DISSIPATIVE NEURAL SDES FOR ROBUST DYNAMICAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

While Neural stochastic differential equations (SDEs) provide a flexible framework for learning stochastic dynamics, their training still lacks dynamical guarantees, and in particular, they often exhibit non-physical behaviors under distribution shifts. This work aims to mitigate such failures by emphasizing the role of the Foster-Lyapunov dissipativity condition, which is commonly assumed in the current literature but is not guaranteed by the standard Neural SDE architectures. In this work, we introduce a flexible unified framework of partially dissipative Neural SDEs, which accommodates a wide class of dynamics, such as globally stable, chaotic, and integrated dynamics. We translate the dissipativity condition into sufficient forms compatible with 2 enforcement mechanisms: PINN-style regularization and structural projection. Theoretically, we show that structural projection is conditionally no-regret, revealing a stability–flexibility trade-off, while our analysis further supports the framework's pathwise approximation flexibility over a moment-bounded class of initial laws. Experimentally, we demonstrate that the methods considerably improve dynamical recovery and robustness. The practical relevance is demonstrated on real-world high-dimensional datasets in terms of forecasting and reconstruction performance under context missingness and noise corruption.

1 INTRODUCTION

1.1 BACKGROUND AND CONTRIBUTIONS

Stochastic differential equations (SDEs) provide a versatile modeling framework for systems subject to uncertainty and have been extensively applied across a wide range of disciplines, including finance, physics, and biology (Shreve et al., 2004; Gardiner, 2009; Øksendal, 2003). More recently, SDEs have also been used to analyze and stabilize deep neural networks (Liu et al., 2019). However, the design of SDEs for a particular application requires substantial domain expertise in both SDEs and the applied domain, which limits their application. Moreover, traditional parametric SDEs can also be restrictive for modeling complex, high-dimensional, and non-stationary data. Neural SDEs offer a powerful alternative in such settings by parameterizing the drift and diffusion functions using neural networks. By the universal approximation property of neural networks (Kidger & Lyons, 2020; Pinkus, 1999), Neural SDEs are therefore capable of representing a broad class of stochastic systems (Kidger et al., 2021a; Kwossek et al., 2025). They therefore, in principle, allow the dynamics to be learned directly from data without requiring demanding parametric design. This has also been facilitated by advances in backpropagation through the SDE solvers (Li et al., 2020a; Kidger et al., 2021b). Variants of Neural SDEs have also been proposed to capture jumps (Jia & Benson, 2019; Zhang et al., 2024; Gao et al., 2026).

Despite their flexibility, learning Neural SDE is challenging. Unlike deterministic dynamics learning (Chen et al., 2018; Grathwohl et al., 2018; Kidger et al., 2020; Kovachki et al., 2023; Li et al., 2020b), Neural SDEs jointly learn a drift and an additional diffusion function, a setting that has long been known to be considerably more difficult, even in the scalar case (Gobet et al., 2004). To introduce stochasticity while maintaining well-conditioned learning, many approaches have introduced fixed diffusion and learn only the drift function (Berman et al., 2024; Neklyudov et al., 2023; Petrović et al., 2026), but note that these are different settings. There are many attempts to improve the learning of Neural SDEs. A variety of training objectives have been proposed to improve the training (Kidger

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.

et al., 2021a; Tzen & Raginsky, 2019; Zhang et al., 2025; Issa et al., 2023), and more recently, there has been a simulation-free training scheme (Bartosh et al., 2025). Stable Neural SDE structures have also been proposed (Oh et al., 2024).

In this work, we identify a gap in Neural SDE training concerning the lack of dynamical guarantees. This is particularly pronounced under distribution shift, where the learned models can exhibit substantial dynamical mismatches. The role of the Foster-Lyapunov type dissipativity condition is emphasized here (Meyn & Tweedie, 1993; 2012) as a fundamental dynamical guarantee. While such conditions are routinely assumed in theoretical analyzes of neural SDEs for stochastic stability (Liu et al., 2019; Oh et al., 2024), we note that they are not automatically satisfied by standard Neural SDE parameterizations or training objectives without explicit enforcement.

The contributions of this work are mainly: (1) we introduce a unified framework of partially dissipative Neural SDEs that covers a broad class of dynamics, including globally stable, chaotic, and integrated dynamics; (2) we translate the conditions into computable forms and present practical enforcement mechanisms via sampled PINN-style regularization (Raissi et al., 2019) and structural projection generalized from the deterministic literature (Min & Azizan, 2024; Kolter & Manek, 2019), together with a scalable normal-tangential decomposition for high-dimensional systems; (3) we establish a conditional generator-level no-regret guarantee for structural projection, revealing a stability–flexibility trade-off, and provide theoretical support for the partially dissipative framework's pathwise approximation flexibility over a moment-bounded class of initial laws; (4) we empirically demonstrate improved dynamical recovery and more robust reconstruction and forecasting under context missingness and noise corruption

1.2 Related Work

Neural learning of dynamical systems is well known to lack dynamical guarantees without explicit enforcement, despite excellent in-distribution performance. For the deterministic models (Chen et al., 2018; Linot et al., 2023; Li et al., 2020b), many solutions have been proposed. In particular, many efforts have been made to utilize deterministic Lyapunov stability to provide better dynamical guarantees (Min & Azizan, 2024; Kolter & Manek, 2019; Li et al., 2022; Tang et al., 2024). For stochastic models, similar problems exist, but considerably fewer works directly address dynamical guarantees for Neural SDEs. Most existing approaches focus on parametric, semi-parametric, or specialized stability-preserving SDE constructions: Urain et al. (2020) enforces stability on a linear SDE and achieves non-linearity through a stability-preserving flow map;Djeumou et al. (2023) considers a physics-constrained semi-parametric Neural SDE in which prior dynamical structure is retained in the drift; Seifner et al. (2026) pretrains a neural foundation inference model for inferring drift and diffusion functions from trajectories generated from prescribed families of synthetic SDEs. The closest to our work is Oh et al. (2024), which proposes more stable variants of Neural SDEs, while we note that our work is complementary to them as a compatible enforcement mechanism. In contrast, our work directly tackles dynamical guarantees for Neural SDEs by imposing a generalized partial Foster–Lyapunov dissipativity condition on the learned stochastic dynamics.

2 Preliminaries

A Neural SDE is defined as:

$$dx(t) = \mu(t, x(t))dt + \sigma(t, x(t))dB(t), \tag{1}$$

where the drift $\mu(t, x(t)) : [0, \infty) \times \mathbb{R}^d \to \mathbb{R}^d$ and diffusion $\sigma(t, x(t)) : [0, \infty) \times \mathbb{R}^d \to \mathbb{R}^{d\times b}$ functions are parameterized by neural networks, and $B(t)$ is standard $b-$dimensional Brownian motion. Let $\mathcal{T} \subseteq [0, \infty)$ denote the declared time domain. Throughout, for the initial laws considered, we assume all SDEs considered, including those induced by structural projection, admit pathwise-unique, non-explosive strong solutions on every finite time horizon of interest. Well-posedness is assumed directly at the solution level, without imposing local Lipschitz or global linear-growth conditions. Additional assumptions are stated in the relevant theoretical results. The focus of this work is a common assumption made in the stochastic analysis of Neural SDEs: the Foster-Lyapunov dissipativity condition, in particular, showing robustness to input distribution shift (Liu et al., 2019; Oh et al., 2024). Throughout this paper, denote $|\cdot|$, $\|\cdot\|$, and $\|\cdot\|_F$ as the absolute value for scalars, the Euclidean norm for vectors, and the Frobenius norm for matrices. Let $c_1 > 0$, $c_2 \geq 0$, and

$V \in C^2(\mathbb{R}^d; [0, \infty))$, with $V(x) \to \infty$ as $\|x\| \to \infty$. The Neural SDE in equation 1 is dissipative if:

$$\langle \nabla_x V(x), \mu(t,x) \rangle + \frac{1}{2} \operatorname{tr}\big(\sigma(t,x)\sigma(t,x)^\top \nabla_x^2 V(x)\big) \leq -c_1 V(x) + c_2. \tag{2}$$

for every $(t, x) \in \mathcal{T} \times \mathbb{R}^d$. When V is quadratic, it becomes the widely applied one-sided dissipativity condition (Fang & Giles, 2016). Our work is essentially complementary to the prior work in showing that the dissipative condition is, in general, not fulfilled by Neural SDEs, as illustrated in Figure 1, where Neural SDEs exhibit diverging paths out of the training distribution. In contrast, the Neural SDEs with the condition enforced by our approaches show the expected robustness to the input shift, as analyzed by the prior work, and behave much closer to the ground truths. This phenomenon has been found to be robust across both different training objectives and model classes, as will be illustrated in our experiments. This robustness to far-field shift can be intuitively attributed to the uniform-in-time Lyapunov-moment control, and hence marginal confinement induced by dissipativity (Mao, 2007; Majda & Tong, 2019; Meyn & Tweedie, 2012).

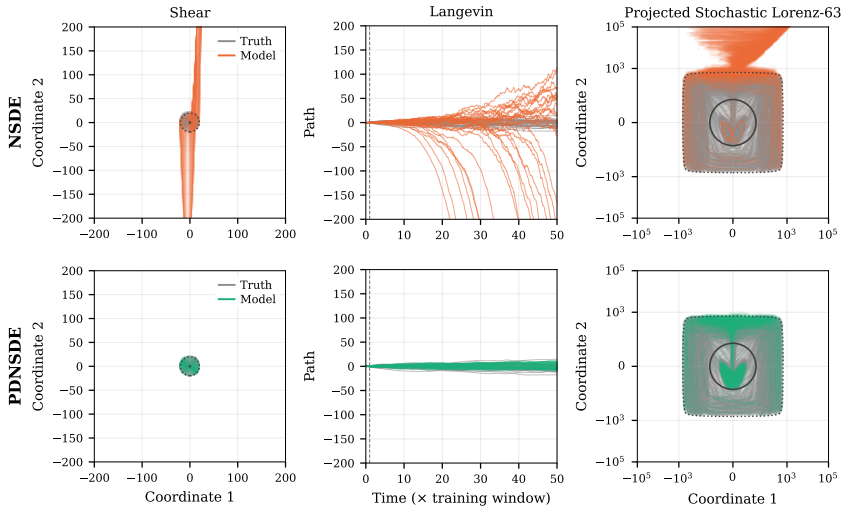

Figure 1: Out-of-distribution dynamical comparison between Neural SDEs (NSDE) and Partially Dissipative Neural SDEs (PDNSDE, ours). Solid and dotted boundaries indicate the training and extrapolation regimes, respectively.

3 Methods

3.1 Partially Dissipative Neural SDEs

Although dissipativity is a useful property, we note that requiring full-state dissipativity is often unnecessarily restrictive, as dissipativity is only required in a subset of states in many models, such as integrated models (Box et al., 2015), some nonlinear stochastic dynamics (Rajpurohit & Haddad, 2017), and chaotic dynamics, which often require dissipativity only outside a bounded set (Geurts et al., 2020; Schmallfuß, 1997). In light of this, a unified notation for the partially constrained dissipative Neural SDEs is introduced to include these features. Let $M = \operatorname{diag}(m_1, \ldots, m_d)$, with $m_i \in \{0, 1\}$, be a fixed coordinate mask, where coordinates with $m_i = 1$ are dissipative, while those with $m_i = 0$ are not. A partially dissipative Neural SDE with drift and diffusion functions (μ, σ) can then decompose into the free component $(\mu_\perp, \sigma_\perp)$ and the dissipative component $(\hat{\mu}, \hat{\sigma})$:

$$\mu_\perp(t,x) := (I - M)\mu(t,x), \qquad \hat{\mu}(t,x) := M\mu(t,x), \qquad \mu(t,x) = \mu_\perp(t,x) + \hat{\mu}(t,x). \tag{3}$$

$$\sigma_\perp(t,x) := (I - M)\sigma(t,x), \qquad \hat{\sigma}(t,x) := M\sigma(t,x), \qquad \sigma(t,x) = \sigma_\perp(t,x) + \hat{\sigma}(t,x). \tag{4}$$

We now define the partially dissipative Neural SDE as follows:

Definition 3.1 ((M, V, c_1, c_2)-partially dissipative Neural SDE)**.** Consider a Neural SDE in equation 1 with coordinate mask M and with drift and diffusion functions defined in equation 3 and equation 4.

Let $c_1 > 0$, $c_2 \geq 0$ and let $V \in C^2(\mathbb{R}^d; [0, \infty))$ satisfy $V(x) = V(Mx)$, with $V(x) \to \infty$ as $\|Mx\| \to \infty$. The Neural SDE is called (M, V, c_1, c_2)-partially dissipative on $\mathcal{T} \subseteq [0, \infty)$ if:

$$\langle \nabla_x V(x), \hat{\mu}(t,x) \rangle + \frac{1}{2} \operatorname{tr}\big(\hat{\sigma}(t,x)\hat{\sigma}(t,x)^\top \nabla_x^2 V(x)\big) \leq -c_1 V(x) + c_2. \tag{5}$$

for every $(t, x) \in \mathcal{T} \times \mathbb{R}^d$.

However, we note that this joint condition does not specify the individual allocation of the drift and diffusion contributions, and so the same dissipativity condition can refer to substantially different dynamics (see Appendix A.1 for an example, where the condition with a quadratic Lyapunov function can accommodate both globally dissipative dynamics and chaotic dynamics). To make this allocation explicit, we first state the following growth assumption on the selected diffusion contribution. (see Appendix A.2 for discussions of how to fulfill this condition, and we also enforce it in experiments).

Assumption 3.2 (Affine diffusion contribution)**.** There exist finite constants $\rho, C_\sigma \geq 0$ such that

$$\frac{1}{2} \operatorname{tr}\big(\hat{\sigma}(t,x)\hat{\sigma}(t,x)^\top \nabla_x^2 V(x)\big) \leq \rho V(x) + C_\sigma \tag{6}$$

for every $(t, x) \in \mathcal{T} \times \mathbb{R}^d$.

Assumption 3.2 states the diffusion contribution and makes the sufficient condition on the drift explicit by subtracting it from the joint condition in equation 5. The specific subtraction term $q : \mathcal{T} \times \mathbb{R}^d \to \mathbb{R}$ may lie in the range:

$$\frac{1}{2} \operatorname{tr}\big(\hat{\sigma}(t,x)\hat{\sigma}(t,x)^\top \nabla_x^2 V(x)\big) \leq q(t,x) \leq \rho V(x) + C_\sigma. \tag{7}$$

Any q in this admissible range leads to the sufficient condition on the drift for partial dissipativity:

$$\langle \nabla_x V(x), \hat{\mu}(t,x) \rangle \leq -c_1 V(x) + c_2 - q(t,x). \tag{8}$$

This is also the form of condition we enforce later on the models. Note that this separation includes a broad class of dynamics, including but not limited to the stochastic extensions of deterministic dissipative systems. To show this, we present the following proposition:

Proposition 3.3 (Dynamical Inclusion Relationship)**.** *For the fixed M, V, and $\mathcal{T}$ in Definition 3.1, denote the selected diffusion contribution as:*

$$\Gamma_V^{\hat{\sigma}}(t,x) := \frac{1}{2} \operatorname{tr}\big(\hat{\sigma}(t,x)\hat{\sigma}(t,x)^\top \nabla_x^2 V(x)\big). \tag{9}$$

Using the parameter ranges in Definition 3.1 and Assumption 3.2, and taking q as in equation 7, where $\alpha > \rho$ and $\beta \geq 0$, the spaces of the dynamics for the stochastic extension of the deterministic dissipative dynamics D_{drift}, the dynamics under Assumption 3.2 and equation 8 D_{comp}, and those under the joint condition in equation 5 D_{FL} are:

$$\mathcal{D}_{\text{drift}} := \big\{(\hat{\mu}, \hat{\sigma}) : \exists \rho, C_\sigma, \alpha, \beta : \quad \Gamma_V^{\hat{\sigma}} \leq \rho V + C_\sigma, \quad \langle \nabla_x V, \hat{\mu} \rangle \leq -\alpha V + \beta \big\}, \tag{10}$$

$$\mathcal{D}_{\text{comp}} := \big\{(\hat{\mu}, \hat{\sigma}) : \exists \rho, C_\sigma, c_1, c_2, q : \quad \Gamma_V^{\hat{\sigma}} \leq q \leq \rho V + C_\sigma, \quad \langle \nabla_x V, \hat{\mu} \rangle \leq -c_1 V + c_2 - q \big\}, \tag{11}$$

$$\mathcal{D}_{\text{FL}} := \big\{(\hat{\mu}, \hat{\sigma}) : \exists c_1, c_2 : \quad \langle \nabla_x V, \hat{\mu} \rangle + \Gamma_V^{\hat{\sigma}} \leq -c_1 V + c_2 \big\}. \tag{12}$$

Then

$$\mathcal{D}_{\text{drift}} \subseteq \mathcal{D}_{\text{comp}} \subseteq \mathcal{D}_{\text{FL}}. \tag{13}$$

The proof is given in Appendix A.3.

3.2 Enforcing Partial Dissipativity

In this section, we describe how to enforce the partial dissipativity condition on Neural SDEs. Under Assumption 3.2, it remains to enforce the sufficient drift condition in equation 8. To begin with, the condition is reparameterized into a more computable form. Let μ_θ^{free} and $\hat{\sigma}_\theta(t,x)$ denote the selected drift before enforcement and the selected diffusion. Define the residual:

$$r_\theta^q(t,x) := \big\langle \nabla_x V(x), \mu_\theta^{\text{free}}(t,x) \big\rangle + c_1 V(x) - c_2 + q_\theta(t,x). \tag{14}$$

The sufficient drift condition in equation 8 is then equivalent to $r_\theta^q(t,x) \leq 0$. For the exact enforcement, it is then required to specify a form of q_θ. Learning q_θ would introduce an additional, non-trivial constrained function-learning problem. Here, the upper and lower bound cases are considered:

$$r_\theta^{\text{upper}}(t,x) = \left\langle \nabla_x V(x), \mu_\theta^{\text{free}}(t,x) \right\rangle + \alpha V(x) - \beta, \quad \alpha > \rho, \quad \beta \geq 0, \tag{15}$$

$$r_\theta^{\text{lower}}(t,x) = \left\langle \nabla_x V(x), \mu_\theta^{\text{free}}(t,x) \right\rangle + c_1 V(x) - c_2 + \Gamma_V^{\hat{\sigma}_\theta}(t,x), \quad c_1 > 0, \quad c_2 \geq 0. \tag{16}$$

The parameters in the upper bound case are reparameterized to $\alpha = c_1 + \rho$ and $\beta = c_2 - C_\sigma$ to remove the redundant degrees of freedom for learning. However, it is noted that for the enforcement of the partial dissipativity condition, the drift enforcement alone may be insufficient due to the presence of the singular set, where the drift is perpendicular to the gradient of the Lyapunov function and has 0 contribution:

$$\mathcal{S}_V := \left\{ x \in \mathbb{R}^d : M \nabla_x V(x) = 0 \right\}. \tag{17}$$

Satisfying partial dissipativity on $\mathcal{S}_V$ requires a corresponding lower bound on c_2. This is straightforward to address in the upper bound compensation, but for the lower bound case, an estimate of the required floor on c_2 is needed. We estimate it at the evaluated singular state-time points, which does not certify feasibility between grid points or beyond the grid's time range. Further details are given in Appendix A.6. Provided enforcement on the singular set, both enforcements are sufficient for partial dissipativity but introduce a stability-flexibility trade-off: although the lower bound has maximal flexibility, it cannot provide guarantee for a globally dissipative drift; the upper bound does so by setting $\beta = 0$ (see Appendix A.4 for more details). The enforcement is now about eliminating the positive residuals while leaving the feasible negative residuals unpenalized. Here, we consider 2 approaches: a Physics-Informed Neural Networks (PINN)-style regularization (Raissi et al., 2019) and hard structural enforcement inspired by the deterministic literature (Kolter & Manek, 2019).

