frac{dv}{dt} = -\omega^2 x - \gamma v + u(t).$$

Embedded physics: kinematic identity $\dot{x} = kv$, $\Lambda = (k)$, $k = 1$ at truth.

Table 3: Forced oscillator parameter and dataset settings.

States	x (position), v (velocity)
Controls	external force $u(t)$ (1D), 5 piecewise-constant switches
True parameters	$\omega = 1,\ \gamma = 0.1$
Control range	$u \in [-3, 3]$
Horizon, samples	$t \in [0, 20]$, $N = 50$
Trajectories	20 train, 3 test
$\Lambda_0, \sigma_\Lambda$ (well-spec.)	$k_0 = 1.0$, $\sigma_\Lambda = 0.1$

Forced Duffing Oscillator.

$$\frac{dx}{dt} = v, \qquad \frac{dv}{dt} = -\gamma v - \alpha x - \beta x^3 + u(t), \quad \alpha, \beta > 0.$$

Embedded physics: kinematic identity $\dot{x} = kv$. The cubic-stiffening restoring force keeps trajectories bounded under random forcing; the bistable $\alpha < 0$ regime is avoided.

Table 4: Duffing oscillator parameter and dataset settings.

States	x (position), v (velocity)
Controls	external force $u(t)$ (1D), 5 piecewise-constant switches
True parameters	$\gamma = 0.2,\ \alpha = 1.0,\ \beta = 0.5$
Control range	$u \in [-0.4, 0.4]$
Horizon, samples	$t \in [0, 45]$, $N = 350$
Trajectories	20 train, 3 test
$\Lambda_0, \sigma_\Lambda$ (well-spec.)	$k_0 = 1.0$, $\sigma_\Lambda = 0.1$

Driven Nonlinear Pendulum.

$$\frac{d\theta}{dt} = \omega, \qquad \frac{d\omega}{dt} = -\omega_0^2 \sin(\theta) - \gamma\,\omega + u(t).$$

Damping is chosen so the pendulum never goes "over the top" under the bounded torque, keeping $|\theta| \lesssim 1$ rad. Embedded physics: kinematic identity $\dot{\theta} = k\omega$.

Table 5: Pendulum parameter and dataset settings.

States	θ (angle), ω (angular velocity)
Controls	external torque $u(t)$ (1D), 5 piecewise-constant switches
True parameters	$\omega_0 = 1.5,\ \gamma = 0.3$
Control range	$u \in [-0.5, 0.5]$
Horizon, samples	$t \in [0, 20]$, $N = 100$ ($\sim$5 periods)
Trajectories	20 train, 3 test
$\Lambda_0, \sigma_\Lambda$ (well-spec.)	$k_0 = 1.0$, $\sigma_\Lambda = 0.1$

Forced Van der Pol Oscillator.

$$\frac{dx}{dt} = v, \qquad \frac{dv}{dt} = \mu(1 - x^2)\,v - x + A\,u(t).$$

Limit-cycle nonlinearity, externally forced. Embedded physics: kinematic identity $\dot{x} = kv$.

Chained Pendulum (coupled chain).

$$\frac{d\theta_k}{dt} = \omega_k, \qquad \frac{d\omega_k}{dt} = -\omega_{0,k}^2 \sin\theta_k - \gamma\,\omega_k + k_c(\theta_{k-1} - 2\theta_k + \theta_{k+1}) + u_k(t),$$

for $k = 0, 1, 2$, giving a 6-dimensional state $(\theta_0, \omega_0, \theta_1, \omega_1, \theta_2, \omega_2)$. The chain has reflective boundaries ($\theta_{-1} := \theta_0$, $\theta_3 := \theta_2$) and per-pendulum natural frequencies $\omega_{0,k} \in \{1.2, 1.5, 1.8\}$ so each pendulum's coupling parameter is identifiable rather than degenerate. Embedded physics: *cumulative*

Table 6: Van der Pol parameter and dataset settings.

States	x (position), v (velocity)
Controls	external force $u(t)$ (1D), 5 piecewise-constant switches
True parameters	$\mu = 1.5,\ A = 1.0$
Control range	$u \in [-2, 2]$
Horizon, samples	$t \in [0, 20]$, $N = 100$ ($\sim$2.5 limit-cycle periods)
Trajectories	20 train, 3 test
$\Lambda_0, \sigma_\Lambda$ (well-spec.)	$k_0 = 1.0$, $\sigma_\Lambda = 0.1$

Table 7: Chained Pendulum parameter and dataset settings.

States	θ_k (angle), ω_k (angular velocity) for $k = 0, 1, 2$
Controls	3D torque $\mathbf{u}(t) = [u_0, u_1, u_2]$, 5 piecewise-constant switches
True parameters	$\omega_{0,k} = (1.2, 1.5, 1.8)$, $\gamma = 0.3$, $k_c = 0.15$
Control range	$u_k \in [-0.5, 0.5]$
Horizon, samples	$t \in [0, 20]$, $N = 100$ ($\sim$4–6 limit-cycle periods per pendulum)
Trajectories	20 train, 3 test
$\Lambda_0, \sigma_\Lambda$ (well-spec.)	$c_k = 1.0$ for all embedded k, $\sigma_\Lambda = 0.1$
Constraint budget	sweep $n \in \{1, 2, 3\}$ (cumulative across pendulums)

kinematic identities $\dot{\theta}_k = c_k\,\omega_k$ for $k = 0, \ldots, n-1$, where $n \in \{1, 2, 3\}$ controls how many pendulums' kinematic rows are constrained at training time.

Table 8: Embedded physical knowledge across the synthetic case studies. C projects onto the constrained observable derivatives, $g(x, u; \Lambda)$ encodes the partial physics, and Λ collects the inferred constraint parameters with Gaussian prior $\mathcal{N}(\Lambda_0, \sigma_\Lambda^2)$. The systematic column lists the structural perturbation applied to g under systematic misspecification.

System	C	$g(x, u; \Lambda)$	Λ_0, σ_Λ (well-spec.)	Systematic mis-spec.
LV *partial RHS*	$[1,\ 0]$	$\alpha x_1 - \beta x_1 x_2 + P(t)$	$[\alpha, \beta] = [\frac{2}{3}, \frac{4}{3}]$, $\sigma = [0.3, 0.3]$	drop predator coupling: $g = \alpha x_1 + P(t)$
Forced Osc. *kinematic*	$[1,\ 0]$	$k\,v$	$k = 1.0, \sigma = 0.1$	$g = k\,xv$
Duffing *kinematic*	$[1,\ 0]$	$k\,v$	$k = 1.0, \sigma = 0.1$	$g = k\,xv$
Pendulum *kinematic*	$[1,\ 0]$	$k\,\omega$	$k = 1.0, \sigma = 0.1$	$g = k\,\theta\,\omega$
Van der Pol *kinematic*	$[1,\ 0]$	$k\,v$	$k = 1.0, \sigma = 0.1$	$g = k\,xv$
High Pendulum *cumul. kinematic, n rows*	$\big[\,\text{rows}\,2k,\ k = 0, \ldots, n-1\,\big]$	$\big[c_k\,\omega_k\big]_{k=0}^{n-1}$	$c_k = 1.0, \sigma = 0.1$	$g_k = c_k\,\theta_k\,\omega_k$

B.3 Hyperparameters

We report the hyperparameters used for all main-paper results. All baselines share architectural settings wherever they overlap, so that comparisons isolate the inference and constraint mechanisms rather than capacity.

Neural ODE families. The latent vector field is a 1-hidden-layer MLP with width 32; encoder and decoder are 1-hidden-layer MLPs with width 32. All MLPs utilize the `swish` activation function. Integration uses Heun's method with a constant step size $\Delta t = 0.05$.

The Bayesian weight posterior is a Neural-SDE with an OU prior, where the diffusion coefficient was swept across the range $[10^{-2}, 10^{0}]$ and set to $\sigma = 0.1$) with a hyper-network posterior drift (1 hidden layer, width 128) sharing the prior diffusion.

The ELBO is estimated with $S = 4$ Monte Carlo samples. The KL coefficients for each of the components are swept over the range $[10^{-6}, 10^{0}]$ with the values set at $\beta = 10^{-6}$ for the weight path, and $\beta_C = 10^{-3}$ for the constraint-parameter prior. The β_{soft} penalty weight was set at 10^{-1} following a grid search, The likelihood was chosen as the full-covariance Gaussian NLL, but this could easily be changed to any other noise model.

Optimizers. Training uses Adam with weight decay 10^{-4} and learning rate of 10^{-3}. These were selected from a grid search over $[10^{-6}, 10^{-2}]$. We set the batch size 128, and trained for 650 epochs for the base case learning; and 1250 for the misspecified-physics regimes so that the posterior has time to traverse a tight or structurally-incorrect prior.

B.4 Dataset Corruption Settings

Robustness sweep settings. We evaluate robustness to data corruption - firstly, with additive Gaussian observation noise and random observation dropout, we evaluate five $(\sigma_{\text{obs}}, \rho_{\text{drop}})$ combinations per system: $(0.05, 0.10)$, $(0.05, 0.30)$, $(0.05, 0.50)$, $(0.10, 0.10)$, and $(0.20, 0.10)$. Here ρ_{drop} is the a given datapoints' dropout probability. We also consider further varying noise profiles. We investigate state-varying heteroscedastic noise, which is generated as a function of the magnitude of a given state $\tilde{y}_{t,j} = y_{t,j} + \sigma h_j z_{t,j}$. We also invoke temporally correlated noise, generated as $\tilde{y}_{t,j} = y_{t,j} + \sigma h_j (t/T) z_{t,j}$, with $z_{t,j} \sim \mathcal{N}(0, 1)$. Each method is evaluated across noise levels, $\sigma = (0.1, 0.2)$ and retrained from scratch at each setting and evaluated on held-out trajectories. We also investigate robustness to SDE-style degration, with noise injected into the data-generating dynamics, $dx_t = F(x_t, u_t)\, dt + \sigma_p B(x_t)\, dW_t$. We consider constant diffusion, swept across $\sigma_p = (0.05, 0.1, 0.2)$ with the resulting dynamics corrupted with additive Gaussian noise at a fixed $\sigma = 0.1$ and same dropout probability.

Partial-observation sweep settings. Another common concept in dynamic systems is observability, where we can only measure a subset of internal states. We distinguish the full decoded state, where we can define our known partial physics, $x_t = D_\phi(h_t)$ from the measured output $y_t = Hx_t + \varepsilon_t$, where H selects the observed coordinates. The observation likelihood is evaluated only on these coordinates,

$$p(y_t \mid h_t, \phi) = \mathcal{N}(y_t;\, HD_\phi(h_t),\, \Sigma_{\text{obs}})\,,$$

while the embedded physical relation may involve the complete decoded state, including unmeasured components. To infer the initial learned latent state from partial measurements, we augment the encoder, to be conditioned on a short history of 5 observations, known inputs, and their sampling times:

$$h_{t_0} = E_\psi\big(\{(y_{t_i}, u_{t_i}, t_i)\}_{i=-L+1}^{0}\big)\,.$$

Only the encoder inputs and the states entering the likelihood change, with everything else identical to our base case. All baselines use the same observation history and likelihood.

For example, when position q_t is measured but velocity Σ_t is unmeasured, the decoder produces both, $D_\phi(h_t) = (q_t, \Sigma_t)$, hence the lifted kinematic relation $J_{D_{\phi,q}}(h_t)\, f_\theta^C(h_t, u_t, t) = D_{\phi,v}(h_t)$ can still be embedded into our framework. We evaluate predictions on just the observed states.

Misspecification sweep settings. We embed the incorrect physics within our latent Neural DE model, to assess robustness to misspecification. Three regimes are run per system (per Section 4) all trained with 1250 training steps:

- *Overconfident:* LV uses $[\alpha, \beta]_0 = [1.5, 0.5]$ with $\sigma_\Lambda = [0.05, 0.05]$; the kinematic systems use $k_0 = 0.5$ with $\sigma_\Lambda = 0.02$; Chained Pendulum uses $c_{k,0} = 0.5$ with $\sigma_\Lambda = 0.02$ for every embedded coupling.
- *Uncertain-but-incorrect:* LV uses $[\alpha, \beta]_0 = [0.5, 1.5]$ with $\sigma_\Lambda = [0.5, 0.5]$; the kinematic systems use $k_0 = 0.85$ with $\sigma_\Lambda = 0.3$. $\beta_C = 10^{-5}$.

- *Informed-but-uncertain*: correct parameter means embedded with standard deviation of $0.3\times$ mean embedded as prior

- *Systematic:* structural perturbation of g such that parameter shift cannot recover the true relation; Λ_0 at the true value with default σ_Λ, $\beta_C = 10^{-5}$.

All three regimes share the same data corruption $(\sigma_{\text{obs}}, \rho) = (0.05, 0.10)$ on the training set with a clean test set, so the parametric-vs-systematic effect is isolated from the corruption axis. Per case study settings can be found in Table 8.

Misinformed physical model. We also assess the robustness to the data generating process becoming more complex than the embedded physical model in two ways. Firstly, we investigate *time-varying physical mismatch.* To introduce a persistent change in the data-generating drift, we scale its constrained component after an onset time t_{on}:

$$[C\dot{x}_t]_{\text{data}} = \big(1 + \kappa s(t)\big)[C\dot{x}_t]_{\text{nom}}, \qquad s(t) = \text{sigmoid}\left(\frac{t - t_{\text{on}}}{\tau}\right).$$

The embedded C and g remain nominal. We set the fractional mismatch $\kappa = 1.5$, holding the onset time and transition width fixed at $t_{\text{on}} = 0.4$ AND $\tau = 0.02$.

We also consider a data generating process, with unmodelled forcing. This consists of adding an unobserved forcing process to each constrained row of the data-generating drift, while retaining the nominal embedded relation:

$$[C\dot{x}_t]_{\text{data}} = g(x_t, u_t; \Lambda_{\text{true}}) + \xi_t, \qquad d\xi_t = -\rho_\xi \xi_t \, dt + \sqrt{2\rho_\xi}\ \text{diag}(\sigma_\xi)\, dW_t.$$

The processes are independent across constrained rows. Their stationary standard deviations $\sigma_{\xi,j}$ are set to 1.5 of the corresponding nominal right-hand side; $\rho_\xi = 0.5$ sets the correlation timescale.

B.5 Extra Experiments

This appendix contains the full per-system results for the corruption and misspecification sweeps reported in aggregate in Section 4, together with extended diagnostic plots and per-method tables.

Synthetic Baseline. Table 9 reports per-system accuracy under correctly specified physics. Latent projection achieves the lowest mean CRPS on Duffing, Forced Oscillator, and Pendulum, and improves both MSE and CRPS over state-space projection on every reported system.

