NEURAL AUTONOMOUS DIFFERENTIAL EQUATIONS: INTEGRATION-FREE LEARNING FOR TIME-INVARIANT DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of learning autonomous dynamics from observed trajectories. Neural ODEs, which model the vector field with a neural network, have emerged as a popular framework for this task. These methods backpropagate through adaptive ODE solvers, which makes them computationally expensive and numerically sensitive. Recently proposed variants accelerate training, but either weaken the ability to constrain autonomous dynamics, introduce periodicity assumptions, or require trajectory truncation. We propose Neural Autonomous Differential Equations (NADE), a simple solver-free training method. NADE separates trajectory reconstruction from vector-field estimation by (i) fitting the trajectory using differentiable splines, and (ii) training the velocity field using analytic derivatives of the fitted trajectory at sampled states. At inference time, the learned autonomous field can be integrated with any standard numerical solver. This preserves the inductive bias of autonomous dynamics while avoiding solver-based training. We evaluate NADE on multistable, periodic, and chaotic systems. Under equal wall-clock training budgets, it achieves substantially faster convergence and lower training error than standard Neural ODEs and accelerated variants.

1 INTRODUCTION

Ordinary differential equations (ODEs) are a standard mathematical framework for describing how quantities change continuously in time, and are used widely across the natural and engineering sciences (Strogatz, 2018; Perko, 1996). Very often the governing equations are unknown, but trajectories from the system can be observed. In this case, a central problem of interest is *learning* the underlying dynamics, so as to enable simulation, forecasting, and scientific inference downstream (Brunton et al., 2016; Raissi et al., 2019). In this work, we focus on autonomous systems,

$$\dot{x}(t) = f(x(t)), \qquad x(t) \in \mathbb{R}^d, \tag{1}$$

where the instantaneous rate, the ODE right-hand side (RHS), depends on the state alone, without an explicit dependence on time. Assuming such time-invariance is natural whenever external forcing is absent or absorbed into an augmented state, and it sharply constrains what a learned model is allowed to do: a single time-invariant field must explain every segment of every trajectory. Yet *autonomy* is not the same as *simplicity* in the time domain. A paradigmatic example is the Lorenz system, whose trajectories are chaotic and sensitive to initial conditions while still arising from a compact three-dimensional autonomous ODE (Lorenz, 1963). In challenging settings such as these, autonomy provides an especially important inductive bias: rather than fitting or memorizing complex temporal patterns directly (which can be brittle under chaos), the model need only explain every observed state and velocity through a single stationary vector field.

Recent approaches to the problem of learning ODEs include Neural Ordinary Differential Equations (NODE), Neural Flow Initial Value Problems (NFLOW), and Fourier Neural Ordinary Differential Equations (FNODE). NODE learns a continuous-time vector field by embedding numerical integration in the training loop (Chen et al., 2018); this preserves ODE structure and can be used to encode autonomous constraints, but is computationally expensive and can be numerically sensitive. NFLOW directly learns“elapsed-time" flow maps from state and time to future state (Biloš et al., 2021); this enables integration-free training and inference, but enforcing stationarity in flow maps is known to be

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.

challenging, making it unsuitable for autonomous systems. FNODE instead estimates derivatives through Fourier analysis and regresses a dynamics model to those targets (Li et al., 2024); like NFLOW this enables integration-free training, but introduces finite-window spectral assumptions that can be restrictive on non-periodic segments. Our aim is to combine the strengths of these methods while avoiding their main drawbacks. Specifically, we seek a method that (i) directly encodes autonomy in the model class, (ii) avoids numerical integration during training, and (iii) does not rely on restrictive spectral assumptions. The goal is fast training without sacrificing stable and accurate roll-outs on challenging autonomous systems.

1.1 Our approach and contributions

Our approach to learning autonomous dynamics, which we call *Neural Autonomous Differential Equations* (NADE), begins from a simple observation about equation 1. If, for any observed trajectory of states $(x_{t_k})_{k=1}^n$ at observation times $(t_k)_{k=1}^n$,we knew the associated velocities, $(\dot{x}_{t_k})_{k=1}^n$, then learning f would reduce to ordinary supervised regression. In particular, one could simply fit $\dot{x} \approx f_\theta(x)$ using all pairs across observation times and trajectories, thus keeping numerical integration out of the optimization loop.

Of course in practice we observe states without derivatives. The remedy is to (i) smooth each trajectory in time by learning a differentiable surrogate $s(t) \approx x_t$ per trajectory (here, using cubic splines), (ii) take analytic derivatives at each observation time and trajectory $(\dot{s}(t))$, and (iii) treat $(x_k, \dot{s}_k)$ as supervised targets for the regression idea outlined above. Training f_θ is thus reduced to regression on imputed state-velocity pairs. The learned f_θ can then be integrated (only) at inference time to predict trajectories from new initial conditions. Later we exploit the fact that this vector field can also be learned component-wise. For d-dimensional trajectory observations $x_t := (x_{1,t}, \ldots, x_{d,t})$, the velocity coordinates $(\dot{s}_1(t), \ldots, \dot{s}_d(t))$ provide separate supervised targets. This allows the problem to be broken into dimension-wise fits, which can be helpful when the full system has many state variables or mixed time scales. Figure 1 illustrates this pipeline from observed samples to roll-out under the learned dynamics, for the case where we observe a single trajectory.

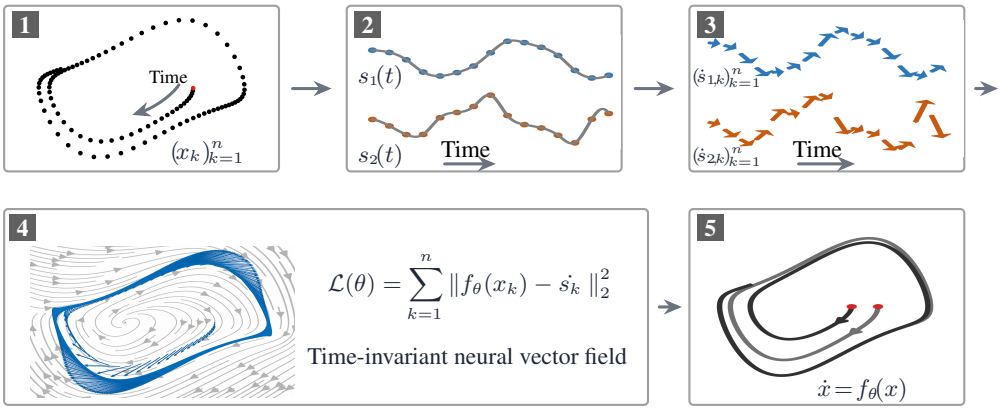


Figure 1: NADE pipeline: (1) The available data consist of discrete observations at observation times t_k, denoted by x_k. (2) We then fit each observed coordinate i with a cubic spline $s_i(t)$ that passes through all observed points, such that $s_i(t_k) = x_{i,k}$; (3) we compute the spline derivatives at the observation times, $\dot{s}_{i,k}$; (4) the resulting state–velocity pairs are then used to supervise a state-only velocity map f_θ, which is a neural network; (5) once the dynamics are learned, rollouts from new initial conditions and over new time intervals are obtained by integrating $f_\theta(x)$.

We evaluate NADE on four core benchmarks: bistable damped Duffing (Strogatz, 2018; Perko, 1996), periodic Lotka–Volterra (Murray, 2002), Van der Pol dynamics with nonuniform local rates of evolution (Hairer & Wanner, 1996), and chaotic Lorenz (Lorenz, 1963). These systems are challenging because they expose distinct failure modes in existing approaches to learning autonomous dynamics, including global spectral bias in derivative targets, instability in time-conditioned flow-map learning, and high training cost. NADE addresses these issues by using spline-based derivative targets that are local in time, thereby avoiding the imposition of a global spectral prior (De Boor,

1978; Li et al., 2024). It further learns a direct state-to-velocity map, rather than a time-conditioned mapping (Biloš et al., 2021), which preserves autonomy by construction. Across these benchmarks, we demonstrate that NADE achieves higher accuracy while converging substantially faster than the compared approaches.

In summary, our main contributions are:

1. a solver-free training procedure for autonomous dynamics that avoids backpropagation through numerical ODE solvers and scales efficiently to long trajectories.
2. a simple way to enforce autonomy by learning a single time-invariant vector field, without relying on periodicity assumptions or trajectory truncations.
3. a consistency result for the two-stage estimator in a well-specified parametric setting, establishing C^1 convergence of the fitted trajectory and vector-field consistency along the true orbit, with uniform rollout convergence under regularity assumptions.
4. an empirical evaluation showing that NADE improves vector-field accuracy and trajectory fidelity over Neural ODEs and scalable approximations across multistable, periodic, and chaotic systems.

The remainder of the paper is organized as follows. Section 2 gives prior-method context. Section 3 formalizes the NADE training pipeline and objective. Section 4 states a basic consistency result for the two-stage estimator. Section 5 reports comparisons on bistable Duffing, Van der Pol, Lotka–Volterra, and Lorenz benchmarks under matched protocols. finally, section 6 discusses empirical takeaways and limitations.

2 Background and Related Work

We first compare the main methods that we aim to build on, and discuss wider related work.

Neural ODE (NODE). The dominant deep-learning route to continuous-depth dynamics culminates in neural ODEs (NODEs), which parameterize a vector field and evaluate predictions by numerically solving an initial value problem (He et al., 2016; Lu et al., 2018; Chen et al., 2018). This alignment with explicit ODE structure is desirable, but training is computationally expensive as gradients are obtained by backpropagating through numerical ODE solvers (or equivalent adjoint sensitivity solves), and for sensitive dynamics like chaotic systems these costs and numerical fragilities typically become more severe (Hairer et al., 1993; Li et al., 2020; Finlay et al., 2020; Zhuang et al., 2020; Kim et al., 2021; Caldana & Hesthaven, 2025; Loya et al., 2025; Hairer & Wanner, 1996). Many extensions improve expressivity or data handling (Dupont et al., 2019; Rubanova et al., 2019; Kidger et al., 2020), yet in many practical workflows a solver still sits inside optimization.

Neural flow (NFLOW). Neural flows (NFLOW) reduce solver-in-the-loop cost by learning maps from initial condition and elapsed time directly to states (Biloš et al., 2021). For autonomous systems, however, the object of interest is one stationary field $f(x)$ whose evolution is composition-consistent across time: stepping by t_1 and then t_2 should agree with stepping once by $t_1 + t_2$. Flexible flow maps need not satisfy this property unless it is enforced by regularization (Biloš et al., 2021). NADE enforces autonomy directly through an explicit state-only field $f_\theta(x)$.

Fourier neural ODE (FNODE). Fourier neural ODEs (FNODEs) similarly avoid inner solves by estimating derivatives through Fourier analysis and regressing to those targets (Li et al., 2024). This is close in spirit to the method proposed in this work, as we elaborate later, but Fourier differentiation on a finite record has two important limitations. First, it implicitly assumes periodic continuation beyond the observation window. When the observed window is not approximately periodic, as can occur for chaotic trajectories, a mismatch between the endpoints introduces an artificial discontinuity in the periodic extension, leading to spurious boundary oscillations and biased derivative estimates near the window edges. Second, even for periodic trajectories, sharp or rapidly varying local features require many Fourier modes to resolve accurately. Truncating the spectrum can smooth out these features, while retaining more high-frequency modes can introduce oscillatory artifacts around them. Thus, Fourier differentiation imposes a global spectral representation that can be poorly suited to strongly localized changes in the dynamics.