PINN-style regularization. The regularized approach uses the free drift $\mu_\theta = \mu_\theta^{\text{free}}$ directly for the Neural SDE, and the enforcement is by regularized training. Here, the regularized term is defined as expected positive residual at sampled state-time points drawn from a specified distribution ν_{sample}:

$$\mathcal{L}_{\text{diss}}(\theta) = \mathbb{E}_{(t,x)\sim\nu_{\text{sample}}} \left[[r_\theta^q(t,x)]_+ \right], \qquad [z]_+ := \max\{z, 0\}. \tag{18}$$

It is trained jointly with the data objective:

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}_{\text{data}}(\theta) + \lambda_{\text{diss}} \mathcal{L}_{\text{diss}}(\theta). \tag{19}$$

The weight λ_{diss} may be fixed or adaptively updated. For adaptive weighting, we use gradient-norm balancing inspired by Wang et al. (2021; 2023), and the exact scheme is given in Appendix A.5. Because the additional parameters in equation 15 and equation 16 enter the training objective, they are considered hyperparameters and tuned by grid search.

Structural enforcement. The structurally constrained approach projects the free drift onto the feasible set of equation 8 in the Euclidean norm. Provided that the singular-set feasibility condition discussed above is satisfied, we define:

$$\mu_\theta^{\text{str}}(t,x) := \begin{cases} \mu_\theta^{\text{free}}(t,x) - \dfrac{[r_\theta^q(t,x)]_+}{\|M \nabla_x V(x)\|^2} M \nabla_x V(x), & x \notin \mathcal{S}_V, \\ \mu_\theta^{\text{free}}(t,x), & x \in \mathcal{S}_V. \end{cases} \tag{20}$$

For $x \notin \mathcal{S}_V$, substitution into equation 14 shows that the condition is fulfilled:

$$\begin{aligned} &\left\langle \nabla_x V(x), M \mu_\theta^{\text{str}}(t,x) \right\rangle + c_1 V(x) - c_2 + q_\theta(t,x) \\ &\quad = r_\theta^q(t,x) - [r_\theta^q(t,x)]_+ = \min\{r_\theta^q(t,x), 0\} \leq 0. \end{aligned} \tag{21}$$

For $x \in \mathcal{S}_V$, the drift contribution vanishes, and the singular-set feasibility condition directly gives the same inequality. Apart from exact enforcement, this structural implementation has an additional practical benefit: the parameters in the residuals in equation 15 and equation 16 can be parameterized within the model and trained jointly, which avoids the cost of additional hyperparameter searches. The structural projection admits a conditional no-regret guarantee while also exposing a stability–flexibility trade-off. To formalize this property, we consider the local discrepancy between the infinitesimal generators of the SDEs. The generator is defined as:

$$\mathcal{L}_t f(x) = \langle \mu(t,x), \nabla_x f(x) \rangle + \frac{1}{2} \left(\sigma(t,x) \sigma(t,x)^\top \right) : \nabla_x^2 f(x), \tag{22}$$

where $A : B = \mathrm{tr}(A^\top B)$ denotes the Frobenius inner product. Because the generator is characterized by the local 2-jet $(\nabla_x f(x), \nabla_x^2 f(x))$, we define the test space for measuring the discrepancy using the following local 2-jet seminorm:

$$\|f\|^2_{\mathcal{J}_{\eta;x}} := \|\nabla_x f(x)\|^2 + \eta \|\nabla_x^2 f(x)\|_F^2, \qquad \eta > 0. \tag{23}$$

Theorem 3.4 (Conditional Generator-Level No-Regret of Structural Projection)**.** *Fix $(t, x) \in \mathcal{T} \times \mathbb{R}^d$. Let $\mathcal{H}_\theta(t, x)$ denote the feasible drift set specified by equation 8, using the compensation $q_\theta(t, x)$ employed in equation 20. For $x \notin \mathcal{S}_V$, this set is a half-space. Suppose that the data-generating drift lies in this feasible set: $\mu^\star(t, x) \in \mathcal{H}_\theta(t, x)$. Let $\mathcal{L}_t^\star$, $\mathcal{L}_{\theta,t}^{\mathrm{free}}$, and $\mathcal{L}_{\theta,t}^{\mathrm{str}}$ denote the generators associated with the data-generating, free, and structurally constrained models, where the free and structurally constrained models share the same diffusion coefficient and the structurally constrained drift is obtained from equation 20. Define their local generator discrepancies by:*

$$E^j_{\mathrm{gen}}(t, x) := \sup_{\substack{f \in C^2 \\ \|f\|_{\mathcal{J}_{\eta;x}} \le 1}} \left| \left(\mathcal{L}^j_{\theta,t} - \mathcal{L}^\star_t \right) f(x) \right|, \qquad j \in \{\mathrm{free}, \mathrm{str}\}. \tag{24}$$

Then the structural projection satisfies

$$E^{\mathrm{str}}_{\mathrm{gen}}(t, x)^2 + \left\| \mu^{\mathrm{str}}_\theta(t, x) - \mu^{\mathrm{free}}_\theta(t, x) \right\|^2 \le E^{\mathrm{free}}_{\mathrm{gen}}(t, x)^2. \tag{25}$$

The proof is given in Appendix A.7. The condition $\mu^\star(t, x) \in \mathcal{H}_\theta(t, x)$ highlights the trade-off: an overly stable constraint may exclude the true dynamics and introduce approximation bias through the projection.

Finally, we provide theoretical support for the path-level approximation capability of the partially dissipative framework. To begin with, we require a controlled growth of the unselected states:

Assumption 3.5 (Controlled growth of the unselected states)**.** Considering a partially dissipative Neural SDE in Definition 3.1, suppose there exists $U \in C^2(\mathbb{R}^d; [0, \infty))$ such that $U(x) = U((I - M)x)$ and $U(x) \to \infty$ as $\|(I - M)x\| \to \infty$. We assume that there exist constants $a_0, a_1, a_2 \ge 0$ such that

$$\mathcal{L}_t U(x) \le a_0 + a_1 V(x) + a_2 U(x), \qquad (t, x) \in [0, T] \times \mathbb{R}^d. \tag{26}$$

This does not impose dissipativity on the unselected states but only requires their auxiliary energy to have at most linear growth. In the integrated models considered in this work, this follows directly from the integration-type interaction with the selected states. The fully dissipative models are included straightforwardly. With partial dissipativity, this yields finite-horizon compact containment of the complete state and supports the following uniform path-approximation flexibility for our framework:

Theorem 3.6 (Uniform Path Approximation over Moment-Bounded Initial Laws)**.** *For $T > 0$, let $X^\star$ and $(X^n)_{n \ge 1}$ denote the data-generating SDE and a sequence of partially dissipative Neural SDEs, respectively, on the common state space $\mathbb{R}^d$, with diffusion factors in $\mathbb{R}^{d \times m}$. Suppose every model satisfies the solution-level well-posedness assumption, the joint partial dissipativity condition in equation 5 for the same M, V, c_1, c_2, and Assumption 3.5 with the same U and common constants a_0, a_1, a_2. Define the auxiliary full-state localization function $W(x) := V(x) + U(x)$. For $0 \le m_0 < \infty$, define:*

$$\mathcal{P}_{m_0}(W) := \left\{ \nu : \int_{\mathbb{R}^d} W(x)\, \nu(dx) \le m_0 \right\}. \tag{27}$$

For $R < \infty$, let $K_R := \{x \in \mathbb{R}^d : W(x) \le R\}$ and $\mathcal{C}_R := [0, T] \times K_R$. Assume that the data-generating coefficients $(\mu_\star, \sigma_\star)$ are locally Lipschitz in the complete state, uniformly in $t \in [0, T]$. Suppose further that, for every fixed $R < \infty$, the Neural SDE sequence satisfies the following compact-local coefficient approximation property:

$$\sup_{(t,x) \in \mathcal{C}_R} \|\mu_n(t, x) - \mu_\star(t, x)\| \longrightarrow 0, \qquad \sup_{(t,x) \in \mathcal{C}_R} \|\sigma_n(t, x) - \sigma_\star(t, x)\|_F \longrightarrow 0. \tag{28}$$

Then the data-generating and learned processes admit synchronous couplings such that

$$\sup_{\nu \in \mathcal{P}_{m_0}(W)} \mathbb{P}_\nu \left(\sup_{0 \le t \le T} \|X^n_t - X^\star_t\| > \epsilon \right) \longrightarrow 0, \qquad \forall \epsilon > 0. \tag{29}$$

The proof is given in Appendix A.8. For continuous coefficients, compact-local approximation in equation 28 is motivated by standard universal approximation results for neural networks and Neural SDE parameterizations (Kidger & Lyons, 2020; Kwossek et al., 2025).

3.3 Scaling to High Dimensions

Direct enforcement of dissipativity can be effective for systems of low or moderate dimension, but it becomes increasingly challenging as dimensionality increases. For PINN-style regularization in equation 18, it shares the well-known scalability problems with PINNs (Hu et al., 2024; Khanra et al., 2026): evaluating the physical constraint requires evaluation over the input domain, so maintaining a fixed expected resolution for the grid requires an exponentially growing number of points. While structural projection bypasses the sampling, enforcing the joint condition presents another scaling difficulty: to satisfy the same dissipativity condition, some dimensions can be excessively dissipative while others can be much less damped, and such ambiguity increases with dimensions. To mitigate the scalability problem, we propose to further decompose the selected drift $\hat{\mu}_\theta$ into the normal and tangential components, $\mu_{\mathrm{diss},\theta}$ and $\mu_{\tan,\theta}$, relative to the Lyapunov gradient direction:

$$\mu_\theta(t,x) = \mu_{\perp,\theta}(t,x) + \hat{\mu}_\theta(t,x) = \mu_{\perp,\theta}(t,x) + \mu_{\mathrm{diss},\theta}(t,x) + \mu_{\tan,\theta}(t,x), \tag{30}$$

where the normal component carries the entire dissipativity requirement. Let $\mathcal{I}_M := \{i : m_i = 1\}$ and we split the joint condition to enforce marginally on each selected coordinate by:

$$V(x) = \sum_{i\in\mathcal{I}_M} V_i(x_i), \qquad q_\theta(t,x) = \sum_{i\in\mathcal{I}_M} q_{\theta,i}(t,x_i), \qquad c_2 = \sum_{i\in\mathcal{I}_M} c_{2,i}. \tag{31}$$

This changes the fixed-resolution sampling complexity from exponential to linear, and can provide uniform dissipativity control over the dimensions. The cross-dimensional interactions are carried by the tangential component, which is orthogonal to the Lyapunov gradient direction and makes no contribution to the dissipativity:

$$\langle M\nabla_x V(x), \mu_{\tan,\theta}(t,x)\rangle = 0. \tag{32}$$

The detailed tangential enforcement constructions are provided in Appendix A.9.

4 Experiments

Our experiments address two questions. First, we study whether enforcing partial dissipativity improves dynamical recovery and robustness under controlled extrapolation beyond the training regime, as motivated by the existing analysis of stable Neural SDEs (Oh et al., 2024). Second, we study whether the resulting models provide practical benefits for forecasting and reconstruction on higher-dimensional real-world data under observation noise and missing context. Considering the recent contributions to Neural SDE training objectives (Zhang et al., 2025; Bartosh et al., 2025; Issa et al., 2023), we treat them together with several standard training objectives as hyperparameters to be selected through the validation set. To isolate the contribution of enforcing the condition, we introduce for comparison both the original Neural SDE (Liu et al., 2019) , a homogeneous variant, and the Langevin-type variant of the Stable Neural SDE (Oh et al., 2024) (the remaining variants impose additional structural restrictions that do not match the dynamical recovery tasks. See Appendix B.1 for more details.). Furthermore, for each partially dissipative Neural SDE, we introduce a counterpart with exactly the same structure but without the partial dissipativity enforcement, allowing the effect of the enforcement to be compared under matched model structures. See Appendix B.1 for details of the objectives, Neural SDE architectures considered, and the partially dissipative Neural SDE architectures. On the real-world datasets, additional baselines are introduced. We also report the sensitivity to the numerical solver in Appendix B.4, where the results are found to be largely insensitive to the solver choice.

4.1 Recovery of Partially Dissipative Stochastic Dynamics

To demonstrate the improved dynamical recovery, we consider the shear model, Langevin dynamics, and stochastic Lorenz-63 (Lemons & Gythiel, 1997; Lorenz & Martin, 1995; Van K., 1992). To evaluate distributional recovery, we use the unbiased squared maximum mean discrepancy (MMD^2) with a median-heuristic RBF kernel (Gretton et al., 2012) for both the validation and test. To study the recovery quality, the selected models are evaluated under distribution shift on the held-out test set. For the shear and stochastic Lorenz–63 models, distribution shift is introduced through far-field initializations outside the training distribution. For the Langevin model, it is introduced through

rollout horizons longer than those used during training. See Appendix B.2 for more details on the datasets, as well as the details of the partial dissipativity enforcement and verification.

Figure 2 reports the summarized distributional recovery under increasing shifts. NSDE refers to the validation-selected Neural SDE baselines without dissipativity enforcement using in-distribution MMD^2. PDNSDE corresponds to the partial-dissipativity enforced models and is reported similarly, additionally with the enforcement conditions and mechanisms selected through the validation set, for a compact comparison to study the effect of the enforcement. The partially dissipative models consistently achieve lower MMD^2 under these shifts, demonstrating improved robustness and dynamical recovery. The corresponding trajectories under the largest distribution shifts, together with the trajectories from the data-generating SDEs, are shown in Figure 1. Notably, the far-field initialization experiments match the form of input shift considered in Oh et al. (2024), providing empirical support for the role of dissipativity discussed in their analysis.

Detailed numerical results and ablations for both the Neural SDE variants and the enforcement mechanisms are reported in Appendix B.2.4. No single configuration uniformly dominates across datasets and evaluation regimes, supporting their treatment as modeling choices selected using the validation set. We therefore follow the same validation-based selection principle in the later applications. For the regularized models, the collocation domains and independent residual diagnostics are reported in Appendix B.4. For the shear and Lorenz–63 far-field experiments, the evaluated initialization shells lie within the broader regularization collocation domains. These experiments , therefore, test extrapolation beyond the empirical training distribution rather than beyond the support used for dissipativity regularization.

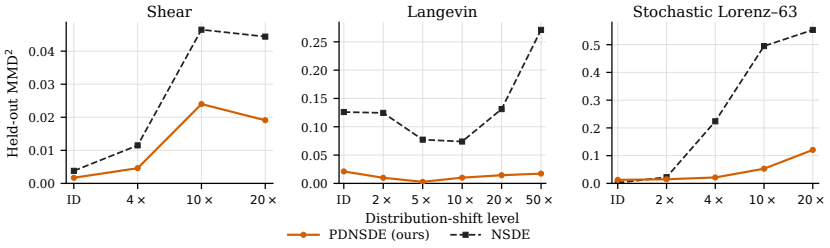

Figure 2: Distributional recovery under increasing far-field (Shear and Lorenz–63) and long-horizon (Langevin) shifts, measured by test mean MMD^2 (lower is better, full results in Appendix B.2.4)

4.2 High-Dimensional Real-World Applications

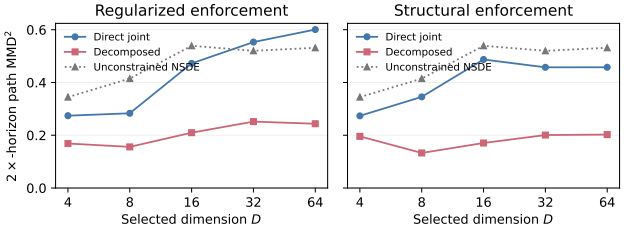

Figure 3: Double training horizon test mean MMD^2 across dimensions under different enforcement settings. Full numerical results are reported in Appendix B.3.4

In this section, we study the impact of enforcing the partial dissipativity condition in higher dimensional real-world settings. We consider multi-agent crowd trajectories (A. et al., 2014; Ivanovic et al., 2023), S&P 500 Stocks (Larxel, 2026), and MuJoCo Hopper (Todorov et al., 2012; Oh et al., 2024). See Appendix B.3.1 for the dataset specifications, protocols, and partial dissipativity enforcement and verification. The regularization diagnostics are also reported in Appendix B.3. To begin with, we use a high-dimensional Langevin generalization as a controlled diagnostic of the scaling behavior of

direct joint and decomposed enforcement. This experiment is not intended as a general benchmark of high-dimensional Neural SDE learning; rather, it illustrates the practical difficulty motivating the decomposed construction in Section 3.3. See Appendix B.3.1 for a detailed definition of the model. Under matched sampling budgets, Figure 3 shows that the direct joint formulation deteriorates with dimension, particularly for regularized enforcement, whereas the decomposed formulation remains substantially more stable.

Following the missing-data robustness evaluation of Oh et al. (2024), we assess forecasting performance under increasing context missingness and additionally evaluate reconstruction under increasing Gaussian observation noise. We use NRMSE and NCRPS (Gneiting & Raftery, 2007) as the evaluation metrics. See Appendix B.3.3 for details of the evaluation protocols and additional baselines. Again, NSDE here refers to the validation-selected best Neural SDE configuration, as stated before. Figure 4 summarizes the results. PDNSDE remains competitive under clean or lightly corrupted observations and generally degrades more gracefully as context missingness or observation noise increases. Under moderate-to-severe observation degradation, it achieves the best or tied-for-best performance across the evaluated datasets and tasks. These results demonstrate practical robustness benefits of partial dissipativity for forecasting and reconstruction under incomplete and noisy observations. Neural CDE is not used for the forecasting task due to the absence of controlled paths.

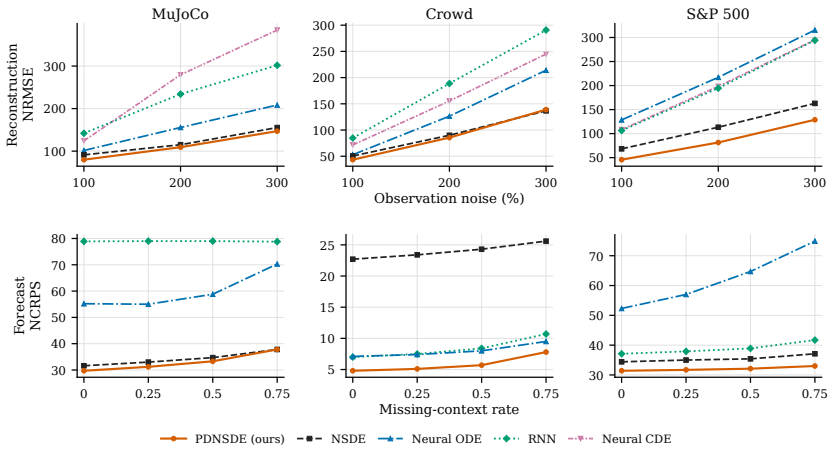

Figure 4: Mean NRMSE and NCRPS across datasets under increasing Gaussian observation noise (reconstruction) and context missingness (forecasting). Full numerical results are reported in Appendix B.3.4.

5 Conclusions, Limitations, and Future Work

We introduce a unified framework for enforcing partial dissipativity in Neural SDEs, demonstrating improved dynamical recovery under controlled extrapolation and practical benefits for forecasting and reconstruction under missing and noisy observations.