Case	Hard (ours)		Soft		NODE	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Duffing	$\mathbf{0.0025 \pm 0.0023}$	$\mathbf{0.1120 \pm 0.0369}$	0.0194 ± 0.0245	0.3627 ± 0.2900	0.0216 ± 0.0092	0.4503 ± 0.0681
Forced Oscillator	$\mathbf{0.4534 \pm 0.2919}$	$\mathbf{0.1136 \pm 0.0278}$	0.4612 ± 0.3655	0.1402 ± 0.1073	0.8194 ± 0.5636	0.1521 ± 0.0479
High Pendulum	0.0184 ± 0.0028	0.5488 ± 0.0896	0.0369 ± 0.0103	0.7607 ± 0.0250	0.0379 ± 0.0109	0.7784 ± 0.0288
Lotka–Volterra	$\mathbf{0.0180 \pm 0.0173}$	$\mathbf{0.3444 \pm 0.3150}$	0.0295 ± 0.0093	0.5549 ± 0.1671	0.0302 ± 0.0091	0.5748 ± 0.2005
Pendulum	$\mathbf{0.0020 \pm 0.0015}$	$\mathbf{0.1627 \pm 0.0728}$	0.0053 ± 0.0034	0.1865 ± 0.0429	0.0042 ± 0.0016	0.2387 ± 0.0345
Van der Pol	$\mathbf{0.1796 \pm 0.0858}$	$\mathbf{0.1319 \pm 0.0443}$	0.3241 ± 0.1420	0.1863 ± 0.0643	0.6133 ± 0.0359	0.2865 ± 0.0285

Case	State NODE		State Soft		Projected	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Duffing	0.0725 ± 0.0345	0.9076 ± 0.2964	0.0530 ± 0.0230	0.7414 ± 0.2063	0.0167 ± 0.0026	0.2951 ± 0.0772
Forced Oscillator	4.8277 ± 3.3117	0.5277 ± 0.2173	5.5661 ± 3.2334	0.5853 ± 0.2257	4.3186 ± 4.8020	0.4118 ± 0.3799
High Pendulum	$22702.3297 \pm 55608.0857$	183.7946 ± 441.5503	0.3216 ± 0.0299	3.0180 ± 0.3408	$51771.8698 \pm 115724.6006$	555.9287 ± 1219.9404
Lotka–Volterra	0.0230 ± 0.0077	0.4188 ± 0.1556	0.0185 ± 0.0082	0.3575 ± 0.0960	$62225.8570 \pm 70269.0276$	580.0310 ± 669.1835
Pendulum	0.1042 ± 0.0295	1.7610 ± 0.3203	0.0926 ± 0.0282	1.6376 ± 0.3761	0.0151 ± 0.0078	0.5096 ± 0.1840
Van der Pol	0.5616 ± 0.1828	0.3073 ± 0.0425	0.5135 ± 0.1059	0.2885 ± 0.0283	0.4978 ± 0.1289	0.2997 ± 0.0479

Case	Stabilized		UDE		HNN (Gaussian weights)	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Duffing	0.0106 ± 0.0035	0.2124 ± 0.0367	0.0595 ± 0.0106	0.6577 ± 0.0500	0.0269 ± 0.0109	0.4246 ± 0.0410
Forced Oscillator	2.7927 ± 3.9737	0.3283 ± 0.3497	9.0049 ± 2.4739	0.7823 ± 0.0388	3.9573 ± 2.2605	0.4795 ± 0.1074
High Pendulum	$49482.9894 \pm 110606.5148$	544.2933 ± 1193.9246	0.0336 ± 0.0083	0.6088 ± 0.0225	0.0343 ± 0.0126	0.6836 ± 0.1602
Lotka–Volterra	$56541.0690 \pm 68655.3227$	471.5798 ± 557.7222	0.0684 ± 0.0335	0.5220 ± 0.0445	–	–
Pendulum	0.0081 ± 0.0008	0.2752 ± 0.0207	0.0237 ± 0.0042	0.5982 ± 0.0575	0.0127 ± 0.0045	0.4209 ± 0.0689
Van der Pol	0.4234 ± 0.0702	0.2817 ± 0.0360	1.8446 ± 0.1715	0.7279 ± 0.0091	0.8479 ± 0.2731	0.3899 ± 0.0892

Case	LNN (Gaussian weights)	
	MSE	CRPS
Duffing	0.0173 ± 0.0084	0.3787 ± 0.0875
Forced Oscillator	0.5517 ± 0.2373	0.1354 ± 0.0128
High Pendulum	$\mathbf{0.0081 \pm 0.0007}$	$\mathbf{0.3166 \pm 0.0265}$
Lotka–Volterra	–	–
Pendulum	0.0022 ± 0.0008	0.1911 ± 0.0210
Van der Pol	0.6835 ± 0.1688	0.3688 ± 0.0718

Table 9: Synthetic baselines: clean-target State MSE and normalised State CRPS.

Case	Hard (ours)	Soft	NODE	State NODE	State Soft
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Duffing	$\mathbf{0.1490 \pm 0.0200}$	0.3082 ± 0.1699	0.3467 ± 0.0201	0.6155 ± 0.1299	0.5290 ± 0.0964
Forced Oscillator	$\mathbf{0.0967 \pm 0.0148}$	0.1166 ± 0.0743	0.1279 ± 0.0258	0.3331 ± 0.1118	0.3612 ± 0.1172
High Pendulum	0.3788 ± 0.0307	0.5023 ± 0.0039	0.5055 ± 0.0095	171.8972 ± 416.6811	1.5421 ± 0.1537
Lotka–Volterra	$\mathbf{0.3109 \pm 0.1863}$	0.4133 ± 0.1268	0.3967 ± 0.1153	0.4137 ± 0.1594	0.3930 ± 0.1411
Pendulum	$\mathbf{0.2049 \pm 0.0599}$	0.2573 ± 0.0082	0.2633 ± 0.0117	0.9971 ± 0.1581	0.9445 ± 0.1777
Van der Pol	$\mathbf{0.1024 \pm 0.0321}$	0.1428 ± 0.0475	0.2021 ± 0.0303	0.2502 ± 0.0331	0.2336 ± 0.0132

Case	Projected	Stabilized	UDE	HNN (Gaussian weights)	LNN (Gaussian weights)
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Duffing	0.3440 ± 0.1225	0.2355 ± 0.0259	0.5932 ± 0.0483	0.3766 ± 0.0372	0.3279 ± 0.0695
Forced Oscillator	0.2776 ± 0.1989	0.2297 ± 0.1798	0.7669 ± 0.0437	0.4406 ± 0.1176	0.1105 ± 0.0099
High Pendulum	547.1881 ± 1204.0649	535.5474 ± 1178.0367	0.5470 ± 0.0289	0.6141 ± 0.1357	$\mathbf{0.3024 \pm 0.0261}$
Lotka–Volterra	568.7785 ± 658.7390	450.7833 ± 535.7139	0.4613 ± 0.0306	–	–
Pendulum	0.5554 ± 0.1975	0.2842 ± 0.0217	0.5375 ± 0.0568	0.3900 ± 0.0568	0.2202 ± 0.0169
Van der Pol	0.2394 ± 0.0272	0.2361 ± 0.0220	0.7083 ± 0.0077	0.3586 ± 0.0873	0.3420 ± 0.0678

Table 10: Synthetic baselines: normalised observation weighted interval score (WIS), penalising both broad intervals and missed observations.

Robustness to Dropout and Noise We report per-system results across varying observation noise and dropout, retraining each method at every corruption setting. These comparisons assess how reduced data quality affects predictive accuracy and the relative benefit of embedded physics.

Firstly, our results with varying noise profiles, show our approach of latent embedding is more robust than state space projections.

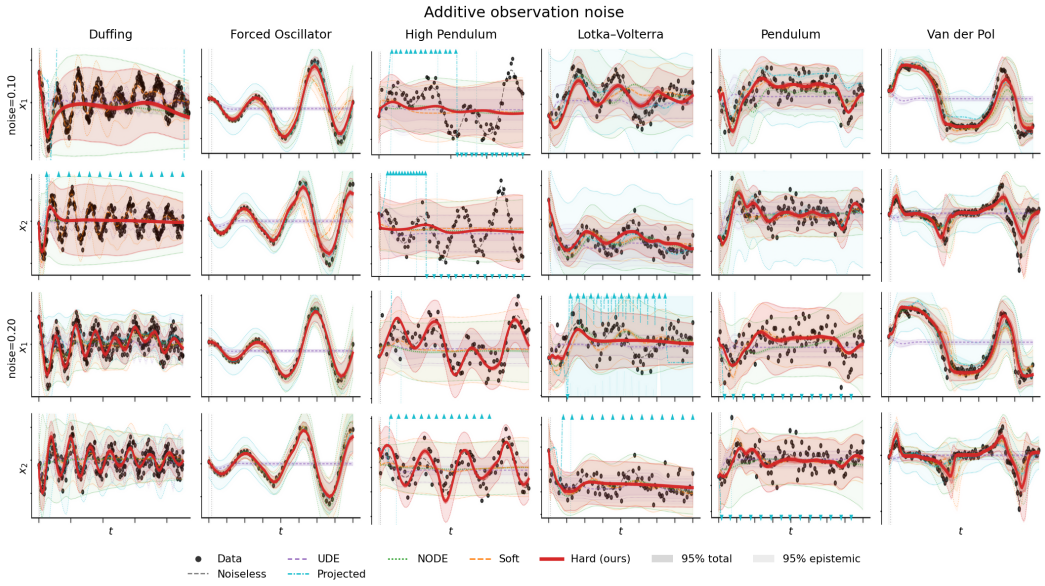


Figure 5: Model rollouts with increasing additive noise

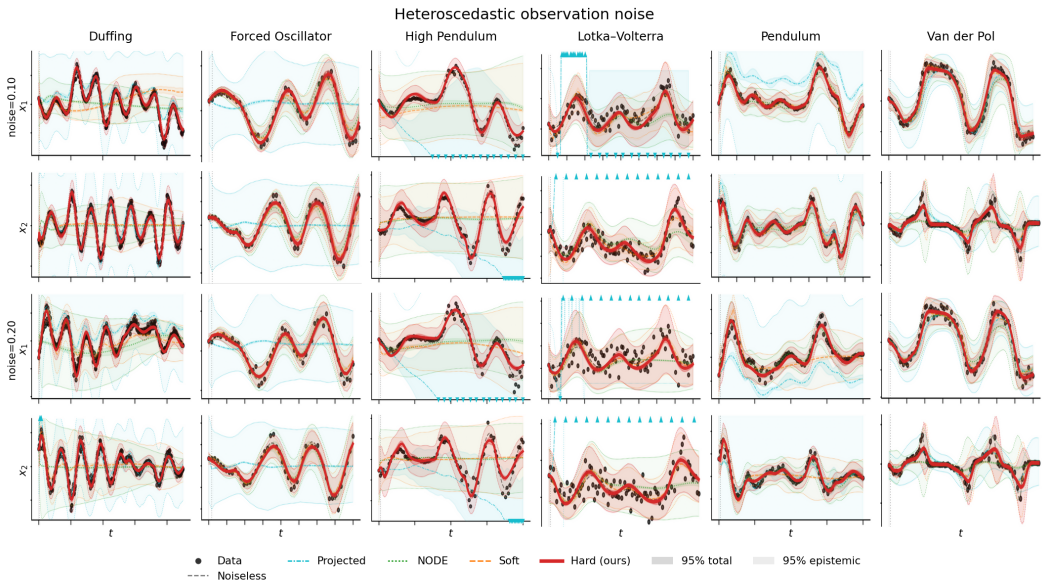


Figure 6: Rollouts with increasing heteroscedastic noise

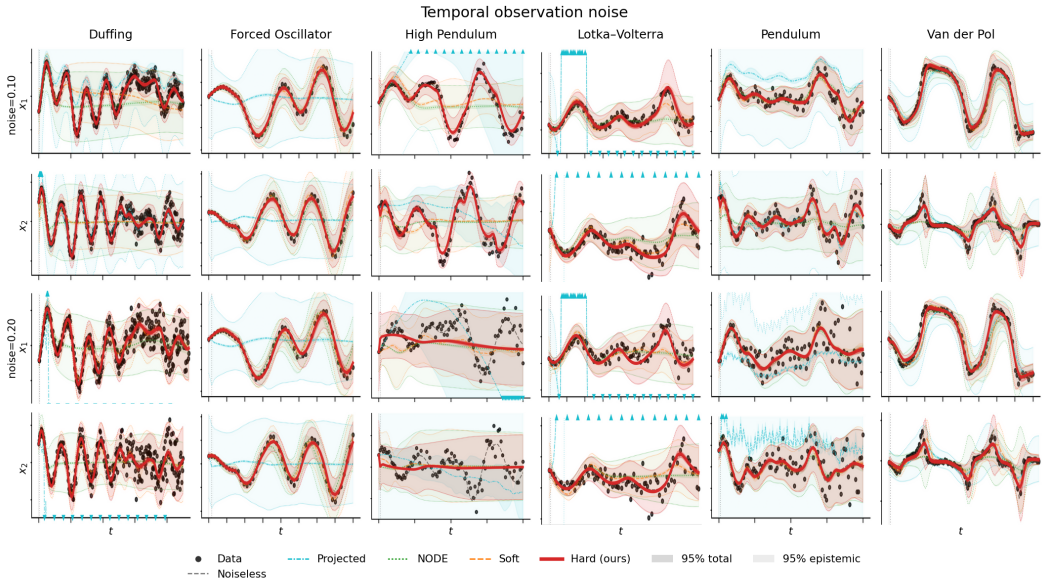


Figure 8: Temporally increasing noise

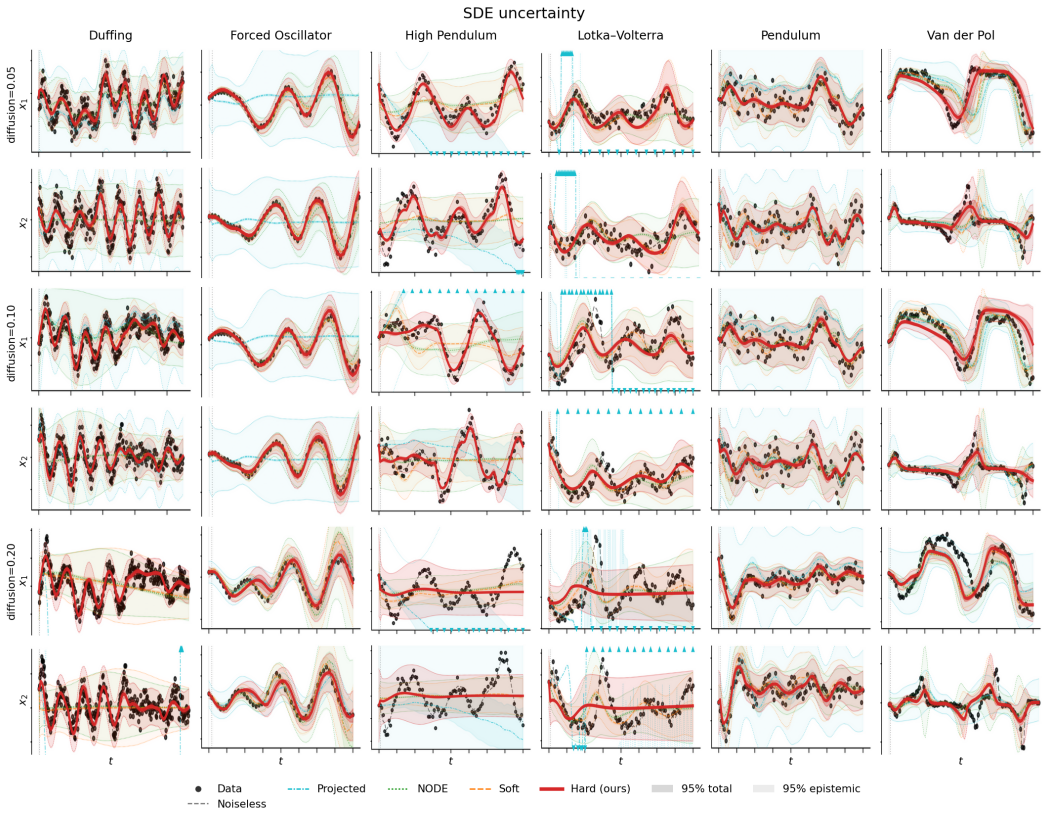


Figure 7: SDE style noise

Additive noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0689	0.7528	0.2805	**0.0782**	0.0420	0.7932	**0.0234**	**0.2709**	0.0065	0.2725	**0.2047**	**0.1600**
Soft	**0.0101**	**0.2164**	3.3863	0.3549	0.0409	0.7803	0.0409	0.4125	**0.0053**	**0.2598**	0.3368	0.2193
NODE	0.0724	0.7601	0.3977	0.1019	**0.0406**	0.7886	0.0738	0.6097	0.0105	0.4444	0.3429	0.2167
Projected	0.0442	0.4821	**0.2727**	0.0890	147758.5781	1884.3776	0.0359	0.3315	0.0160	0.5828	0.4077	0.2709

Additive noise, 20%

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	**0.0056**	**0.1590**	0.4585	0.1208	**0.0139**	**0.4743**	0.0680	0.5887	0.0087	0.3342	**0.1824**	**0.1553**
Soft	0.0234	0.3409	**0.1935**	**0.0766**	0.0414	0.7731	**0.0466**	**0.4212**	**0.0074**	**0.3256**	0.2807	0.2043
NODE	0.0421	0.5685	0.2519	0.0937	0.0433	0.7809	0.0785	0.6204	0.0224	0.6277	0.2713	0.1972
Projected	0.0267	0.3948	0.2615	0.0927	–	–	118739.5859	809.4227	–	–	0.3433	0.2373

Table 11: Additive noise ablation.