Wider related work Our viewpoint is complementary to methods that encode additional physics or coordinates before learning f, including Hamiltonian neural networks (Greydanus et al., 2019), physics-informed neural networks and residuals (Raissi et al., 2019), sparse identification of nonlinear dynamics (Brunton et al., 2016), and Koopman-operator perspectives on data-driven dynamics (Brunton et al., 2022). Those lines ask how much inductive structure can be built into f_θ; NADE asks how far one can get with an unstructured autonomous field when the estimator itself enforces autonomy and avoids solver-in-the-loop training.

3 Integration-Free Learning for Time-Invariant Dynamics

For autonomous dynamics, the vector field is time-invariant: when the dynamics are unknown, the object to learn is therefore the $x \mapsto \dot{x}$ map, with no t involved. The method we introduce here, NADE, does this by first estimating the velocities from smooth trajectory fits to the observations, then learning the map from state–velocity pairs. This will provide a fast method to learn the dynamics as it does not require solver-in-the-loop training.

3.1 Spline-based Recovery of Observation Derivatives

Suppose we are given data in the form of observed trajectories. For trajectory l, we denote its observation times by $(t_k)_{k=1}^{n_l}$ and the corresponding observation states by $x_k^{(l)} \in \mathbb{R}^d$, with components denoted by $x_{i,k}^{(l)} \in \mathbb{R}$ for $i \in \{1, \dots, d\}$. Different trajectories may therefore contain different numbers of observations n_l; throughout, t_k simply denotes the observation times of the trajectory under consideration. We first convert these discrete observations into continuous-time objects. For each trajectory l and coordinate i, we construct a cubic spline which interpolates the observed data,

$$s_i^{(l)}(t_k) = x_{i,k}^{(l)}. \tag{2}$$

Because spline segments are piecewise polynomials, their derivatives are available in closed form. We denote the derivative at each observation time by

$$\dot{s}_{i,k}^{(l)} := \frac{d}{dt} s_i^{(l)}(t_k), \tag{3}$$

and treat these values as surrogates for the unknown velocities $\dot{x}_i^{(l)}(t_k)$. Cubic splines for clean observations are a natural choice here because they are local, fast to fit, and avoid imposing a global spectral prior. For noisy observations, we use a smoothing spline with the same analytic-derivative construction (Appendix I).

3.2 Time-Invariant Learning for Neural Velocity Fields

The spline derivatives yield supervised state–velocity pairs

$$\left\{ \left(x_k^{(l)}, \dot{s}_k^{(l)} \right) \right\}_{l,k}, \tag{4}$$

from which we learn the autonomous vector field. We consider two parameterizations. In the coordinate-wise formulation (NADE-c), each network $f_{\theta_i} : \mathbb{R}^d \to \mathbb{R}$ receives the full state but predicts only the ith velocity component. To account for differences in scale across coordinates, we introduce learnable positive factors α_i and minimize

$$\mathcal{L}_i(\theta_i, \alpha_i) = \sum_{l=1}^{L} \sum_{k=1}^{n_l} \left(\frac{f_{\theta_i}(x_k^{(l)})}{\alpha_i} - \dot{s}_{i,k}^{(l)} \right)^2. \tag{5}$$

The learned NADE-c field is therefore

$$f_\theta(x) = \begin{bmatrix} \alpha_1^{-1} f_{\theta_1}(x) \\ \vdots \\ \alpha_d^{-1} f_{\theta_d}(x) \end{bmatrix}, \tag{6}$$

which depends only on the state.

Alternatively, the joint formulation (NADE-j) uses a single network $f_\theta : \mathbb{R}^d \to \mathbb{R}^d$ and minimizes

$$\mathcal{L}(\theta) = \sum_{l=1}^{L} \sum_{k=1}^{n_l} \left\| f_\theta(x_k^{(l)}) - \dot{s}_k^{(l)} \right\|_2^2. \tag{7}$$

Both formulations learn the same state-to-velocity map; they differ only in its parameterization. NADE-j learns all velocity components jointly, whereas NADE-c separates the output coordinates while retaining cross-coordinate dependence through the full-state input. The latter can be useful in higher dimensions because smaller coordinate networks can be trained independently and in parallel, and coordinate-specific scaling can aid optimization when velocity components differ substantially in magnitude or local rate of variation.

3.3 Inference with NADE

At deployment time, we generate trajectories by integrating the learned system $\dot{x}(t) = f_\theta(x(t))$, using a standard explicit method such as Runge–Kutta. In this way, derivative estimation is separated from numerical simulation: training operates purely on supervised state and velocity pairs, and integration is performed only after the vector field has been learned. As with other data-driven field estimators, accuracy is strongest in regions of $\mathbb{R}^d$ that are well-covered by the observed trajectories. Details on the optimization and validations are given in AppendixE.

4 Theoretical Analysis

Having derived NADE, we now give a basic consistency result for the two-stage procedure in a parametric setting. The result formalizes the following intuition: if the first stage asymptotically recovers the underlying trajectory, then, under smoothness conditions, its derivatives provide asymptotically valid targets for the second-stage vector-field regression. This in turn propagates to consistency of the full procedure under regularity conditions.

In this section, we consider a single trajectory observed over a fixed time interval $\mathcal{T} = [0, \tau]$. The true trajectory $s_0 : \mathcal{T} \to \mathbb{R}^d$ is assumed to solve the autonomous system

$$\dot{s}_0(t) = f_{\theta_0}(s_0(t)), \qquad t \in \mathcal{T}, \tag{8}$$

for some parameter $\theta_0 \in \Theta_f \subset \mathbb{R}^p$. We observe this trajectory at equispaced times $t_{k,n} := \frac{k\tau}{n}$, $k = 0, \ldots, n$, through possibly noisy measurements

$$Y_{k,n} = s_0(t_{k,n}) + \varepsilon_{k,n}, \qquad k = 0, \ldots, n. \tag{9}$$

Thus, the only source of randomness in our analysis comes from measurement errors; the sampling times are fixed. We assume $\mathbb{E}[\varepsilon_{k,n}] = 0$ and $\sup_{k,n} \mathbb{E}\|\varepsilon_{k,n}\|^2 < \infty$.

Let $\{s_\eta : \eta \in \Theta_s \subset \mathbb{R}^l\}$ be a parametric trajectory class, and let $\{f_\theta : \theta \in \Theta_f\}$ be a parametric class of autonomous vector fields. Recall that the first stage of NADE fits the trajectory by least squares,

$$\hat{\eta}_n \in \operatorname*{argmin}_{\eta \in \Theta_s} \frac{1}{n+1} \sum_{k=0}^{n} \|Y_{k,n} - s_\eta(t_{k,n})\|^2 \tag{10}$$

while the second stage trains the vector field against the derivatives of the fitted trajectory:

$$\hat{\theta}_n \in \operatorname*{argmin}_{\theta \in \Theta_f} \frac{1}{n+1} \sum_{k=0}^{n} \|f_\theta(s_{\hat{\eta}_n}(t_{k,n})) - \dot{s}_{\hat{\eta}_n}(t_{k,n})\|^2 . \tag{11}$$

Under several regularity conditions stated in Assumption C.1 of Appendix C (i.e., compactness of the parameter spaces, continuity of $\eta \mapsto s_\eta$ into $C^1(\mathcal{T}; \mathbb{R}^d)$, regularity of the vector-field class, and an identifiability condition ensuring that the observed trajectory is sufficiently informative to identify θ_0 within Θ_f), we obtain the following consistency result.

Theorem 4.1 (Parametric consistency of NADE). *Suppose the regularity and identifiability conditions in Theorem C.1 hold. Then*

$$\hat{\eta}_n \to_p \eta_0, \qquad \hat{\theta}_n \to_p \theta_0. \tag{12}$$

Moreover, the fitted trajectory and its derivative are uniformly consistent:

$$\|s_{\hat{\eta}_n} - s_0\|_\infty + \|\dot{s}_{\hat{\eta}_n} - \dot{s}_0\|_\infty \to_p 0. \tag{13}$$

Consequently, the learned vector field is consistent along the true trajectory:

$$\int_0^\tau \|f_{\hat{\theta}_n}(s_0(t)) - f_{\theta_0}(s_0(t))\|^2 \, dt \to_p 0. \tag{14}$$

Theorem C.2 in Appendix C extends the last result to integrated rollouts: under uniform local consistency of the learned vector field, the trajectories generated by integrating NADE converge uniformly to the true trajectory.

This result establishes consistency of NADE's two-stage estimator in the parametric setting considered here. If the fitted trajectory converges in C^1, then its derivatives become valid supervision for learning the autonomous vector field along the observed path. The second stage therefore approaches the supervised regression problem we would use if true state-velocity pairs were available. This recovery is limited to regions covered by the data, as captured by the identifiability condition. The rollout result additionally requires local Lipschitz control and uniform consistency near the true orbit, which together yield uniform convergence of integrated trajectories.

5 Experiments

We benchmark NADE against baseline methods on systems (details in Appendix F) for which the exact velocity field is available. This enables evaluation using MSE_{vec}, i.e., the error between the learned and true vector fields (Appendix K); importantly, this metric is used only for evaluation, not training. The main benchmarks are intentionally more demanding than short-horizon trajectory fitting, where most baseline methods are expected to perform well (as discussed further in Appendix B); they include long-horizon prediction, extrapolation beyond the training interval, and generalisation to unseen initial conditions.

While our main focus is on clean, low-dimensional systems, we also report results with corrupted observations and a high-dimensional chaotic system in Appendices I and J, respectively. These results demonstrate that, even in these more challenging settings, NADE learns substantially faster while achieving higher accuracy.

Evaluation Protocol. We compare all methods under matched 30-minute wall-clock budgets in the experiments reported below. Each hyperparameter candidate receives the same wall-clock stopping budget. Within each trial, the checkpoint with the lowest native training objective is retained, and hyperparameters are selected using each method's native validation objective on a separate $0.95x_0$ validation trajectory. The ground-truth vector-field error, MSE_{vec}, is used only for evaluation. Tables aggregate results over three seeds. Full hyperparameter grids, method-specific validation objectives, and systematic scaled-initial-condition tests are provided in Appendix E, and the evaluation metrics are defined in Appendix K.

Figures 2–5 compare NADE with the baseline methods across benchmarks. The accompanying tables report the evaluation-only vector-field error MSE_{vec}, the held-out trajectory Wasserstein distance (see AppendixK) over $[T, 2T]$, and the time required to reach the indicated vector-field-error threshold. The vector-field panels visualize the learned dynamics using blue velocity vectors, with the ground-truth in black and the learned streamlines in gray. The close agreement between the learned and ground-truth velocity vectors illustrates the accuracy of the recovered vector field. We also show rollouts from unseen initial conditions beyond the training horizon. Blue denotes predictions within $[0, T]$, while orange denotes extrapolation beyond T. Figure 6 shows the training histories of all methods across all systems.