Several limitations motivate future work. First, specifying the Lyapunov function and selected state subspace requires prior structural knowledge. In real-world settings where a more specific Lyapunov structure is unavailable, we use a quadratic Lyapunov function and a globally dissipative condition on the selected coordinates as a modeling prior. Learning or adapting these structures from data would broaden the applicability of the framework. Second, our partial-state applications primarily consider integrated systems, while the decomposed high-dimensional construction introduces additional structural assumptions for scalability. Generalizations beyond these structures are therefore important directions. Third, sampled regularization provides empirical constraint control rather than a global certificate, while the lower-bound singular-set floor is evaluated on a finite time grid and does not certify feasibility at unevaluated singular state–time points. Finally, overly stable constraints may exclude the true dynamics and introduce approximation bias, reflecting the stability–flexibility trade-off in Theorem 3.4.

AI USE STATEMENT

Generative AI tools, primarily ChatGPT and Claude, were used to assist in the development of this work. For paper writing, they were used to polish the language and improve the presentation of the manuscript. For retrieval and discovery, they were used to assist in searching for relevant literature and clarifying the positioning of this work relative to prior studies. For research ideation and execution, AI tools were also used to assist with software implementation and debugging. The resulting code and experimental outputs were inspected and validated by the authors. For drafting the paper, they were used extensively to help organize the presentation and the experimental details in the appendices, with the aim of improving clarity and transparency. For mathematical claims, AI tools were used extensively to check derivations, proposing refinements and steps, identify and check relevant assumptions, and clarify the appropriate scope of theoretical statements. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We have provided detailed model architectures, training objectives, optimization settings, dataset configuration and preprocessing, validation and evaluation protocols, and complete numerical results in Appendix B. All experiments are repeated over three independently trained seeds unless otherwise stated. The source code and scripts required to reproduce the experiments will be released publicly upon acceptance.

REFERENCES

Alexandre A., Vignesh R., and Li F. Socially-aware Large-scale Crowd Forecasting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2203–2210, 2014.

G. Bartosh, D. Vetrov, and C. A. Naesseth. SDE Matching: Scalable and Simulation-free Training of Latent Stochastic Differential Equations. *arXiv preprint arXiv:2502.02472*, 2025.

J. Berman, T. Blickhan, and B. Peherstorfer. Parametric Model Reduction of Mean-Field and Stochastic Systems via Higher-Order Action Matching. *Advances in neural information processing systems*, 37:56588–56618, 2024.

G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 2015.

R. TQ Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural Ordinary Differential Equations. *Advances in neural information processing systems*, 31, 2018.

F. Djeumou, C. Neary, and U. Topcu. How to Learn and Generalize from Three Minutes of Data: Physics-constrained and Uncertainty-aware Neural Stochastic Differential Equations. *arXiv preprint arXiv:2306.06335*, 2023.

G. Evensen. Sequential Data Assimilation with a Nonlinear Quasi-Geostrophic Model using Monte Carlo Methods to Forecast Error Statistics. *Journal of Geophysical Research: Oceans*, 99(C5): 10143–10162, 1994.

W. Fang and M. B. Giles. Adaptive Euler–Maruyama Method for SDEs with Non-globally Lipschitz Drift. In *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pp. 217–234. Springer, 2016.

Y. Gao, Q. Yan, Y. Leng, and R. Liao. Neural MJD: Neural Non-stationary Merton Jump Diffusion for Time Series Prediction. *Advances in Neural Information Processing Systems*, 38:131580–131614, 2026.

C. Gardiner. *Stochastic Methods*, volume 4. Springer Berlin Heidelberg, 2009.

B. J. Geurts, D. D. Holm, and E. Luesink. Lyapunov Exponents of Two Stochastic Lorenz 63 Systems. *Journal of statistical physics*, 179(5):1343–1365, 2020.

T. Gneiting and A. E. Raftery. Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

E. Gobet, M. Hoffmann, and M. Reiß. Nonparametric Estimation of Scalar Diffusions based on Low Frequency Data. 2004.

W. Grathwohl, R. TQ Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud. Ffjord: Free-form Continuous Dynamics for Scalable Reversible Generative Models. *arXiv preprint arXiv:1810.01367*, 2018.

A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A Kernel Two-Sample Test. *The journal of machine learning research*, 13(1):723–773, 2012.

P. L. Houtekamer and H. L. Mitchell. Data Assimilation using an Ensemble Kalman Filter Technique. *Monthly weather review*, 126(3):796–811, 1998.

Z. Hu, K. Shukla, G. E. Karniadakis, and K. Kawaguchi. Tackling the Curse of Dimensionality with Physics-informed Neural Networks. *Neural Networks*, 176:106369, 2024.

Z. Issa, B. Horvath, M. Lemercier, and C. Salvi. Non-adversarial Training of neural SDs with Signature Kernel Scores. *Advances in Neural Information Processing Systems*, 36:11102–11126, 2023.

B. Ivanovic, G. Song, I. Gilitschenski, and M. Pavone. trajdata: A Unified Interface to Multiple Human Trajectory Datasets. *Advances in Neural Information Processing Systems*, 36:27582–27593, 2023.

J. Jia and A. R. Benson. Neural Jump Stochastic Differential Equations. *Advances in neural information processing systems*, 32, 2019.

S. Khanra, V. K. Kukreja, and I. Bala. Physics-informed Neural Networks for Differential Equation Solutions: A Comprehensive Review. *Neurocomputing*, pp. 133317, 2026.

P. Kidger and T. Lyons. Universal Approximation with Deep Narrow Networks. In *Conference on learning theory*, pp. 2306–2327. PMLR, 2020.

P. Kidger, J. Morrill, J. Foster, and T. Lyons. Neural Controlled Differential Equations for Irregular Time Series. *Advances in neural information processing systems*, 33:6696–6707, 2020.

P. Kidger, J. Foster, X. Li, and T. J. Lyons. Neural SDEs as Infinite-dimensional GANs. In *International conference on machine learning*, pp. 5453–5463. PMLR, 2021a.

P. Kidger, J. Foster, X. C. Li, and T. Lyons. Efficient and Accurate Gradients for Neural SDEs. *Advances in Neural Information Processing Systems*, 34:18747–18761, 2021b.

J. Z. Kolter and G. Manek. Learning Stable Deep Dynamics Models. *Advances in neural information processing systems*, 32, 2019.

N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural Operator: Learning Maps between Function Spaces with Applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

A. P. Kwossek, D. J. Prömel, and J. Teichmann. Universal Approximation Property of Neural Stochastic Differential Equations. *arXiv preprint arXiv:2503.16696*, 2025.

Larxel. S&P 500 Stocks (Daily Updated). Kaggle dataset, 2026. URL `https://www.kaggle.com/datasets/andrewmvd/sp-500-stocks`. Version accessed 15 May 2026; CC0: Public Domain.

D. S. Lemons and A. Gythiel. Paul Langevin's 1908 Paper"On the Theory of Brownian Motion"["Sur la théorie du mouvement brownien,"C Acad. Sci.(Paris) 146, 530-533 (1908)]. *American Journal of Physics*, 65(11):1079–1081, 1997.

X. Li, T. Leonard Wong, R. T. Q. Chen, and D. Duvenaud. Scalable Gradients for Stochastic Differential Equations. In *International Conference on Artificial Intelligence and Statistics*, pp. 3870–3882. PMLR, 2020a.

Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, and A. Stuart, A.and Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations. *arXiv preprint arXiv:2010.08895*, 2020b.

Z. Li, M. Liu-Schiaffini, N. Kovachki, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar. Learning Dissipative Dynamics in Chaotic Systems, 2022. *URL https://arxiv.org/abs/2106.06898*, 2022.

A. J. Linot, J. W. Burby, Q. Tang, P. Balaprakash, M. D. Graham, and R. Maulik. Stabilized Neural Ordinary Differential Equations for Long-time Forecasting of Dynamical Systems. *Journal of Computational Physics*, 474:111838, 2023.

X. Liu, T. Xiao, S. Si, Q. Cao, S. Kumar, and C. Hsieh. Neural SDE: Stabilizing Neural ODE Networks with Stochastic Noise. *arXiv preprint arXiv:1906.02355*, 2019.

E. N Lorenz and P. Martin. The Essence of Chaos, 1995.

A. J. Majda and X. T. Tong. Simple Nonlinear Models with Rigorous Extreme Events and Heavy Tails. *Nonlinearity*, 32(5):1641, 2019.

Xuerong Mao. *Stochastic differential equations and applications*. Elsevier, 2007.

S. P. Meyn and R. L. Tweedie. Stability of Markovian Processes III: Foster–Lyapunov Criteria for Continuous-Time Processes. *Advances in Applied Probability*, 25(3):518–548, 1993.

S. P. Meyn and R. L. Tweedie. *Markov Chains and Stochastic Stability*. Springer Science & Business Media, 2012.

Y. Min and N. Azizan. Hardnet: Hard-Constrained Neural Networks with Universal Approximation Guarantees. *arXiv preprint arXiv:2410.10807*, 2024.

K. Neklyudov, R. Brekelmans, D. Severo, and A. Makhzani. Action Matching: Learning Stochastic Dynamics from Samples. In *International conference on machine learning*, pp. 25858–25889. PMLR, 2023.

Y. Oh, D. Lim, and S. Kim. Stable Neural Stochastic Differential Equations in Analyzing Irregular Time Series Data. *arXiv preprint arXiv:2402.14989*, 2024.

B. Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Universitext. Springer Berlin, Heidelberg, 6 edition, 2003. ISBN 978-3-540-04758-2. doi: 10.1007/978-3-642-14394-6.

K. Petrović, L. Atanackovic, V. Moro, K. Kapusniak, I. I. Ceylan, M. Bronstein, J. Bose, and A. Tong. Curly Flow Matching for Learning Non-Gradient Field Dynamics. *Advances in Neural Information Processing Systems*, 38:121544–121573, 2026.

A. Pinkus. Approximation Theory of the MLP Model in Neural Networks. *Acta numerica*, 8:143–195, 1999.

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, 2019.

T. Rajpurohit and W. M. Haddad. Partial-State Stabilization and Optimal Feedback Control for Stochastic Dynamical Systems. *Journal of Dynamic Systems, Measurement, and Control*, 139(9): 091001, 2017.

B. Schmallfuß. The Random Attractor of the Stochastic Lorenz System. *Zeitschrift für angewandte Mathematik und Physik ZAMP*, 48(6):951–975, 1997.

P. Seifner, K. Cvejoski, D. Berghaus, C. A. Ojeda Marin, and R. J. Sánchez. In-context Learning of Stochastic Differential Equations with Foundation Inference Models. *Advances in Neural Information Processing Systems*, 38:152636–152676, 2026.

S. E. Shreve et al. *Stochastic Calculus for Finance II: Continuous-time Models*, volume 11. Springer, 2004.

S. Tang, T. Sapsis, and N. Azizan. Learning Dissipative Chaotic Dynamics with Boundedness Guarantees. *arXiv preprint arXiv:2410.00976*, 2024.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A Physics Engine for Model-based Control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

B. Tzen and M. Raginsky. Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit. *arXiv preprint arXiv:1905.09883*, 2019.

J. Urain, M. Ginesi, D. Tateo, and J. Peters. Imitationflow: Learning Deep Stable Stochastic Dynamic Systems by Normalizing Flows. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5231–5237. IEEE, 2020.

N. G. Van K. *Stochastic Processes in Physics and Chemistry*, volume 1. Elsevier, 1992.

S. Wang, Y. Teng, and P. Perdikaris. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

S. Wang, S. Sankaran, H. Wang, and P. Perdikaris. An Expert's Guide to Training Physics-Informed Neural Networks. *arXiv preprint arXiv:2308.08468*, 2023.

J. Zhang, J.and Viktorov, D. Jung, and E. Pitler. Efficient Training of Neural Stochastic Differential Equations by Matching Finite Dimensional Distributions. In *International Conference on Learning Representations*, volume 2025, pp. 36730–36742, 2025.

S. Zhang, C. Zhou, Y. A. Liu, P. Zhang, X. Lin, and Z. Ma. Neural Jump-diffusion Temporal Point Processes. In *Forty-first International Conference on Machine Learning*, 2024.

A Method Appendix

A.1 Joint-Condition Ambiguity Example for Globally Inward and Chaos-Compatible Drifts

Let $r = Mx$ and $V(x) = \|r\|^2$. Since M is symmetric and idempotent, $\nabla_x V(x) = 2r$, $\nabla_x^2 V(x) = 2M$, and $M\hat{\sigma} = \hat{\sigma}$. The selected diffusion contribution is therefore:

$$\frac{1}{2}\operatorname{tr}\big(\hat{\sigma}\hat{\sigma}^\top \nabla_x^2 V\big) = \|\hat{\sigma}\|_F^2, \tag{33}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, and the joint Foster-Lyapunov condition becomes:

$$2\,\langle r, \hat{\mu}(t,x)\rangle + \|\hat{\sigma}(t,x)\|_F^2 \le -c_1\|r\|^2 + c_2. \tag{34}$$

Choose any $\beta, C_\sigma \ge 0$ that satisfies $\beta + C_\sigma = c_2$. A sufficient way to satisfy the joint condition in equation 34 is:

$$\|\hat{\sigma}(t,x)\|_F^2 \le C_\sigma, \qquad 2\,\langle r, \hat{\mu}(t,x)\rangle \le -c_1\|r\|^2 + \beta. \tag{35}$$

When $\beta = 0$, equivalently $C_\sigma = c_2$, the selected drift is globally dissipative. When $\beta > 0$, non-dissipative drift is permitted inside the ball:

$$\|r\| \le \sqrt{\frac{\beta}{c_1}}, \tag{36}$$

which can accommodate chaotic dynamics, while the drift must be dissipative outside the ball. Thus, identical values of (c_1, c_2) in the joint condition do not identify how much of c_2 is allocated to diffusion through C_σ and how much is allocated to drift through β, and therefore cannot distinguish globally dissipative drift dynamics from dynamics that are dissipative only outside a bounded region.

A.2 Compatibility of Linear-Growth Diffusion and the Lyapunov Function

In this section, we provide sufficient conditions under which the selected diffusion and Lyapunov function satisfy Assumption 3.2.

First, suppose the selected diffusion satisfies the subsystem-aligned linear-growth condition, which is automatically fulfilled if the selected diffusion networks receive only the selected states as inputs:

$$\|\hat{\sigma}(t,x)\|_F^2 \le K_0 + K_1\|Mx\|^2, \qquad K_0, K_1 \ge 0. \tag{37}$$

For a symmetric matrix H, define:

$$\lambda_+(H) := \max\{\lambda_{\max}(H), 0\}. \tag{38}$$

Here $\lambda_{\max}(H)$ denotes the largest eigenvalue of H. Because $\hat{\sigma}\hat{\sigma}^\top$ is positive semidefinite,

$$\begin{aligned}\frac{1}{2}\operatorname{tr}\big(\hat{\sigma}\hat{\sigma}^\top\nabla_x^2 V\big) &\le \frac{1}{2}\|\hat{\sigma}\|_F^2\lambda_+\big(\nabla_x^2 V\big)\\ &\le \frac{1}{2}\big(K_0 + K_1\|Mx\|^2\big)\lambda_+\big(\nabla_x^2 V\big)\,.\end{aligned} \tag{39}$$

Consequently, a sufficient compatibility condition on the Lyapunov function is

$$\big(K_0 + K_1\|Mx\|^2\big)\lambda_+\big(\nabla_x^2 V(x)\big) \le 2\rho V(x) + 2C_\sigma. \tag{40}$$

We now show that quadratic Lyapunov functions and a broader class with polynomial growth satisfy this condition. For the positive-definite quadratic Lyapunov function

$$V(x) = (Mx)^\top P(Mx), \qquad P \succ 0 \tag{41}$$

on the selected coordinates, $\nabla_x^2 V = 2MPM$ and

$$\begin{aligned}\frac{1}{2}\operatorname{tr}\big(\hat{\sigma}\hat{\sigma}^\top\nabla_x^2 V\big) &\le \lambda_{\max}(P)\|\hat{\sigma}\|_F^2\\ &\le \frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}K_1 V(x) + \lambda_{\max}(P)K_0.\end{aligned} \tag{42}$$

Thus one valid choice is

$$\rho = \frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}K_1, \qquad C_\sigma = \lambda_{\max}(P)K_0. \tag{43}$$

Every selected linear-growth diffusion therefore satisfies the affine contribution assumption under a positive-definite quadratic Lyapunov function. More generally, let $r = \|Mx\|$ and suppose that, for some $p \ge 2$ and positive constants a, b, H_0, H_1,

$$V(x) \ge ar^p - b, \qquad \lambda_+\big(\nabla_x^2 V(x)\big) \le H_0 + H_1 r^{p-2}. \tag{44}$$

Substitution into the left-hand side of equation 40 gives

$$\begin{aligned}&\frac{1}{2}(K_0 + K_1 r^2)(H_0 + H_1 r^{p-2})\\ &\quad= \frac{1}{2}\left(K_0H_0 + K_1H_0 r^2 + K_0H_1 r^{p-2} + K_1H_1 r^p\right) = \mathcal{O}(1 + r^p).\end{aligned} \tag{45}$$

Since $V(x) \ge ar^p - b$, this expression is bounded above by $\rho V(x) + C_\sigma$ for suitable finite $\rho, C_\sigma \ge 0$.

Next, we show that a wider range of Lyapunov functions can be used under a stricter uniform bound on the selected diffusion:

$$\|\hat{\sigma}(t,x)\|_F^2 \le K_0. \tag{46}$$

The compatibility condition then becomes:

$$\lambda_+\big(\nabla_x^2 V(x)\big) \le AV(x) + B, \qquad A, B \ge 0. \tag{47}$$

Scaled $\tanh$ or sigmoid output heads provide the required uniform diffusion bound.

A.3 Proof of Proposition 3.3

The affine diffusion bound in equation 6 ensures that the interval in equation 7 is nonempty. Let $(\hat{\mu}, \hat{\sigma}) \in \mathcal{D}_{\text{drift}}$. By definition, there exist $\alpha > \rho$ and $\beta \geq 0$ such that the drift inequality in equation 10 holds. Choose:

$$c_1 = \alpha - \rho > 0, \qquad c_2 = \beta + C_\sigma \geq 0. \tag{48}$$

and take the admissible envelope $q(t, x) = \rho V(x) + C_\sigma$. Then

$$\begin{aligned} -c_1 V(x) + c_2 - q(t, x) &= -(\alpha - \rho)V(x) + \beta + C_\sigma - \rho V(x) - C_\sigma \\ &= -\alpha V(x) + \beta. \end{aligned} \tag{49}$$

It follows that

$$\langle \nabla_x V(x), \hat{\mu}(t, x) \rangle \leq -c_1 V(x) + c_2 - q(t, x), \tag{50}$$

so $(\hat{\mu}, \hat{\sigma}) \in \mathcal{D}_{\text{comp}}$. This proves the first inclusion.

Now let $(\hat{\mu}, \hat{\sigma}) \in \mathcal{D}_{\text{comp}}$, and let q be the finite measurable function in its definition. Its lower bound gives

$$\begin{aligned} \langle \nabla_x V(x), \hat{\mu}(t, x) \rangle + \frac{1}{2} \operatorname{tr}\big(\hat{\sigma}(t, x)\hat{\sigma}(t, x)^\top \nabla_x^2 V(x)\big) &\leq \langle \nabla_x V(x), \hat{\mu}(t, x) \rangle + q(t, x) \\ &\leq -c_1 V(x) + c_2. \end{aligned} \tag{51}$$

Hence $(\hat{\mu}, \hat{\sigma}) \in \mathcal{D}_{\text{FL}}$ proves the second inclusion.

