Benchmarking Plan: Neural ODE, PINN, and SINDy on Chen-2000 Dynamical System

This document outlines a detailed experiment plan for benchmarking Neural ODEs, Physics-Informed Neural Networks (PINNs), and Sparse Identification of Nonlinear Dynamics (SINDy) on forecasting and out-of-distribution (OOD) forecasting tasks for the Chen-2000 dynamical system (Equations 11–15). The goal is to rigorously compare forecasting accuracy, generalization, robustness, and computational efficiency.

1. Objectives

- Compare forecasting and OOD forecasting performance of Neural ODE, PINN, and SINDy. - Evaluate accuracy, robustness to noise, and generalization under parameter and initial-condition shifts. - Quantify computational trade-offs (runtime, function evaluations, and memory).

2. Ground Truth Model and Data Generation

Use the Chen-2000 dynamical system (Eqs. 11–15) as ground truth. The system models coupled amplitude dynamics and is integrated numerically using a 4th-order Runge-Kutta (RK4) or adaptive solver.

Simulation Settings: - Time step (dt): 0.001 - Simulation horizon: 100 time units - Integrator: RK4 (reference), optional `torchdiffeq` for Neural ODE - Parameter set: baseline from Chen (2000), varied ±{20%, 50%, 100%} for OOD tests - Number of trajectories: - Training: 1000 trajectories × 1000 steps - Validation: 200 trajectories - In-distribution test: 200 trajectories - OOD test: 200 trajectories per OOD regime

Noise models: - Additive Gaussian noise with $\sigma \in \{0.0, 0.01, 0.05, 0.1\}$ × std(state) - Optional dropout of 5% measurements to test irregular sampling robustness.

3. Out-of-Distribution (OOD) Scenarios

Each OOD dataset introduces one type of distributional shift: 1. **Parameter Shift:** Modify key coupling or damping parameters by ±20%, ±50%, ±100%. 2. **Initial Condition Shift:** Sample initial states from a larger region of phase space. 3. **Forcing Shift:** Add small periodic or stochastic forcing to a state equation. 4. **Temporal Extrapolation:** Train for horizon 10, evaluate on horizon 100.

4. Model Implementations

Neural ODE - Architecture: MLP (2–3 hidden layers, 64 neurons, tanh activation) - Integration: adaptive RK45 via `torchdiffeq` - Training: adjoint method, Adam optimizer, lr=1e-3 - Regularization: L2 weight decay 1e-5 - Evaluation metrics include number of function evaluations (NFE) and solver tolerance sensitivity.

Physics-Informed Neural Network (PINN) - Architecture: 3–4 hidden layers × 128 neurons - Loss: weighted sum of data loss + ODE residual loss - Hyperparameters: residual weight $\lambda \in \{0.1, 1, 10\}$ - Optimizer: Adam + L-BFGS refinement

SINDy - Library: polynomial features up to 3rd order (including cross terms) - Sparse regression: sequential thresholded least squares - Sparsity parameter $\lambda \in \{1e\text{-}3, 1e\text{-}2, 1e\text{-}1\}$ - Evaluate on both

noise-free and noisy data.

5. Evaluation Metrics

Forecasting Metrics: - RMSE, MAE over time horizons {1, 5, 10, 50, 100} - Relative error (normalized RMSE) - Trajectory divergence rate - Computational cost: runtime, NFE, memory usage **OOD Metrics:** - RMSE under parameter/initial shifts - Error vs. shift magnitude curve - Calibration error (if probabilistic) - Fraction of unstable trajectories

6. Experimental Protocol

1. Generate all datasets and save in standardized format (NumPy/HDF5). 2. Train each model with 5 random seeds. 3. Tune hyperparameters on validation RMSE (random search, 50 trials per model). 4. Evaluate on in-distribution and OOD test sets. 5. Record performance metrics, NFE, and runtime. 6. Plot: - Error vs. lead time - Example trajectory overlays - Phase-space plots - Error vs. parameter shift magnitude

7. Statistical Analysis

- Report mean ± std across seeds. - Paired Wilcoxon signed-rank tests for RMSE differences. - Compute effect sizes (Cohen’s d) between models. - Analyze error variance vs. OOD magnitude.

8. Deliverables and Reporting

- Benchmark table: forecasting and OOD RMSE, NFE, runtime. - Diagnostic plots: trajectories, phase portraits, error curves. - Discussion: robustness, interpretability, and computational trade-offs. - Release reproducible code, seeds, and config files.

9. Reproducibility Checklist

- Fixed random seeds and library versions. - Publish Chen-2000 simulator code. - Document solver choices and tolerances. - Save hyperparameter grids and best configurations. - Provide all metrics as CSV/JSON.

End of Plan