5.1 Bistable dynamics

We evaluate the damped Duffing oscillator, which has two attractors. The long-term dynamics are determined by the initial condition: trajectories evolve toward the attractor associated with the basin

containing that initial condition.

$$\dot{x}_1 = x_2, \qquad \dot{x}_2 = x_1 - x_1^3 - \delta x_2. \tag{15}$$

Here, δ is a damping coefficient. The two training trajectories start in opposite basins, testing whether one learned autonomous field remains coherent across both regions. Figures 2 and 8 show that both NADE variants maintain low vector-field error and accurate held-out rollouts in the two basins. For NFLOW and FNODE, we display the trajectories for the best-performing seed. However, the large Wasserstein-distance errors for the other seeds suggest that the learned dynamics do not consistently drive the trajectory toward the correct attractor.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{4.2\times10^{-6}\pm5.7\times10^{-9}}$	$\mathbf{4.2\times10^{-6}\pm6.2\times10^{-10}}$	0.38 ± 0.12	0.036 ± 0.0019	0.065 ± 0.015
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	$\mathbf{0.083\pm0.031}$	$\mathbf{0.063\pm0.018}$	$1.6\times10^{3}\pm4.9\times10^{2}$	$4.3\times10^{2}\pm9.1\times10^{2}$	$2.3\times10^{3}\pm1.2\times10^{3}$
$t_{\mathrm{MSE}_{\mathrm{vec}}\leq0.01}$	$\mathbf{2.0\pm0.83}\ s$	$\mathbf{1.4\pm0.11}\ s$	–	$0.14\ s$	–

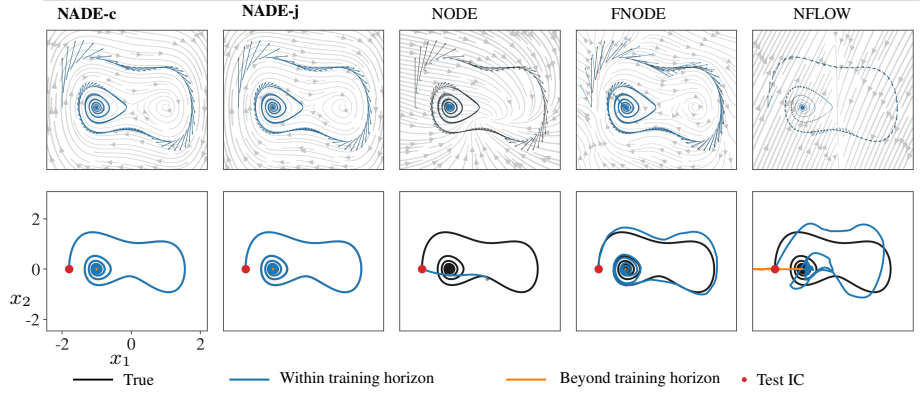


Figure 2: Duffing bistable comparison under the same training budget and comparable model capacity. The table reports vector-field error, trajectory Wasserstein distance over $[T, 2T]$, and the time required to reach the indicated vector-field accuracy. The NADE variants reach the low-error regime rapidly and maintain accurate long-horizon dynamics, while most baselines do not attain the same vector-field-error threshold within the training budget. The lower row shows a held-out rollout from one basin of attraction; results for the complementary basin are shown in Fig. 8.

5.2 Periodic Systems

We consider two systems with qualitatively different periodic dynamics. Van der Pol has an attracting limit cycle with nonuniform local rates of evolution, whereas the classical Lotka–Volterra system has an initial-condition-dependent family of neutrally stable closed orbits. Together, they test both recovery of an attracting periodic orbit and generalization across neighboring periodic orbits.

Van der Pol. We consider the Van der Pol oscillator

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = \mu(1 - x_1^2)x_2 - x_1. \tag{16}$$

At the $\mu = 1$ setting used here, the system exhibits an attracting limit cycle with nonuniform local rates of evolution: the magnitude and direction of the velocity vary substantially along the orbit. This provides a periodic benchmark for assessing whether the learned vector field remains accurate across regions with different local rates of change, as shown in Figure 3.

Lotka–Volterra. We next consider the classical two-dimensional Lotka–Volterra system

$$\dot{x}_1 = \alpha x_1 - \beta x_1 x_2, \quad \dot{x}_2 = -\gamma x_2 + \delta x_1 x_2. \tag{17}$$

Unlike Van der Pol, the classical Lotka–Volterra system has a family of neutrally stable closed orbits around the coexistence fixed point, with the particular orbit determined by the initial condition. Because training observes only one orbit, a rollout from a new initial condition tests whether the learned vector field generalizes away from the training trajectory to a neighboring, previously unobserved orbit. Small vector-field errors can cause trajectories to drift across these closed orbits, making accurate long-horizon recovery particularly sensitive to off-trajectory field accuracy. As shown in Figure 4, the NADE variants recover the local vector-field structure closely having smallMSE_{vec}, with NADE-j most closely preserving the unseen closed orbit in the representative held-out rollout.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{2.5\times10^{-4}\pm8.2\times10^{-6}}$	$\mathbf{2.7\times10^{-4}\pm6.2\times10^{-6}}$	3.7 ± 1	$0.029\pm3.9\times10^{-4}$	1.8 ± 0.21
$\mathrm{Wasserstein}_{\mathrm{traj}}[T,2T]$	$\mathbf{1.9\times10^{-3}\pm8.2\times10^{-5}}$	$\mathbf{2.1\times10^{-3}\pm3.9\times10^{-4}}$	$0.87\pm1.2\times10^{-4}$	0.61 ± 0.44	5.4 ± 4.4
$t_{\mathrm{MSE}_{\mathrm{vec}}\leq0.01}$	$\mathbf{5.3\pm0.54}\,\boldsymbol{s}$	$\mathbf{1.1\pm0.63}\,\boldsymbol{s}$	–	–	–

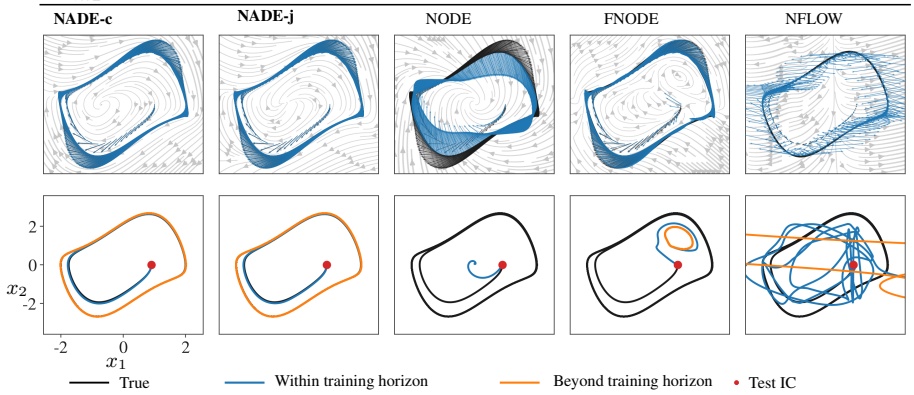


Figure 3: Van der Pol comparison under the same training budget and parameter-matched setting. The table reports vector-field error, long-horizon trajectory error over $[T, 2T]$, and the time required to reach the indicated vector-field accuracy. Both NADE variants reach the low-error regime substantially earlier than the baselines, which do not attain the same threshold, even under the full training budget. The upper row shows the learned vector fields and the lower row shows a representative held-out rollout.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{1.3\times10^{-4}\pm4.8\times10^{-6}}$	$\mathbf{1.6\times10^{-4}\pm1.5\times10^{-6}}$	7.9 ± 3.1	0.56 ± 0.074	4.6 ± 0.54
$\mathrm{Wasserstein}_{\mathrm{traj}}[T,2T]$	$\mathbf{0.55\pm0.19}$	$\mathbf{0.47\pm0.25}$	0.78 ± 0.03	1.9 ± 0.4	12 ± 8
$t_{\mathrm{MSE}_{\mathrm{vec}}\leq0.01}$	$\mathbf{8.2\pm4.0}\,\boldsymbol{s}$	$\mathbf{0.67\pm0.71}\,\boldsymbol{s}$	–	–	–

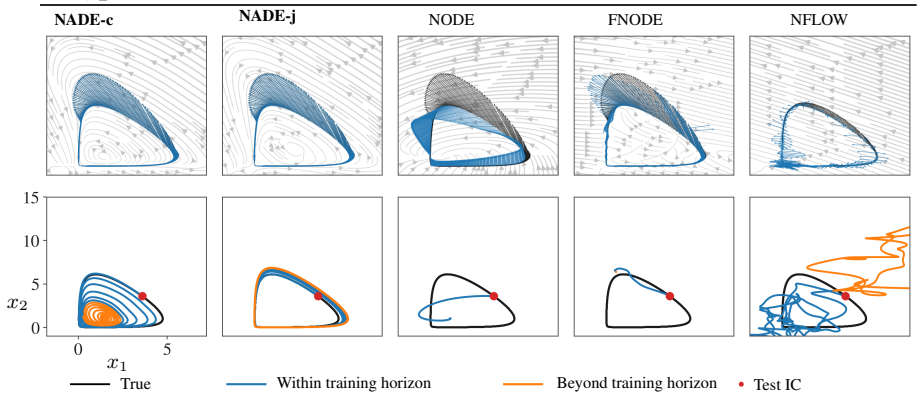


Figure 4: Lotka–Volterra comparison under a matched wall-clock training budget and comparable model capacity. The table reports vector-field error, trajectory Wasserstein distance over $[T, 2T]$, and the first time at which the vector-field error reaches the indicated threshold. NADE-c and NADE-j reach this low-error regime substantially earlier, while the other methods do not reach the same threshold even within the full available training budget. The upper row visualizes the learned vector fields, and the lower row shows a representative rollout from a held-out initial condition.

5.3 Chaotic Dynamics

We evaluate chaotic dynamics using the Lorenz system:

$$\dot{x}_1 = \sigma(x_2 - x_1), \qquad \dot{x}_2 = x_1(\rho - x_3) - x_2, \quad \dot{x}_3 = x_1 x_2 - \beta x_3. \tag{18}$$

The Lorenz system combines nonlinear folding with sensitive dependence on initial conditions: a single stationary vector field $f(x)$ must consistently explain states that revisit similar regions of space with different orientations and velocities. This places strong demands on global coherence of the

learned dynamics, where unconstrained flow maps can achieve good local fit without corresponding to a single integrable autonomous system.

Under the same per-trial wall-clock protocol, both NADE variants achieve markedly lower vector-field error and better preserve the long-horizon attractor geometry (Figure 5).