A.4 Compensation choices and permitted dynamics

In this section, we discuss the stability–flexibility trade-off between upper bound and exact pointwise compensation. Under upper bound compensation, define $\alpha = c_1 + \rho$ and $\beta = c_2 - C_\sigma$. The drift condition in equation 8 becomes:

$$\langle \nabla_x V, \hat{\mu} \rangle \leq -\alpha V + \beta. \tag{52}$$

When $\beta = 0$, the selected drift guarantees global dissipativity. When $\beta > 0$, the model is allowed to increase the Lyapunov potential within:

$$\left\{ x : V(x) \leq \frac{\beta}{\alpha} \right\}, \tag{53}$$

while dissipativity is required outside this set, accommodating chaotic dynamics.

The lower bound compensation, instead, imposes:

$$\langle \nabla_x V, \hat{\mu} \rangle \leq -c_1 V + c_2 - \frac{1}{2} \operatorname{tr}\big(\hat{\sigma}\hat{\sigma}^\top \nabla_x^2 V\big), \tag{54}$$

so the selected drift is guaranteed not to increase V at a point with $V(x) > 0$ only when:

$$\Gamma_V^{\hat{\sigma}}(t, x) = \frac{1}{2} \operatorname{tr}\big(\hat{\sigma}\hat{\sigma}^\top \nabla_x^2 V\big) \geq c_2 - c_1 V(x), \tag{55}$$

which is not satisfied automatically. Therefore, the lower bound enforcement cannot by itself guarantee global dissipativity.

A.5 PINN-style Regularization and Gradient Balancing

The expectation in equation 18 is approximated using finitely many sampled state-time points from ν_{sample}. However, the data and the physical (dissipativity) losses can have substantially different gradient scales, and so manual tuning of the scales can be difficult. Here, we make use of the gradient-balancing scheme similar to the PINN literature.

Let Θ_{bal} denote the parameters with respect to which the two gradients are balanced. At an optimization step k, define:

$$g_{\text{data},k} := \|\nabla_{\Theta_{\text{bal}}} \mathcal{L}_{\text{data}}\|_2, \qquad g_{\text{diss},k} := \|\nabla_{\Theta_{\text{bal}}} \mathcal{L}_{\text{diss}}\|_2. \tag{56}$$

For a target weighted gradient ratio $\tau_g > 0$, the proposed weight is:

$$\widehat{\lambda}_k := \tau_g \frac{g_{\text{data},k}}{g_{\text{diss},k} + \varepsilon_g}, \qquad \varepsilon_g > 0. \tag{57}$$

The operative weight is an exponentially smoothed and clipped update:

$$\lambda_{\text{diss},k} = \operatorname{clip}\Big((1 - \eta)\lambda_{\text{diss},k-1} + \eta\widehat{\lambda}_k, \lambda_{\min}, \lambda_{\max}\Big), \qquad 0 < \eta \leq 1. \tag{58}$$

A.6 SINGULAR-SET FEASIBILITY

For any $x \in \mathcal{S}_V$, the selected drift satisfies:

$$\langle \nabla_x V(x), \hat{\mu}(t,x) \rangle = 0. \tag{59}$$

Substitution into equation 8 shows that feasibility on the singular set is equivalent to:

$$q_\theta(t,x) \le c_2 - c_1 V(x), \qquad (t,x) \in \mathcal{T} \times \mathcal{S}_V, \tag{60}$$

which transfers to the following lower bound on c_2 that requires enforcement:

$$c_2 \ge c_{2,\min} := \max\left\{0, \sup_{(t,x)\in\mathcal{T}\times\mathcal{S}_V} \left(c_1 V(x) + q_\theta(t,x)\right)\right\}. \tag{61}$$

Moreover, equation 7 gives the validity upper bound:

$$c_{2,\min} \le \max\left\{0, (c_1+\rho) \sup_{x\in\mathcal{S}_V} V(x) + C_\sigma\right\}. \tag{62}$$

We now discuss its enforcement for the two residual choices in equation 15 and equation 16.

For upper bound compensation, let:

$$V_{\mathcal{S}} := \sup_{x\in\mathcal{S}_V} V(x). \tag{63}$$

Because $q_\theta(t,x) = \rho V(x) + C_\sigma$ and $\alpha = c_1 + \rho$, the singular-set condition is $c_2 \ge C_\sigma + \alpha V_{\mathcal{S}}$ and transfers to the reparameterized parameter β as:

$$\beta := c_2 - C_\sigma \ge \alpha V_{\mathcal{S}}. \tag{64}$$

In the common case where every singular point is a normalized minimum of V, we have $V_{\mathcal{S}} = 0$, so the lower bound reduces to $\beta \ge 0$ and requires no additional treatment.

For exact pointwise compensation, $q_\theta = \Gamma_V^{\hat{\sigma}_\theta}$ and c_2 remains explicit in the residual in equation 16. For trainable c_2, we use:

$$c_2 = c_{2,\min} + s_2 \,\mathrm{softplus}(\eta_2), \qquad s_2 > 0, \tag{65}$$

which trains η_2 and preserves the lower bound throughout training. We obtain $c_{2,\min}$ from the model's own diffusion over the known singular set and a fixed time grid on the training domain, providing a sampled-time guarantee.

A.7 PROOF OF THEOREM 3.4

To begin with, we show that projection is conditionally guaranteed not to worsen the drift learned, similar to (Min & Azizan, 2024). First consider $x \in \mathcal{S}_V$. Since the data-generating drift lies in the feasible set $\mathcal{H}_\theta(t,x)$, the drift constraint is feasible at (t,x). By equation 20:

$$\mu_\theta^{\mathrm{str}}(t,x) = \mu_\theta^{\mathrm{free}}(t,x), \tag{66}$$

and hence the claimed no-regret inequality holds trivially with zero projection-distance term. It therefore remains to consider $x \notin \mathcal{S}_V$. Since the data-generating drift lies in the same enforced half-space, its residual is guaranteed to be non-positive:

$$r_\star^q(t,x) := \langle \nabla_x V(x), M\mu^\star(t,x) \rangle + c_1 V(x) - c_2 + q_\theta(t,x) \le 0. \tag{67}$$

The inner product of the drift errors for the free drift and projected drift is also non-positive:

$$\begin{aligned} &\left\langle \mu_\theta^{\mathrm{free}}(t,x) - \mu_\theta^{\mathrm{str}}(t,x), \mu^\star(t,x) - \mu_\theta^{\mathrm{str}}(t,x) \right\rangle \\ &\quad = \frac{[r_\theta^q(t,x)]_+}{\|M\nabla_x V(x)\|^2} r_\star^q(t,x) \le 0. \end{aligned} \tag{68}$$

This then leads to the guarantee:

$$\begin{aligned}
&\left\|\mu_\theta^{\text{free}}(t,x) - \mu^\star(t,x)\right\|^2 \\
&\quad = \left\|\mu_\theta^{\text{free}}(t,x) - \mu_\theta^{\text{str}}(t,x)\right\|^2 + \left\|\mu_\theta^{\text{str}}(t,x) - \mu^\star(t,x)\right\|^2 \\
&\qquad + 2\left\langle \mu_\theta^{\text{free}}(t,x) - \mu_\theta^{\text{str}}(t,x), \mu_\theta^{\text{str}}(t,x) - \mu^\star(t,x)\right\rangle \\
&\quad \geq \left\|\mu_\theta^{\text{free}}(t,x) - \mu_\theta^{\text{str}}(t,x)\right\|^2 + \left\|\mu_\theta^{\text{str}}(t,x) - \mu^\star(t,x)\right\|^2 .
\end{aligned} \tag{69}$$

We next evaluate the local generator discrepancy. For any $g \in \mathbb{R}^d$ and any symmetric $H \in \mathbb{R}^{d\times d}$, construct a test function:

$$f_{g,H}(y) := g^\top (y-x) + \frac{1}{2}(y-x)^\top H (y-x), \tag{70}$$

which has $\nabla f_{g,H}(x) = g$ and $\nabla^2 f_{g,H}(x) = H$. Therefore, this construction realizes the entire local 2-jet space. The supremum in equation 24 can then be rewritten as:

$$\begin{aligned}
E^j_{\text{gen}}(t,x) = \sup_{\substack{g\in\mathbb{R}^d,\ H=H^\top \\ \|g\|^2+\eta\|H\|_F^2\leq 1}} \Bigg| &\left\langle \mu_\theta^j(t,x) - \mu^\star(t,x), g\right\rangle \\
&+ \frac{1}{2}\left(\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top\right) : H \Bigg|, \qquad j \in \{\text{free}, \text{str}\}.
\end{aligned} \tag{71}$$

Note that in the local 2-jet space $\mathbb{R}^d \oplus \mathbb{S}^d$, where $\mathbb{S}^d$ is the space of symmetric $d \times d$ matrices, the inner product is defined as:

$$\left\langle (g,H), (\widetilde{g},\widetilde{H})\right\rangle_{\mathcal{J}_\eta} := \langle g, \widetilde{g}\rangle + \eta H : \widetilde{H}. \tag{72}$$

The generator difference can then be expressed as a single inner product in this weighted 2-jet space:

$$\begin{aligned}
&\left(\mathcal{L}^j_{\theta,t} - \mathcal{L}^\star_t\right) f(x) \\
&\quad = \left\langle \left(\mu_\theta^j(t,x) - \mu^\star(t,x), \frac{\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top}{2\eta}\right), \left(\nabla_x f(x), \nabla_x^2 f(x)\right)\right\rangle_{\mathcal{J}_\eta}.
\end{aligned} \tag{73}$$

Therefore, the Cauchy–Schwarz inequality can be applied in this weighted 2-jet space to give:

$$\begin{aligned}
&\left|\left(\mathcal{L}^j_{\theta,t} - \mathcal{L}^\star_t\right) f(x)\right| \\
&\quad \leq \left(\left\|\mu_\theta^j(t,x) - \mu^\star(t,x)\right\|^2 + \frac{1}{4\eta}\left\|\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top\right\|_F^2\right)^{1/2} \\
&\qquad \times \left(\|\nabla_x f(x)\|^2 + \eta\|\nabla_x^2 f(x)\|_F^2\right)^{1/2}.
\end{aligned} \tag{74}$$

Define:

$$\begin{aligned}
D_j(t,x) := \Bigg(&\left\|\mu_\theta^j(t,x) - \mu^\star(t,x)\right\|^2 \\
&+ \frac{1}{4\eta}\left\|\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top\right\|_F^2\Bigg)^{1/2}.
\end{aligned} \tag{75}$$

Equation equation 74 implies $E^j_{\text{gen}}(t,x) \leq D_j(t,x)$ because the test function term is at most 1 in the test space. When $D_j(t,x) = 0$, both coefficient errors are zero and the identity holds trivially. When $D_j(t,x) > 0$, we show that the upper bound is attained by choosing:

$$\begin{aligned}
g_j^{\max} &:= \frac{\mu_\theta^j(t,x) - \mu^\star(t,x)}{D_j(t,x)}, \\
H_j^{\max} &:= \frac{\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top}{2\eta D_j(t,x)}.
\end{aligned} \tag{76}$$

The corresponding test function is

$$f_j^{\max}(y) := \left(g_j^{\max}\right)^\top (y-x) + \frac{1}{2}(y-x)^\top H_j^{\max}(y-x), \tag{77}$$

which attains the unit sphere:

$$\begin{aligned}
&\left\|\nabla f_j^{\max}(x)\right\|^2 + \eta \left\|\nabla^2 f_j^{\max}(x)\right\|_F^2 \\
&\quad = \left\|g_j^{\max}\right\|^2 + \eta \left\|H_j^{\max}\right\|_F^2 \\
&\quad = \frac{1}{D_j(t,x)^2}\left(\left\|\mu_\theta^j(t,x) - \mu^\star(t,x)\right\|^2 + \frac{1}{4\eta}\left\|\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top\right\|_F^2 \right) \\
&\quad = 1.
\end{aligned} \tag{78}$$

Evaluating the generator difference at this test function attains the upper bound:

$$\begin{aligned}
&\left(\mathcal{L}_{\theta,t}^j - \mathcal{L}_t^\star\right) f_j^{\max}(x) \\
&\quad = \frac{\left\|\mu_\theta^j(t,x) - \mu^\star(t,x)\right\|^2}{D_j(t,x)} + \frac{\left\|\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top\right\|_F^2}{4\eta D_j(t,x)} \\
&\quad = D_j(t,x).
\end{aligned} \tag{79}$$

This shows that:

$$E_{\text{gen}}^j(t,x) = D_j(t,x). \tag{80}$$

Squaring leads to:

$$\begin{aligned}
E_{\text{gen}}^j(t,x)^2 = &\left\|\mu_\theta^j(t,x) - \mu^\star(t,x)\right\|^2 \\
&+ \frac{1}{4\eta}\left\|\sigma_\theta(t,x)\sigma_\theta(t,x)^\top - \sigma^\star(t,x)\sigma^\star(t,x)^\top\right\|_F^2, \qquad j \in \{\text{free}, \text{str}\}.
\end{aligned} \tag{81}$$

Structural projection modifies only the learned drift. Hence, the diffusion-covariance error in equation 81 is identical for the free and structurally constrained models. We then have:

$$\begin{aligned}
&E_{\text{gen}}^{\text{str}}(t,x)^2 + \left\|\mu_\theta^{\text{str}}(t,x) - \mu_\theta^{\text{free}}(t,x)\right\|^2 \\
&\qquad \le E_{\text{gen}}^{\text{free}}(t,x)^2,
\end{aligned} \tag{82}$$

which proves the claim.

A.8 Proof of Theorem 3.6

To show the uniform path approximation capability of the partial dissipativity framework, first consider the data-generating SDE $X^\star$ and let the sequence of learned Neural SDEs $(X^n)_{n\ge 1}$ approximate the data-generating SDE:

$$dX_t^j = \mu_j(t, X_t^j)\, dt + \sigma_j(t, X_t^j)\, dB_t, \qquad j \in \{\star, n\}. \tag{83}$$

To obtain localization in the complete state, define the auxiliary function:

$$W(x) := V(x) + U(x), \tag{84}$$

where U is the unselected-state growth function in Assumption 3.5. Since $V(x) \to \infty$ as $\|Mx\| \to \infty$ and $U(x) \to \infty$ as $\|(I-M)x\| \to \infty$, while $\|x\|^2 = \|Mx\|^2 + \|(I-M)x\|^2$, we have

$$W(x) \to \infty \qquad \text{as} \qquad \|x\| \to \infty. \tag{85}$$

Hence every sublevel set of W is compact.

The set of initial laws where the uniform path approximation is established is: for $0 \le m_0 < \infty$,

$$\mathcal{P}_{m_0}(W) := \left\{ \nu : \int_{\mathbb{R}^d} W(x)\, \nu(dx) \le m_0 \right\}. \tag{86}$$

Note that this still permits initial laws with unbounded support, while requiring a common moment bound on both the selected-state Lyapunov energy and the unselected-state growth function.

For $R < \infty$, define the full-state localization region and exit times by:

$$\mathcal{C}_R := \left\{(t,x) \in [0,T] \times \mathbb{R}^d : W(x) \le R\right\}, \qquad \tau_R^j := \inf\left\{t \in [0,T] : W(X_t^j) \ge R\right\}. \tag{87}$$

By the coercivity of W, $\mathcal{C}_R$ is compact for every $R < \infty$.

By assumption, the data-generating and learned SDEs satisfy the joint partial dissipativity condition in equation 5 with the same M, V, c_1, and c_2:

$$\mathcal{L}_t V(x) \le -c_1 V(x) + c_2. \tag{88}$$

The unselected-state growth assumption further gives:

$$\mathcal{L}_t U(x) \le a_0 + a_1 V(x) + a_2 U(x). \tag{89}$$

Therefore,

$$\begin{aligned} \mathcal{L}_t W(x) &= \mathcal{L}_t V(x) + \mathcal{L}_t U(x) \\ &\le a_0 + c_2 + (a_1 - c_1) V(x) + a_2 U(x) \\ &\le C_W + \kappa W(x), \end{aligned} \tag{90}$$

where

$$C_W := a_0 + c_2, \qquad \kappa := \max\{(a_1 - c_1)_+, a_2\}. \tag{91}$$

Applying Itô's formula to the stopped process, after standard localization of the local martingale term, and then applying Grönwall's inequality gives, for $0 \le t \le T$,

$$\begin{aligned} \mathbb{E}_\nu W(X^j_{t\wedge\tau_R^j}) &\le \mathbb{E}_\nu W(X_0) + C_W t + \kappa \int_0^t \mathbb{E}_\nu W(X^j_{s\wedge\tau_R^j})\, ds \\ &\le e^{\kappa t}\left(\mathbb{E}_\nu W(X_0) + C_W t\right). \end{aligned} \tag{92}$$

For $\nu \in \mathcal{P}_{m_0}(W)$, we then have:

$$\mathbb{E}_\nu W(X^j_{t\wedge\tau_R^j}) \le \Gamma_T(m_0), \qquad \Gamma_T(m_0) := e^{\kappa T}(m_0 + C_W T). \tag{93}$$

On the event $\{\tau_R^j \le T\}$, $W(X^j_{\tau_R^j}) \ge R$, and therefore

$$R\,\mathbb{P}_\nu(\tau_R^j \le T) \le \mathbb{E}_\nu W(X^j_{T\wedge\tau_R^j}) \le \Gamma_T(m_0), \tag{94}$$

which implies:

$$\sup_{j\in\{\star,1,2,\dots\}} \sup_{\nu\in\mathcal{P}_{m_0}(W)} \mathbb{P}_\nu(\tau_R^j \le T) \le \frac{\Gamma_T(m_0)}{R}. \tag{95}$$

On each $\mathcal{C}_R$, suppose that the data-generating SDE is Lipschitz in the complete state for its drift and diffusion functions, uniformly in time, and so for some $L_R < \infty$ and all $(t,x), (t,y) \in \mathcal{C}_R$:

$$\|\mu_\star(t,x) - \mu_\star(t,y)\| \le L_R \|x - y\|, \tag{96}$$

$$\|\sigma_\star(t,x) - \sigma_\star(t,y)\|_F \le L_R \|x - y\|. \tag{97}$$

Define also the localized drift and diffusion approximation errors by:

$$\varepsilon_{\mu,n,R} := \sup_{(t,x)\in\mathcal{C}_R} \|\mu_n(t,x) - \mu_\star(t,x)\|, \tag{98}$$

$$\varepsilon_{\sigma,n,R} := \sup_{(t,x)\in\mathcal{C}_R} \|\sigma_n(t,x) - \sigma_\star(t,x)\|_F. \tag{99}$$

Fix R, n, and $\nu \in \mathcal{P}_{m_0}(W)$. Under the synchronous coupling, define the joint stopping time, path error, and no-exit event by:

$$\tau_R := \tau_R^n \wedge \tau_R^\star, \qquad Z_t := X_t^n - X_t^\star, \qquad A_R := \{\tau_R > T\}. \tag{100}$$

To bound the complete path error, we separate the paths in A_R, where both processes remain in $\mathcal{C}_R$, from the paths in A_R^c, where at least one process exits $\mathcal{C}_R$. For $\varepsilon > 0$:

$$\begin{aligned}
&\mathbb{P}_\nu\left(\sup_{0\le t\le T}\|X_t^n - X_t^\star\|^2 > \varepsilon\right)\\
&\quad= \mathbb{P}_\nu\left(\left\{\sup_{0\le t\le T}\|Z_t\|^2 > \varepsilon\right\}\cap A_R\right) + \mathbb{P}_\nu\left(\left\{\sup_{0\le t\le T}\|Z_t\|^2 > \varepsilon\right\}\cap A_R^c\right)\\
&\quad\le \mathbb{P}_\nu(A_R^c) + \mathbb{P}_\nu\left(\sup_{0\le t\le T}\|Z_{t\wedge\tau_R}\|^2 > \varepsilon\right)\\
&\quad\le \mathbb{P}_\nu(A_R^c) + \frac{1}{\varepsilon}\mathbb{E}_\nu\left[\sup_{0\le t\le T}\|Z_{t\wedge\tau_R}\|^2\right],
\end{aligned} \tag{101}$$

where the last line is obtained by Markov's inequality, and the first term is the exit probability controlled by equation 95. It therefore remains to bound the second term of the expected stopped pathwise error.