Heteroscedastic noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0033 \pm 0.0024}$	$\mathbf{0.1158 \pm 0.0317}$	0.3869 ± 0.4323	$\mathbf{0.0897 \pm 0.0379}$	$\mathbf{0.0305 \pm 0.0186}$	$\mathbf{0.6370 \pm 0.2809}$
Soft	0.0258 ± 0.0263	0.4713 ± 0.3757	$\mathbf{0.2189 \pm 0.0198}$	0.1003 ± 0.0227	0.0327 ± 0.0042	0.7445 ± 0.0095
NODE	0.0395 ± 0.0162	0.6650 ± 0.1992	0.7772 ± 0.5582	0.1581 ± 0.0125	0.0350 ± 0.0009	0.7703 ± 0.0318
Projected	0.0253 ± 0.0052	0.4300 ± 0.1084	4.6731 ± 4.9186	0.4949 ± 0.3727	27.0259 ± 8.0402	14.0369 ± 2.7816

Heteroscedastic noise, 10%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0184 \pm 0.0202}$	0.3658 ± 0.2989	0.0028 ± 0.0043	0.1986 ± 0.2164	$\mathbf{0.2227 \pm 0.0817}$	$\mathbf{0.1453 \pm 0.0322}$
Soft	0.0185 ± 0.0007	0.3922 ± 0.1096	0.0137 ± 0.0035	0.3709 ± 0.3224	0.3763 ± 0.1608	0.2058 ± 0.0796
NODE	0.0270 ± 0.0054	0.4882 ± 0.0742	$\mathbf{0.0025 \pm 0.0018}$	$\mathbf{0.1786 \pm 0.0356}$	0.6502 ± 0.1249	0.3092 ± 0.0816
Projected	0.0212 ± 0.0065	$\mathbf{0.3619 \pm 0.0869}$	0.0250 ± 0.0042	0.7706 ± 0.1307	0.4603 ± 0.2134	0.2730 ± 0.0627

Heteroscedastic noise, 20%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0036 \pm 0.0029}$	$\mathbf{0.1257 \pm 0.0304}$	$\mathbf{0.3212 \pm 0.1864}$	$\mathbf{0.0996 \pm 0.0218}$	$\mathbf{0.0182 \pm 0.0167}$	$\mathbf{0.5078 \pm 0.3718}$
Soft	0.0210 ± 0.0120	0.4200 ± 0.1702	0.3654 ± 0.3326	0.1114 ± 0.0341	0.0330 ± 0.0041	0.7499 ± 0.0120
NODE	0.0641 ± 0.0124	0.8362 ± 0.1413	0.5531 ± 0.5024	0.1378 ± 0.0286	0.0332 ± 0.0036	0.7488 ± 0.0001
Projected	0.0355 ± 0.0103	0.4972 ± 0.0657	4.5869 ± 4.2583	0.5348 ± 0.3471	26.6146 ± 7.6885	13.9435 ± 2.7354

Heteroscedastic noise, 20%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0208 \pm 0.0072}$	$\mathbf{0.3902 \pm 0.1388}$	$\mathbf{0.0010 \pm 0.0004}$	$\mathbf{0.1131 \pm 0.0228}$	$\mathbf{0.2687 \pm 0.0810}$	$\mathbf{0.1665 \pm 0.0374}$
Soft	0.0222 ± 0.0061	0.4039 ± 0.1578	0.0080 ± 0.0074	0.4156 ± 0.3434	0.3040 ± 0.0777	0.1890 ± 0.0524
NODE	0.0321 ± 0.0032	0.5631 ± 0.1081	0.0032 ± 0.0027	0.2028 ± 0.0731	0.7137 ± 0.1511	0.3254 ± 0.0859
Projected	$67374.1877 \pm 95281.4674$	553.6147 ± 782.3527	0.0248 ± 0.0018	0.7652 ± 0.1028	0.4773 ± 0.1917	0.2787 ± 0.0646

Table 12: Heteroscedastic noise ablation.

Temporal noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0021 \pm 0.0013}$	$\mathbf{0.0964 \pm 0.0198}$	$\mathbf{0.3058 \pm 0.1743}$	$\mathbf{0.0869 \pm 0.0213}$	$\mathbf{0.0200 \pm 0.0092}$	$\mathbf{0.5452 \pm 0.1770}$
Soft	0.0078 ± 0.0007	0.2127 ± 0.0188	0.7523 ± 0.6893	0.1653 ± 0.0674	0.0304 ± 0.0015	0.7254 ± 0.0239
NODE	0.0357 ± 0.0105	0.6183 ± 0.1457	0.4804 ± 0.3864	0.1213 ± 0.0210	0.0330 ± 0.0008	0.7502 ± 0.0469
Projected	0.0352 ± 0.0120	0.4892 ± 0.0393	4.6546 ± 4.7462	0.5196 ± 0.3756	27.5209 ± 8.5055	14.1812 ± 2.8751

Temporal noise, 10%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0194 ± 0.0123	$\mathbf{0.3729 \pm 0.2205}$	0.0038 ± 0.0059	0.2240 ± 0.2431	$\mathbf{0.1644 \pm 0.0144}$	$\mathbf{0.1234 \pm 0.0152}$
Soft	$\mathbf{0.0184 \pm 0.0030}$	0.3758 ± 0.1610	$\mathbf{0.0010 \pm 0.0005}$	$\mathbf{0.1140 \pm 0.0068}$	0.2723 ± 0.0821	0.1707 ± 0.0483
NODE	0.0244 ± 0.0032	0.4447 ± 0.0648	0.0059 ± 0.0070	0.2628 ± 0.1801	0.6075 ± 0.0684	0.2999 ± 0.0663
Projected	$92124.7568 \pm 79831.9564$	809.8868 ± 710.2774	0.0406 ± 0.0247	1.0203 ± 0.5197	0.3524 ± 0.0218	0.2467 ± 0.0213

Temporal noise, 20%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0022 \pm 0.0013}$	$\mathbf{0.0993 \pm 0.0140}$	0.2725 ± 0.1246	$\mathbf{0.0854 \pm 0.0117}$	$\mathbf{0.0210 \pm 0.0143}$	$\mathbf{0.5702 \pm 0.2915}$
Soft	0.0071 ± 0.0027	0.2050 ± 0.0254	$\mathbf{0.2586 \pm 0.2404}$	0.0901 ± 0.0274	0.0227 ± 0.0092	0.6040 ± 0.1654
NODE	0.0328 ± 0.0101	0.5819 ± 0.1422	0.6805 ± 0.2531	0.1552 ± 0.0052	0.0278 ± 0.0055	0.7048 ± 0.1032
Projected	0.0312 ± 0.0128	0.4199 ± 0.0647	4.5512 ± 4.5928	0.4998 ± 0.3579	27.5583 ± 8.5243	14.2222 ± 2.9098

Temporal noise, 20%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0196 \pm 0.0158}$	$\mathbf{0.3758 \pm 0.2681}$	$\mathbf{0.0010 \pm 0.0005}$	$\mathbf{0.1072 \pm 0.0150}$	$\mathbf{0.1650 \pm 0.0597}$	$\mathbf{0.1279 \pm 0.0270}$
Soft	0.0285 ± 0.0088	0.4677 ± 0.0221	0.0010 ± 0.0007	0.1156 ± 0.0149	0.3163 ± 0.0337	0.1857 ± 0.0042
NODE	0.0263 ± 0.0014	0.4728 ± 0.0821	0.0045 ± 0.0047	0.2292 ± 0.1144	0.4913 ± 0.0212	0.2538 ± 0.0390
Projected	$46992.0517 \pm 81392.5950$	367.2629 ± 635.5632	0.0367 ± 0.0256	0.9522 ± 0.5251	0.3370 ± 0.0130	0.2373 ± 0.0099

Table 13: Temporal noise ablation.

SDE diffusion, 5%; observation noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0126 \pm 0.0134}$	$\mathbf{0.2459 \pm 0.1717}$	1.0233 ± 0.6386	$\mathbf{0.1314 \pm 0.0346}$	$\mathbf{0.0226 \pm 0.0183}$	$\mathbf{0.5224 \pm 0.2564}$
Soft	0.0136 ± 0.0003	0.2945 ± 0.0413	$\mathbf{0.7553 \pm 0.7186}$	0.1329 ± 0.0131	0.0415 ± 0.0013	0.7619 ± 0.0328
NODE	0.0436 ± 0.0055	0.6537 ± 0.1115	1.7189 ± 0.8859	0.2052 ± 0.0022	0.0420 ± 0.0065	0.7741 ± 0.0886
Projected	0.0332 ± 0.0111	0.4543 ± 0.0459	8.4670 ± 6.4689	0.6052 ± 0.2549	28.3027 ± 8.9806	13.1162 ± 2.6818

SDE diffusion, 5%; observation noise, 10%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0393 ± 0.0276	0.4444 ± 0.1664	$\mathbf{0.0046 \pm 0.0015}$	$\mathbf{0.2295 \pm 0.0398}$	$\mathbf{0.4486 \pm 0.1565}$	$\mathbf{0.2234 \pm 0.0551}$
Soft	$\mathbf{0.0265 \pm 0.0068}$	$\mathbf{0.4154 \pm 0.1043}$	0.0070 ± 0.0017	0.3159 ± 0.0362	0.6221 ± 0.0324	0.2755 ± 0.0083
NODE	0.0307 ± 0.0041	0.4587 ± 0.0386	0.0056 ± 0.0006	0.2785 ± 0.0739	0.6479 ± 0.0739	0.2939 ± 0.0141
Projected	$34965.8566 \pm 60562.5730$	322.7897 ± 558.3569	0.0276 ± 0.0069	0.7561 ± 0.2136	0.6196 ± 0.1070	0.3243 ± 0.0437

SDE diffusion, 10%; observation noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0133 \pm 0.0082}$	$\mathbf{0.2305 \pm 0.0284}$	$\mathbf{1.6895 \pm 1.0017}$	$\mathbf{0.1872 \pm 0.0414}$	$\mathbf{0.0282 \pm 0.0137}$	$\mathbf{0.6154 \pm 0.2174}$
Soft	0.0158 ± 0.0087	0.2976 ± 0.0654	1.9479 ± 2.0716	0.2115 ± 0.0429	0.0410 ± 0.0023	0.7620 ± 0.0298
NODE	0.1066 ± 0.0884	0.8331 ± 0.2036	2.5045 ± 2.8408	0.2311 ± 0.0934	0.0403 ± 0.0009	0.7633 ± 0.0603
Projected	0.0353 ± 0.0109	0.4701 ± 0.0552	7.8830 ± 8.0662	0.5654 ± 0.3423	28.3012 ± 8.9828	12.9843 ± 2.7853

SDE diffusion, 10%; observation noise, 10%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0738 ± 0.0386	$\mathbf{0.5245 \pm 0.0895}$	$\mathbf{0.0048 \pm 0.0016}$	$\mathbf{0.2374 \pm 0.0593}$	$\mathbf{0.9175 \pm 0.3064}$	$\mathbf{0.3443 \pm 0.0938}$
Soft	0.0597 ± 0.0061	0.5709 ± 0.0382	0.0074 ± 0.0015	0.3225 ± 0.1025	1.1685 ± 0.2134	0.3759 ± 0.0134
NODE	$\mathbf{0.0540 \pm 0.0158}$	0.5555 ± 0.0924	0.0080 ± 0.0042	0.3291 ± 0.1567	1.0300 ± 0.0618	0.3631 ± 0.0192
Projected	$98092.9102 \pm 55188.3971$	699.3368 ± 287.9517	0.0263 ± 0.0068	0.7138 ± 0.1969	1.0030 ± 0.1537	0.4128 ± 0.0194

SDE diffusion, 20%; observation noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0271 \pm 0.0267}$	$\mathbf{0.3030 \pm 0.0797}$	5.6679 ± 2.8775	0.3135 ± 0.0507	$\mathbf{0.0447 \pm 0.0055}$	$\mathbf{0.7306 \pm 0.0499}$
Soft	0.0284 ± 0.0192	0.3879 ± 0.0730	5.6357 ± 5.2317	0.3579 ± 0.0420	0.0474 ± 0.0042	0.7457 ± 0.0710
NODE	0.0924 ± 0.0570	0.7954 ± 0.0442	$\mathbf{5.0245 \pm 5.6592}$	$\mathbf{0.3032 \pm 0.0421}$	0.0490 ± 0.0072	0.7823 ± 0.0386
Projected	0.0444 ± 0.0175	0.4720 ± 0.0024	11.1735 ± 11.3091	0.6431 ± 0.2453	28.2999 ± 8.9845	12.4516 ± 3.1851

SDE diffusion, 20%; observation noise, 10%

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.2886 ± 0.2249	0.6773 ± 0.1167	$\mathbf{0.0096 \pm 0.0042}$	$\mathbf{0.3331 \pm 0.1073}$	$\mathbf{1.7059 \pm 0.4657}$	$\mathbf{0.4719 \pm 0.0905}$
Soft	$\mathbf{0.1362 \pm 0.0135}$	$\mathbf{0.5669 \pm 0.0045}$	0.0142 ± 0.0057	0.4188 ± 0.1581	2.3939 ± 1.4786	0.4846 ± 0.0828
NODE	0.1471 ± 0.0247	0.6142 ± 0.0121	0.0175 ± 0.0113	0.5009 ± 0.2968	2.3520 ± 1.0064	0.5100 ± 0.0122
Projected	15.1062 ± 21.0702	3.9998 ± 4.8089	0.0306 ± 0.0111	0.6933 ± 0.2762	1.8894 ± 0.7099	0.5171 ± 0.0052

Table 14: SDE noise ablation.

