DO LEARNED MODELS OF DYNAMICAL SYSTEMS ANSWER INTERVENTIONAL QUESTIONS? A BENCHMARK WITH ORACLE REFERENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

Models of dynamical systems learned from data are used to predict the effects of interventions and to infer which variables drive which, yet they are evaluated mainly on forecasts or recovered equations. We pose five questions to every fitted model: a held-out forecast, a new initial condition, a state perturbation, a clamp do-intervention that replaces one variable's equation, and the causal graph. Each trajectory and intervention score is placed between a trivial reference and an oracle that knows the true functional form and fits only its constants to the same data; graph scores are compared with the complete graph and with chance. Twelve methods that see only 105 noisy samples of one trajectory (sparse regression, Koopman operators, symbolic regression, neural ODEs and a conditional-independence test) are evaluated on all 63 ODEBench systems and on two new sparse-coupling suites, one generated under an analysis plan fixed before any method was run on it. Given the form, the data suffice at low noise: the trajectory-fit oracle generalises to a new initial condition on 98/88/63% of the predictable ODEBench systems at noise 0/1/5%, whereas the best learned method reaches 62% without noise and at most 37% with it; without noise, the vector fields of SINDy, Weak SINDy and PySR fit the training orbit to within 1% (median) yet miss the truth by 11–181% along the new trajectory. With noise, on the 82 systems with an absent edge, all with three to five variables, no learned method's median error under a clamp at the end of the training window is detectably below 1, the error of predicting no response, while the oracle's lies between 0.01 and 0.30. Graph F1 cannot separate any method from the complete graph on ODEBench, which scores 0.955; balanced accuracy and a threshold-free AUROC can, and AUROC exposes read-outs that rank edges well but threshold badly.

1 INTRODUCTION

Models of dynamical systems are often fit to data to address three questions: how the system will evolve, how it responds when a variable is perturbed or held constant (a gene knocked out, a dose fixed), and which variables influence which others. Neural ODEs, sparse regression and symbolic regression are used in machine learning for all three (Chen et al., 2018; Brunton et al., 2016; Cranmer, 2023; Aliee et al., 2021). The first question can be answered by a model that merely reproduces the observed trajectory. The other two require a model of the vector field, as they concern states not represented in the data and the dependencies that generate the dynamics, and one trajectory need not determine that field off its orbit (Stanhope et al., 2014). Evaluations nonetheless score trajectories or graphs separately: equation-discovery benchmarks score reconstructions or recovered expressions (La Cava et al., 2021; d'Ascoli et al., 2024), forecasting benchmarks score rollouts (Gilpin, 2021), and causal-discovery benchmarks score the graphs that causal-discovery algorithms return (Runge et al., 2019a; Herdeanu et al., 2025), not the graphs implied by the fitted models practitioners use for prediction. Interventions have been posed to regularised neural ODEs (Aliee et al., 2021) and to temporal causal models (Thumm et al., 2026); what is missing is to pose them, and the graph, on an external equation-discovery benchmark, across method classes, against references that show what knowing nothing, and knowing the equation's form, would score.

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.

We evaluate twelve *blind* methods, which see only the training samples, from five method classes (a conditional-independence test, four sparse-regression variants, three Koopman operators, two symbolic regressors and two neural ODEs), plus a Granger test of the graph and four methods reported in the appendix only, on all 63 ODEBench systems and on two sparse-coupling suites: 12 systems we wrote and 60 drawn by a fixed generator. Each method is fitted to 105 noisy samples of one trajectory and asked five questions (Figure 1). Our contributions are (i) a protocol that poses the five questions to any fitted model through one interface and places every trajectory and intervention score between a trivial reference and a form-aware oracle, and every graph score against the complete graph and chance (Section 3); (ii) two sparse-coupling suites in ODEBench's format, one drawn under an analysis plan fixed before any method was run on it (Section 5.4); (iii) a harness that runs each method from its released code where usable as released (Appendix A lists the exceptions), with the per-system scores and a script that checks every number we quote; and (iv) four findings, which methods designed against these failures leave in place, each with the reporting practice it implies.