To analyze the stopped process path error, fix R, n, and $\nu \in \mathcal{P}_{m_0}(W)$. Because both processes use the same initial random variable, $Z_0 = 0$ under the synchronous coupling. Subtracting the two SDEs and stopping at τ_R gives:

$$\begin{aligned}
Z_{t\wedge\tau_R} = &\int_0^t \mathbf{1}_{\{s<\tau_R\}}\left[\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\right]ds\\
&+ \int_0^t \mathbf{1}_{\{s<\tau_R\}}\left[\sigma_n(s,X_s^n) - \sigma_\star(s,X_s^\star)\right]dB_s,
\end{aligned} \tag{102}$$

which has 2 contributions; the first term is the drift error, while the second term is the diffusion error. For $s < \tau_R$, both processes remain in $\mathcal{C}_R$. Therefore, the Euclidean norm of the drift error and the Frobenius norm of the diffusion error are bounded by the localized approximation errors in equation 98 and equation 99, and Lipschitz continuity in equation 96 and equation 97:

$$\begin{aligned}
\|\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\| &\le \|\mu_n(s,X_s^n) - \mu_\star(s,X_s^n)\|\\
&\quad + \|\mu_\star(s,X_s^n) - \mu_\star(s,X_s^\star)\|\\
&\le \varepsilon_{\mu,n,R} + L_R\|Z_s\|,
\end{aligned} \tag{103}$$

and

$$\|\sigma_n(s,X_s^n) - \sigma_\star(s,X_s^\star)\|_F \le \varepsilon_{\sigma,n,R} + L_R\|Z_s\|. \tag{104}$$

For the drift contribution, first apply the triangle inequality, followed by the Cauchy–Schwarz inequality, which gives, for every $0 \le u \le t$:

$$\begin{aligned}
&\left\|\int_0^u \mathbf{1}_{\{s<\tau_R\}}\left[\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\right]ds\right\|\\
&\quad\le \int_0^u \mathbf{1}_{\{s<\tau_R\}}\|\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\|\,ds\\
&\quad\le \left(\int_0^u \mathbf{1}_{\{s<\tau_R\}}ds\right)^{1/2}\left(\int_0^u \mathbf{1}_{\{s<\tau_R\}}\|\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\|^2\,ds\right)^{1/2}\\
&\quad= (u\wedge\tau_R)^{1/2}\left(\int_0^u \mathbf{1}_{\{s<\tau_R\}}\|\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\|^2\,ds\right)^{1/2}\\
&\quad\le u^{1/2}\left(\int_0^u \mathbf{1}_{\{s<\tau_R\}}\|\mu_n(s,X_s^n) - \mu_\star(s,X_s^\star)\|^2\,ds\right)^{1/2}.
\end{aligned} \tag{105}$$

Squaring, taking the supremum over $0 \le u \le t$, and substituting equation 103 therefore gives the following bound on the drift contribution:

$$\begin{aligned}
&\sup_{0\le u\le t}\left\|\int_0^u \mathbf{1}_{\{s<\tau_R\}}\left[\mu_n(s,X_s^n)-\mu_\star(s,X_s^\star)\right]ds\right\|^2 \\
&\quad\le 2t\varepsilon_{\mu,n,R}^2\int_0^t \mathbf{1}_{\{s<\tau_R\}}ds + 2tL_R^2\int_0^t \mathbf{1}_{\{s<\tau_R\}}\|Z_s\|^2 ds \\
&\quad\le 2t^2\varepsilon_{\mu,n,R}^2 + 2tL_R^2\int_0^t \mathbf{1}_{\{s<\tau_R\}}\|Z_s\|^2 ds. \qquad (106)
\end{aligned}$$

Taking expectations and applying Tonelli's theorem to the nonnegative integrand gives the following upper bound on the drift contribution:

$$\begin{aligned}
&\mathbb{E}_\nu\left[\sup_{0\le u\le t}\left\|\int_0^u \mathbf{1}_{\{s<\tau_R\}}\left[\mu_n(s,X_s^n)-\mu_\star(s,X_s^\star)\right]ds\right\|^2\right] \\
&\quad\le 2t^2\varepsilon_{\mu,n,R}^2 + 2tL_R^2\int_0^t \mathbb{E}_\nu\left[\mathbf{1}_{\{s<\tau_R\}}\|Z_s\|^2\right]ds \qquad (107)\\
&\quad\le 2t^2\varepsilon_{\mu,n,R}^2 + 2tL_R^2\int_0^t \mathbb{E}_\nu\left[\sup_{0\le v\le s}\|Z_{v\wedge\tau_R}\|^2\right]ds. \qquad (108)
\end{aligned}$$

For the diffusion contribution, Doob's L^2 maximal inequality provides an upper bound on the expected supremum:

$$\begin{aligned}
&\mathbb{E}_\nu\left[\sup_{0\le u\le t}\left\|\int_0^u \mathbf{1}_{\{s<\tau_R\}}\left[\sigma_n(s,X_s^n)-\sigma_\star(s,X_s^\star)\right]dB_s\right\|^2\right] \\
&\quad\le 4\mathbb{E}_\nu\left[\left\|\int_0^t \mathbf{1}_{\{s<\tau_R\}}\left[\sigma_n(s,X_s^n)-\sigma_\star(s,X_s^\star)\right]dB_s\right\|^2\right]. \qquad (109)
\end{aligned}$$

Itô isometry further converts the right-hand side into an ordinary time integral:

$$\begin{aligned}
&\mathbb{E}_\nu\left[\sup_{0\le u\le t}\left\|\int_0^u \mathbf{1}_{\{s<\tau_R\}}\left[\sigma_n(s,X_s^n)-\sigma_\star(s,X_s^\star)\right]dB_s\right\|^2\right] \\
&\quad\le 4\mathbb{E}_\nu\int_0^t \mathbf{1}_{\{s<\tau_R\}}\left\|\sigma_n(s,X_s^n)-\sigma_\star(s,X_s^\star)\right\|_F^2 ds. \qquad (110)
\end{aligned}$$

Substituting the Frobenius-norm bound in equation 104, using $(a+b)^2 \le 2a^2+2b^2$, and applying Tonelli's theorem leads to the following bound on the diffusion contribution:

$$\begin{aligned}
&\mathbb{E}_\nu\left[\sup_{0\le u\le t}\left\|\int_0^u \mathbf{1}_{\{s<\tau_R\}}\left[\sigma_n(s,X_s^n)-\sigma_\star(s,X_s^\star)\right]dB_s\right\|^2\right] \\
&\quad\le 8t\varepsilon_{\sigma,n,R}^2 + 8L_R^2\int_0^t \mathbb{E}_\nu\left[\mathbf{1}_{\{s<\tau_R\}}\|Z_s\|^2\right]ds \\
&\quad\le 8t\varepsilon_{\sigma,n,R}^2 + 8L_R^2\int_0^t \mathbb{E}_\nu\left[\sup_{0\le v\le s}\|Z_{v\wedge\tau_R}\|^2\right]ds. \qquad (111)
\end{aligned}$$

Now, we may substitute the drift contribution bound in equation 108 and the diffusion contribution bound in equation 111 into the expected stopped pathwise error in equation 102, and using $\|a+b\|^2 \le$

$2\|a\|^2 + 2\|b\|^2$, gives:

$$\begin{aligned}
&\mathbb{E}_\nu\left[\sup_{0\le u\le t}\|Z_{u\wedge\tau_R}\|^2\right] \\
&\quad\le 4t^2\varepsilon_{\mu,n,R}^2 + 16t\varepsilon_{\sigma,n,R}^2 \\
&\qquad + \left(4tL_R^2 + 16L_R^2\right)\int_0^t \mathbb{E}_\nu\left[\sup_{0\le v\le s}\|Z_{v\wedge\tau_R}\|^2\right] ds \\
&\quad\le 4T^2\varepsilon_{\mu,n,R}^2 + 16T\varepsilon_{\sigma,n,R}^2 \\
&\qquad + \left(4TL_R^2 + 16L_R^2\right)\int_0^t \mathbb{E}_\nu\left[\sup_{0\le v\le s}\|Z_{v\wedge\tau_R}\|^2\right] ds, \qquad 0\le t\le T.
\end{aligned} \tag{112}$$

Grönwall's inequality then yields the exponential bound

$$\begin{aligned}
&\mathbb{E}_\nu\left[\sup_{0\le t\le T}\|Z_{t\wedge\tau_R}\|^2\right] \\
&\quad\le \left(4T^2\varepsilon_{\mu,n,R}^2 + 16T\varepsilon_{\sigma,n,R}^2\right)\exp\left[\left(4TL_R^2 + 16L_R^2\right)T\right].
\end{aligned} \tag{113}$$

Substituting equation 113 into equation 101, the containment bound and the union bound for the two exit events give, for every $\varepsilon > 0$:

$$\begin{aligned}
&\sup_{\nu\in\mathcal{P}_{m_0}(W)}\mathbb{P}_\nu\left(\sup_{0\le t\le T}\|X_t^n - X_t^\star\|^2 > \varepsilon\right) \\
&\quad\le \frac{2\Gamma_T(m_0)}{R} + \frac{1}{\varepsilon}\left(4T^2\varepsilon_{\mu,n,R}^2 + 16T\varepsilon_{\sigma,n,R}^2\right)\exp\left[\left(4TL_R^2 + 16L_R^2\right)T\right].
\end{aligned} \tag{114}$$

For every fixed R, the compact-local approximation property gives $\varepsilon_{\mu,n,R}\to 0$ and $\varepsilon_{\sigma,n,R}\to 0$ as $n\to\infty$. Hence:

$$\limsup_{n\to\infty}\sup_{\nu\in\mathcal{P}_{m_0}(W)}\mathbb{P}_\nu\left(\sup_{0\le t\le T}\|X_t^n - X_t^\star\|^2 > \varepsilon\right) \le \frac{2\Gamma_T(m_0)}{R}. \tag{115}$$

Letting the Lyapunov energy level $R\to\infty$ expands the compact localization regions to the entire state space and makes the right-hand side vanish. Replacing ε with ε^2 then gives:

$$\sup_{\nu\in\mathcal{P}_{m_0}(W)}\mathbb{P}_\nu\left(\sup_{0\le t\le T}\|X_t^n - X_t^\star\| > \varepsilon\right)\longrightarrow 0, \tag{116}$$

which proves uniform convergence in probability over the set of initial laws.

A.9 Scalable Tangential Neural Parameterizations

In this section, we describe methods to enforce the tangential condition in equation 32.

The first construction uses a matrix-valued neural network $A_\theta : \mathcal{T}\times\mathbb{R}^d\to\mathbb{R}^{d\times d}$. Define:

$$K_\theta(t,x) := M\left(A_\theta(t,x) - A_\theta(t,x)^\top\right)M \tag{117}$$

and we have:

$$\mu_{\tan,\theta}(t,x) := K_\theta(t,x)M\nabla_x V(x). \tag{118}$$

Since $K_\theta(t,x)^\top = -K_\theta(t,x)$:

$$\langle M\nabla_x V(x), \mu_{\tan,\theta}(t,x)\rangle = 0, \tag{119}$$

for every (t,x). Thus, the tangential condition is satisfied for every network parameter θ and does not require an additional penalty.

A dense matrix-valued output has d^2 degrees of freedom. A potentially more scalable alternative is obtained from two neural networks $U_\theta, V_\theta : \mathcal{T}\times\mathbb{R}^d\to\mathbb{R}^{d\times r}$, where $r\ll d$ for a low-rank representation. Define:

$$K_\theta(t,x) := M\left(U_\theta(t,x)V_\theta(t,x)^\top - V_\theta(t,x)U_\theta(t,x)^\top\right)M \tag{120}$$

and use equation 118. This reduces the matrix output from order d^2 to order dr while preserving exact orthogonality. A fixed small rank introduces a low-rank interaction prior; increasing r recovers a progressively richer class of skew-symmetric maps.

The second construction starts from an unrestricted vector-valued neural network $h_\theta : \mathcal{T} \times \mathbb{R}^d \to \mathbb{R}^d$ and projects its selected output onto the tangent space. Whenever $\|M\nabla_x V(x)\| > 0$, define:

$$\mu_{\tan,\theta}(t,x) := Mh_\theta(t,x) - \frac{\langle M\nabla_x V(x), Mh_\theta(t,x)\rangle}{\|M\nabla_x V(x)\|^2} M\nabla_x V(x). \tag{121}$$

Taking the inner product with $M\nabla_x V(x)$ shows directly that equation 32 holds. On the singular set $\{x : M\nabla_x V(x) = 0\}$, every selected vector is orthogonal to $M\nabla_x V(x)$, so the tangential output may be defined directly as $Mh_\theta(t,x)$. This construction requires only a vector-valued network and has output complexity of order d.

For high-dimensional models, the initial magnitude of a dense or low-rank tangential field can increase with dimension. We therefore use dimension-normalized initialization so that the initial tangential magnitude is comparable across dimensions. All network parameters remain trainable and unbounded after initialization; consequently, this normalization changes the optimization geometry but does not restrict the represented function class.

B Experiment Appendix

B.1 Objectives, Architectures, and Shared Protocols for Both Neural SDEs and Partially Dissipative Neural SDEs

In this section, we present the main training objectives and architectures considered for Neural SDEs in this work, an overview of the common training practices, and the enforcement practices for the partial dissipativity conditions. We consider both the training objectives and architectures as hyperparameters to be selected on the validation set, such that the reported results are not prescribed to a particular training objective or architecture. The specific validation-selection procedures are task-based and are described in the later sections. Furthermore, unless otherwise specified, all the experiments have been repeated over 3 random seeds, and we report the mean and standard deviation for the results.

Observation Framework and Notation: All of the Neural SDEs considered here, including both the unconstrained and partially dissipative variants, follow the possibly partially observed structure:

$$dX_t = \mu_\theta(t, X_t)\, dt + \sigma_\theta(t, X_t)\, dB_t, \qquad Y_k = HX_{t_k}, \tag{122}$$

where H is the observation map. Let

$$\begin{aligned} \mathbf{Y}^{\text{path}} &:= (Y_0, \dots, Y_L), \\ \widetilde{\mathbf{Y}}^{\text{path}} &:= (\widetilde{Y}_0, \dots, \widetilde{Y}_L), \qquad \widetilde{Y}_k := H\widetilde{X}_{t_k}, \end{aligned} \tag{123}$$

denote an observed data trajectory and a trajectory simulated from the Neural SDE on the same time axis. The notation $\widetilde{\mathbf{Y}}^{\text{path}\prime}$ denotes an independent simulation from the same model.

B.1.1 Training Objectives and Shared Protocols

Here, we define the training objectives used for Neural SDEs. We consider three recently proposed objectives: the signature-kernel score (Issa et al., 2023), SDE Matching (Bartosh et al., 2025), and finite-dimensional matching (FDM) (Zhang et al., 2025), together with several additional standard objectives.

Signature-kernel score: The signature-kernel score proposed in (Issa et al., 2023) compares the pathwise distributions of the observed trajectory $\mathbf{Y}^{\text{path}}$ and the simulated trajectory $\widetilde{\mathbf{Y}}^{\text{path}}$. For the signature kernel k_{sig}, the training objective is defined as:

$$\mathcal{L}_{\text{Sig}}(\theta) := \mathbb{E}k_{\text{sig}}\left(\widetilde{\mathbf{Y}}^{\text{path}}, \widetilde{\mathbf{Y}}^{\text{path}\prime}\right) - 2\mathbb{E}k_{\text{sig}}\left(\widetilde{\mathbf{Y}}^{\text{path}}, \mathbf{Y}^{\text{path}}\right). \tag{124}$$

The omitted data–data term is independent of the training parameters and is therefore ignored.

SDE Matching: The SDE Matching objective proposed in (Bartosh et al., 2025) is a simulation-free objective for training Neural SDEs. SDE Matching introduces a separate inference network F_ϕ, trained jointly with the Neural SDE, which induces approximate posterior marginals $q_\phi(Z_t \mid Y_{0:L})$. These marginals are realized by the conditional posterior SDE:

$$dZ_t = b_{\theta,\phi}(t, Z_t; Y_{0:L})\, dt + \sigma_\theta(t, Z_t)\, dW_t, \tag{125}$$

where $b_{\theta,\phi}$ is the induced posterior drift and the posterior SDE shares the same diffusion coefficient as the Neural SDE being trained. Let $p_\theta(Z_0)$ and $p_\theta(Y_t \mid Z_t)$ denote the prior initial distribution and emission distribution, respectively. For the invertible diffusion parameterization used by this objective, the three-term objective is:

$$\begin{aligned}\mathcal{L}_{\mathrm{SDEM}}(\theta,\phi) := \; & \mathrm{KL}(q_\phi(Z_0 \mid Y_{0:L}) \,\|\, p_\theta(Z_0)) \\ & + \mathbb{E}_{t, Z_t \sim q_\phi}\left[\frac{1}{2}\left\|\sigma_\theta(t,Z_t)^{-1}\big(b_{\theta,\phi}(t,Z_t;Y_{0:L}) - \mu_\theta(t,Z_t)\big)\right\|^2\right] \\ & + \mathbb{E}_{I, Z_{t_I} \sim q_\phi}\left[-\log p_\theta(Y_I \mid Z_{t_I})\right].\end{aligned} \tag{126}$$

The expectations are evaluated by Monte Carlo sampling. This is the mean-normalized Monte Carlo form used by the authors' reference implementation.

Finite-dimensional matching: In this work, we use a full-observation-grid variant motivated by FDM (Zhang et al., 2025) to accommodate the observation grids considered here. Define the concatenated simulated and observed trajectory vectors as:

$$\widetilde{\mathbf{y}} := \mathrm{vec}\left(\widetilde{Y}_0, \widetilde{Y}_1, \ldots, \widetilde{Y}_L\right), \qquad \mathbf{y} := \mathrm{vec}\left(Y_0, Y_1, \ldots, Y_L\right). \tag{127}$$

Using an RBF kernel k with a median-heuristic bandwidth, the objective is defined as:

$$\mathcal{L}_{\mathrm{FDM-full}}(\theta) := \frac{1}{2}\mathbb{E}k(\widetilde{\mathbf{y}}, \widetilde{\mathbf{y}}') - \mathbb{E}k(\widetilde{\mathbf{y}}, \mathbf{y}), \tag{128}$$

where $\widetilde{\mathbf{y}}'$ is an independent simulated trajectory vector. Again, the omitted data–data term is independent of θ and is ignored.

One-step Conditional-mean MSE: The one-step conditional-mean MSE first uses the Euler approximation to compute the conditional-mean one-step forecast from the current observation:

$$\widehat{Y}^{\mathrm{mean}}_{k+1} := Y_k + \mu_\theta(t_k, Y_k)\Delta t_k, \tag{129}$$

and is defined by:

$$\mathcal{L}_{\mathrm{1step}}(\theta) := \frac{1}{L}\sum_{k=0}^{L-1}\left\|\widehat{Y}^{\mathrm{mean}}_{k+1} - Y_{k+1}\right\|^2. \tag{130}$$

This objective is only applicable to fully observed first-order models.

Stochastic Rollout MSE: The stochastic rollout MSE computes the MSE of the entire simulated trajectory compared with the observation trajectory and is defined as:

$$\mathcal{L}_{\mathrm{roll}}(\theta) := \frac{1}{L}\sum_{k=1}^{L}\mathbb{E}\left[\left\|\widetilde{Y}_k - Y_k\right\|^2\right], \tag{131}$$

where the expectation is approximated by Monte Carlo simulation during training.