Additive noise, 10%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	0.5172	**0.0731**	0.5808	**0.2602**	0.3901	**0.1358**
Soft	**0.2668**	0.2755	**0.5540**	0.3279	**0.3727**	0.1678
NODE	0.5182	0.1034	0.5628	0.4139	0.4385	0.1760
Projected	–	0.1063	1815.9978	0.3569	0.6017	0.2417

Additive noise, 20%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	**0.3880**	0.0928	**0.6654**	0.4844	**0.6686**	**0.1453**
Soft	0.4596	**0.0863**	0.7626	**0.4429**	0.6697	0.1827
NODE	0.5212	0.1180	0.7653	0.5086	0.7979	0.1814
Projected	0.5225	0.1072	–	786.2899	–	0.2357

Table 15: Additive noise calibration. Normalised observation weighted interval score (WIS).

Heteroscedastic noise, 10%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1131 \pm 0.0220}$	$\mathbf{0.0926 \pm 0.0260}$	$\mathbf{0.4039 \pm 0.1506}$	$\mathbf{0.3231 \pm 0.1600}$	$\mathbf{0.1533 \pm 0.1237}$	$\mathbf{0.1232 \pm 0.0269}$
Soft	0.3227 ± 0.2095	0.1051 ± 0.0199	0.4751 ± 0.0042	0.3411 ± 0.0503	0.3105 ± 0.1364	0.1666 ± 0.0640
NODE	0.4222 ± 0.1222	0.1465 ± 0.0020	0.4892 ± 0.0279	0.3883 ± 0.0419	0.1646 ± 0.0170	0.2214 ± 0.0625
Projected	0.4285 ± 0.0642	0.3286 ± 0.2117	11.6536 ± 3.4062	0.3843 ± 0.0934	0.7595 ± 0.2837	0.2386 ± 0.0447

Heteroscedastic noise, 20%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1418 \pm 0.0131}$	$\mathbf{0.1175 \pm 0.0082}$	$\mathbf{0.3407 \pm 0.1744}$	$\mathbf{0.4749 \pm 0.0762}$	$\mathbf{0.1301 \pm 0.0086}$	$\mathbf{0.1560 \pm 0.0273}$
Soft	0.3129 ± 0.1142	0.1343 ± 0.0050	0.4818 ± 0.0137	0.5030 ± 0.0913	0.3200 ± 0.2006	0.1660 ± 0.0385
NODE	0.5612 ± 0.0991	0.1576 ± 0.0087	0.4804 ± 0.0032	0.5420 ± 0.0698	0.1938 ± 0.0349	0.2447 ± 0.0642
Projected	0.4488 ± 0.0503	0.3670 ± 0.1888	11.5908 ± 3.3863	497.0389 ± 702.1161	0.7601 ± 0.2839	0.2552 ± 0.0439

Table 16: Heteroscedastic noise calibration. Normalised observation weighted interval score (WIS).

SDE diffusion, 5%; observation noise, 10%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.2779 \pm 0.1091}$	$\mathbf{0.1085 \pm 0.0164}$	$\mathbf{0.4513 \pm 0.1180}$	0.3713 ± 0.0771	$\mathbf{0.3401 \pm 0.0392}$	$\mathbf{0.1827 \pm 0.0458}$
Soft	0.3115 ± 0.0177	0.1129 ± 0.0065	0.5565 ± 0.0123	$\mathbf{0.3557 \pm 0.0521}$	0.3919 ± 0.0503	0.2266 ± 0.0095
NODE	0.4869 ± 0.0470	0.1651 ± 0.0105	0.5537 ± 0.0312	0.3692 ± 0.0255	0.3860 ± 0.0652	0.2213 ± 0.0078
Projected	0.4580 ± 0.0625	0.3964 ± 0.1296	10.7829 ± 3.1255	310.2704 ± 536.7302	0.7445 ± 0.2454	0.2675 ± 0.0269

SDE diffusion, 10%; observation noise, 10%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.2625 \pm 0.0072}$	$\mathbf{0.1468 \pm 0.0219}$	$\mathbf{0.4705 \pm 0.0962}$	0.4108 ± 0.0521	$\mathbf{0.3342 \pm 0.0420}$	0.2785 ± 0.0823
Soft	0.3144 ± 0.0229	0.1762 ± 0.0299	0.5454 ± 0.0025	0.4458 ± 0.0035	0.3850 ± 0.0777	0.3157 ± 0.0193
NODE	0.6096 ± 0.1458	0.1872 ± 0.0530	0.5367 ± 0.0115	$\mathbf{0.3956 \pm 0.0494}$	0.3964 ± 0.1121	$\mathbf{0.2782 \pm 0.0054}$
Projected	0.4623 ± 0.0604	0.3646 ± 0.1918	10.6692 ± 3.1796	665.3074 ± 340.3920	0.7168 ± 0.2256	0.3137 ± 0.0115

SDE diffusion, 20%; observation noise, 10%

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.3039 \pm 0.0250}$	$\mathbf{0.2407 \pm 0.0287}$	$\mathbf{0.5284 \pm 0.0252}$	0.4927 ± 0.0888	$\mathbf{0.3631 \pm 0.0728}$	0.4858 ± 0.0701
Soft	0.3558 ± 0.0237	0.2872 ± 0.0092	0.5464 ± 0.0181	0.4275 ± 0.0333	0.4104 ± 0.1008	0.4162 ± 0.0862
NODE	0.5466 ± 0.0194	0.2533 ± 0.0220	0.5572 ± 0.0236	$\mathbf{0.4199 \pm 0.0184}$	0.4488 ± 0.1623	0.4006 ± 0.0215
Projected	0.4429 ± 0.0258	0.4268 ± 0.1405	10.2302 ± 3.4183	3.1614 ± 3.8369	0.6916 ± 0.2690	$\mathbf{0.3822 \pm 0.0100}$

Table 17: Sde noise calibration. Normalised observation weighted interval score (WIS).

Temporal noise, 10%

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	Obs.	WIS	Obs.	WIS	Obs.	WIS	Obs.	WIS	Obs.	WIS	Obs.	WIS
Hard (ours)	**0.1447 ± 0.0054**		**0.0779 ± 0.0103**		**0.4023 ± 0.0980**		0.2953 ± 0.1187		0.2519 ± 0.1364		**0.1011 ± 0.0154**	
Soft	0.2156 ± 0.0228		0.1350 ± 0.0258		0.5088 ± 0.0080		**0.2950 ± 0.0699**		**0.2187 ± 0.0252**		0.1312 ± 0.0383	
NODE	0.4299 ± 0.0896		0.1142 ± 0.0012		0.5236 ± 0.0197		0.3317 ± 0.0355		0.2880 ± 0.0511		0.2038 ± 0.0372	
Projected	0.4538 ± 0.0465		0.3315 ± 0.2128		11.7356 ± 3.4642		796.3079 ± 703.0875		0.7950 ± 0.3194		0.2104 ± 0.0154	

Temporal noise, 20%

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	Obs.	WIS	Obs.	WIS	Obs.	WIS	Obs.	WIS	Obs.	WIS	Obs.	WIS
Hard (ours)	**0.2430 ± 0.0211**		**0.0789 ± 0.0069**		**0.5081 ± 0.1351**		**0.3669 ± 0.1233**		**0.3591 ± 0.0521**		**0.1168 ± 0.0232**	
Soft	0.2940 ± 0.0085		0.0947 ± 0.0016		0.5425 ± 0.0790		0.4347 ± 0.0148		0.4019 ± 0.0509		0.1528 ± 0.0008	
NODE	0.4639 ± 0.0759		0.1305 ± 0.0042		0.5796 ± 0.0480		0.4159 ± 0.0572		0.4363 ± 0.0127		0.1894 ± 0.0291	
Projected	0.4490 ± 0.0151		0.3225 ± 0.2060		11.7641 ± 3.4897		352.2894 ± 609.4940		0.8560 ± 0.3223		0.2096 ± 0.0111	

Table 18: Temporal noise calibration. Normalised observation weighted interval score (WIS)

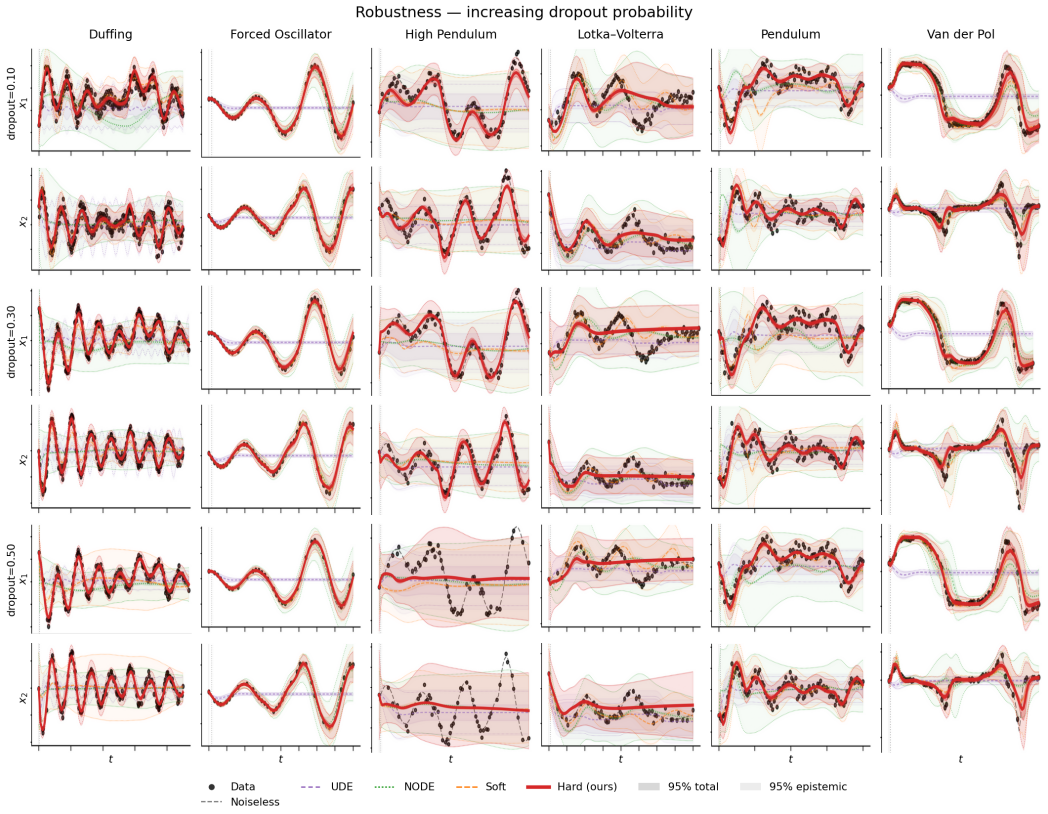


Figure 9: Model rollouts with increasing dropout probability

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	**0.0016**	**0.0843**	0.4803	0.1028	**0.0106**	**0.4173**	0.0650	0.5953	**0.0032**	**0.1771**	0.2364	**0.1641**
Soft	0.0128	0.2766	**0.1863**	**0.0731**	0.0415	0.7822	**0.0475**	**0.3872**	0.0142	0.5231	**0.2033**	0.1683
NODE	0.0878	0.8553	0.3001	0.0895	0.0394	0.7695	0.0720	0.5955	0.0233	0.6936	0.2997	0.2090
Projected	0.0648	0.6065	10.2586	0.8025	0.0420	0.5870	0.0987	0.5601	0.0208	0.5183	1.7559	0.7347

Table 19: Robustness with noise=0.05, dropout probability=0.30: state MSE and normalised state CRPS.

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	**0.1174**	0.0875	**0.3121**	0.3656	**0.2169**	0.1347
Soft	0.2273	**0.0810**	0.5044	**0.2879**	0.3998	**0.1313**
NODE	0.5772	0.0928	0.4783	0.3737	0.4813	0.1640
Projected	0.5250	0.7716	0.5069	0.4772	0.4710	0.7122

Table 20: Normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	**0.0023**	**0.1051**	**0.1832**	**0.0684**	**0.0403**	0.7766	0.0685	0.6172	**0.0013**	**0.1247**	**0.1673**	**0.1407**
Soft	0.0764	0.7920	0.3635	0.0931	0.0412	0.7724	**0.0404**	**0.3860**	0.0032	0.1990	0.2616	0.1945
NODE	0.0834	0.8360	0.3833	0.0992	0.0406	0.7858	0.0715	0.5667	0.0158	0.5756	0.4449	0.2633
Projected	0.0652	0.6128	10.2479	0.8020	0.0421	**0.5880**	0.0983	0.5566	0.0209	0.5200	1.7465	0.7323

Table 21: Robustness with noise=0.05, dropout probability=0.50: state MSE and normalised state CRPS.

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	**0.1285**	**0.0671**	0.4978	0.4067	**0.1838**	**0.1123**
Soft	0.5131	0.0714	0.4978	**0.2963**	0.2245	0.1509
NODE	0.5416	0.1027	**0.4923**	0.3999	0.4199	0.2072
Projected	0.5186	0.7589	0.5118	0.5059	0.4725	0.7065

Table 22: Normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0689	0.7528	**0.2805**	**0.0782**	0.0420	0.7932	**0.0234**	**0.2709**	0.0065	0.2725	**0.2047**	**0.1600**
Soft	**0.0101**	**0.2164**	3.3863	0.3549	0.0409	0.7803	0.0409	0.4125	**0.0053**	**0.2598**	0.3368	0.2193
NODE	0.0724	0.7601	0.3977	0.1019	**0.0406**	0.7886	0.0738	0.6097	0.0105	0.4444	0.3429	0.2167
Projected	0.0647	0.6078	10.2458	0.8016	147758.5781	1884.3776	0.0359	0.3315	0.0160	0.5828	0.4077	0.2709

Table 23: Robustness with noise=0.10, dropout probability=0.10: state MSE and normalised state CRPS.

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	0.5172	**0.0731**	0.5808	**0.2602**	0.3901	**0.1358**
Soft	**0.2668**	0.2755	**0.5540**	0.3279	**0.3727**	0.1678
NODE	0.5182	0.1034	0.5628	0.4139	0.4385	0.1760
Projected	0.5477	0.1063	1815.9978	0.3569	0.6017	0.2417

Table 24: Normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum		Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	**0.0056**	**0.1590**	0.4585	0.1208	**0.0139**	**0.4743**	0.0680	0.5887	0.0087	0.3342	**0.1824**	**0.1553**
Soft	0.0234	0.3409	**0.1935**	**0.0766**	0.0414	0.7731	**0.0466**	**0.4212**	**0.0074**	**0.3256**	0.2807	0.2043
NODE	0.0421	0.5685	0.2519	0.0937	0.0433	0.7809	0.0785	0.6204	0.0224	0.6277	0.2713	0.1972
Projected	0.0267	0.3948	0.2615	0.0927	0.0426	0.6023	118739.5859	809.4227	0.0209	0.5252	0.3433	0.2373

Table 25: Robustness with noise=0.20, dropout probability=0.10: state MSE and normalised state CRPS.