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{0.35 \pm 0.018}$	$\mathbf{0.37 \pm 0.016}$	$4.0 \times 10^3 \pm 11$	$4.1 \times 10^2 \pm 2.4$	$3.8 \times 10^3 \pm 2.1 \times 10^2$
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	$\mathbf{0.13 \pm 0.10}$	$\mathbf{0.23 \pm 0.33}$	0.81 ± 0.017	2.5 ± 2	10 ± 11
$t_{\mathrm{MSE}_{\mathrm{vec}} \leq 0.5}$	$\mathbf{591 \pm 328\ s}$	$\mathbf{62 \pm 50\ s}$	–	–	–

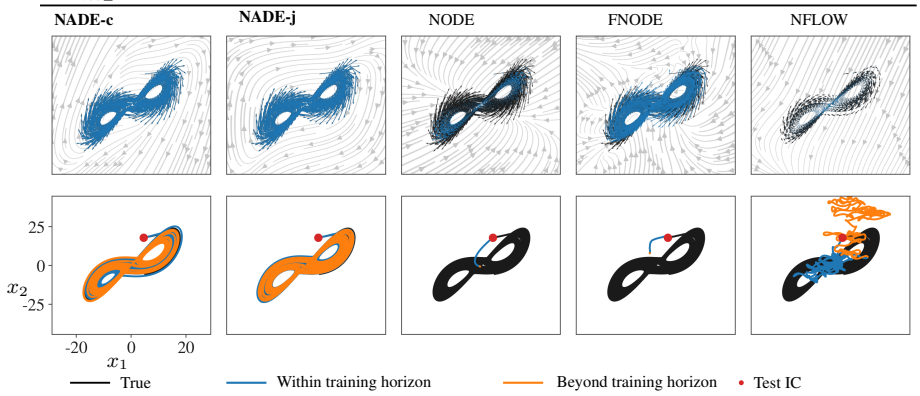


Figure 5: Lorenz comparison under the same wall-clock training budget. The table reports vector-field error, trajectory Wasserstein distance over $[T, 2T]$, and the first time at which the vector-field error reaches the indicated threshold. The Lorenz system is three-dimensional; we visualize the first two state dimensions. The upper row compares the learned and true vector fields, while the lower row shows a representative held-out rollout. The NADE models obtain substantially lower vector-field error and closely reproduce the long-term attractor geometry.

6 Discussion and Limitations

NADE turns learning autonomous dynamics into supervised state-to-velocity regression. Crucially, this computational efficiency relative to NODE is achieved without sacrificing the autonomous structure of the learned dynamics. Because the learned model has the form $f_\theta(x)$, with no explicit dependence on time, the same state is always assigned the same velocity. This distinguishes NADE from time-conditioned flow approaches such as NFLOW, where elapsed time enters the learned map and autonomy is not enforced directly. Compared with FNODE, NADE retains the same integration-free motivation while avoiding Fourier differentiation and its global spectral and periodic-continuation assumptions. The method also separates trajectory reconstruction from vector-field learning, allowing the preprocessing stage to be adapted to the observation setting: we use cubic interpolation for clean data and smoothing splines for noisy data (Appendix I).

Limitations. The method depends on the quality of the trajectory reconstruction used to obtain velocity targets. Consequently, very sparse sampling or strong measurement noise can degrade derivative estimates, and more robust or probabilistic reconstruction methods may be preferable in such regimes. Our noisy-observation experiments provide supporting evidence that NADE remains accurate under these conditions (Appendix I). A more systematic study of noise levels, observation density, trajectory coverage, and alternative reconstruction methods is left for future work. The theoretical guarantees should also be interpreted locally. The consistency analysis establishes vector-field recovery along the observed trajectory and supports rollout convergence under additional local regularity and stability conditions, but it does not guarantee generalization to arbitrary initial conditions far from the training data. Results for more distant or scaled initial conditions presented in the experiments section should therefore be viewed as empirical evidence of off-trajectory generalization rather than consequences of the theory. If physical structure is known in advance, models that encode it directly may be more appropriate than an unconstrained neural vector field.

AI USE STATEMENT

AI tools were used only for proof checking, grammar and language refinement, and coding-related support. The authors are responsible for the scientific ideas, methodology, implementation, interpretation of results, verification of all technical content, and all final decisions regarding the manuscript.

REPRODUCIBILITY STATEMENT

We provide the experimental details required to reproduce the results in the appendix. Appendix E describes the trajectory fitting, optimization, validation, hyperparameter selection, and training protocol; Appendix F specifies the benchmark equations, parameters, initial conditions, observation settings, and data-generation procedure; and Appendix D reports the model architectures and parameter counts. The evaluation metrics are defined in Appendix K. Additional settings for the corrupted-observation and high-dimensional experiments are provided in Appendices I and J, respectively.

REFERENCES

Marin Biloš, Johanna Sommer, Syama Sundar Rangapuram, Tim Januschowski, and Stephan Günnemann. Neural flows: Efficient alternative to neural ODEs. In *Advances in Neural Information Processing Systems*, volume 34, pp. 21325–21337, 2021.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamics. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016. doi: 10.1073/pnas.1517384113.

Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. Modern Koopman theory for dynamical systems. *SIAM Review*, 64(2):229–340, 2022. doi: 10.1137/21M1401243.

Matteo Caldana and Jan S. Hesthaven. Neural ordinary differential equations for model order reduction of stiff systems. *International Journal for Numerical Methods in Engineering*, 126(12), 2025. doi: 10.1002/nme.70060.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6571–6583, 2018.

Carl De Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer, New York, 1978.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs. In *Advances in Neural Information Processing Systems*, volume 32, pp. 3134–3144, 2019.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M. Oberman. How to train your neural ODE: The world of Jacobian and kinetic regularization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3154–3164. PMLR, 2020.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 15353–15363, 2019.

Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, volume 14 of *Springer Series in Computational Mathematics*. Springer, 1996.

Ernst Hairer, Syvert Paul Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Computational Mathematics*. Springer, Berlin, 2 edition, 1993.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6696–6706, 2020.

Suyong Kim, Weiqi Ji, Sili Deng, Yingbo Ma, and Christopher Rackauckas. Stiff neural ordinary differential equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9):093122, 2021. doi: 10.1063/5.0060697.

Xin Li, Jingdong Zhang, Qunxi Zhu, Chengli Zhao, Xue Zhang, Xiaojun Duan, and Wei Lin. From Fourier to neural ODEs: Flow matching for modeling complex systems. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 29390–29405, 2024.

Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3870–3882. PMLR, 26–28 Aug 2020.

Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2): 130–141, 1963.

Allen Alvarez Loya, Daniel A. Serino, J. W. Burby, and Qi Tang. Structure-preserving neural ordinary differential equations for stiff systems, 2025.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3276–3285. PMLR, 2018.

James D. Murray. *Mathematical Biology I: An Introduction*. Springer, New York, 3 edition, 2002.

Lawrence Perko. *Differential Equations and Dynamical Systems*. Springer, 3 edition, 1996.

Maziar Raissi, Paris Perdikaris, and George 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. doi: 10.1016/j.jcp.2018.10.045.

Yulia Rubanova, Ricky T. Q. Chen, and David K. Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, volume 32, pp. 5320–5330, 2019.

Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. Westview Press, 2 edition, 2018.

Juntang Zhuang, Nicha Dvornek, Xiaoxiao Li, Sekhar Tatikonda, Xenophon Papademetris, and James S. Duncan. Adaptive checkpoint adjoint method for gradient estimation in neural ODE. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11639–11649. PMLR, 2020.

A Training loss summaries

B NODE: Short Trajectory Comparison

We would like to emphasise that the benchmarks in the main paper are intentionally designed to evaluate all methods, including NADE and the baselines, under more demanding conditions than short-horizon trajectory fitting alone. These tests include learning difficult nonlinear dynamics, predicting over long trajectories, generalising beyond the training horizon, and generalising to previously unseen initial conditions. Additionally, to ensure a fair comparison, all methods were assigned the same wall-clock budget of 30 minutes per hyperparameter trial, corresponding to approximately 3–6 hours of total optimisation time for each method, depending on the size of the hyperparameter search.

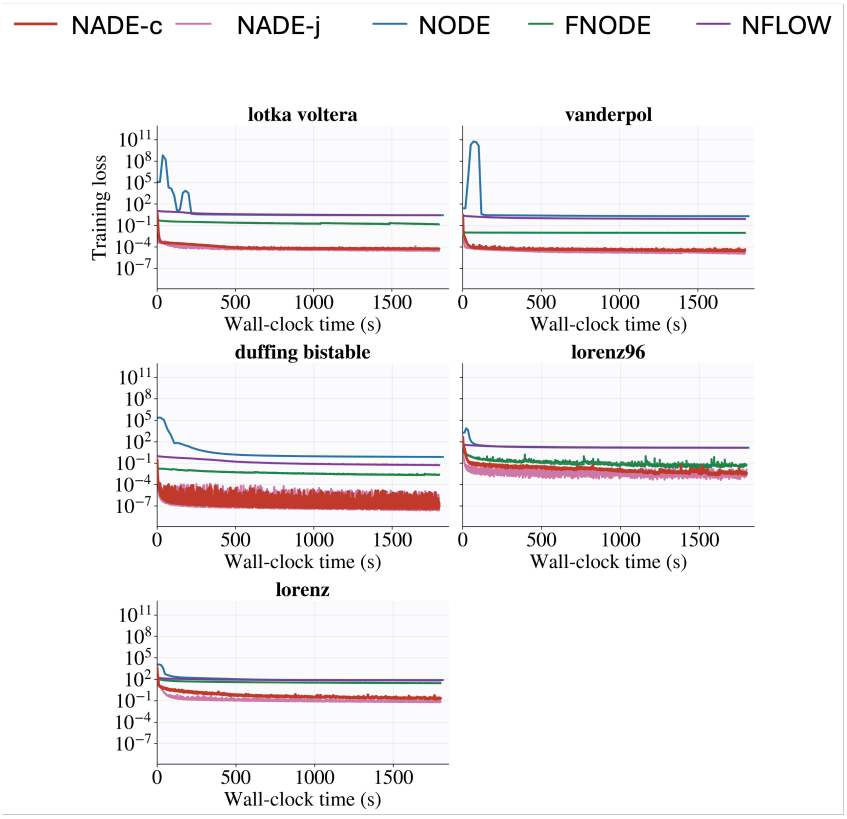


Figure 6: Training objective vs. wall-clock (averaged over seeds, and for NADE-c, averaged over dimensions).

In the main experiments, we observed that, for some systems, FNODE and NFLOW can achieve reasonable predictive accuracy. Figure 2 provides one such example, where NADE reaches a lower error substantially faster.

Here, in Figure 7, we consider the Lorenz system as a representative example and show that, for a very short trajectory ($T = 1$), NODE is also able, given sufficient training time, to approximate the observed vector field and reproduce the observed trajectory with reasonable accuracy.

C Mathematical Proofs

Assumption C.1 (Regularity and identifiability)**.** Throughout, let $\mathcal{T} = [0, \tau]$, let $t_{k,n} = k\tau/n$ for $k = 0, \dots, n$, and suppose that

$$Y_{k,n} = s_0(t_{k,n}) + \varepsilon_{k,n}.$$

We assume the following.

1. **Noise.** The errors $\{\varepsilon_{k,n} : k = 0, \dots, n\}$ are independent, mean-zero, and uniformly square-integrable:
$$\mathbb{E}[\varepsilon_{k,n}] = 0, \qquad \sup_{k,n} \mathbb{E}\|\varepsilon_{k,n}\|^2 < \infty.$$

2. **Compact parameter spaces.** The trajectory and vector-field parameter spaces $\Theta_s \subset \mathbb{R}^{p_s}$ and $\Theta_f \subset \mathbb{R}^{p_f}$ are compact.