Transition Likelihood: This is another training objective that applies only to fully observed first-order Neural SDEs. It uses a local Gaussian approximation to the conditional mean and covariance based on the current Neural SDE structure:

$$m^Y_{\theta,k} = Y_k + \mu_\theta(t_k, Y_k)\Delta t_k, \tag{132}$$

$$S^Y_{\theta,k} = \sigma_\theta(t_k, Y_k)\sigma_\theta(t_k, Y_k)^\top \Delta t_k + R_{\mathrm{NLL}}, \tag{133}$$

where R_{NLL} is the fixed likelihood covariance used by this objective. Up to an additive constant independent of the parameters, the Gaussian transition negative log-likelihood is defined as:

$$\begin{aligned}\mathcal{L}_{\mathrm{NLL}}(\theta) := \frac{1}{2L}\sum_{k=0}^{L-1}\Big[& \left(Y_{k+1} - m^Y_{\theta,k}\right)^\top \left(S^Y_{\theta,k}\right)^{-1}\left(Y_{k+1} - m^Y_{\theta,k}\right) \\ & + \log\det S^Y_{\theta,k}\Big].\end{aligned} \tag{134}$$

Increment Distribution Matching: Finally, increment distribution matching compares the increments of simulated and observed paths. Let:

$$U := \widetilde{Y}_{k+1} - \widetilde{Y}_k, \qquad V := Y_{k+1} - Y_k. \tag{135}$$

This training objective is then defined as the energy distance:

$$\mathcal{L}_{\mathrm{DM}}(\theta) := 2\mathbb{E}\|U - V\| - \mathbb{E}\|U - U'\| - \mathbb{E}\|V - V'\|, \tag{136}$$

where U' and V' are independent copies of U and V, respectively.

All compatible objectives listed above are considered in the controlled dynamical-recovery experiments. In the high-dimensional application experiments, we use FDM as the scalable kernel-based distributional objective and do not additionally evaluate the more computationally demanding signature-kernel score. We also do not repeat SDE Matching throughout the high-dimensional application grid: in the controlled recovery experiments and the high-dimensional Langevin objective ablation, it does not improve distributional recovery over the selected simulation-based objectives. This is consistent with the original report in Bartosh et al. (2025), which presents performance comparable to simulation-based training while primarily improving training-time and memory complexity.

Within each dataset, all of the Neural SDE variants, including both the unconstrained ones and the dissipativity enforced ones, use the same data batches, base learning rate, weight decay, gradient-clipping threshold, epoch budget, learning rate scheduler, and early-stopping rule, with exceptions mentioned later. Unless otherwise specified, these models are optimized using AdamW with a `ReduceLROnPlateau` scheduler and global gradient-norm clipping at 10. Thus, the comparisons match the data and optimization budgets, although the regularized model has an additional computational cost for evaluating the dissipativity residual at collocation points and balancing its gradient contribution against the data objective. Training with SDE Matching follows its method-specific fixed-iteration Adam and exponential-moving-average protocol.

B.1.2 Neural SDE Architectures

In this section, we describe the Neural SDE structures considered in this paper, including the general time-inhomogeneous Neural SDE structure (Liu et al., 2019), a homogeneous variant, the Langevin-type Neural SDE structure in Stable Neural SDE (Oh et al., 2024), and the integrated variants. We emphasize that our work is complementary to these prior architectures through compatible enforcement mechanisms. Among the three stable Neural SDE variants proposed in Oh et al. (2024), we consider only the Langevin-type variant because its additive, state-independent diffusion is most directly suited to the dynamical-learning setting considered here. The other two variants impose multiplicative noise structures, and the geometric variant additionally has a compulsory absorbing point at zero. These restrictions do not match many of the dynamics considered in this work and are therefore outside its scope.

We also provide a complementary architectural design by integrating Neural SDEs. This is particularly important for modeling integrated dynamics, which appear in a wide range of time-series applications such as stock prices or physical trajectories (Box et al., 2015), where the time series may grow through the integration of a stable derivative.

General Neural SDE: This is the general time-inhomogeneous Neural SDE structure considered in Liu et al. (2019) and is defined as:

$$dX_t = \mu_\theta(t, X_t)\, dt + \sigma_\theta(t, X_t)\, dB_t. \tag{137}$$

Homogeneous Neural SDE: This is the time-homogeneous version of equation 137, which also commonly appears in the literature, and is defined as:

$$dX_t = \mu_\theta(X_t)\, dt + \sigma_\theta(X_t)\, dB_t. \tag{138}$$

Langevin-type Neural SDE: This is the Langevin-type Neural SDE variant proposed in Oh et al. (2024) that is suitable for the dynamical recovery task and is defined as:

$$dX_t = \mu_\theta(X_t)\, dt + \sigma_\theta(t)\, dB_t. \tag{139}$$

Given the previous definitions of the different Neural SDE variants, we now provide a complementary structural modification by integrating Neural SDEs.

Integrated Neural SDEs: For an integrated Neural SDE, define its observed dimensions $i = 1, \dots, I$ to have integration orders $\{p_i \geq 1\}_{i=1}^{I}$. To model the integration, each observed dimension is augmented with derivative states $X_{t,i}^{(0)}, \dots, X_{t,i}^{(p_i)}$. Here $X_{t,i}^{(0)}$ is the observed, most-integrated state, and $X_{t,i}^{(p_i)}$ is its least-integrated, highest-order derivative. We use the dynamics:

$$\begin{aligned} dX_{t,i}^{(\ell)} &= X_{t,i}^{(\ell+1)}\, dt, && \ell = 0, \dots, p_i - 1, \\ dX_{t,i}^{(p_i)} &= \mu_{\theta,i}(t, X_t)\, dt + \sum_{j=1}^{m} \sigma_{\theta,ij}(t, X_t)\, dB_t^{(j)}. \end{aligned} \tag{140}$$

The Neural SDE is defined on the highest-order derivative $X_{t,i}^{(p_i)}$ to provide stability control, while the lower-order states are generated through deterministic integration as part of the complete augmented system. This is also directly compatible with the three Neural SDE variants defined above. Their network inputs need not contain only the highest-order derivative states on which the Neural SDE is defined, but may also include the complete augmented state. In this work, we mainly consider integrated dissipative dynamics as the primary form of partial dissipativity. Therefore, to show that it is the enforcement, but not the integrated structure, that leads to the performance improvement, for each partially dissipative Neural SDE, we introduce an integrated Neural SDE with the exact same structure but without dissipative condition enforcement as a baseline.

B.1.3 Partially Dissipative Neural SDE Architectures

In this section, we describe the practice of enforcing partial dissipativity and the enforcement details across the datasets considered. We emphasize again that our methods are complementary to the Neural SDE architectures as compatible enforcement mechanisms. Our enforcement is compatible with every unconstrained Neural SDE parameterization mentioned before, combined with any integration order. In the integrated applications of this paper, partial dissipativity is enforced only on the highest-derivative states in equation 140. This provides Lyapunov containment of the selected derivative states, while the unselected integrated states corresponding to the observations may grow through integration at a controlled rate, as in Assumption 3.5. We provide the following example to show that this holds conveniently. For the finite-order integrated models in equation 140, consider:

$$U(x) = \sum_{i=1}^{I} \sum_{\ell=0}^{p_i-1} \|X_i^{(\ell)}\|^2, \qquad V(x) = \|R(x)\|^2, \tag{141}$$

where $R(x)$ collects the highest-order selected states. Then:

$$\mathcal{L}_t U(x) = 2 \sum_{i=1}^{I} \sum_{\ell=0}^{p_i-1} \left\langle X_i^{(\ell)}, X_i^{(\ell+1)} \right\rangle \leq 2U(x) + V(x). \tag{142}$$

Hence Assumption 3.5 holds with $a_0 = 0$, $a_1 = 1$, and $a_2 = 2$. The growth of the unselected state is therefore controlled.

To enforce the partially dissipative conditions, we consider both the upper and lower bound enforcements, as in equation 15 and equation 16, which introduce additional parameters to the Neural networks. The additional parameters include the potential extra parameters in the Lyapunov function, α, and potentially also β in the upper bound case (β is only set to be non-zero when chaos is desired; otherwise, α is the only additional parameter), and c_1 and c_2 in the lower bound case (the additional trace term is evaluated pointwise from the diffusion networks and does not require additional treatment). In the structurally constrained methods, both additional parameters are treated as trainable parameters. To maintain the condition in the singular set for the lower bound enforcement, the lower bound is enforced by the following reparameterization, estimated by the sample-time grid, and the unconstrained parameter ξ_2 is trained:

$$\begin{aligned} c_{2,\mathrm{floor}}(\theta, V) &:= \sup_{\{(t,x):\, M\nabla V(x)=0\}} \left[\Gamma_V^{\hat{\sigma}_\theta}(t,x) + c_1 V(x)\right]_+, \\ c_2 &= c_{2,\mathrm{floor}}(\theta, V) + \mathrm{softplus}(\xi_2). \end{aligned} \tag{143}$$

In the regularized methods, the additional parameters are selected on the validation set via grid searches in the feasible domains.

B.2 Experimental Details of Recovery of Partially Dissipative Stochastic Dynamics

In this section, we provide the additional experimental details for Section 4.1: Recovery of Partially Dissipative Stochastic Dynamics. This section includes the detailed dataset specifications, protocols for training, validation, and testing, and the full numerical results for the experiments.

B.2.1 Dataset Specifications

Shear Model The shear model generates two-dimensional observations (X_t, Y_t). In terms of Definition 3.1, both coordinates are selected, so $M = I_2$, and dissipativity is characterized by the Lyapunov function:

$$V_{\mathrm{sh},k}(x, y) := x^2 + \left(y + kx^2\right)^2. \tag{144}$$

The dynamics are:

$$\begin{aligned} dX_t &= -aX_t\, dt + \sigma_x(X_t, Y_t)\, dB_t^{(1)}, \\ dY_t &= \left[-bY_t + (2a - b)kX_t^2\right] dt + \sigma_y(X_t, Y_t)\, dB_t^{(2)}, \end{aligned} \tag{145}$$

where $B^{(1)}$ and $B^{(2)}$ are independent Brownian motions. For data generation, we use the bounded state-dependent diffusion functions and parameters:

$$\begin{aligned} &\sigma_x(x, y) = 0.20 + 0.05 \tanh(y + kx^2), \\ &\sigma_y(x, y) = 0.20 + 0.05 \tanh(x), \\ &(a, b, k) = (1, 1, 1). \end{aligned} \tag{146}$$

We generate independent training, validation, and held-out test sets, each containing 60 trajectories with 200 observations. The time intervals are sampled independently from $\mathrm{Unif}(0.01, 0.10)$.

Langevin dynamics: This is the classical one-dimensional integrated Ornstein–Uhlenbeck process. Its latent state (Q_t, V_t) consists of position and velocity, but only the position Q_t is observed. The selected coordinate is the latent velocity, so $M = \mathrm{diag}(0, 1)$, and partial dissipativity is characterized by:

$$V_{\mathrm{Lan}}(q, v) := v^2. \tag{147}$$

The dynamics are:

$$\begin{aligned} dQ_t &= V_t\, dt, \\ dV_t &= \vartheta V_t\, dt + \sigma\, dB_t, \qquad \vartheta < 0, \quad \sigma > 0. \end{aligned} \tag{148}$$

The data-generating parameters are:

$$(\vartheta, \sigma) = (-1, 1). \tag{149}$$

We generate a training set of 60 trajectories and an independent evaluation set of 60 trajectories, each containing 200 observations. The evaluation set is divided equally into 30 validation and 30 held-out test trajectories. As in the shear model, the time steps between observations are sampled independently from $\mathrm{Unif}(0.01, 0.10)$.

Stochastic Lorenz–63: The stochastic Lorenz–63 model injects multiplicative noise into the classical Lorenz system. It is a three-dimensional model in which all states are selected and observed, so $M = I_3$. The system is chaotic and is characterized by the Lyapunov function:

$$V_{\mathrm{L63}}(x, y, z) := x^2 + y^2 + (z - C)^2, \qquad C := \rho + \sigma_{\mathrm{L}}. \tag{150}$$

The resulting trajectories exhibit the characteristic two-lobed butterfly geometry of Lorenz dynamics. The model is defined by:

$$\begin{aligned} dX_t &= \sigma_{\mathrm{L}}(Y_t - X_t)\, dt + \eta X_t\, dB_t^{(1)}, \\ dY_t &= [X_t(\rho - Z_t) - Y_t]\, dt + \eta Y_t\, dB_t^{(2)}, \\ dZ_t &= (X_t Y_t - \beta_{\mathrm{L}} Z_t)\, dt + \eta Z_t\, dB_t^{(3)}, \end{aligned} \tag{151}$$

where the three Brownian motions are independent. The data-generating parameters are:

$$(\sigma_{\mathrm{L}}, \rho, \beta_{\mathrm{L}}, \eta, C) = \left(10, 28, \frac{8}{3}, 0.15, 38\right). \tag{152}$$

We generate independent training, validation, and held-out test sets, each containing 120 trajectories with 2400 observations. The observations are recorded on a regular time grid with a step size of $\Delta t = 0.005$.

B.2.2 Partial Dissipativity Enforcement Details

For the dynamical models, the functional class of the Lyapunov function is known, and we apply the joint partial dissipativity condition directly using that class. Parameters of the Lyapunov functions are treated as unknowns and are optimized with the models or selected by validation. The integration order is also set to match the data-generating dynamics exactly.

Upper Bound Enforcement Additional Details: For the upper-bound enforcement in equation 15, we need to consider the affine diffusion contribution bound in Assumption 3.2 for a lower bound on α. This is determined jointly by the specific Lyapunov function class and the diffusion network architectures.

For the shear model, the Lyapunov function Hessian is:

$$\nabla^2 V_{\mathrm{sh},k}(x, y) = \begin{pmatrix} 2 + 4kz + 8k^2x^2 & 4kx \\ 4kx & 2 \end{pmatrix}, \tag{153}$$

where z is the shear coordinate defined as $z := y + kx^2$. We employ a diagonal diffusion network $\sigma_\theta(t, x, y) = \mathrm{diag}(\sigma_{x,\theta}(t, x, y), \sigma_{y,\theta}(t, x, y))$, capped by $\sigma_{\max}$. The diffusion contribution is then:

$$\begin{aligned} \Gamma^{\sigma_\theta}_{V_{\mathrm{sh},k}}(t, x, y) &= \frac{1}{2}\sigma^2_{x,\theta}\left(2 + 4kz + 8k^2x^2\right) + \frac{1}{2}\sigma^2_{y,\theta}(2) \\ &= \sigma^2_{x,\theta}\left(1 + 2kz + 4k^2x^2\right) + \sigma^2_{y,\theta}, \end{aligned} \tag{154}$$

which is bounded by:

$$\begin{aligned} \Gamma^{\sigma_\theta}_{V_{\mathrm{sh},k}}(t, x, y) &\le \sigma^2_{\max}\left[4k^2(x^2 + z^2) + \frac{5}{4}\right] + \sigma^2_{\max} \\ &= 4k^2\sigma^2_{\max} V_{\mathrm{sh},k}(x, y) + \frac{9}{4}\sigma^2_{\max}. \end{aligned} \tag{155}$$

This then leads to the closed-form formula for ρ as a function of k:

$$\rho(k) = 4k^2\sigma^2_{\max}. \tag{156}$$

For the shear model experiments, we consider $k \in [0.5, 1.5]$ with $\sigma_{\max} = 0.4$ above the data-generating diffusion value. The worst case $\rho_{\max} = 1.44$ is enforced as a lower bound on α.

On the stochastic Lorenz-63 model, we set the diffusion network to be of the following capped linear-growth form:

$$\sigma_{i,\theta}(x) = b_{i,\theta}(x) + x_i m_{i,\theta}(x), \tag{157}$$

and the two neural networks are capped by:

$$|b_{i,\theta}(x)| \le B, \qquad |m_{i,\theta}(x)| \le L. \tag{158}$$

Writing $u = x - c$, where $c = (0, 0, C)$, Young's inequality gives, for every fixed $\varepsilon > 0$:

$$\|\sigma_\theta(x)\|_F^2 \le (1 + \varepsilon)L^2\|u\|^2 + (1 + \varepsilon^{-1})\sum_{i=1}^{3}(B + L|c_i|)^2, \tag{159}$$

and so we have:

$$\rho = (1 + \varepsilon)L^2. \tag{160}$$

In our implementation, $L = 0.5$ and because ε can be chosen arbitrarily, we pick $\varepsilon = 0.25$.

For the Langevin dynamic, the selected diffusion contribution is:

$$\Gamma^{\sigma_\theta}_{V_{\mathrm{Lan}}}(t,q,v) = \sigma_{v,\theta}(t,q,v)^2. \tag{161}$$

We impose the architectural bound: $|\sigma_{v,\theta}(t,q,v)| \le S_\sigma$, which gives

$$\Gamma^{\sigma_\theta}_{V_{\mathrm{Lan}}}(t,q,v) \le S_\sigma^2. \tag{162}$$

Thus, the affine diffusion bound holds with $\rho = 0$ and $C_\sigma = S_\sigma^2$. We parameterize the generator residual as $\beta = c_2 - C_\sigma$ and use $\beta = 0$. Consequently, C_σ is absorbed into c_2 and does not affect the projected drift. We simply pick a sufficiently large value in our experiments that is never active during the experiments.

B.2.3 Experimental Protocol

In this section, we provide the additional details about the experimental protocol in Section 4.1 for training and validation.

The training follows the general protocols in Appendix B.1.1. The drift and diffusion networks have two hidden layers of width 64 with LipSwish activations. Unless otherwise specified, training uses AdamW with a learning rate 2×10^{-3}, weight decay 10^{-5}, a `ReduceLROnPlateau` scheduler with patience 12, global gradient-norm clipping at 10, a batch size of 256, and the Euler–Maruyama solver with one solver step per observation interval. For training, each trajectory in the training dataset is further split into overlapping rolling windows. Shear is trained for at most 500 epochs using windows of length 20 and stride 5; Lorenz-63 for at most 400 epochs using windows of length 200 and stride 100; and Langevin for at most 300 epochs using windows of length 20 and stride 5. Early-stopping patience is 100 for shear and Lorenz-63 and 60 for Langevin.

On the shear and stochastic Lorenz datasets, due to training difficulties for some of the Neural SDE baselines under certain objectives, as widely reported in the existing literature, for instance in (Oh et al., 2024; Bartosh et al., 2025), we use a common 300-epoch warm start on the Stochastic Lorenz model and a common 400-epoch warm start on the shear model using transition negative log likelihood on the unconstrained matched coefficient variant. The pretrain uses the training split only and is shared by the unconstrained, regularized, and structural arms, so arms differ only in the enforcement and not in initialization.

For validation selection, in-distribution maximum mean discrepancy (MMD) with a median-heuristic RBF kernel is used as the selection metric for dynamical recovery. The search ranges carry only minimal prior knowledge about the nature of the dynamics and the roles of the constants. For upper-bound enforcement, smaller positive values of α impose weaker contraction and preserve a larger admissible dynamical class, so α is searched over $(0, 10]$, while β depends on the nature of the dynamics. For globally dissipative dynamics, β is set to 0. For stochastic Lorenz–63, following Li et al. (2022), the dissipative boundary is placed on shells far outside the empirical training attractor, and the shell level β/α is searched over $[10^3, 10^4]$. Lower-bound enforcement is treated similarly: smaller positive values of c_1 likewise preserve a larger admissible class, so c_1 is searched over $(0, 10]$, while c_2 must satisfy the singular-set requirement $c_2 \ge c_{2,\min}$. For globally dissipative dynamics, c_2 is searched over $[c_{2,\min}, 10c_{2,\min}]$; for stochastic Lorenz–63, the shell level c_2/c_1 is searched over $[10^3, 10^4]$. The unconstrained Neural SDEs select the training objective and architecture, the partially dissipative models additionally select the enforcement mechanism, and the regularized models further select the Lyapunov function parameters.