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	**0.3880**	0.0928	**0.6654**	0.4844	**0.6686**	**0.1453**
Soft	0.4596	**0.0863**	0.7626	**0.4429**	0.6697	0.1827
NODE	0.5212	0.1180	0.7653	0.5086	0.7979	0.1814
Projected	0.5225	0.1072	0.8139	786.2899	0.6942	0.2357

Table 26: Normalised observation weighted interval score (WIS)

Robustness to Partial Observability We assess the model's ability to deal with partial observability

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0027 \pm 0.0012}$	$\mathbf{0.1010 \pm 0.0185}$	$\mathbf{0.2260 \pm 0.1620}$	$\mathbf{0.0803 \pm 0.0201}$	$\mathbf{0.0217 \pm 0.0198}$	$\mathbf{0.5319 \pm 0.3255}$
Soft	0.0239 ± 0.0279	0.3342 ± 0.2035	1.6235 ± 2.2177	0.2495 ± 0.2837	0.0232 ± 0.0117	0.5979 ± 0.2006
NODE	0.0251 ± 0.0171	0.3818 ± 0.1297	0.3046 ± 0.2750	0.1078 ± 0.0524	0.0230 ± 0.0138	0.6043 ± 0.1482
Projected	0.0142 ± 0.0063	0.2887 ± 0.0912	9.1744 ± 7.1175	0.7139 ± 0.3617	0.3116 ± 0.0788	3.5051 ± 0.3048

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0221 \pm 0.0176}$	$\mathbf{0.3485 \pm 0.2303}$	$\mathbf{0.0009 \pm 0.0002}$	$\mathbf{0.1102 \pm 0.0213}$	$\mathbf{0.1546 \pm 0.0470}$	$\mathbf{0.1162 \pm 0.0231}$
Soft	0.0349 ± 0.0195	0.4105 ± 0.0875	0.0019 ± 0.0008	0.1649 ± 0.0331	0.3840 ± 0.3703	0.1857 ± 0.1112
NODE	0.0404 ± 0.0112	0.5579 ± 0.0655	0.0050 ± 0.0023	0.2605 ± 0.0514	0.5568 ± 0.4383	0.2477 ± 0.1351
Projected	6919.7830 ± 13913.7229	202.1498 ± 270.7013	0.0111 ± 0.0053	0.4611 ± 0.1884	0.5344 ± 0.2462	0.2976 ± 0.0920

Table 27: State MSE and State CRPS (normalised) for observability, nominal

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1238 \pm 0.0125}$	$\mathbf{0.0750 \pm 0.0094}$	$\mathbf{0.3753 \pm 0.1471}$	$\mathbf{0.2629 \pm 0.1211}$	$\mathbf{0.1787 \pm 0.0320}$	$\mathbf{0.0913 \pm 0.0165}$
Soft	0.2554 ± 0.1212	0.1689 ± 0.1453	0.4165 ± 0.0996	0.3121 ± 0.0431	0.2032 ± 0.0326	0.1465 ± 0.0795
NODE	0.2820 ± 0.0687	0.1013 ± 0.0235	0.4425 ± 0.0763	0.3929 ± 0.0390	0.2448 ± 0.0106	0.1880 ± 0.0994
Projected	0.3287 ± 0.0735	0.4043 ± 0.1870	1.6488 ± 0.1120	156.3102 ± 216.1786	0.4873 ± 0.1331	0.2084 ± 0.0583

Table 28: Normalised observation weighted interval score (WIS).

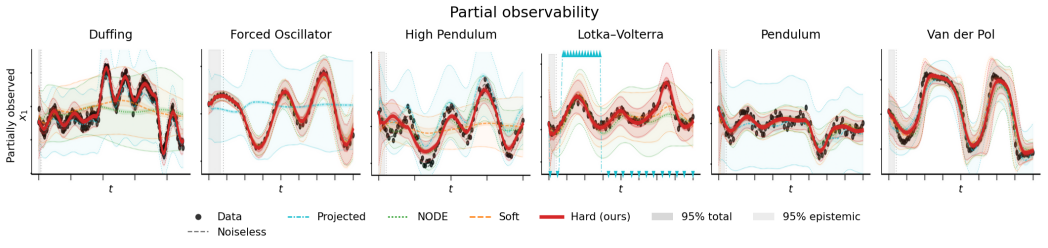


Figure 10: Robustness to observability - only one state is able to be measured and compared with the data, but the internal states that hold the physical relation is embedded into the model

Physical Mis-specification. We report per-system results for the informed, overconfident, uncertain, and systematically misspecified regimes, assessing sensitivity to parameter-prior specification and errors in the embedded relation.

Our results are more successful when the physical parameters are allowed to also be inferred.

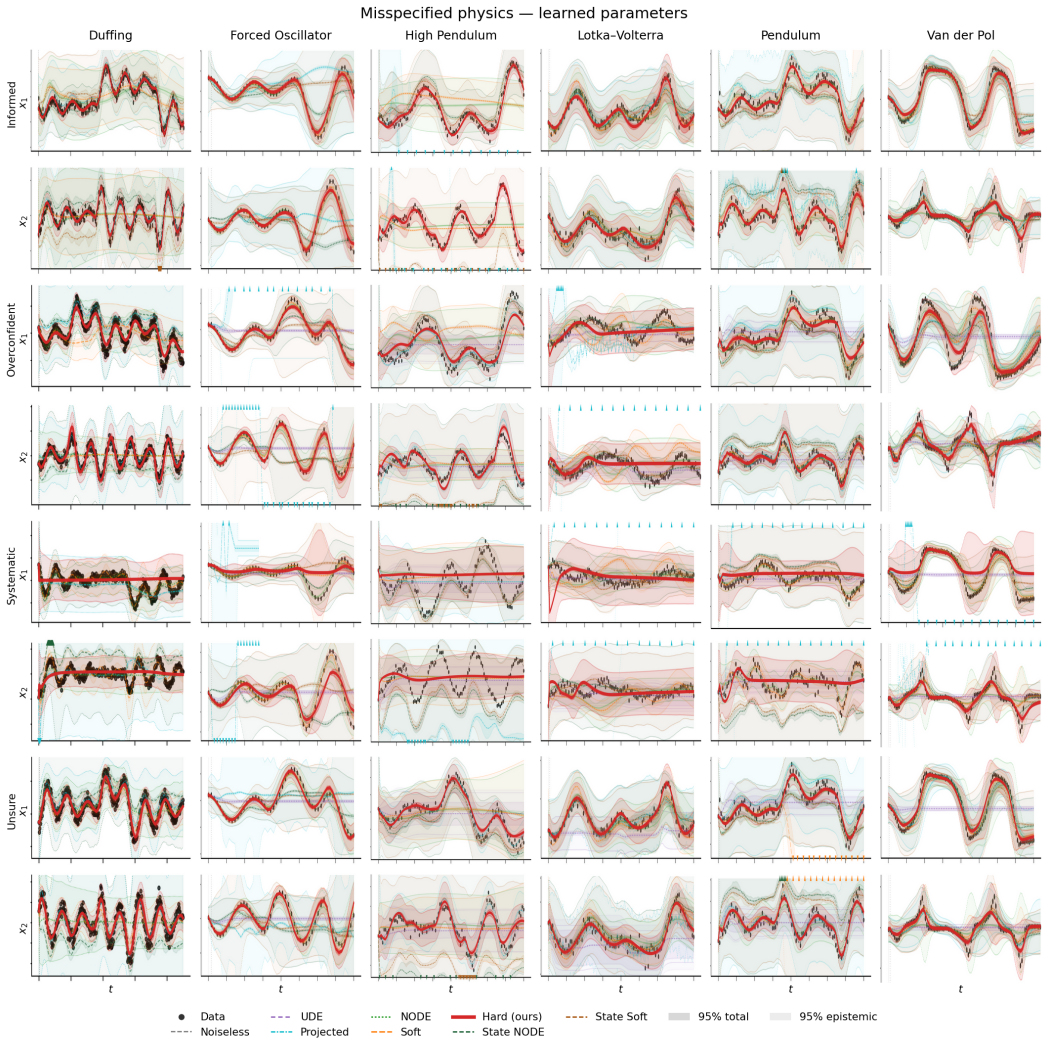


Figure 11: Sweep of the rollouts under embedded physics misspecification, with physics parameters are set as learnable

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0178 ± 0.0292	$\mathbf{0.2912 \pm 0.2927}$	$\mathbf{0.2240 \pm 0.1776}$	0.1021 ± 0.0489	$\mathbf{0.0058 \pm 0.0008}$	$\mathbf{0.2983 \pm 0.0219}$
Soft	0.0346 ± 0.0239	0.5423 ± 0.2783	0.3006 ± 0.3898	$\mathbf{0.0976 \pm 0.0539}$	3.2017 ± 5.4962	4.3049 ± 6.1144
NODE	0.0321 ± 0.0122	0.5738 ± 0.0949	0.4233 ± 0.4508	0.1216 ± 0.0355	0.0316 ± 0.0023	0.7793 ± 0.0419
State NODE	0.0645 ± 0.0362	0.8707 ± 0.2279	5.1841 ± 3.9294	0.6431 ± 0.0964	0.4436 ± 0.0347	3.5640 ± 0.2679
State Soft	0.0515 ± 0.0229	0.7775 ± 0.2377	6.5687 ± 3.1008	0.7572 ± 0.0399	0.2625 ± 0.0263	2.8843 ± 0.3400
Projected	$\mathbf{0.0134 \pm 0.0057}$	0.3091 ± 0.0434	6.8354 ± 2.2396	0.7804 ± 0.0770	0.7031 ± 0.2076	3.7683 ± 0.0469

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0053 \pm 0.0001}$	$\mathbf{0.1618 \pm 0.0214}$	$\mathbf{0.0024 \pm 0.0025}$	$\mathbf{0.1690 \pm 0.1129}$	$\mathbf{0.2221 \pm 0.0998}$	$\mathbf{0.1594 \pm 0.0450}$
Soft	0.0353 ± 0.0087	0.5915 ± 0.1757	0.0650 ± 0.1094	0.1871 ± 0.0587	0.2967 ± 0.0937	0.1856 ± 0.0378
NODE	0.0290 ± 0.0054	0.5295 ± 0.1549	0.0034 ± 0.0002	0.2138 ± 0.0127	0.5758 ± 0.1613	0.2936 ± 0.0467
State NODE	0.0178 ± 0.0080	0.3519 ± 0.1753	0.0952 ± 0.0311	1.6590 ± 0.4275	0.4469 ± 0.0611	0.2842 ± 0.0313
State Soft	0.0171 ± 0.0040	0.3293 ± 0.0626	0.0697 ± 0.0387	1.2943 ± 0.4539	0.4457 ± 0.1330	0.2696 ± 0.0357
Projected	0.0076 ± 0.0039	0.1990 ± 0.0215	0.0239 ± 0.0170	0.6625 ± 0.3017	0.4139 ± 0.0904	0.2766 ± 0.0529

Table 29: Fixed parameters — Informed misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.2441 \pm 0.1531}$	0.0902 ± 0.0286	$\mathbf{0.2988 \pm 0.0231}$	$\mathbf{0.1670 \pm 0.0120}$	$\mathbf{0.2218 \pm 0.0875}$	$\mathbf{0.1244 \pm 0.0340}$
Soft	0.4010 ± 0.1738	$\mathbf{0.0862 \pm 0.0277}$	6.7531 ± 10.8164	0.4446 ± 0.1242	0.3051 ± 0.1552	0.1445 ± 0.0303
NODE	0.4012 ± 0.0548	0.1184 ± 0.0129	0.5134 ± 0.0216	0.3758 ± 0.1005	0.2423 ± 0.0301	0.2095 ± 0.0362
State NODE	0.5908 ± 0.0980	0.3969 ± 0.0371	1.7880 ± 0.1533	0.3904 ± 0.1908	0.9512 ± 0.2117	0.2304 ± 0.0228
State Soft	0.5464 ± 0.0984	0.4465 ± 0.0243	1.4732 ± 0.1816	0.3663 ± 0.1412	0.7799 ± 0.2042	0.2260 ± 0.0229
Projected	0.4451 ± 0.0835	0.4645 ± 0.0424	1.9249 ± 0.1236	0.3137 ± 0.1267	0.6661 ± 0.1253	0.2203 ± 0.0364

Table 30: Fixed parameters — Informed misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0063 \pm 0.0022}$	$\mathbf{0.1982 \pm 0.0092}$	0.8735 ± 0.3402	0.2286 ± 0.0246	$\mathbf{0.0229 \pm 0.0082}$	$\mathbf{0.6344 \pm 0.1750}$
Soft	0.0371 ± 0.0258	0.5556 ± 0.2754	$\mathbf{0.1373 \pm 0.1378}$	$\mathbf{0.0687 \pm 0.0140}$	0.0307 ± 0.0019	0.7494 ± 0.0366
NODE	0.0337 ± 0.0151	0.5891 ± 0.1165	0.2592 ± 0.0778	0.1164 ± 0.0315	0.0326 ± 0.0024	0.7782 ± 0.0438
State NODE	0.0765 ± 0.0321	1.0106 ± 0.2820	4.7454 ± 3.5152	0.6037 ± 0.0836	0.3327 ± 0.0499	3.1214 ± 0.2090
State Soft	0.0521 ± 0.0257	0.7824 ± 0.2774	5.5632 ± 3.0706	0.6854 ± 0.0497	0.2411 ± 0.0231	2.6717 ± 0.1873
Projected	0.0179 ± 0.0067	0.3670 ± 0.0314	7.5359 ± 3.2105	0.7886 ± 0.0234	518.0436 ± 896.2406	61.7644 ± 100.8096

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0513 ± 0.0118	0.7311 ± 0.0705	0.0107 ± 0.0104	0.4187 ± 0.2893	0.3054 ± 0.1009	0.2106 ± 0.0417
Soft	0.0317 ± 0.0119	0.4945 ± 0.1820	0.0160 ± 0.0228	0.1897 ± 0.0017	$\mathbf{0.2691 \pm 0.0616}$	$\mathbf{0.1767 \pm 0.0436}$
NODE	0.0268 ± 0.0126	0.4899 ± 0.2205	$\mathbf{0.0028 \pm 0.0005}$	$\mathbf{0.1864 \pm 0.0100}$	0.6089 ± 0.1300	0.3040 ± 0.0331
State NODE	0.0143 ± 0.0068	0.2880 ± 0.0717	0.1094 ± 0.0447	1.7435 ± 0.5241	0.5173 ± 0.0882	0.3065 ± 0.0165
State Soft	$\mathbf{0.0131 \pm 0.0045}$	$\mathbf{0.2682 \pm 0.0376}$	0.0781 ± 0.0305	1.4583 ± 0.4369	0.5366 ± 0.1818	0.3117 ± 0.0465
Projected	595.1223 ± 850.0465	59.4193 ± 55.6849	0.0179 ± 0.0111	0.5692 ± 0.1518	0.5328 ± 0.1257	0.3274 ± 0.0497