3. **Trajectory regularity.** The map $\eta \mapsto s_\eta$ is continuous from Θ_s into $C^1(\mathcal{T}; \mathbb{R}^d)$. Equivalently, if $\eta_m \to \eta$, then
$$\|s_{\eta_m} - s_\eta\|_\infty + \|\dot{s}_{\eta_m} - \dot{s}_\eta\|_\infty \to 0.$$

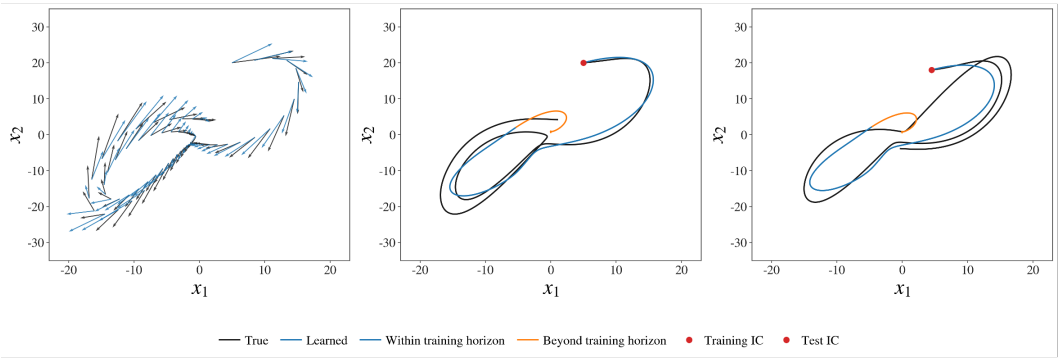


Figure 7: **Short-trajectory NODE performance for the Lorenz system.** NODE is trained on a short observed trajectory with $T = 1$. Given sufficient training time, NODE is able to approximate the observed vector field and reproduce the observed trajectory with reasonable accuracy. However, achieving this fit requires substantially more training time than NADE, as discussed in the main results.

4. **Trajectory well-specification and identifiability.** There exists a unique $\eta_0 \in \Theta_s$ such that $s_{\eta_0} = s_0$ on $\mathcal{T}$. Equivalently, η_0 is the unique minimizer of
$$M_s(\eta) := \frac{1}{\tau} \int_0^\tau \|s_\eta(t) - s_0(t)\|^2 \, dt.$$

5. **Vector-field regularity.** There exists a compact set $K \subset \mathbb{R}^d$ such that
$$s_\eta(t) \in K \qquad \text{for all } \eta \in \Theta_s,\ t \in \mathcal{T}.$$
The map $(\theta, x) \mapsto f_\theta(x)$ is continuous on $\Theta_f \times K$, and the vector fields are uniformly Lipschitz in x on K: there exists $L < \infty$ such that
$$\|f_\theta(x) - f_\theta(x')\| \le L\|x - x'\|$$
for all $\theta \in \Theta_f$ and all $x, x' \in K$.

6. **Vector-field well-specification and persistent excitation.** The true trajectory satisfies
$$\dot{s}_0(t) = f_{\theta_0}(s_0(t)), \qquad t \in \mathcal{T},$$
for some $\theta_0 \in \Theta_f$. Moreover, θ_0 is the unique minimizer of
$$M_f(\theta) := \frac{1}{\tau} \int_0^\tau \|f_\theta(s_0(t)) - \dot{s}_0(t)\|^2 \, dt.$$

Proof of Theorem 4.1. We prove the result by analyzing the two stages separately. The first stage is a parametric least-squares fit of the trajectory, while the second stage is a least-squares regression of the vector field on the derivative of the fitted trajectory.

Define the empirical trajectory objective
$$M_{s,n}(\eta) := \frac{1}{n+1} \sum_{k=0}^{n} \|Y_{k,n} - s_\eta(t_{k,n})\|^2.$$

Since $Y_{k,n} = s_0(t_{k,n}) + \varepsilon_{k,n}$, we can write
$$M_{s,n}(\eta) = \frac{1}{n+1} \sum_{k=0}^{n} \|s_0(t_{k,n}) - s_\eta(t_{k,n})\|^2 + \frac{2}{n+1} \sum_{k=0}^{n} \langle \varepsilon_{k,n}, s_0(t_{k,n}) - s_\eta(t_{k,n}) \rangle + \frac{1}{n+1} \sum_{k=0}^{n} \|\varepsilon_{k,n}\|^2.$$

The final term does not depend on η, and therefore does not affect the argmin. It remains to compare the first two terms with the population trajectory objective M_s.

First, because $\eta \mapsto s_\eta$ is continuous into $C(\mathcal{T}; \mathbb{R}^d)$ and Θ_s is compact, the class

$$\left\{ t \mapsto \|s_0(t) - s_\eta(t)\|^2 : \eta \in \Theta_s \right\}$$

is uniformly bounded and equicontinuous. Writing $\Delta t = \tau/n$, for any such function g_η,

$$\frac{1}{n+1} \sum_{k=0}^{n} g_\eta(t_{k,n}) = \frac{n}{n+1} \left(\frac{1}{\tau} \sum_{k=0}^{n-1} g_\eta(t_{k,n})\, \Delta t \right) + \frac{g_\eta(\tau)}{n+1}.$$

Hence, by uniform Riemann-sum convergence,

$$\sup_{\eta \in \Theta_s} \left| \frac{1}{n+1} \sum_{k=0}^{n} \|s_0(t_{k,n}) - s_\eta(t_{k,n})\|^2 - M_s(\eta) \right| \to 0.$$

Second, the noise cross term is uniformly negligible. Indeed, define

$$h_\eta(t) := s_0(t) - s_\eta(t).$$

The set $\{h_\eta : \eta \in \Theta_s\}$ is compact in $C(\mathcal{T}; \mathbb{R}^d)$, hence totally bounded in the sup norm. For any finite δ-net $\{h_{\eta_1}, \dots, h_{\eta_N}\}$, independence and the uniform second-moment bound imply

$$\frac{1}{n+1} \sum_{k=0}^{n} \left\langle \varepsilon_{k,n}, h_{\eta_j}(t_{k,n}) \right\rangle \to_p 0 \qquad \text{for each } j = 1, \dots, N.$$

Taking a union bound over the finite net and then letting $\delta \downarrow 0$ gives

$$\sup_{\eta \in \Theta_s} \left| \frac{1}{n+1} \sum_{k=0}^{n} \langle \varepsilon_{k,n}, h_\eta(t_{k,n}) \rangle \right| \to_p 0.$$

Therefore

$$\sup_{\eta \in \Theta_s} |M_{s,n}(\eta) - M_s(\eta) - R_n| \to_p 0,$$

where

$$R_n := \frac{1}{n+1} \sum_{k=0}^{n} \|\varepsilon_{k,n}\|^2$$

does not depend on η. Since adding an η-independent term does not change the argmin, and since M_s has the unique minimizer η_0, the argmin theorem yields

$$\hat{\eta}_n \to_p \eta_0.$$

By continuity of $\eta \mapsto s_\eta$ into $C^1(\mathcal{T}; \mathbb{R}^d)$, this immediately implies

$$\|s_{\hat{\eta}_n} - s_0\|_\infty + \|\dot{s}_{\hat{\eta}_n} - \dot{s}_0\|_\infty \to_p 0.$$

We now analyze the second stage. For $\eta \in \Theta_s$, define

$$M_{f,n}(\theta; \eta) := \frac{1}{n+1} \sum_{k=0}^{n} \|f_\theta(s_\eta(t_{k,n})) - \dot{s}_\eta(t_{k,n})\|^2.$$

We claim that

$$\sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \hat{\eta}_n) - M_f(\theta)| \to_p 0.$$

By the triangle inequality,

$$\begin{aligned} &\sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \hat{\eta}_n) - M_f(\theta)| \\ &\le \sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \hat{\eta}_n) - M_{f,n}(\theta; \eta_0)| \\ &\quad + \sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \eta_0) - M_f(\theta)|. \end{aligned}$$

The second term vanishes by uniform Riemann-sum convergence. To see this, note that the functions

$$t \mapsto \|f_\theta(s_0(t)) - \dot{s}_0(t)\|^2, \qquad \theta \in \Theta_f,$$

are uniformly bounded and equicontinuous on $\mathcal{T}$. This follows from compactness of Θ_f, continuity of $(\theta, x) \mapsto f_\theta(x)$, uniform Lipschitzness of f_θ in x, and continuity of s_0 and $\dot{s}_0$. Hence

$$\sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \eta_0) - M_f(\theta)| \to 0.$$

It remains to control the effect of replacing the true trajectory s_0 and derivative $\dot{s}_0$ by their fitted counterparts. By compactness and continuity, there are constants $B, D < \infty$ such that

$$\sup_{\theta \in \Theta_f} \sup_{x \in K} \|f_\theta(x)\| \le B, \qquad \sup_{\eta \in \Theta_s} \|\dot{s}_\eta\|_\infty \le D.$$

For any $\theta \in \Theta_f$, $\eta \in \Theta_s$, and $t \in \mathcal{T}$, set

$$a_{\theta,\eta}(t) := f_\theta(s_\eta(t)) - \dot{s}_\eta(t), \qquad a_{\theta,0}(t) := f_\theta(s_0(t)) - \dot{s}_0(t).$$

Then

$$\begin{aligned} \left| \|a_{\theta,\eta}(t)\|^2 - \|a_{\theta,0}(t)\|^2 \right| &\le \left(\|a_{\theta,\eta}(t)\| + \|a_{\theta,0}(t)\| \right) \|a_{\theta,\eta}(t) - a_{\theta,0}(t)\| \\ &\le 2(B + D) \left(L\|s_\eta(t) - s_0(t)\| + \|\dot{s}_\eta(t) - \dot{s}_0(t)\| \right). \end{aligned}$$

Taking averages over $t_{k,n}$ and then a supremum over θ gives a constant $C < \infty$ such that

$$\sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \eta) - M_{f,n}(\theta; \eta_0)| \le C \left(\|s_\eta - s_0\|_\infty + \|\dot{s}_\eta - \dot{s}_0\|_\infty \right).$$

Applying this bound with $\eta = \hat{\eta}_n$, and using the first-stage consistency already proved, yields

$$\sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \hat{\eta}_n) - M_{f,n}(\theta; \eta_0)| \to_p 0.$$

Combining the two terms, we obtain

$$\sup_{\theta \in \Theta_f} |M_{f,n}(\theta; \hat{\eta}_n) - M_f(\theta)| \to_p 0.$$

Since M_f has the unique minimizer θ_0, another application of the argmin theorem gives

$$\hat{\theta}_n \to_p \theta_0.$$

Finally, the map

$$\theta \mapsto \int_0^\tau \|f_\theta(s_0(t)) - f_{\theta_0}(s_0(t))\|^2 \, dt$$

is continuous on Θ_f, because $(\theta, x) \mapsto f_\theta(x)$ is continuous on the compact set $\Theta_f \times K$. It equals zero at $\theta = \theta_0$. Therefore, by the continuous mapping theorem and $\hat{\theta}_n \to_p \theta_0$,

$$\int_0^\tau \|f_{\hat{\theta}_n}(s_0(t)) - f_{\theta_0}(s_0(t))\|^2 \, dt \to_p 0.$$

This proves the theorem. □

The following result extends Theorem 4.1 to consistency of the integrated prediction path.