B.2.4 Full Numerical Results

In this section, we present the full numerical results for the experiments in Section 4.1. In addition, we provide the details of the additional ablation experiments for the enforcement mechanisms and the model architectures across the datasets.

Table 1 shows the numerical results corresponding to Figure 2.

Shear		
Evaluation condition	NSDE	PDNSDE (ours)
ID	0.0038 ± 0.0032	$\mathbf{0.0017 \pm 0.0003}$
$4\times$ shell	0.0115 ± 0.0072	$\mathbf{0.0046 \pm 0.0025}$
$10\times$ shell	0.0465 ± 0.0287	$\mathbf{0.0240 \pm 0.0118}$
$20\times$ shell	0.0444 ± 0.0290	$\mathbf{0.0191 \pm 0.0137}$

Langevin		
Evaluation condition	NSDE	PDNSDE (ours)
ID	0.1260 ± 0.0333	$\mathbf{0.0211 \pm 0.0077}$
$2\times$ horizon	0.1245 ± 0.0455	$\mathbf{0.0099 \pm 0.0065}$
$5\times$ horizon	0.0772 ± 0.0621	$\mathbf{0.0028 \pm 0.0023}$
$10\times$ horizon	0.0739 ± 0.0900	$\mathbf{0.0101 \pm 0.0046}$
$20\times$ horizon	0.1312 ± 0.1716	$\mathbf{0.0144 \pm 0.0082}$
$50\times$ horizon	0.2711 ± 0.3445	$\mathbf{0.0173 \pm 0.0085}$

Stochastic Lorenz–63		
Evaluation condition	NSDE	PDNSDE (ours)
ID	$\mathbf{0.0024 \pm 0.0001}$	0.0127 ± 0.0148
$2\times$ shell	0.0222 ± 0.0023	$\mathbf{0.0148 \pm 0.0074}$
$4\times$ shell	0.2237 ± 0.0634	$\mathbf{0.0214 \pm 0.0037}$
$10\times$ shell	0.4949 ± 0.0152	$\mathbf{0.0529 \pm 0.0145}$
$20\times$ shell	0.5532 ± 0.0354	$\mathbf{0.1207 \pm 0.0330}$

Table 1: Held-out distributional recovery after validation selection. Entries are MMD^2 means $\pm$ sample standard deviations over three independently trained seeds; lower is better.

Table 2 reports the ablation results for the enforcement mechanism. No enforcement or compensation choice uniformly dominates across datasets and shift levels, motivating validation-based selection.

Shear

Condition	NSDE	Reg-Upper	Reg-Lower	Struct-Upper	Struct-Lower
ID	0.0038 ± 0.0032	$\mathbf{0.0015 \pm 0.0001}$	0.0017 ± 0.0003	0.0038 ± 0.0032	0.0038 ± 0.0032
$4\times$ shell	0.0115 ± 0.0072	0.0056 ± 0.0026	$\mathbf{0.0046 \pm 0.0025}$	0.0097 ± 0.0072	0.0104 ± 0.0072
$10\times$ shell	0.0465 ± 0.0287	0.0275 ± 0.0099	$\mathbf{0.0240 \pm 0.0118}$	0.0389 ± 0.0331	0.0399 ± 0.0336
$20\times$ shell	0.0444 ± 0.0290	0.0229 ± 0.0121	$\mathbf{0.0191 \pm 0.0137}$	0.0355 ± 0.0367	0.0357 ± 0.0369

Langevin

Condition	NSDE	Reg-Upper	Reg-Lower	Struct-Upper	Struct-Lower
ID	0.1260 ± 0.0333	$\mathbf{0.0065 \pm 0.0053}$	0.0273 ± 0.0164	0.0211 ± 0.0077	0.0431 ± 0.0198
$2\times$ horizon	0.1245 ± 0.0455	0.0115 ± 0.0080	0.0251 ± 0.0296	$\mathbf{0.0099 \pm 0.0065}$	0.0265 ± 0.0216
$5\times$ horizon	0.0772 ± 0.0621	0.0151 ± 0.0083	0.0313 ± 0.0264	$\mathbf{0.0028 \pm 0.0023}$	0.0224 ± 0.0198
$10\times$ horizon	0.0739 ± 0.0900	0.0207 ± 0.0092	0.0534 ± 0.0276	$\mathbf{0.0101 \pm 0.0046}$	0.0711 ± 0.0374
$20\times$ horizon	0.1312 ± 0.1716	0.0180 ± 0.0093	0.0653 ± 0.0329	$\mathbf{0.0144 \pm 0.0082}$	0.1402 ± 0.0822
$50\times$ horizon	0.2711 ± 0.3445	0.0224 ± 0.0115	0.1040 ± 0.0546	$\mathbf{0.0173 \pm 0.0085}$	0.2540 ± 0.2199

Stochastic Lorenz–63

Condition	NSDE	Reg-Upper	Reg-Lower	Struct-Upper	Struct-Lower
ID	$\mathbf{0.0024 \pm 0.0001}$	0.0336 ± 0.0146	0.0127 ± 0.0148	0.2150 ± 0.0272	0.0325 ± 0.0046
$2\times$ shell	0.0222 ± 0.0023	0.0379 ± 0.0185	$\mathbf{0.0148 \pm 0.0074}$	0.1346 ± 0.0219	0.0977 ± 0.0877
$4\times$ shell	0.2237 ± 0.0634	0.0387 ± 0.0100	$\mathbf{0.0214 \pm 0.0037}$	0.1340 ± 0.0215	0.3548 ± 0.2257
$10\times$ shell	0.4949 ± 0.0152	0.0574 ± 0.0114	$\mathbf{0.0529 \pm 0.0145}$	0.0872 ± 0.0286	0.4804 ± 0.1959
$20\times$ shell	0.5532 ± 0.0354	0.0873 ± 0.0072	0.1207 ± 0.0330	$\mathbf{0.0729 \pm 0.0209}$	0.5073 ± 0.1178

Table 2: Held-out Upper/Lower enforcement comparison. Entries are MMD^2 means $\pm$ sample standard deviations over three seeds. Every repeated checkpoint uses the same canonical test record as Table 1.

NSDE (structure-matched)

Condition	Naive (Roll.)	Homogeneous (NLL)	Langevin (1-step)
ID	0.0687 ± 0.0131	0.0018 ± 0.0004	0.0038 ± 0.0032
$4\times$ shell	0.1061 ± 0.0246	0.0298 ± 0.0036	0.0115 ± 0.0072
$10\times$ shell	0.2630 ± 0.0498	0.0898 ± 0.0166	0.0465 ± 0.0287
$20\times$ shell	0.2730 ± 0.0464	0.0935 ± 0.0144	0.0444 ± 0.0290

PDNSDE-reg

Condition	Naive (Roll.)	Homogeneous (NLL)	Langevin (NLL)
ID	0.0639 ± 0.0111	$\mathbf{0.0008 \pm 0.0004}$	0.0017 ± 0.0003
$4\times$ shell	0.1003 ± 0.0380	0.0135 ± 0.0127	$\mathbf{0.0046 \pm 0.0025}$
$10\times$ shell	0.2400 ± 0.0660	0.0462 ± 0.0348	$\mathbf{0.0240 \pm 0.0118}$
$20\times$ shell	0.2496 ± 0.0630	0.0425 ± 0.0378	$\mathbf{0.0191 \pm 0.0137}$

PDNSDE-struct

Condition	Naive (Roll.)	Homogeneous (NLL)	Langevin (1-step)
ID	0.0687 ± 0.0131	0.0018 ± 0.0004	0.0038 ± 0.0032
$4\times$ shell	0.0890 ± 0.0140	0.0260 ± 0.0076	0.0097 ± 0.0072
$10\times$ shell	0.2212 ± 0.0285	0.0678 ± 0.0364	0.0389 ± 0.0331
$20\times$ shell	0.2344 ± 0.0317	0.0650 ± 0.0412	0.0355 ± 0.0367

Table 3: Full model-structure sweep for Shear. Within each fixed model structure and enforcement mechanism, the training objective and compensation choice are selected using validation MMD. Entries are held-out MMD^2 means $\pm$ sample standard deviations over three seeds. Bold indicates the lowest unrounded mean in each evaluation condition.

NSDE (structure-matched)

Condition	Naive (DM)	Homogeneous (Roll.)	Langevin (DM)
ID	0.0398 ± 0.0171	$\mathbf{0.0024 \pm 0.0001}$	0.0323 ± 0.0015
$2\times$ shell	0.1286 ± 0.0770	0.0222 ± 0.0023	0.1208 ± 0.1397
$4\times$ shell	0.3131 ± 0.2068	0.2237 ± 0.0634	0.2663 ± 0.2480
$10\times$ shell	0.4831 ± 0.1478	0.4949 ± 0.0152	0.4088 ± 0.1837
$20\times$ shell	0.5471 ± 0.1070	0.5532 ± 0.0354	0.5458 ± 0.0696

PDNSDE-reg

Condition	Naive (NLL)	Homogeneous (NLL)	Langevin (NLL)
ID	0.0336 ± 0.0146	0.0127 ± 0.0148	0.0337 ± 0.0018
$2\times$ shell	0.0379 ± 0.0185	$\mathbf{0.0148 \pm 0.0074}$	0.0517 ± 0.0062
$4\times$ shell	0.0387 ± 0.0100	$\mathbf{0.0214 \pm 0.0037}$	0.0697 ± 0.0099
$10\times$ shell	0.0574 ± 0.0114	$\mathbf{0.0529 \pm 0.0145}$	0.0829 ± 0.0315
$20\times$ shell	0.0873 ± 0.0072	0.1207 ± 0.0330	0.0916 ± 0.0256

PDNSDE-struct

Condition	Naive (DM)	Homogeneous (DM)	Langevin (DM)
ID	0.0325 ± 0.0046	0.0503 ± 0.0295	0.2150 ± 0.0272
$2\times$ shell	0.0977 ± 0.0877	0.0898 ± 0.0133	0.1346 ± 0.0219
$4\times$ shell	0.3548 ± 0.2257	0.3003 ± 0.1463	0.1340 ± 0.0215
$10\times$ shell	0.4804 ± 0.1959	0.5079 ± 0.1772	0.0872 ± 0.0286
$20\times$ shell	0.5073 ± 0.1178	0.5374 ± 0.0913	$\mathbf{0.0729 \pm 0.0209}$

Table 4: Full model-structure sweep for Stochastic Lorenz–63. Within each fixed model structure and enforcement mechanism, the training objective and compensation choice are selected using validation MMD. Entries are held-out MMD^2 means $\pm$ sample standard deviations over three seeds. Bold indicates the lowest unrounded mean in each evaluation condition.

NSDE (structure-matched)

Condition	Naive (DM)	Homogeneous (Sig.)	Langevin (Roll.)
ID	0.1233 ± 0.0505	0.1260 ± 0.0333	0.1762 ± 0.0003
$2\times$ horizon	0.1251 ± 0.0699	0.1245 ± 0.0455	0.1956 ± 0.0002
$5\times$ horizon	0.0710 ± 0.0598	0.0772 ± 0.0621	0.1833 ± 0.0007
$10\times$ horizon	0.0323 ± 0.0291	0.0739 ± 0.0900	0.1811 ± 0.0031
$20\times$ horizon	0.3425 ± 0.1704	0.1312 ± 0.1716	0.1504 ± 0.0231
$50\times$ horizon	0.5250 ± 0.0432	0.2711 ± 0.3445	0.0533 ± 0.0771

NSDE (unmatched order)

Condition	Naive (Sig.)	Homogeneous (Sig.)	Langevin (Sig.)
ID	0.0241 ± 0.0091	0.0502 ± 0.0178	0.0534 ± 0.0622
$2\times$ horizon	0.0403 ± 0.0216	$\mathbf{0.0077 \pm 0.0041}$	0.0684 ± 0.0524
$5\times$ horizon	0.0368 ± 0.0128	0.0488 ± 0.0421	0.0459 ± 0.0218
$10\times$ horizon	0.0840 ± 0.0883	0.1201 ± 0.0306	0.0413 ± 0.0561
$20\times$ horizon	0.5099 ± 0.5381	0.2444 ± 0.1974	0.1102 ± 0.0970
$50\times$ horizon	0.5237 ± 0.0295	0.3019 ± 0.2297	0.2315 ± 0.2615

PDNSDE-reg

Condition	Naive (DM)	Homogeneous (DM)	Langevin (Roll.)
ID	0.0866 ± 0.0550	$\mathbf{0.0065 \pm 0.0053}$	0.1763 ± 0.0002
$2\times$ horizon	0.0776 ± 0.0498	0.0115 ± 0.0080	0.1956 ± 0.0002
$5\times$ horizon	0.0128 ± 0.0040	0.0151 ± 0.0083	0.1832 ± 0.0007
$10\times$ horizon	0.0306 ± 0.0246	0.0207 ± 0.0092	0.1813 ± 0.0030
$20\times$ horizon	0.1964 ± 0.1765	0.0180 ± 0.0093	0.1523 ± 0.0207
$50\times$ horizon	0.4351 ± 0.1362	0.0224 ± 0.0115	0.0511 ± 0.0619

PDNSDE-struct

Condition	Naive (Roll.)	Homogeneous (DM)	Langevin (Roll.)
ID	0.1765 ± 0.0000	0.0211 ± 0.0077	0.1764 ± 0.0003
$2\times$ horizon	0.1959 ± 0.0000	0.0099 ± 0.0065	0.1958 ± 0.0002
$5\times$ horizon	0.1836 ± 0.0001	$\mathbf{0.0028 \pm 0.0023}$	0.1835 ± 0.0003
$10\times$ horizon	0.1718 ± 0.0093	$\mathbf{0.0101 \pm 0.0046}$	0.1840 ± 0.0005
$20\times$ horizon	0.0339 ± 0.0287	$\mathbf{0.0144 \pm 0.0082}$	0.1733 ± 0.0058
$50\times$ horizon	0.3283 ± 0.0906	$\mathbf{0.0173 \pm 0.0085}$	0.1232 ± 0.0537

Table 5: Full model-structure sweep for Langevin. Within each fixed model structure and enforcement mechanism, the training objective and compensation choice are selected using validation MMD. Entries are held-out MMD^2 means $\pm$ sample standard deviations over three seeds. Bold indicates the lowest unrounded mean in each evaluation condition.

Table 3, 5, and 4 report the ablation results across model variants for the shear, Langevin, and stochastic Lorenz-63 datasets. Similarly, the best Neural SDE architecture varies with the dynamics and evaluation regime; however, the benefits of dissipativity are not confined to a single architecture. These results support treating the model architecture as modeling choices and indicate that there is not a universally optimal configuration. Therefore, in the later experiments, we treat both the enforcements and model architectures as hyperparameters to be selected over the validation datasets. Notably, using the warm start, the drift-only training continuation using 1-step conditional-mean MSE achieves good performance on the shear dataset.

B.3 Experimental Details of High-Dimensional Application

B.3.1 Dataset Specifications

High-dimensional generalization of Langevin Dynamics: We generalize the classical Langevin dynamics to high dimensions as a clean testbed for the scaling behavior of the methods with respect to dimension:

$$\begin{aligned} dQ_t &= V_t\, dt, \\ dV_t &= \left(-\Gamma_D V_t + \frac{1}{\sqrt{D}}\Omega_D V_t\right) dt + \Sigma_D\, dB_t, \end{aligned} \tag{163}$$

where B_t is a D-dimensional Brownian motion, $\Gamma_D = \mathrm{diag}(\gamma_1, \ldots, \gamma_D)$ with $\gamma_i > 0$ carrying the dissipative behavior in each coordinate, $\Omega_D^\top = -\Omega_D$, and $\Sigma_D = \mathrm{diag}(\sigma_1, \ldots, \sigma_D)$ with $\sigma_i > 0$. The factor $D^{-1/2}$ keeps the scale of the random cross-coordinate interaction comparable across dimensions.

In terms of the partially dissipative model definition, we have $M = \mathrm{diag}(0_D, I_D)$ as only the velocity coordinates are selected, and the Lyapunov function is:

$$V_{\mathrm{HDLan}}(q, v) := \|v\|^2. \tag{164}$$

For data generation, we use:

$$\begin{aligned} \Omega_D &:= A_D - A_D^\top, & (A_D)_{ij} &\overset{\text{iid}}{\sim} \mathcal{N}(0, 0.5^2), \\ \log \gamma_i &\sim \mathrm{Unif}(\log(0.5), \log(2))\,, & \log \sigma_i &\sim \mathrm{Unif}(\log(0.7), \log(1.4))\,. \end{aligned} \tag{165}$$

We use $D \in \{4, 8, 16, 32, 64\}$. For each dimension, the training and evaluation sets contain 60 trajectories with 100 observations on a regular grid with $\Delta t = 0.1$; the evaluation set is divided equally into validation and held-out test trajectories.

S&P 500 stocks: We use daily closing prices from the S&P 500 stock dataset (Larxel, 2026). The data taken covers January 4, 2010, to December 20, 2024. Taking stocks with at least 80% coverage over time, 156 stocks are retained, though still with missingness. The dataset split is chronological: observations through 2019 are used for training, 2020–2021 for validation, and 2022–2024 for held-out testing. We remove the first 20 trading days from both evaluation periods as an embargo. The resulting dataset contains 116 training windows, 30 validation windows, and 30 test windows, each with 200 observations.

Multi-agent crowd trajectories: This is the TrajNet++ CFF station trajectories dataset from (A. et al., 2014; Ivanovic et al., 2023). Each trajectory contains six fixed agent slots in a 2-dimensional map. The dataset split is agent-disjoint and contains 3576 training trajectories, 344 validation trajectories, and 344 held-out test trajectories. Each trajectory contains 60 observations on a normalized regular time axis with unit interval $\Delta t = 1$.

MuJoCo Hopper: This is the MuJoCo Hopper trajectories dataset from the Stable Neural SDE benchmark (Todorov et al., 2012; Oh et al., 2024). Each trajectory contains 100 observations with 14 channels, comprising seven position channels $Q_t \in \mathbb{R}^7$ and seven separately observed velocity channels $V_t \in \mathbb{R}^7$. The 100 trajectories are randomly divided into 70 training trajectories and 30 evaluation trajectories; the evaluation trajectories are further divided equally into 15 validation and 15 held-out test trajectories. Each observed coordinate is standardized using training-split statistics. The time axis is again a normalized regular grid with a unit time interval.

B.3.2 Partial Dissipativity Details

Enforcement Details on the High-dimensional Datasets: Except in the joint-versus-decomposed scaling ablation, we enforce the decomposed condition in equation 30 for high-dimensional Langevin, S&P 500, crowd, and MuJoCo. For the high-dimensional Langevin dataset, both the integration order and Lyapunov function class are known, so they are set to match exactly. For the real datasets, we use the quadratic Lyapunov function with a globally dissipative radial drift on the selected coordinates, corresponding to $\beta = 0$ in equation 15:

$$V_R(r) := \|r\|^2. \tag{166}$$

We argue this to be a minimal dynamical prior: it controls the radial stability of the selected derivative subsystem but does not prescribe its invariant law, modes, circulation, or tangential interactions. Together with the enforced inward-drift inequality, this quadratic choice yields a standard form of one-sided dissipativity used in SDE analysis (Fang & Giles, 2016). More elaborate conditions would require considerable additional prior knowledge of the underlying dynamics, which is available in the controlled synthetic systems but, in general, are not available for the real datasets. The condition is enforced in an isotropic fashion with the same dissipativity coefficient across all the selected coordinates.