Table 31: Fixed parameters — Overconfident misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1841 \pm 0.0112}$	0.1540 ± 0.0230	$\mathbf{0.4375 \pm 0.0893}$	0.4675 ± 0.0381	0.3506 ± 0.1839	0.1569 ± 0.0337
Soft	0.4117 ± 0.1845	$\mathbf{0.0671 \pm 0.0051}$	0.5105 ± 0.0115	0.3863 ± 0.1176	0.3233 ± 0.1554	$\mathbf{0.1357 \pm 0.0303}$
NODE	0.4103 ± 0.0683	0.1068 ± 0.0132	0.5196 ± 0.0305	0.3641 ± 0.1321	$\mathbf{0.2299 \pm 0.0207}$	0.2178 ± 0.0295
State NODE	0.6490 ± 0.1186	0.3678 ± 0.0345	1.5749 ± 0.1213	0.3357 ± 0.1095	0.9824 ± 0.2448	0.2442 ± 0.0237
State Soft	0.5494 ± 0.1108	0.4166 ± 0.0147	1.3540 ± 0.1033	$\mathbf{0.3082 \pm 0.0791}$	0.8650 ± 0.2104	0.2441 ± 0.0265
Projected	0.4510 ± 0.0863	0.4647 ± 0.0009	52.0933 ± 87.1430	39.0027 ± 43.6832	0.6201 ± 0.1609	0.2477 ± 0.0376

Table 32: Fixed parameters — Overconfident misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.1200 ± 0.1525	0.8884 ± 0.2269	2.5000 ± 0.7629	0.3100 ± 0.0086	5.7413 ± 9.8934	4.8014 ± 6.9281
Soft	$\mathbf{0.0278 \pm 0.0181}$	$\mathbf{0.4667 \pm 0.2195}$	1.4325 ± 0.7982	0.2271 ± 0.0518	1.6108 ± 2.7367	3.4377 ± 4.6419
NODE	0.0329 ± 0.0102	0.5907 ± 0.0883	$\mathbf{0.2112 \pm 0.1010}$	$\mathbf{0.1052 \pm 0.0274}$	$\mathbf{0.0322 \pm 0.0017}$	$\mathbf{0.7772 \pm 0.0433}$
State NODE	0.0807 ± 0.0321	1.0451 ± 0.2477	4.7118 ± 3.3739	0.6051 ± 0.0746	0.3468 ± 0.0555	3.1502 ± 0.3468
State Soft	0.0542 ± 0.0230	0.8063 ± 0.2375	5.8415 ± 3.0744	0.7217 ± 0.0468	0.2737 ± 0.0150	2.7627 ± 0.2733
Projected	90.8864 ± 157.3287	19.8250 ± 32.8633	4362.2921 ± 6156.4581	16.5470 ± 22.2938	318.4422 ± 535.4260	37.0795 ± 55.8937

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0405 ± 0.0053	0.6450 ± 0.0662	0.0192 ± 0.0145	0.7064 ± 0.0921	1.6309 ± 0.7680	0.6208 ± 0.2293
Soft	0.0313 ± 0.0179	0.5252 ± 0.2735	$\mathbf{0.0024 \pm 0.0000}$	$\mathbf{0.1730 \pm 0.0162}$	$\mathbf{0.4449 \pm 0.1264}$	$\mathbf{0.2359 \pm 0.0318}$
NODE	0.0301 ± 0.0078	0.5266 ± 0.1738	0.0033 ± 0.0011	0.2069 ± 0.0286	0.5804 ± 0.1470	0.2953 ± 0.0363
State NODE	$\mathbf{0.0145 \pm 0.0051}$	$\mathbf{0.2826 \pm 0.0627}$	0.1053 ± 0.0353	1.7301 ± 0.4381	0.4747 ± 0.1280	0.2880 ± 0.0296
State Soft	0.0150 ± 0.0037	0.2903 ± 0.0272	0.0854 ± 0.0306	1.5196 ± 0.3893	0.5129 ± 0.1928	0.2923 ± 0.0558
Projected	1334.8361 ± 44.5563	113.1358 ± 17.3644	$64282.4188 \pm 111340.3151$	748.9013 ± 1294.9927	$51329.8432 \pm 88893.0429$	78.0958 ± 132.7047

Table 33: Fixed parameters — Systematic misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	0.5914 ± 0.2219	0.1847 ± 0.0026	5.0103 ± 7.7846	0.4394 ± 0.0509	0.5361 ± 0.0412	0.3681 ± 0.1395
Soft	$\mathbf{0.3666 \pm 0.1433}$	0.1440 ± 0.0273	2.8004 ± 3.9739	0.4241 ± 0.2100	$\mathbf{0.2299 \pm 0.0329}$	$\mathbf{0.1797 \pm 0.0307}$
NODE	0.4088 ± 0.0562	$\mathbf{0.1063 \pm 0.0119}$	$\mathbf{0.5174 \pm 0.0275}$	0.3864 ± 0.1104	0.2460 ± 0.0348	0.2088 ± 0.0258
State NODE	0.6582 ± 0.1115	0.3668 ± 0.0273	1.5890 ± 0.1809	0.3351 ± 0.1091	0.9745 ± 0.2128	0.2367 ± 0.0236
State Soft	0.5538 ± 0.1014	0.4195 ± 0.0137	1.3920 ± 0.1411	$\mathbf{0.3188 \pm 0.0757}$	0.8903 ± 0.1995	0.2381 ± 0.0303
Projected	13.2690 ± 21.8151	14.9035 ± 20.3807	32.2742 ± 47.6259	83.1239 ± 12.5536	733.6347 ± 1269.2671	75.9261 ± 129.5761

Table 34: Fixed parameters — Systematic misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0025 \pm 0.0009}$	$\mathbf{0.1184 \pm 0.0254}$	$\mathbf{0.2366 \pm 0.0896}$	0.1009 ± 0.0247	3.6812 ± 6.3450	3.6974 ± 5.4526
Soft	0.0365 ± 0.0240	0.5416 ± 0.2649	0.4130 ± 0.6004	$\mathbf{0.0968 \pm 0.0564}$	$\mathbf{0.0304 \pm 0.0021}$	$\mathbf{0.7555 \pm 0.0346}$
NODE	0.0351 ± 0.0079	0.5979 ± 0.0727	0.2924 ± 0.1889	0.1077 ± 0.0093	0.0315 ± 0.0022	0.7789 ± 0.0457
State NODE	0.0788 ± 0.0370	1.0069 ± 0.2987	5.3604 ± 4.0027	0.6548 ± 0.0915	0.4291 ± 0.0510	3.4877 ± 0.2555
State Soft	0.0479 ± 0.0270	0.7164 ± 0.3093	6.5599 ± 3.1606	0.7577 ± 0.0381	0.2592 ± 0.0165	2.8491 ± 0.3228
Projected	0.0143 ± 0.0048	0.3176 ± 0.0368	9.6432 ± 5.179	0.6987 ± 0.151	291.9413 ± 412.1397	41.9978 ± 54.1559

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0242 ± 0.0049	0.5864 ± 0.1386	$\mathbf{0.0022 \pm 0.0020}$	$\mathbf{0.1621 \pm 0.1016}$	$\mathbf{0.1818 \pm 0.0502}$	$\mathbf{0.1458 \pm 0.0324}$
Soft	0.0341 ± 0.0023	0.5742 ± 0.0768	0.0024 ± 0.0002	0.1783 ± 0.0130	0.3148 ± 0.0546	0.1949 ± 0.0339
NODE	0.0280 ± 0.0048	0.5196 ± 0.1609	0.0026 ± 0.0004	0.1824 ± 0.0051	0.5755 ± 0.1132	0.2951 ± 0.0279
State NODE	0.0202 ± 0.0083	0.3911 ± 0.2000	0.1099 ± 0.0415	1.7717 ± 0.5029	0.4338 ± 0.0422	0.2794 ± 0.0074
State Soft	$\mathbf{0.0157 \pm 0.0033}$	$\mathbf{0.3063 \pm 0.0319}$	0.0737 ± 0.0303	1.3997 ± 0.4348	0.5191 ± 0.1907	0.2962 ± 0.0462
Projected	0.0193 ± 0.0103	0.4078 ± 0.1953	0.0219 ± 0.0122	0.6624 ± 0.2051	0.4093 ± 0.0932	0.2685 ± 0.0684

Table 35: Fixed parameters — Unsure misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1510 \pm 0.0268}$	$\mathbf{0.0926 \pm 0.0145}$	3.7218 ± 5.7475	0.3695 ± 0.0996	$\mathbf{0.2162 \pm 0.0720}$	$\mathbf{0.1132 \pm 0.0214}$
Soft	0.4029 ± 0.1649	0.0952 ± 0.0388	$\mathbf{0.5044 \pm 0.0190}$	0.4360 ± 0.0592	0.2221 ± 0.0323	0.1514 ± 0.0296
NODE	0.4118 ± 0.0434	0.1096 ± 0.0047	0.5104 ± 0.0239	0.3712 ± 0.0978	0.2304 ± 0.0249	0.2111 ± 0.0251
State NODE	0.6485 ± 0.1311	0.3998 ± 0.0402	1.7527 ± 0.1422	0.3978 ± 0.1901	0.9877 ± 0.2390	0.2276 ± 0.0157
State Soft	0.5339 ± 0.1108	0.4473 ± 0.0228	1.4536 ± 0.1694	$\mathbf{0.3642 \pm 0.1341}$	0.8482 ± 0.2068	0.2397 ± 0.0341
Projected	0.4441 ± 0.0812	0.4145 ± 0.3512	33.9317 ± 45.2273	0.3909 ± 0.1766	0.6471 ± 0.1221	0.2238 ± 0.0403

Table 36: Fixed parameters — Unsure misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0028 \pm 0.0019}$	$\mathbf{0.1263 \pm 0.0301}$	0.2034 ± 0.1761	0.0860 ± 0.0112	3.6124 ± 3.1675	5.0851 ± 4.2022
Soft	0.0309 ± 0.0210	0.5103 ± 0.2631	$\mathbf{0.1239 \pm 0.0567}$	$\mathbf{0.0768 \pm 0.0125}$	$\mathbf{0.0306 \pm 0.0021}$	$\mathbf{0.7545 \pm 0.0329}$
NODE	0.0321 ± 0.0122	0.5738 ± 0.0949	0.4233 ± 0.4508	0.1216 ± 0.0355	0.0316 ± 0.0023	0.7793 ± 0.0419
State NODE	0.0645 ± 0.0362	0.8707 ± 0.2279	5.1841 ± 3.9294	0.6431 ± 0.0964	0.4436 ± 0.0347	3.5640 ± 0.2679
State Soft	0.0610	0.7826	6.3771 ± 3.3437	0.7450 ± 0.0675	0.2528 ± 0.0262	2.8350 ± 0.3011
Projected	0.0099 ± 0.0020	0.2780 ± 0.0367	6.9357 ± 2.1284	0.7910 ± 0.0796	287.1272 ± 405.3202	39.6467 ± 50.8754

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0112 ± 0.0089	0.2615 ± 0.1615	$\mathbf{0.0016 \pm 0.0005}$	$\mathbf{0.1577 \pm 0.0446}$	$\mathbf{0.2501 \pm 0.1045}$	$\mathbf{0.1624 \pm 0.0340}$
Soft	0.0332 ± 0.0138	0.5264 ± 0.1593	0.0021 ± 0.0002	0.1769 ± 0.0160	0.3416 ± 0.1149	0.1964 ± 0.0415
NODE	0.0290 ± 0.0054	0.5295 ± 0.1549	0.0034 ± 0.0002	0.2138 ± 0.0127	0.5758 ± 0.1613	0.2936 ± 0.0467
State NODE	0.0178 ± 0.0080	0.3519 ± 0.1753	0.0952 ± 0.0311	1.6590 ± 0.4275	0.4469 ± 0.0611	0.2842 ± 0.0313
State Soft	0.0185 ± 0.0058	0.3464 ± 0.1319	0.0813 ± 0.0300	1.4959 ± 0.4163	0.4384 ± 0.1424	0.2654 ± 0.0389
Projected	$\mathbf{0.0096 \pm 0.0047}$	$\mathbf{0.2270 \pm 0.0688}$	0.0398 ± 0.0318	0.9790 ± 0.5605	0.5512 ± 0.1845	0.3190 ± 0.0584

Table 37: Learned parameters — Informed misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1561 \pm 0.0213}$	0.0859 ± 0.0016	4.4630 ± 3.6273	$\mathbf{0.2389 \pm 0.1097}$	$\mathbf{0.2127 \pm 0.0360}$	$\mathbf{0.1289 \pm 0.0251}$
Soft	0.3808 ± 0.1645	$\mathbf{0.0790 \pm 0.0122}$	$\mathbf{0.5035 \pm 0.0147}$	0.3937 ± 0.1091	0.2297 ± 0.0197	0.1535 ± 0.0306
NODE	0.4012 ± 0.0548	0.1184 ± 0.0129	0.5134 ± 0.0216	0.3758 ± 0.1005	0.2423 ± 0.0301	0.2095 ± 0.0362
State NODE	0.5908 ± 0.0980	0.3969 ± 0.0371	1.7880 ± 0.1533	0.3904 ± 0.1908	0.9512 ± 0.2117	0.2304 ± 0.0228
State Soft	0.5360	0.4412 ± 0.0298	1.4499 ± 0.1696	0.3730 ± 0.1657	0.8716 ± 0.1992	0.2259 ± 0.0229
Projected	0.4599 ± 0.0901	0.4707 ± 0.0471	31.2384 ± 41.4276	0.3111 ± 0.0971	0.7361 ± 0.1747	0.2577 ± 0.0460

Table 38: Learned parameters — Informed misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0081 \pm 0.0032}$	$\mathbf{0.2009 \pm 0.0269}$	0.6374 ± 0.5176	0.1713 ± 0.0611	$\mathbf{0.0172 \pm 0.0114}$	$\mathbf{0.5270 \pm 0.2282}$
Soft	–	–	$\mathbf{0.2198 \pm 0.1324}$	$\mathbf{0.0988 \pm 0.0185}$	0.0397 ± 0.0146	0.7633 ± 0.0410
NODE	0.0337 ± 0.0151	0.5891 ± 0.1165	0.2592 ± 0.0778	0.1164 ± 0.0315	0.0326 ± 0.0024	0.7782 ± 0.0438
State NODE	0.0765 ± 0.0321	1.0106 ± 0.2820	4.7454 ± 3.5152	0.6037 ± 0.0836	0.3327 ± 0.0499	3.1214 ± 0.2090
State Soft	–	–	5.7614 ± 3.0259	0.7088 ± 0.0394	0.2568 ± 0.0231	2.7798 ± 0.1816
Projected	0.0240 ± 0.0097	0.4092 ± 0.0095	7.4501 ± 3.4489	0.7788 ± 0.0399	518.3186 ± 896.7101	61.8469 ± 100.9321
UDE	–	–	10.2796 ± 0.0000	0.8097 ± 0.0000	0.0409 ± 0.0000	0.5803 ± 0.0000