Corollary C.2 (Rollout consistency)**.** *Let*

$$\mathcal{O}_0 := \{s_0(t) : t \in \mathcal{T}\}$$

denote the true orbit. In addition to the conditions of Theorem 4.1, suppose that there exist $\rho > 0$ and an open set $U \subset \mathbb{R}^d$ containing the compact tube

$$K_\rho := \{x \in \mathbb{R}^d : \operatorname{dist}(x, \mathcal{O}_0) \le \rho\}$$

such that all vector fields $\{f_\theta : \theta \in \Theta_f\}$ are locally Lipschitz on U and uniformly Lipschitz on K_ρ. Suppose further that the learned vector field is uniformly consistent on this tube:

$$\sup_{x \in K_\rho} \|f_{\hat{\theta}_n}(x) - f_{\theta_0}(x)\| \to_p 0.$$

Let $\hat{x}_n$ be the maximal solution of

$$\dot{\hat{x}}_n(t) = f_{\hat{\theta}_n}(\hat{x}_n(t)), \qquad \hat{x}_n(0) = \hat{x}_{0,n},$$

and assume that

$$\hat{x}_{0,n} \to_p s_0(0).$$

Then, with probability tending to one, $\hat{x}_n$ is defined on all of $\mathcal{T}$, and

$$\sup_{t\in\mathcal{T}} \|\hat{x}_n(t) - s_0(t)\| \to_p 0.$$

Proof. The proof is a standard continuous-dependence argument for ODEs, but we spell out the localization step because the uniform control of the vector field is only assumed on a neighbourhood of the true orbit.

Define the error process

$$e_n(t) := \hat{x}_n(t) - s_0(t),$$

and let

$$\Delta_n := \|\hat{x}_{0,n} - s_0(0)\| + \tau \sup_{x\in K_\rho} \|f_{\hat{\theta}_n}(x) - f_{\theta_0}(x)\|.$$

By assumption, $\Delta_n \to_p 0$.

Let

$$t_n^\star := \inf\{t \in [0,\tau] : \|e_n(t)\| \geq \rho\},$$

with the convention that $t_n^\star = \tau$ if the set is empty. For $t < t_n^\star$, both $s_0(t)$ and $\hat{x}_n(t)$ lie in K_ρ. Hence, using $\dot{s}_0(t) = f_{\theta_0}(s_0(t))$, we have

$$\begin{aligned}
\|e_n(t)\| &\leq \|\hat{x}_{0,n} - s_0(0)\| + \int_0^t \|f_{\hat{\theta}_n}(\hat{x}_n(u)) - f_{\theta_0}(s_0(u))\|\, du \\
&\leq \|\hat{x}_{0,n} - s_0(0)\| + \int_0^t \|f_{\hat{\theta}_n}(\hat{x}_n(u)) - f_{\hat{\theta}_n}(s_0(u))\|\, du \\
&\quad + \int_0^t \|f_{\hat{\theta}_n}(s_0(u)) - f_{\theta_0}(s_0(u))\|\, du.
\end{aligned}$$

Using the uniform Lipschitz constant L on K_ρ, this gives

$$\|e_n(t)\| \leq \Delta_n + L\int_0^t \|e_n(u)\|\, du, \qquad t < t_n^\star.$$

Gronwall's inequality therefore implies

$$\sup_{0\leq t<t_n^\star} \|e_n(t)\| \leq e^{L\tau}\Delta_n.$$

Now consider the event

$$A_n := \{e^{L\tau}\Delta_n < \rho\}.$$

Since $\Delta_n \to_p 0$, we have $\mathbb{P}(A_n) \to 1$. On A_n, the above bound prevents the error from reaching ρ. Hence the solution cannot exit the tube K_ρ before time τ, so $t_n^\star = \tau$, the solution is defined on all of $\mathcal{T}$, and

$$\sup_{t\in\mathcal{T}} \|\hat{x}_n(t) - s_0(t)\| \leq e^{L\tau}\Delta_n.$$

Since $\Delta_n \to_p 0$, this proves

$$\sup_{t\in\mathcal{T}} \|\hat{x}_n(t) - s_0(t)\| \to_p 0.$$

□

System	Method	Parameters	Final training loss
Lotka–Volterra	NADE-c	**133636**	$\mathbf{5\times10^{-5}\pm6.4\times10^{-6}}$
	NADE-j	**265218**	$\mathbf{2.5\times10^{-5}\pm1.2\times10^{-6}}$
	NODE	265218	2.7 ± 0.095
	FNODE	265218	0.12 ± 0.083
	NFLOW	265762	5.6 ± 5.4
Van der Pol	NADE-c	**133636**	$\mathbf{2.6\times10^{-5}\pm7.5\times10^{-6}}$
	NADE-j	**265218**	$\mathbf{9\times10^{-6}\pm4.7\times10^{-6}}$
	NODE	265218	2 ± 0.052
	FNODE	265218	0.009 ± 0.00063
	NFLOW	265762	0.81 ± 0.09
Lorenz	NADE-c	**201222**	$\mathbf{0.11\pm0.038}$
	NADE-j	**266243**	$\mathbf{0.057\pm0.022}$
	NODE	266243	77 ± 1.5
	FNODE	266243	27 ± 3
	NFLOW	266803	65 ± 25
Duffing bistable	NADE-c	**133636**	$\mathbf{4\times10^{-8}\pm7.5\times10^{-9}}$
	NADE-j	**265218**	$\mathbf{2.2\times10^{-8}\pm4.6\times10^{-9}}$
	NODE	265218	0.71 ± 0.15
	FNODE	265218	0.0019 ± 0.0019
	NFLOW	265762	0.051 ± 0.034
Lorenz–96	NADE-c	**244520**	$\mathbf{7.4\times10^{-4}\pm7.6\times10^{-4}}$
	NADE-j	**273418**	$\mathbf{2.6\times10^{-4}\pm8.7\times10^{-5}}$
	NODE	273418	14 ± 0.27
	FNODE	273418	0.016 ± 0.012
	NFLOW	274090	22 ± 16

Table 1: Model capacity and final training loss across the benchmark systems. We report the total number of trainable parameters and the final training loss. Values are reported as mean $\pm$ standard deviation across seeds.

D Model variants and capacity

The experiments report two NADE parameterizations. NADE-c uses d independent two-hidden-layer ReLU MLPs, each mapping the full state $\mathbb{R}^d$ to a single velocity component. NADE-j uses one joint two-hidden-layer ReLU MLP, $f_\theta : \mathbb{R}^d \to \mathbb{R}^d$. NODE and FNODE use the same joint MLP width as NADE-j. NFLOW uses the ResNetFlow implementation with one outer flow block and two hidden layers, with the same hidden width as the NADE-j network.

For the low-dimensional systems, each NADE-c head has width 256, while for the 10-dimensional Lorenz–96 system we use width 150 to keep the total coordinate-wise parameter count close to that of the width-512 joint models. NADE-j, NODE, FNODE, and NFLOW use width 512. The comparison therefore uses comparable, although not exactly identical, model capacities: the NADE variants always have no more parameters than the corresponding baselines. Exact parameter counts are reported in Table 1.

For a scalar NADE-c head of width w,

$$P_{\text{head}} = dw + w^2 + 3w + 1,$$

and the complete coordinate-wise model contains dP_{head} network parameters together with d learned scale parameters. A joint d-output MLP of width W contains

$$P_{\text{joint}} = dW + W^2 + (d+2)W + d.$$

E Implementation

Trajectory fitting and derivative targets. For clean observations and for noise-free sparse/irregular observations, each coordinate is fit with a cubic spline on the retained observation times and differentiated analytically at those times. When additive noise is present, the final code uses a cubic smoothing spline with SciPy smoothing parameter $s = \max\{10^{-8},\ \sigma_{\text{rel}}^2 a_i^2 n_{\text{obs}}\}$, where σ_{rel} is the configured relative noise level and a_i is the clean max-absolute scale of coordinate i. States from all training trajectories are concatenated into the regression inputs.

Optimization. Each scalar head is a two-hidden-layer ReLU MLP. The scale in equation 5 is

$$\alpha_i = 0.1 + 9.9\,\sigma(u_i),$$

and the learned derivative is $f_{\theta_i}(x)/\alpha_i$. Network parameters use Adam at the candidate learning rate, while u_i uses one quarter of that rate. The d heads are trained concurrently with `torch.func.vmap`; each head receives the full wall-clock budget, gradients are clipped independently to norm 10, and each head restores the checkpoint with its lowest native training loss. Learning rates are selected separately per coordinate using held-out NADE validation MSE.

NADE-j directly regresses the d-vector of spline derivatives with a width-512 joint MLP. NODE uses a width-512 vector field. FNODE uses a width-512 joint derivative-regression MLP with Fourier derivative targets. NFLOW uses the ResNetFlow implementation with one outer flow block, two width-512 hidden layers, ReLU activations, with eight time features, and invertibility enabled.

Checkpointing and hyperparameter selection. Within every hyperparameter trial, the checkpoint with the lowest native training objective is restored. Hyperparameters are then selected on a held-out validation initial condition $0.95x_0$, with the same observation model as training:

- NADE-c: per-coordinate spline-derivative MSE;
- NADE-j and FNODE: joint spline/Fourier-derivative MSE;
- NODE: validation trajectory MSE;
- NFLOW: validation trajectory MSE plus the selected autonomous-flow semigroup penalty.

Search grids and wall-clock control. All methods use Adam learning-rate candidates

$$\{10^{-4},\ 3{\cdot}10^{-4},\ 10^{-3}\}.$$

NADE-c, NADE-j and NODE search learning rate only. FNODE searches learning rate jointly with Fourier modes $K \in \{50, 100\}$. NFLOW searches learning rate jointly with semigroup weight $\gamma \in \{0, 0.1, 1\}$. Every candidate combination receives the same per-trial wall-clock cap for a given runtime condition; runtime tags and the actual seconds used are stored with the checkpoints. Because candidate-grid sizes differ, this per-trial budget should not be interpreted as equal total hyperparameter-search compute.

Validation and test ICs. The training initial condition (IC) is x_0, validation uses $0.95x_0$, and the systematic held-out test set uses scales

$$0.90x_0,\ 0.85x_0,\ 0.80x_0,\ 0.75x_0,\ 0.70x_0.$$

Representative paper panels use $0.90x_0$.