To apply the partially dissipative methods to real datasets, integration orders are needed, which require only minimal domain knowledge. Financial log prices are commonly modeled as integrated log returns, while physical trajectory channels naturally admit position, velocity, and acceleration orders. As an auxiliary diagnostic supporting these choices, we compare persistence across derivative orders on the training split using the mean coordinatewise lag-one correlation, with lower absolute correlation indicating less persistent dynamics. For S&P 500, this correlation decreases from 0.992 for the integrated level to -0.022 for standardized log returns. For crowd trajectories, it decreases from 1.000 for position to 0.921 for velocity and -0.259 for acceleration. For MuJoCo, it decreases from 1.000 to 0.993 and then to 0.550 along the Q_t derivative chain, and from 0.990 to 0.773 along the separately observed V_t chain. The S&P 500 and crowd diagnostics are consistent with selecting their highest derivatives. The MuJoCo derivatives remain moderately persistent, but this does not invalidate the integration-order construction because partial dissipativity is used there as a boundedness and extrapolation prior rather than for stationarity control.

The following table gives the augmented states, observations, and selected subsystems for the datasets. Here Q_t denotes the common direct observations, while the MuJoCo dataset has an additional, separately observed velocity channel V_t with a different integration order:

Dataset	X_t	HX_t	$R_t = MX_t$
High-dimensional Langevin	$(Q_t, \dot{Q}_t)$	Q_t	$\dot{Q}_t$
S&P 500	$(Q_t, \dot{Q}_t)$	Q_t	$\dot{Q}_t$
Crowd	$(Q_t, \dot{Q}_t, \ddot{Q}_t)$	Q_t	$\ddot{Q}_t$
MuJoCo	$(Q_t, \dot{Q}_t, \ddot{Q}_t, V_t, \dot{V}_t)$	(Q_t, V_t)	$(\ddot{Q}_t, \dot{V}_t)$

(167)

For MuJoCo, V_t denotes a separately observed velocity channel rather than the latent derivative $\dot{Q}_t$. We augment the Q_t and V_t channels separately and do not impose the identity $\dot{Q}_t = V_t$.

In this case, the selected diffusion contribution is again:

$$\Gamma^{\sigma_\theta}_{V_{\mathrm{HDLan}}}(t, q, v) = \|\sigma_{v,\theta}(t, q, v)\|_F^2. \tag{168}$$

Using the same architectural bound $\|\sigma_{v,\theta}(t, q, v)\|_F \le S_\sigma$, the affine diffusion bound in Assumption 3.2 again has $\rho = 0$ and $C_\sigma = S_\sigma^2$. As in the previous case, we have set $\beta := c_2 - C_\sigma = 0$, so C_σ is absorbed into c_2 and does not affect the projected drift. We again use a fixed, sufficiently large value of S_σ that is not attained on any of the training, validation, or test trajectories. The same cap is used for the unconstrained, directly constrained, and decomposably constrained Neural SDEs.

B.3.3 Training, Validation, and Test Protocols

Again, both the unconstrained Neural SDEs and the partially dissipative Neural SDEs follow the shared protocol in Appendix B.1.1. Their drift and diffusion networks have two hidden layers of width 64 with LipSwish activations. Unless otherwise stated, they are trained using AdamW with a learning rate 10^{-3}, weight decay 10^{-5}, a `ReduceLROnPlateau` scheduler with a patience of 12, global gradient-norm clipping at 10, batch size 128, and Euler–Maruyama with one solver step per observation interval. Due to the absence of further dynamical priors, we enforce the globally dissipative condition with a quadratic Lyapunov function, and the only additional hyper-parameter for the regularized method is α, which follows the same philosophy as the previous dynamical experiments and is tuned on the same small-value grid for a larger admissible model space. On the high-dimensional Langevin dataset, the validation protocol follows exactly as before in Appendix B.2, while on the real dataset, the metric for the validation selection is the mean forecast MSE on the validation split.

We consider 2 tasks on the real datasets: reconstruction under Gaussian observation noise and forecasting under missing context.

For the reconstruction task, Gaussian noise is injected into the test trajectories to introduce a controlled distribution shift via the noise injection level, and reconstruction is applied to the corrupted test set. An ensemble Kalman filter (Evensen, 1994; Houtekamer & Mitchell, 1998) is used as the common inference tool for all models. The noise levels of the injected Gaussian noise are computed relative to the standard deviation of the validation dataset trajectories. It is applicable to deterministic baselines: the ensemble still represents the uncertainty around the initial state and the observation noise, and the transition remains deterministic. The initial state distribution is taken to be the empirical distribution of the initial noisy observations. Let y_{ikr} and $\widehat{y}^{\mathrm{rec}}_{ikr}$ denote the clean target and reconstructed mean for test trajectory i, observation time k, and observed coordinate r. We report:

$$\mathrm{NRMSE}(\%) := \frac{100}{s_q}\left(\frac{1}{NKd_y}\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{r=1}^{d_y}\left(\widehat{y}^{\mathrm{rec}}_{ikr}-y_{ikr}\right)^2\right)^{1/2}, \tag{169}$$

where N, K, and d_y are the numbers of test trajectories, observation times, and observed coordinates, respectively, and s_q is the pooled standard deviation of the observed validation trajectories.

For the forecasting task, we randomly remove the observation context to introduce a controlled observation shift via the missing percentage, and for each test trajectory, we take the first 70% as the forecast context, while the remaining 30% serves as the forecast target. Ensemble Kalman filters are used to assimilate the available context for all the models by running through the entire forecast context. At each missing data point, the models perform only their transition steps without observation updates. After the context, the models are then run forward to make the predictions. We evaluate the forecast using the continuous ranked probability score (CRPS) (Gneiting & Raftery, 2007) as a unified metric for both models. For a scalar predictive distribution F and target value y, it is defined as:

$$\mathrm{CRPS}(F,y) := \int_{-\infty}^{\infty}\left(F(z)-\mathbf{1}_{\{y\le z\}}\right)^2 dz = \mathbb{E}|Y-y| - \frac{1}{2}\mathbb{E}|Y-Y'|, \tag{170}$$

where Y and Y' are independent draws from F. We average the marginal CRPS over the N test trajectories, H forecast steps, and d_y observed coordinates and report:

$$\mathrm{NCRPS}(\%) := \frac{100}{s_q NHd_y}\sum_{i=1}^{N}\sum_{h=1}^{H}\sum_{r=1}^{d_y}\mathrm{CRPS}(F_{ihr}, y_{ihr}), \tag{171}$$

where s_q is the pooled standard deviation of the observed validation trajectories. For a deterministic forecast, $F_{ihr} = \delta_{\widehat{y}_{ihr}}$ and CRPS reduces exactly to $|\widehat{y}_{ihr} - y_{ihr}|$. Thus, CRPS provides one proper-score scale for both probabilistic and deterministic models.

For comparison on the real datasets, we also consider common deterministic baselines, including Neural ODEs (Chen et al., 2018; Grathwohl et al., 2018), Neural CDEs (Kidger et al., 2020), and an RNN, in addition to the unconstrained Neural SDEs (Liu et al., 2019), as stated in Appendix B.1.1. Due to the deterministic nature of these baselines, they are trained directly under their standard supervised objectives. A Neural ODE is a continuous-time model defined by:

$$dh_t = f_\theta(t,h_t)\,dt, \qquad \widehat{Y}_t = Hh_t, \tag{172}$$

where f_θ uses the same two-hidden-layer, width-64 LipSwish MLP as the Neural SDE drift. The Neural ODE evolves the augmented physical state directly and therefore uses the fixed observation map H, rather than a learned readout network. It is trained by multi-step rollout MSE in the observation space. A Neural CDE is similarly defined but evolves its hidden state along an interpolated observation path Y:

$$dh_t = f_\theta(h_t)\,dY_t, \qquad \widehat{Y}_t = g_\theta(h_t). \tag{173}$$

For the Neural CDE, the initial observation is mapped to a 64-dimensional hidden state by a linear layer followed by LipSwish. The vector field f_θ uses two hidden layers of width 64 with LipSwish activations and a final bounded matrix-valued output of the dimension required to act on dY_t. Its readout is the learned linear map $g_\theta(h) = W_g h + b_g$ from the hidden state to the observed coordinates. It is trained using the teacher-forced one-step MSE. For the RNN, it is defined as:

$$h_{k+1} = F_\theta(h_k, Y_k), \qquad \widehat{Y}_{k+1} = g_\theta(h_{k+1}), \tag{174}$$

where $h_k \in \mathbb{R}^{64}$, F_θ is a two-hidden-layer, width-64 LipSwish MLP applied to (h_k, Y_k), and $g_\theta(h) = W_g h + b_g$ is a learned linear map from the hidden state to the observed coordinates. Depending on the validation-selected configuration, the readout predicts either the next observation or its increment. It is trained by the standard autoregressive rollout MSE.

B.3.4 Results for Experiments in High-Dimensional Applications

In this section, we present the full numerical results in Section 4.2.

Table 6 presents the complete numerical results in Figure 3:

Enforcement	Dimension	NSDE	Joint	Decomposed
Regularized	$D = 4$	0.3445 ± 0.0498	0.2742 ± 0.0687	$\mathbf{0.1687 \pm 0.0378}$
	$D = 8$	0.4148 ± 0.0182	0.2833 ± 0.0121	$\mathbf{0.1559 \pm 0.0573}$
	$D = 16$	0.5392 ± 0.0141	0.4724 ± 0.0603	$\mathbf{0.2097 \pm 0.0539}$
	$D = 32$	0.5201 ± 0.0026	0.5532 ± 0.0499	$\mathbf{0.2517 \pm 0.0983}$
	$D = 64$	0.5315 ± 0.0117	0.6007 ± 0.0054	$\mathbf{0.2436 \pm 0.0484}$
Structural	$D = 4$	0.3445 ± 0.0498	0.2735 ± 0.0123	$\mathbf{0.1958 \pm 0.1045}$
	$D = 8$	0.4148 ± 0.0182	0.3455 ± 0.0141	$\mathbf{0.1329 \pm 0.0241}$
	$D = 16$	0.5392 ± 0.0141	0.4872 ± 0.0162	$\mathbf{0.1706 \pm 0.0150}$
	$D = 32$	0.5201 ± 0.0026	0.4575 ± 0.0119	$\mathbf{0.2007 \pm 0.0490}$
	$D = 64$	0.5315 ± 0.0117	0.4577 ± 0.0091	$\mathbf{0.2025 \pm 0.0228}$

Table 6: Double training horizon test mean MMD^2 across dimensions under different enforcement settings. Values are means $\pm$ standard deviations over three random seeds.

Table 7 reports the complete numerical results for the reconstruction task in Figure 4.

Dataset	Model	Noise: 100%	200%	300%
MuJoCo	PDNSDE (ours)	$\mathbf{79.9 \pm 11.1}$	$\mathbf{109.3 \pm 7.8}$	$\mathbf{147.0 \pm 1.7}$
	NSDE	91.3 ± 3.9	115.1 ± 2.4	155.4 ± 8.4
	Neural ODE	101.6 ± 5.0	155.6 ± 4.2	208.4 ± 11.9
	RNN	142.0 ± 5.1	234.1 ± 3.6	301.8 ± 5.8
	Neural CDE	124.1 ± 4.7	279.5 ± 6.9	384.6 ± 41.8
Crowd	PDNSDE (ours)	$\mathbf{43.4 \pm 0.6}$	$\mathbf{85.5 \pm 1.0}$	138.9 ± 4.7
	NSDE	50.5 ± 0.9	90.0 ± 1.5	$\mathbf{136.6 \pm 5.8}$
	Neural ODE	52.9 ± 1.2	126.4 ± 0.7	214.0 ± 5.7
	RNN	84.7 ± 1.1	188.7 ± 1.3	290.7 ± 7.4
	Neural CDE	71.5 ± 0.6	155.4 ± 4.2	244.9 ± 9.0
S&P 500	PDNSDE (ours)	$\mathbf{45.7 \pm 1.7}$	$\mathbf{81.5 \pm 1.6}$	$\mathbf{128.8 \pm 5.5}$
	NSDE	68.3 ± 2.5	113.3 ± 4.0	162.9 ± 2.4
	Neural ODE	128.5 ± 3.8	217.1 ± 3.8	315.5 ± 6.3
	RNN	106.4 ± 0.3	194.6 ± 1.1	294.2 ± 3.7
	Neural CDE	108.3 ± 0.8	198.3 ± 3.0	295.1 ± 2.9

Table 7: Reconstruction NRMSE (%) under increasing Gaussian observation noise. Noise magnitudes are expressed as percentages relative to the standard deviation of the observed validation trajectories.

Table 8 presents the complete numerical results for the forecasting task in Figure 4.

Dataset	Model	Missing: 0%	25%	50%	75%
MuJoCo	PDNSDE (ours)	$\mathbf{29.7 \pm 0.9}$	$\mathbf{31.2 \pm 2.1}$	$\mathbf{33.3 \pm 2.2}$	$\mathbf{37.8 \pm 4.1}$
	NSDE	31.6 ± 1.7	33.0 ± 2.1	34.7 ± 1.3	$\mathbf{37.8 \pm 3.6}$
	Neural ODE	55.2 ± 1.6	55.0 ± 4.0	58.8 ± 5.9	70.3 ± 3.5
	RNN	78.9 ± 4.9	79.0 ± 4.9	79.0 ± 4.8	78.8 ± 4.8
Crowd	PDNSDE (ours)	$\mathbf{4.8 \pm 0.1}$	$\mathbf{5.1 \pm 0.0}$	$\mathbf{5.7 \pm 0.2}$	$\mathbf{7.8 \pm 0.5}$
	NSDE	22.7 ± 0.8	23.4 ± 0.8	24.3 ± 0.8	25.6 ± 0.6
	Neural ODE	7.1 ± 0.0	7.4 ± 0.0	8.0 ± 0.1	9.5 ± 0.1
	RNN	7.0 ± 0.1	7.5 ± 0.2	8.4 ± 0.2	10.7 ± 0.2
S&P 500	PDNSDE (ours)	$\mathbf{31.4 \pm 1.4}$	$\mathbf{31.7 \pm 1.2}$	$\mathbf{32.1 \pm 1.3}$	$\mathbf{33.0 \pm 1.2}$
	NSDE	34.4 ± 1.1	35.0 ± 1.7	35.4 ± 1.4	37.1 ± 2.6
	Neural ODE	52.3 ± 2.9	57.0 ± 4.8	64.7 ± 0.6	74.9 ± 0.4
	RNN	37.1 ± 0.1	37.9 ± 0.1	38.9 ± 0.2	41.7 ± 0.2

Table 8: Forecast NCRPS under increasing context missingness.

B.4 Additional Diagnostics and Ablation Studies

B.4.1 Regularization Diagnostics

For the regularized methods, the dissipativity residual is evaluated at sampled collocation points with a sampling distribution ν_{sample}. Except for the geometry-adapted shell samplers described below, it factorizes as follows:

$$\nu_{\mathrm{sample}}(dt, dx) = \mathrm{Unif}[0, T_{\mathrm{col}}](dt)\, \nu_x(dx), \tag{175}$$

where ν_x is a uniform distribution over a some bounded domain. For a shell sampler, we first draw

$$G \sim \mathcal{N}(0, I_d), \qquad U = \frac{G}{\|G\|}, \qquad X = C + RU, \tag{176}$$

so that U is uniform on the unit sphere. New collocation points are sampled at every optimization update. The distributions used in the experiments are summarized in Table 9:

Dataset	State sampling distribution ν_x	T_{col}	Points/update
Shear	$2 \le \sqrt{x^2 + (y + kx^2)^2} \le 32$; uniform in area	5	4096
Langevin	$\mathrm{Unif}([-4, 4]^2)$ over (q, v)	5	512
Lorenz–63	$60 \le \|x - (0, 0, 38)\| \le 700$; log-uniform radius	5	1024
High-dimensional Langevin	$\mathrm{Unif}([-4, 4]^{2D})$ over (q, v)	10	512
S&P 500	$\mathrm{Unif}([-4, 4]^{312})$ over $(q, \dot{q})$	200	512
Crowd	Training-scaled anisotropic box in $\mathbb{R}^{36}$	60	512
MuJoCo	$\mathrm{Unif}([-1, 1]^{35})$ over the augmented state	100	512

Table 9: Collocation distributions used by PINN-style dissipativity regularization.

We compare the ranges of the regularization domains in Table 9 with the initialization ranges used in the far-field experiments. As shown in Table 10, all far-field initializations are covered by the corresponding regularization domains.

Dataset	Test condition	Initial $\sqrt{V}$ interval	Regularization shell
Shear	$4\times$	$[3.736, 3.932]$	$[2, 32]$
	$10\times$	$[9.339, 9.831]$	$[2, 32]$
	$20\times$	$[18.678, 19.661]$	$[2, 32]$
Stochastic Lorenz–63	$2\times$	$[60.790, 63.989]$	$[60, 700]$
	$4\times$	$[121.579, 127.978]$	$[60, 700]$
	$10\times$	$[303.949, 319.946]$	$[60, 700]$
	$20\times$	$[607.897, 639.892]$	$[60, 700]$

Table 10: Comparison between the far-field initialization ranges and the corresponding regularization domains. All reported initialization ranges are covered by the regularization domains.

We now report the residual reduction in Table 11 as diagnostics for the regularization approaches. Consider an independently sampled evaluation region $\{(t_b, x_b)\}_{b=1}^{N}$ and measure the threshold-free mean positive residual:

$$\overline{\varphi}_{+} := \frac{1}{N} \sum_{b=1}^{N} [\varphi(t_b, x_b)]_{+} . \tag{177}$$

We report the relative residual reduction between the regularized method and its corresponding unconstrained model:

$$\text{Reduction}(\%) := 100 \left(1 - \frac{\overline{\varphi}_{+,\text{reg}}}{\overline{\varphi}_{+,\text{unc}}} \right) . \tag{178}$$

Dataset	Median	Minimum	Maximum
S&P 500	100.0	100.0	100.0
Crowd	100.0	99.3	100.0
MuJoCo	99.6	96.2	99.9
Shear	89.7	86.7	97.7
Lorenz–63	100.0	100.0	100.0
Langevin	100.0	100.0	100.0

Table 11: Reduction (%) in the mean positive dissipativity residual under PINN-style regularization.

B.4.2 Solver Sensitivity and Choice

This section presents our experiments on solver sensitivity, and consistent with (Oh et al., 2024), we have found the performance to be largely insensitive to the solvers, as reported in Table 12. Thus, we use the EM solver for the lowest computational cost.

Model	Training solver	EM	Drift-Heun	Itô SRK
NSDE	EM	0.5370 ± 0.0139	0.5360 ± 0.0143	0.5360 ± 0.0142
	Drift-Heun	0.5370 ± 0.0139	0.5360 ± 0.0143	0.5360 ± 0.0142
	Itô SRK	0.5370 ± 0.0139	0.5360 ± 0.0142	0.5360 ± 0.0141
PDNSDE-Struct	EM	0.1160 ± 0.0084	0.1120 ± 0.0081	0.1110 ± 0.0081
	Drift-Heun	0.1150 ± 0.0084	0.1110 ± 0.0080	0.1100 ± 0.0080
	Itô SRK	0.1150 ± 0.0082	0.1110 ± 0.0079	0.1110 ± 0.0079
PDNSDE-Reg	EM	0.0748 ± 0.0027	0.0738 ± 0.0024	0.0729 ± 0.0024
	Drift-Heun	0.0735 ± 0.0023	0.0725 ± 0.0019	0.0716 ± 0.0019
	Itô SRK	0.0748 ± 0.0032	0.0738 ± 0.0028	0.0728 ± 0.0028

Table 12: Held-out path MMD^2 for the complete crossed training-solver and evaluation-solver comparison on high-dimensional Langevin at $D = 64$ (lower is better). Values are means $\pm$ sample standard deviations over three random seeds.