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0391 ± 0.0074	0.6433 ± 0.0878	0.0043 ± 0.0017	0.2296 ± 0.0357	$\mathbf{0.2808 \pm 0.1065}$	$\mathbf{0.1871 \pm 0.0414}$
Soft	0.0338 ± 0.0219	0.5262 ± 0.2270	$\mathbf{0.0025 \pm 0.0006}$	$\mathbf{0.1791 \pm 0.0378}$	0.3625 ± 0.1072	0.2040 ± 0.0398
NODE	0.0268 ± 0.0126	0.4899 ± 0.2205	0.0028 ± 0.0005	0.1864 ± 0.0100	0.6089 ± 0.1300	0.3040 ± 0.0331
State NODE	$\mathbf{0.0143 \pm 0.0068}$	0.2880 ± 0.0717	0.1094 ± 0.0447	1.7435 ± 0.5241	0.5173 ± 0.0882	0.3065 ± 0.0165
State Soft	0.0149 ± 0.0029	$\mathbf{0.2875 \pm 0.0240}$	0.0801 ± 0.0281	1.4703 ± 0.3718	0.5104 ± 0.1580	0.3047 ± 0.0371
Projected	1507.3824 ± 87.5377	118.5129 ± 1.3928	0.0239 ± 0.0083	0.7408 ± 0.1794	0.6331 ± 0.2640	0.3690 ± 0.0759
UDE	0.0976	0.5876	0.0219 ± 0.0000	0.5457 ± 0.0000	1.7632 ± 0.0000	0.7382 ± 0.0000

Table 39: Learned parameters — Overconfident misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1802 \pm 0.0042}$	0.1300 ± 0.0295	$\mathbf{0.3910 \pm 0.1143}$	0.4216 ± 0.0591	0.2299 ± 0.0291	$\mathbf{0.1420 \pm 0.0280}$
Soft	–	$\mathbf{0.0815 \pm 0.0103}$	0.5487 ± 0.0813	0.4265 ± 0.1611	$\mathbf{0.2230 \pm 0.0416}$	0.1581 ± 0.0315
NODE	0.4103 ± 0.0683	0.1068 ± 0.0132	0.5196 ± 0.0305	0.3641 ± 0.1321	0.2299 ± 0.0207	0.2178 ± 0.0295
State NODE	0.6490 ± 0.1186	0.3678 ± 0.0345	1.5749 ± 0.1213	0.3357 ± 0.1095	0.9824 ± 0.2448	0.2442 ± 0.0237
State Soft	–	0.4249 ± 0.0087	1.4034 ± 0.1040	$\mathbf{0.3184 \pm 0.0797}$	0.8727 ± 0.1910	0.2392 ± 0.0261
Projected	0.4841 ± 0.1339	0.4618 ± 0.0046	52.1709 ± 87.2715	99.0323 ± 15.3677	0.7405 ± 0.2468	0.2741 ± 0.0439
UDE	–	0.7984 ± 0.0000	0.5121 ± 0.0000	0.5147	0.5014 ± 0.0000	0.7184 ± 0.0000

Table 40: Learned parameters — Overconfident misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0625 ± 0.0243	0.8263 ± 0.0165	3.8064 ± 3.6107	0.4508 ± 0.2218	$\mathbf{0.0291 \pm 0.0003}$	0.7979 ± 0.0080
Soft	–	–	1.5033 ± 0.8845	0.2315 ± 0.0516	0.0304 ± 0.0022	0.7608 ± 0.0321
NODE	$\mathbf{0.0329 \pm 0.0102}$	$\mathbf{0.5907 \pm 0.0883}$	$\mathbf{0.2112 \pm 0.1010}$	$\mathbf{0.1052 \pm 0.0274}$	0.0322 ± 0.0017	0.7772 ± 0.0433
State NODE	0.0807 ± 0.0321	1.0451 ± 0.2477	4.7118 ± 3.3739	0.6051 ± 0.0746	0.3468 ± 0.0555	3.1502 ± 0.3468
State Soft	–	–	5.7555 ± 3.2921	0.7092 ± 0.0653	0.2642 ± 0.0283	2.6688 ± 0.3396
Projected	0.0520 ± 0.0111	0.7815 ± 0.0299	52.6910	1.5736	318.4140 ± 535.5043	37.0515 ± 55.9244
UDE	–	–	10.2796 ± 0.0000	0.8097 ± 0.0000	0.0409	$\mathbf{0.5803}$

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	0.0491 ± 0.0078	0.7389 ± 0.0413	21.9195 ± 30.9608	17.5450 ± 23.7906	1.5787 ± 0.7442	0.6142 ± 0.2226
Soft	0.0336 ± 0.0173	0.5595 ± 0.2599	0.0036 ± 0.0013	0.2125 ± 0.0558	$\mathbf{0.4364 \pm 0.1400}$	$\mathbf{0.2369 \pm 0.0594}$
NODE	0.0301 ± 0.0078	0.5266 ± 0.1738	$\mathbf{0.0033 \pm 0.0011}$	$\mathbf{0.2069 \pm 0.0286}$	0.5804 ± 0.1470	0.2953 ± 0.0363
State NODE	0.0145 ± 0.0051	0.2826 ± 0.0627	0.1053 ± 0.0353	1.7301 ± 0.4381	0.4747 ± 0.1280	0.2880 ± 0.0296
State Soft	$\mathbf{0.0133 \pm 0.0045}$	$\mathbf{0.2789 \pm 0.0666}$	0.0833 ± 0.0344	1.4863 ± 0.4651	0.4563 ± 0.1312	0.2761 ± 0.0442
Projected	1323.1635 ± 37.9733	107.1016 ± 19.7388	$44266.1033 \pm 88532.0770$	535.0161 ± 1067.2016	$38903.0467 \pm 65167.7543$	54.1579 ± 86.1544
UDE	0.0976	0.5876	0.0219 ± 0.0000	0.5457 ± 0.0000	1.7632 ± 0.0000	0.7382 ± 0.0000

Table 41: Learned parameters — Systematic misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	0.5362 ± 0.0193	0.2712 ± 0.1184	0.5199 ± 0.0060	0.5101 ± 0.0226	16.9219 ± 23.2217	0.3438 ± 0.1152
Soft	–	0.1470 ± 0.0256	$\mathbf{0.5044 \pm 0.0162}$	0.4523 ± 0.1979	$\mathbf{0.2442 \pm 0.0510}$	$\mathbf{0.1775 \pm 0.0434}$
NODE	$\mathbf{0.4088 \pm 0.0562}$	$\mathbf{0.1063 \pm 0.0119}$	0.5174 ± 0.0275	0.3864 ± 0.1104	0.2460 ± 0.0348	0.2088 ± 0.0258
State NODE	0.6582 ± 0.1115	0.3668 ± 0.0273	1.5890 ± 0.1809	0.3351 ± 0.1091	0.9745 ± 0.2128	0.2367 ± 0.0236
State Soft	–	0.4148 ± 0.0157	1.3602 ± 0.1642	$\mathbf{0.3101 \pm 0.0866}$	0.8790 ± 0.2202	0.2251 ± 0.0085
Projected	0.6589 ± 0.0686	1.3957	32.2658 ± 47.6315	78.6419 ± 14.5298	523.1812 ± 1044.3997	52.5333 ± 83.8533
UDE	–	0.7984 ± 0.0000	0.5121	0.5147	0.5014 ± 0.0000	0.7184 ± 0.0000

Table 42: Learned parameters — Systematic misspecification: normalised observation weighted interval score (WIS)

Method	Duffing		Forced Oscillator		High Pendulum	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0021 \pm 0.0004}$	$\mathbf{0.1068 \pm 0.0125}$	0.3478 ± 0.2766	0.1118 ± 0.0129	1.1869 ± 2.0237	2.5617 ± 3.4872
Soft	–	–	$\mathbf{0.2253 \pm 0.1559}$	$\mathbf{0.0883 \pm 0.0218}$	$\mathbf{0.0314 \pm 0.0021}$	0.7627 ± 0.0344
NODE	0.0351 ± 0.0079	0.5979 ± 0.0727	0.2924 ± 0.1889	0.1077 ± 0.0093	0.0315 ± 0.0022	0.7789 ± 0.0457
State NODE	0.0788 ± 0.0370	1.0069 ± 0.2987	5.3604 ± 4.0027	0.6548 ± 0.0915	0.4291 ± 0.0510	3.4877 ± 0.2555
State Soft	–	–	6.4882 ± 3.5084	0.7435 ± 0.0486	0.2824 ± 0.0360	2.9830 ± 0.3244
Projected	0.0107 ± 0.0037	0.2986 ± 0.0457	8.1687 ± 2.3639	0.8006 ± 0.1459	193.2072 ± 333.5244	29.1783 ± 44.2853
UDE	–	–	10.2343 ± 0.0000	0.8012 ± 0.0000	0.0420 ± 0.0000	$\mathbf{0.5875 \pm 0.0000}$

Method	Lotka–Volterra		Pendulum		Van der Pol	
	MSE	CRPS	MSE	CRPS	MSE	CRPS
Hard (ours)	$\mathbf{0.0137 \pm 0.0128}$	$\mathbf{0.3007 \pm 0.2142}$	$\mathbf{0.0015 \pm 0.0003}$	$\mathbf{0.1378 \pm 0.0276}$	$\mathbf{0.2091 \pm 0.0328}$	$\mathbf{0.1600 \pm 0.0125}$
Soft	0.0306 ± 0.0095	0.5307 ± 0.1700	1.1495 ± 1.9855	1.4542 ± 2.1448	0.2778 ± 0.0689	0.1731 ± 0.0274
NODE	0.0280 ± 0.0048	0.5196 ± 0.1609	0.0026 ± 0.0004	0.1824 ± 0.0051	0.5755 ± 0.1132	0.2951 ± 0.0279
State NODE	0.0202 ± 0.0083	0.3911 ± 0.2000	0.1099 ± 0.0415	1.7717 ± 0.5029	0.4338 ± 0.0422	0.2794 ± 0.0074
State Soft	0.0180 ± 0.0002	0.3553 ± 0.1224	0.0810 ± 0.0298	1.4866 ± 0.4160	0.5633 ± 0.2222	0.3123 ± 0.0504
Projected	0.0190 ± 0.0069	0.3819 ± 0.1599	0.0197 ± 0.0091	0.6527 ± 0.2245	0.3904 ± 0.0416	0.2650 ± 0.0404
UDE	0.0982	0.5568	0.0209 ± 0.0000	0.5192 ± 0.0000	1.7515 ± 0.0000	0.7338 ± 0.0000

Table 43: Learned parameters — Unsure misspecification: State MSE and State CRPS (normalised)

Method	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS	Obs. WIS
Hard (ours)	$\mathbf{0.1436 \pm 0.0172}$	0.0963 ± 0.0046	4.4872 ± 7.1004	$\mathbf{0.2652 \pm 0.1489}$	$\mathbf{0.1998 \pm 0.0317}$	$\mathbf{0.1272 \pm 0.0142}$
Soft	–	$\mathbf{0.0837 \pm 0.0154}$	0.5113 ± 0.0190	0.4067 ± 0.1254	1.9137 ± 2.8766	0.1372 ± 0.0225
NODE	0.4118 ± 0.0434	0.1096 ± 0.0047	$\mathbf{0.5104 \pm 0.0239}$	0.3712 ± 0.0978	0.2304 ± 0.0249	0.2111 ± 0.0251
State NODE	0.6485 ± 0.1311	0.3998 ± 0.0402	1.7527 ± 0.1422	0.3978 ± 0.1901	0.9877 ± 0.2390	0.2276 ± 0.0157
State Soft	–	0.4406 ± 0.0211	1.5090 ± 0.1714	0.4028 ± 0.1722	0.8680 ± 0.2023	0.2490 ± 0.0366
Projected	0.4957 ± 0.1025	0.4842 ± 0.0962	23.1311 ± 36.8636	0.3701 ± 0.1397	0.7231 ± 0.2522	0.2227 ± 0.0278
UDE	–	0.7881 ± 0.0000	0.5139 ± 0.0000	0.4785	0.4712 ± 0.0000	0.7136 ± 0.0000

Table 44: Learned parameters — Unsure misspecification: normalised observation weighted interval score (WIS)

However, interestingly, the results are still reasonably robust when the physical parameters are fixed, suggesting the latent embedding still assists, even under misspecification.

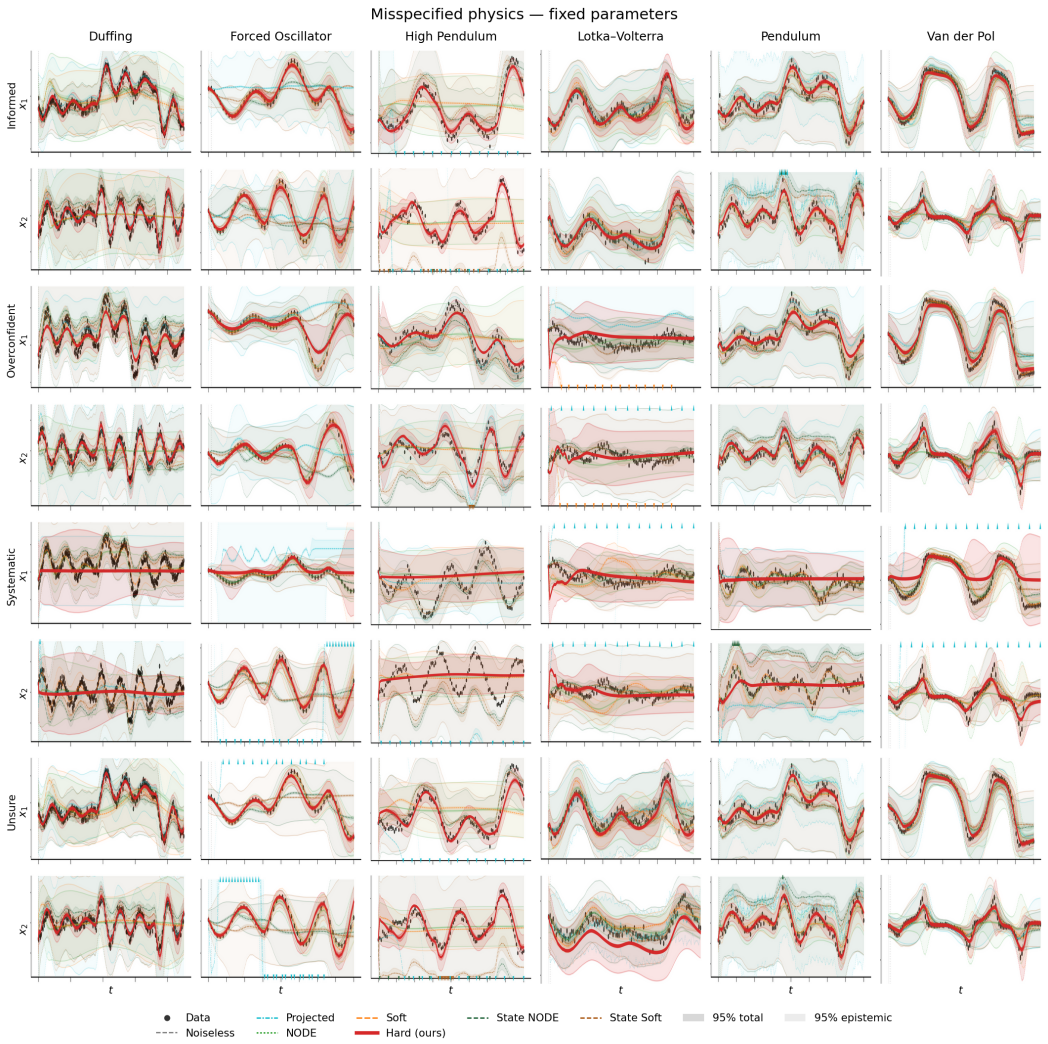


Figure 12: Sweep of rollouts, but the parameters are fixed and *cannot* be updated

Latent Projection. To understand why latent projection is more robust than state-space projection, we examine how the decoder adapts to imperfect physics or data. We measure the shift in decoder Jacobian norms relative to a reference model trained under ideal conditions, with correct physics and noiseless observations, and compare this to models trained with added noise or misspecified physics. A larger shift indicates that the decoder has reshaped the latent-to-observation map to accommodate the discrepancy.