F Benchmark data generation

System	d	T	Δt_{obs}	Training IC(s)	Fixed parameters
Lotka–Volterra	2	50	0.01	$(4, 4)$	$\alpha = \beta = 2/3,\ \gamma = \delta = 1$
Van der Pol	2	40	0.01	$(1, 0)$	$\mu = 1$
Lorenz	3	30	0.01	$(5, 20, 20)$	$\sigma = 10,\ \rho = 28,\ \beta = 8/3$
Duffing bistable	2	60	0.02	$(-2, 0), (2, 0)$	$\delta = 0.2$
Lorenz–96	10	30	0.01	$(8.01, 8, \ldots, 8)$	$F = 8$

G Second Basin of Duffing oscillator

To complement the first-basin results in the main text, Figure 8 reports the corresponding evaluation from a held-out initial condition in the second basin of attraction. This provides an additional test of whether a single learned autonomous vector field can represent the dynamics consistently across both basins.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{4.2\times10^{-6}\pm5.7\times10^{-9}}$	$\mathbf{4.2\times10^{-6}\pm6.2\times10^{-10}}$	0.38 ± 0.12	0.036 ± 0.0019	0.065 ± 0.015
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	$\mathbf{0.083\pm0.031}$	$\mathbf{0.063\pm0.018}$	$1.6\times10^{3}\pm4.9\times10^{2}$	$4.3\times10^{2}\pm9.1\times10^{2}$	$2.3\times10^{3}\pm1.2\times10^{3}$
$t_{\mathrm{MSE}_{\mathrm{vec}}\leq0.01}$	$\mathbf{2.0\pm0.83}\ s$	$\mathbf{1.4\pm0.11}\ s$	–	$0.14\ s$	–

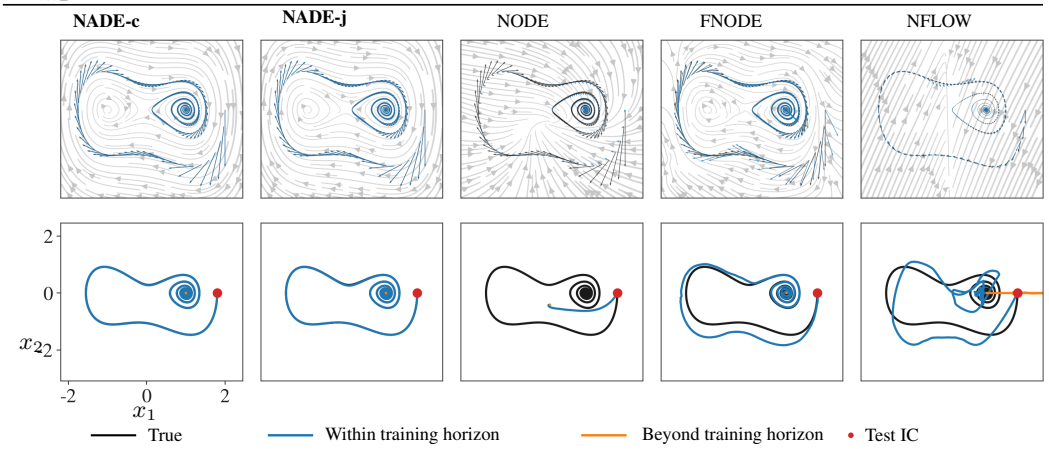


Figure 8: Duffing bistable results for a held-out initial condition in the complementary basin of attraction. The quantitative metrics are the same system-level measurements reported in Fig. 2; the lower row tests whether the learned dynamics also preserve trajectories initialized in the second basin.

H Compute Resources

Experiments are launched through the same SLURM-managed GPU workflow. Each model/system/seed/hyperparameter trial uses one GPU worker and an explicit wall-clock stop; no experiment uses multi-GPU training.

I Noisy, Sparse and Irregular Observations

This experiment considers three simultaneous perturbations: additive Gaussian noise, sparse observations, and irregular sampling. Specifically, we add Gaussian noise with standard deviation 0.02 times the clean maximum absolute value of each coordinate, randomly remove 20% of the interior observation times while retaining the endpoints, and perturb each remaining interior observation time by a uniformly sampled offset of up to $\pm0.25\Delta t_{\mathrm{obs}}$. The same retained and jittered observation-time grid is shared across training trajectories, while additive noise is sampled independently for the observation states of each trajectory. When noise is present, NADE-c and NADE-j obtain their velocity targets from smoothing splines rather than interpolating cubic splines. FNODE is omitted from this experiment because the irregular observation times are incompatible with the regular sampling assumed by our FNODE implementation.

Under simultaneous noise, sparsity, and irregular sampling, evaluation using MSE_{vec} shows that NADE retains a strong vector-field-recovery advantage under the same per-trial wall-clock and comparable-capacity protocol (2). As in the clean experiments, MSE_{vec} is used only for evaluation and is computed against the true vector field, not used for training or model selection. Figures 9–13 show the corresponding robustness results for the systems considered in the main text.

J High Dimensional System

As a concrete higher-dimensional stress test, we evaluate the 10-dimensional Lorenz–96 system, described as

$$\dot{x}_i = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F, \qquad F = 8.$$

For this experiment shown in Figure14, NADE-c uses width 150 per coordinate in this experiment; NADE-j and the joint baselines use width 512. Numerical metrics are computed in all 10 dimensions. For visualisation, the vector field and trajectories are projected to the x_1–x_2 plane, with the remaining

System	Method	Parameters	Final training loss
Lotka–Volterra	NADE-c	**133636**	**0.049±0.011**
	NADE-j	**265218**	**0.039±0.0059**
	NODE	265218	2.7±0.083
	NFLOW	265762	2.3±1.7
Van der Pol	NADE-c	**133636**	**0.0077±0.00022**
	NADE-j	**265218**	**0.0059±0.00097**
	NODE	265218	1.4±1
	NFLOW	265762	1.4±0.91
Lorenz	NADE-c	**201222**	**42±4.9**
	NADE-j	**266243**	**22±4.9**
	NODE	266243	77±0.63
	NFLOW	266803	66±20
Duffing bistable	NADE-c	**133636**	**0.0018±0.00021**
	NADE-j	**265218**	**0.0015±0.00018**
	NODE	265218	0.67±0.15
	NFLOW	265762	0.071±0.049

Table 2: Model capacity and final training loss under simultaneous noisy, sparse, and irregular observations. Values are reported as mean ± standard deviation across seeds.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	**0.03±0.0029**	**0.035±0.0072**	7.4±2.6	0.56±0.074	4±0.91
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	**0.86±0.44**	**1.3±0.68**	0.77±0.018	1.9±0.4	24±13
$t_{\mathrm{MSE}_{\mathrm{vec}} \le 0.01}$	**3.2±0.77 s**	**1.2±0.66 s**	–	–	–

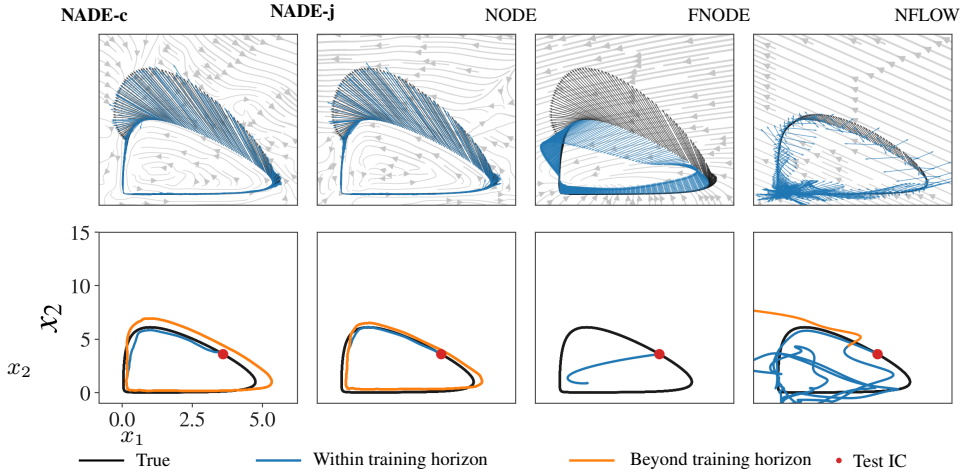


Figure 9: Lotka–Volterra under noisy, sparse, and irregular observations.

coordinates fixed to their coordinate-wise medians for broad vector-field slices. This example demonstrates that the same training construction applies beyond the low-dimensional visual benchmarks; it is not intended as an exhaustive scaling study.

K Evaluation metrics

We use complementary metrics to evaluate vector-field recovery and trajectory prediction. These metrics are used only for evaluation and are not training objectives.

Vector-field error. For clean synthetic data, the true vector field f is known and can therefore be compared directly with the learned vector field f_θ. Let $N = \sum_{l=1}^{L} n_l$ denote the total number of

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
MSE_{vec}	$\mathbf{0.0052 \pm 0.00084}$	$\mathbf{0.0055 \pm 0.001}$	0.38 ± 0.12	0.036 ± 0.0019	0.075 ± 0.01
$\text{Wasserstein}_{\text{traj}}[T, 2T]$	$\mathbf{2.8 \times 10^2 \pm 6.4 \times 10^2}$	$\mathbf{3.4 \times 10^2 \pm 7.2 \times 10^2}$	$1.6 \times 10^3 \pm 5.6 \times 10^2$	$4.3 \times 10^2 \pm 9.1 \times 10^2$	$2.2 \times 10^3 \pm 1.2 \times 10^3$
$t_{\text{MSE}_{\text{vec}} \leq 0.01}$	$\mathbf{1.8 \pm 1.4\ s}$	$\mathbf{0.99 \pm 0.55\ s}$	–	$0.14\ s$	–

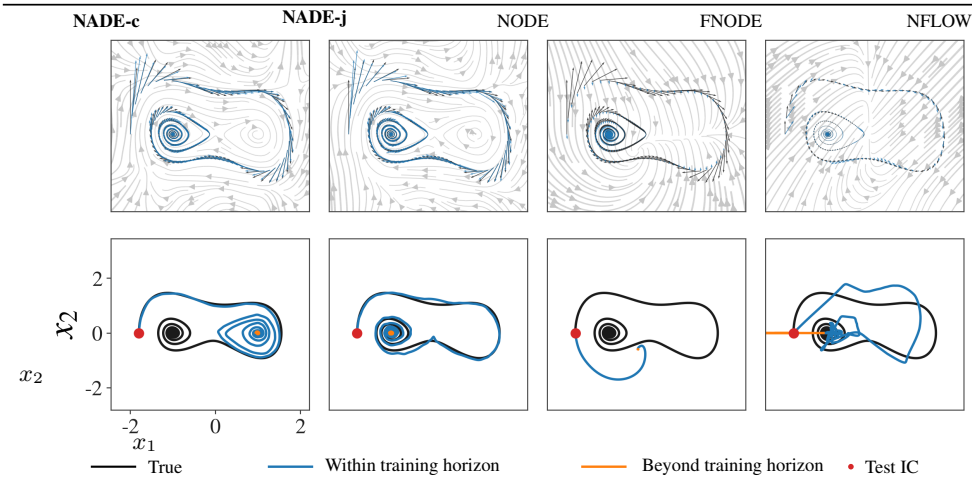


Figure 10: Duffing bistable dynamics in the first basin under noisy, sparse, and irregular observations.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
MSE_{vec}	$\mathbf{0.0052 \pm 0.00084}$	$\mathbf{0.0055 \pm 0.001}$	0.38 ± 0.12	0.036 ± 0.0019	0.075 ± 0.01
$\text{Wasserstein}_{\text{traj}}[T, 2T]$	$\mathbf{2.8 \times 10^2 \pm 6.4 \times 10^2}$	$\mathbf{3.4 \times 10^2 \pm 7.2 \times 10^2}$	$1.6 \times 10^3 \pm 5.6 \times 10^2$	$4.3 \times 10^2 \pm 9.1 \times 10^2$	$2.2 \times 10^3 \pm 1.2 \times 10^3$
$t_{\text{MSE}_{\text{vec}} \leq 0.01}$	$\mathbf{1.8 \pm 1.4\ s}$	$\mathbf{0.99 \pm 0.55\ s}$	–	$0.14\ s$	–