Figure 13 shows this for the Duffing oscillator under systematic misspecification, where the embedded physics has the wrong functional form and no parameter shift can recover the true relation. The decoder shifts substantially relative to the correctly specified case, indicating that it helps absorb the structural error and reduce the degradation of the method.

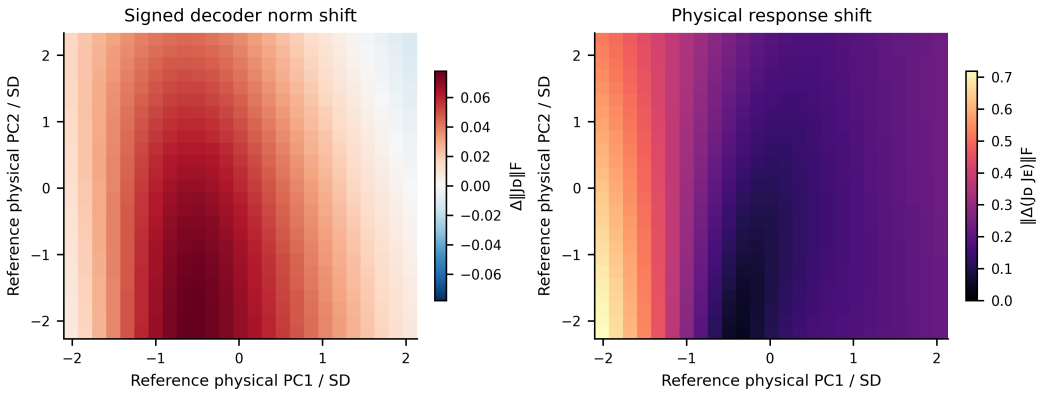


Figure 13: Duffing decoder norm shift under systematic misspecification

As a result, although there is a small degradation, the prediction remains reasonably robust as a result, compared to the direct state projection:

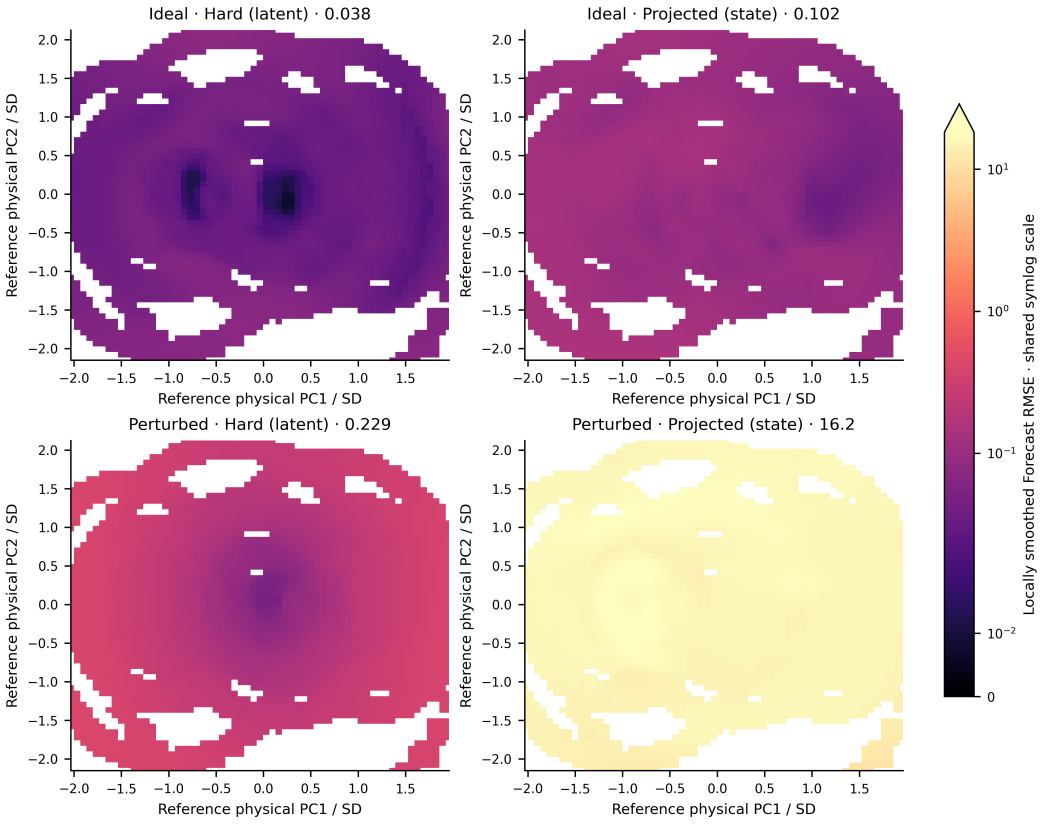


Figure 14: RMSE under misspecification, and how the embedded relation propagates from the latent to the state space as a result

We see a similar effect for noise propagation too, for example on the case of the Pendulum under SDE-style noise, at 20% diffusion and 10% additive observation noise, the decoder shifts significantly:

As a result, the propagated constraint again, although is degraded slightly, remains robust to this misspecification:

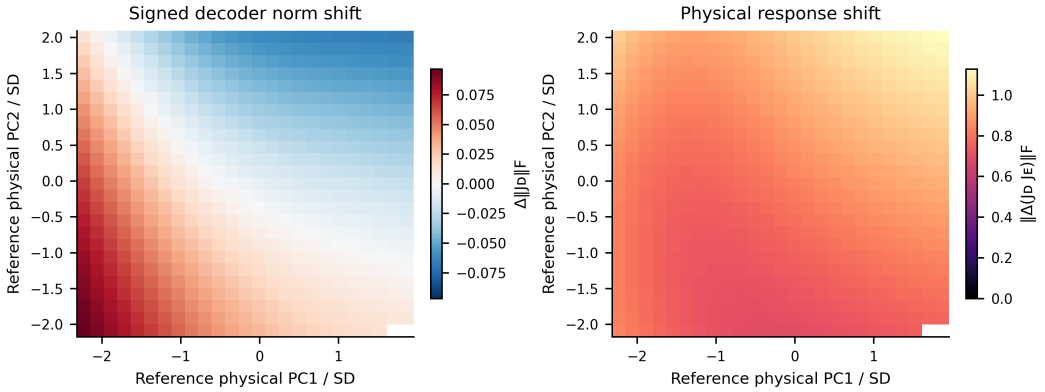


Figure 15: Decoder shift for the pendulum under noise misspecification

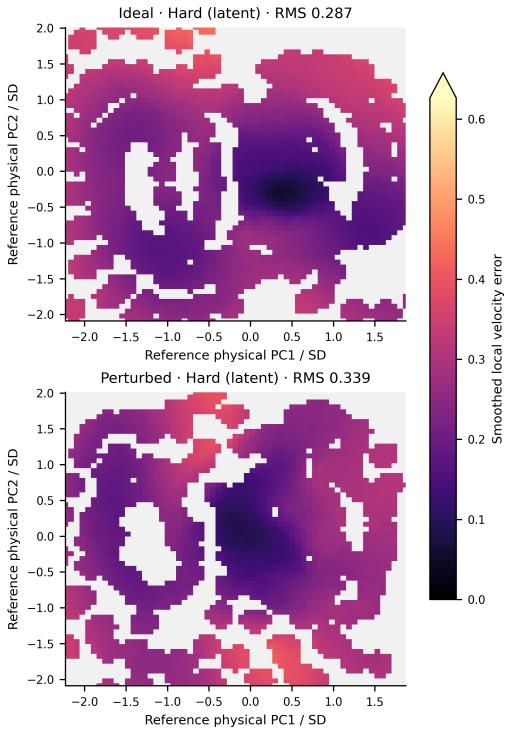


Figure 16: Propagated physical knowledge through the decoder

B.6 Ablation Studies

We ablate components introduced in Section 3 that are not isolated by the soft/hard/no-constraint baseline grid. Each ablation is run across all case studies with all other settings held at the defaults from Appendix B.3.

Stabilizing $\hat{Q}$ in the projection. The latent projection metric $\hat{Q}_t = J_\phi^\top J_\phi + q_{\text{reg}} I$ uses Tikhonov regularization q_{reg} so the closed-form Kalman update remains well-conditioned when the decoder Jacobian J_ϕ is rank-deficient. We sweep $q_{\text{reg}} \in \{10^{-6}, 10^{-4}, 10^{-2}, 10^{0}\}$.

At $q_{\text{reg}} = 10^{-6}$ the regularization is too weak to stabilize the linear solve at points where J_ϕ is rank-deficient. The plateau $q_{\text{reg}} \in [10^{-4}, 10^{-2}]$ is effectively flat on all three metrics across all six systems. At $q_{\text{reg}} = 10^{0}$ the Tikhonov term dominates the natural metric and the projection's effectiveness degrades. Hence we adopt $q_{\text{reg}} = 10^{-4}$ throughout, sitting in the centre of the stable plateau.

An alternative is to use to the Moore-Penrose pseudoinverse $Q_t^\dagger$ when Q_t is rank-deficient. We compare this against the default Cholesky path on $\hat{Q}_t$ at $q_{\text{reg}} = 10^{-4}$, with all else equal. Initially we believed the pseudoinverse could exhibit discontinuities in the vector field, which could harm training. In practice we observe minimal difference between the pseudoinverse and well calibrated Tikhonov regularization.

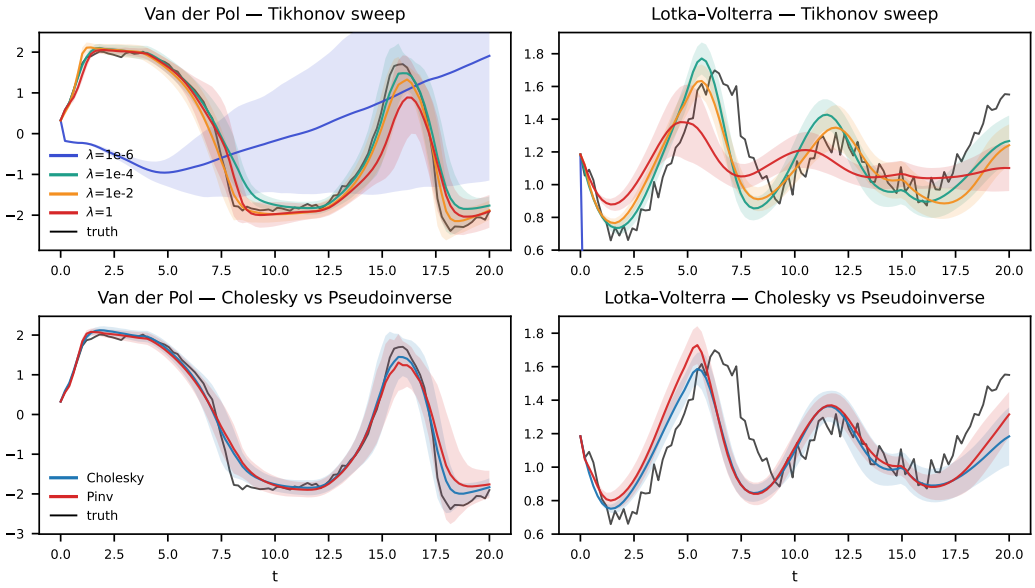


Figure 17: Top row: Tikhonov q_{reg} sweep on Van der Pol (test 0) and Lotka–Volterra (test 1). Bottom row: Fixed Tikhonov at 10^{-4} vs. Moore–Penrose pseudoinverse on the same test trajectories.

Variant	Duffing	Forced Oscillator	High Pendulum	Lotka–Volterra	Pendulum	Van der Pol
Pseudoinverse	0.034 ± 0.007	0.452 ± 0.058	$\mathbf{0.092 \pm 0.013}$	0.157 ± 0.030	0.060 ± 0.047	0.404 ± 0.145
Tikhonov λ=1	0.212 ± 0.045	0.533 ± 0.044	0.194 ± 0.014	0.154 ± 0.031	0.118 ± 0.042	0.515 ± 0.153
Tikhonov λ=1e-2	$\mathbf{0.029 \pm 0.009}$	0.379 ± 0.045	0.186 ± 0.015	0.121 ± 0.024	0.044 ± 0.027	$\mathbf{0.335 \pm 0.125}$
Tikhonov λ=1e-4	0.048 ± 0.007	$\mathbf{0.368 \pm 0.066}$	0.159 ± 0.009	$\mathbf{0.111 \pm 0.021}$	$\mathbf{0.044 \pm 0.029}$	0.372 ± 0.095
Tikhonov λ=1e-6	–	3.390 ± 0.079	3.600 ± 0.059	9.624 ± 1.226	3.142 ± 0.072	2.378 ± 0.094

Table 45: Pinv / Tikhonov ablation — RMSE per case (mean ± std).

Bayesian encoder/decoder. We found when we also inferred posterior distributions over the encoder/decoder parameters this made training a lot more unstable and led to intervals that blow up in time, hence we reduced to inference over the subset of the model.

Case	Bayesian enc/dec			Hard (point)		
	RMSE	Cov_{95}	Width_{95}	RMSE	Cov_{95}	Width_{95}
Forced Oscillator	3.230 ± 0.039	0.893 ± 0.021	12.423 ± 3.065	$\mathbf{0.555 \pm 0.091}$	$\mathbf{0.977 \pm 0.012}$	$\mathbf{3.109 \pm 0.369}$
High Pendulum	0.893 ± 0.246	1.000 ± 0.000	41.168 ± 2.583	$\mathbf{0.145 \pm 0.048}$	$\mathbf{0.922 \pm 0.030}$	$\mathbf{0.543 \pm 0.089}$
Lotka–Volterra	1.939 ± 0.475	$\mathbf{1.000 \pm 0.000}$	67.175 ± 1.334	$\mathbf{0.127 \pm 0.033}$	0.890 ± 0.028	$\mathbf{0.450 \pm 0.078}$
Pendulum	0.128 ± 0.028	1.000 ± 0.000	1.391 ± 0.005	$\mathbf{0.109 \pm 0.038}$	$\mathbf{0.912 \pm 0.063}$	$\mathbf{0.487 \pm 0.144}$
Van der Pol	1.337 ± 0.039	0.960 ± 0.000	5.417 ± 0.148	$\mathbf{0.419 \pm 0.053}$	$\mathbf{0.953 \pm 0.014}$	$\mathbf{1.245 \pm 0.060}$

Table 46: Bayesian encoder/decoder ablation against standard formulation with point encoder/decoder (mean $\pm$ std).