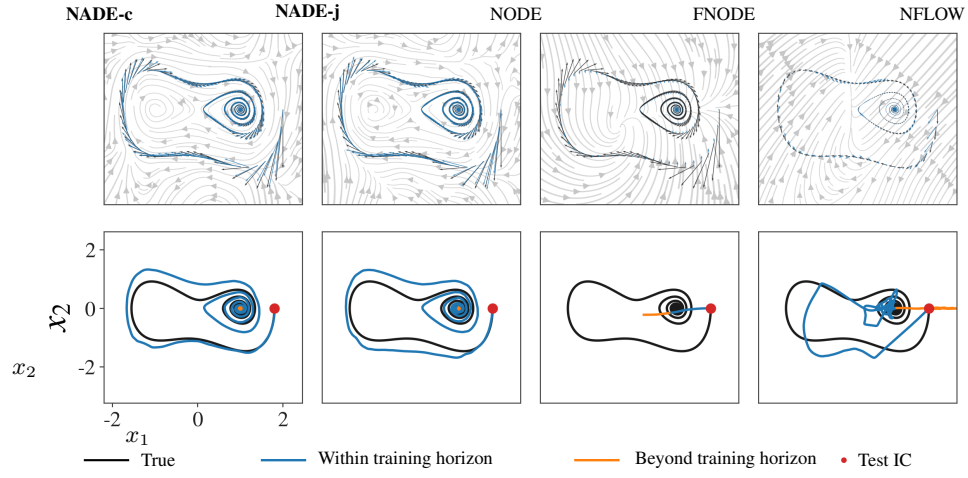


Figure 11: Duffing bistable dynamics in the second basin under noisy, sparse, and irregular observations.

observed states. We define

$$MSE_{\text{vec}} = \frac{1}{dN} \sum_{l=1}^{L} \sum_{k=1}^{n_l} \left\| f_\theta\left(x_k^{(l)}\right) - f\left(x_k^{(l)}\right) \right\|_2^2. \tag{19}$$

Thus, MSE_{vec} measures the mean squared error of the learned velocity field at the clean observation states.

Trajectory Wasserstein distance. This metric measures how well the predicted rollout reproduces the distribution of states visited by the true trajectory. For each coordinate i, let σ_i denote the standard

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{46 \pm 9}$	$\mathbf{75 \pm 16}$	$4.2 \times 10^3 \pm 4.1$	$4.1 \times 10^2 \pm 2.4$	$3.9 \times 10^3 \pm 3 \times 10^2$
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	$\mathbf{3.9 \pm 2.8}$	$\mathbf{0.41 \pm 0.39}$	0.83 ± 0.02	2.5 ± 2	4.7 ± 2.1
$t_{\mathrm{MSE}_{\mathrm{vec}} \leq 0.5}$	–	–	–	–	–

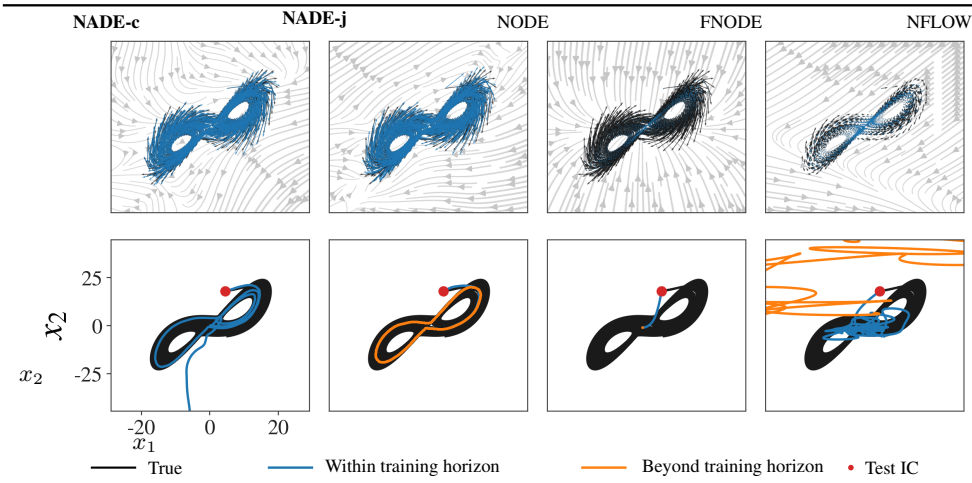


Figure 12: Lorenz under noisy, sparse, and irregular observations.

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{0.005 \pm 0.00057}$	$\mathbf{0.0068 \pm 0.00052}$	2.7 ± 1.2	0.029 ± 0.00039	2.3 ± 0.64
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	$\mathbf{0.34 \pm 0.46}$	$\mathbf{0.58 \pm 0.46}$	0.64 ± 0.34	0.61 ± 0.44	4.7 ± 4.8
$t_{\mathrm{MSE}_{\mathrm{vec}} \leq 0.01}$	$\mathbf{3.4 \pm 2.2}\ s$	$\mathbf{1.1 \pm 0.50}\ s$	–	–	–

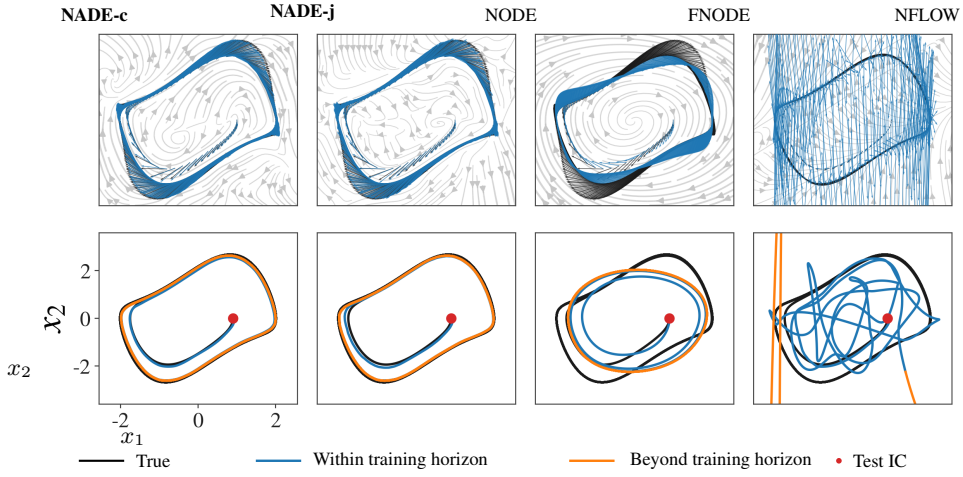


Figure 13: Van der Pol under noisy, sparse, and irregular observations.

deviation of the true trajectory over the evaluation interval, and define the empirical distributions

$$\mu_i = \frac{1}{M} \sum_{q=1}^{M} \delta_{x_i(t_q)/\sigma_i}, \qquad \hat{\mu}_i = \frac{1}{M} \sum_{q=1}^{M} \delta_{\hat{x}_i(t_q)/\sigma_i}.$$

We report the mean coordinate-wise 1-Wasserstein distance

$$W_{\mathrm{traj}} = \frac{1}{d} \sum_{i=1}^{d} W_1(\mu_i, \hat{\mu}_i)\,, \tag{20}$$

Metric	NADE-c	NADE-j	NODE	FNODE	NFLOW
$\mathrm{MSE}_{\mathrm{vec}}$	$\mathbf{0.013 \pm 0.00086}$	$\mathbf{0.013 \pm 0.00038}$	$4.4 \times 10^{2} \pm 9.9$	$1.5 \times 10^{2} \pm 0.022$	$4.3 \times 10^{2} \pm 15$
$\mathrm{Wasserstein}_{\mathrm{traj}}[T, 2T]$	$\mathbf{0.31 \pm 0.06}$	$\mathbf{0.30 \pm 0.17}$	1.1 ± 0.23	$9.3 \times 10^{31} \pm 2.3 \times 10^{32}$	5.9 ± 3.5
$t_{\mathrm{MSE}_{\mathrm{vec}} \leq 0.5}$	$\mathbf{20 \pm 3.3\ s}$	$\mathbf{6.0 \pm 3.4\ s}$	–	–	–

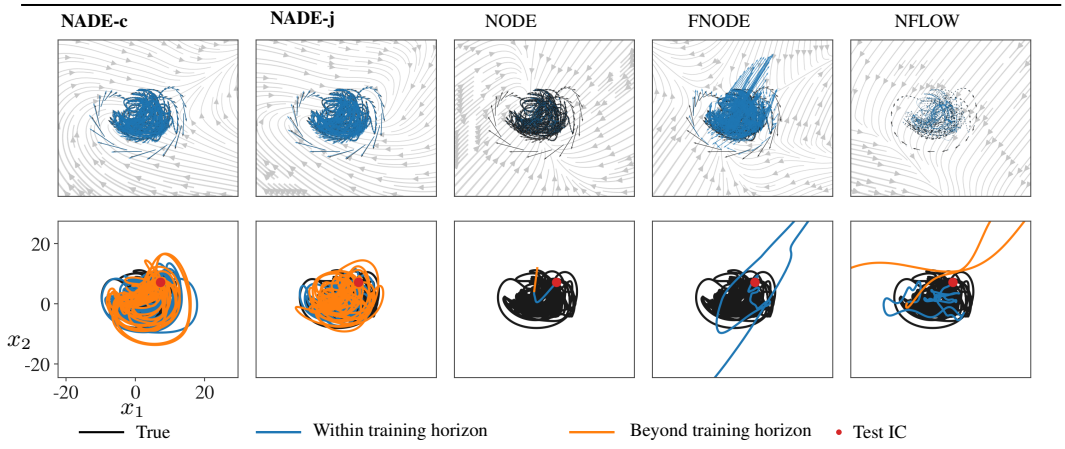


Figure 14: Lorenz96 comparison under the same wall-clock training budget. The table reports vector-field error, trajectory Wasserstein distance over $[T, 2T]$, and the first time at which the vector-field error reaches the indicated threshold. The upper row visualizes two-dimensional projections of the learned vector fields, while the lower row shows a representative held-out rollout in the same projection. The NADE models retain substantially lower vector-field and long-horizon trajectory errors.

where W_1 is the one-dimensional Wasserstein distance. In the experiments, this metric is evaluated over the extrapolation interval $[T, 2T]$. Unlike time-aligned trajectory errors W_{traj} compares the distributions of visited states and is therefore useful for assessing whether a rollout preserves the statistical geometry of the dynamics even when trajectories become phase-shifted or diverge pointwise.