1. **Blind methods fit the training orbit but do not identify the form; given the form, the data suffice at low noise.** The trajectory-fit oracle generalises to a new initial condition on 98/88/63% of the 60 predictable ODEBench systems at noise 0/1/5%, whether its constants start within $\pm 50\%$ of the truth or within a factor of ten; the best blind method reaches 62% without noise and at most 37% with it. Without noise, the vector fields of SINDy, Weak SINDy and PySR match the truth on the training orbit to within 1% (median) but miss it by 11–181% along the new trajectory; with noise, blind fits also miss on the orbit (SINDy, Weak SINDy and EDMD by 9–15% at $\sigma = 0.01$). *Recommendation:* where the form is known, report a form-aware oracle.
2. **With noise, no blind method answers a clamp detectably better than predicting no response on the systems with an absent edge.** On ODEBench's 10 such systems and on the 12 hand-written and 60 generated ones, all with $d = 3$–5, the best blind median error with noise is at least 0.96, against 0.01–0.30 for the oracle; the blind method with the most wins, at either noise level, beats no response on 4 of 10, 8 of 12 and 32 of 60 systems, consistent with chance (sign test $p = 0.75$, 0.39, 0.70). Several methods do beat no response on the 30 fully coupled ODEBench systems, but 28 of those are two-dimensional, so this contrast does not separate sparsity from dimension. On the ten ODEBench systems the verdict holds across clamp sizes but not for a clamp inside the training window, and on all three suites under both Koopman adapters. *Recommendation:* pose mechanism-changing interventions and report the response's size and shape, not only its error.
3. **Graph F1 rewards the complete graph; balanced accuracy and AUROC do not.** Every two-dimensional ODEBench system is fully coupled, so the complete graph scores F1 0.955 and no read-out beats it by more than 0.014. On the generated suite the complete graph scores 0.67; read-outs beat it without noise (the horseshoe group test 0.86), but at 5% noise none by more than 0.001, while balanced accuracy, 0.5 for the complete graph, reaches 0.65 and pooled AUROC separates seven read-outs from chance. *Recommendation:* score graphs by balanced accuracy or specificity and a threshold-free AUROC, on benchmarks with enough absent edges.
4. **A threshold-free AUROC separates a failing threshold from a failing model.** The Neural ODE's Jacobian ranks true edges above absent ones on ODEBench at $\sigma \leq 0.01$ and on the generated suite at every noise level, although its thresholded graph is near-complete; Weak SINDy's group contributions do so on ODEBench at $\sigma \leq 0.01$ but on the generated suite only at $\sigma = 0$; SINDy's are indistinguishable from chance on ODEBench and reach a pooled 0.54 at most on the generated suite. *Recommendation:* report the statistic a read-out thresholds, not only its graph.

2 Related work

Equation discovery and its benchmarks. Symbolic regression recovers closed-form laws from data (Schmidt & Lipson, 2009; Petersen et al., 2021; Cranmer, 2023), recently with language models proposing the forms (Shojaee et al., 2025a). SINDy (Brunton et al., 2016) makes this a sparse linear problem over a fixed library, and later work learns coordinates (Champion et al., 2019), adds weak forms, ensembles, sparsifying priors or implicit, rational libraries (Messenger & Bortz, 2021; Fasel et al., 2022; Hirsh et al., 2022; Kaheman et al., 2020), or emits equations with transformers (Kamienny et al., 2022; d'Ascoli et al., 2024). SRBench (La Cava et al., 2021) and `dysts` (Gilpin, 2021) score reconstruction and forecasting, and Kaptanoglu et al. (2023) compare SINDy variants on the latter's chaotic systems. ODEBench (d'Ascoli et al., 2024) already separates reconstruction

from generalisation to a new initial condition; we do not claim that gap as a finding but use an oracle and an ablation to locate it. Whether one trajectory identifies a system at all is studied for linear-in-parameters models (Stanhope et al., 2014).

Causality in dynamical systems. For ODEs and SDEs, the structural graph and interventions that replace an equation are well defined (Mooij et al., 2013; Hansen & Sokol, 2014; Rubenstein et al., 2018; Bongers et al., 2018; Peters et al., 2017); our clamp is such an intervention. Aliee et al. (2021) is the closest prior work: regularised neural ODEs that recover dynamics and causal structure and predict intervention effects. We ask the same questions of many method classes on a common external benchmark and run their regulariser as one of the methods; Bellot et al. (2022) learn a continuous-time graph with the vector field in the same spirit. Pfister et al. (2019) select kinetic models by stability across conditions. Time-series causal discovery, from Granger causality (Granger, 1969; Tank et al., 2022) to convergent cross mapping (Sugihara et al., 2012) and PCMCI+ (Runge et al., 2019b; Runge, 2020), is benchmarked by CauseMe (Runge et al., 2019a), CausalTime (Cheng et al., 2024), CausalRivers (Stein et al., 2025) and CausalDynamics (Herdeanu et al., 2025), which score the graphs of causal-discovery algorithms; concurrent work generates interventional trajectories from temporal structural causal models (Thumm et al., 2026). We evaluate identification methods on graphs, interventions and trajectories together, against trivial and oracle references.

Koopman operators and surrogates. Koopman theory (Koopman, 1931) represents nonlinear dynamics as a linear operator on observables, approximated by DMD (Schmid, 2010), EDMD (Williams et al., 2015a), kernel DMD (Williams et al., 2015b) and learned embeddings (Lusch et al., 2018); filters in the lifted space are established (Surana & Banaszuk, 2016; Netto & Mili, 2018), and Rupe et al. (2024) define causal mechanisms through Koopman flows. Gaussian processes condition on ODE structure through gradient matching (Calderhead et al., 2008; Wenk et al., 2020) or learn a vector field (Heinonen et al., 2018); FlowGP (Moss et al., 2026) recasts GP sampling as a guided probability-flow ODE (Song et al., 2021). DAPredDNN (Sun et al., 2026) and SE-TFT (Sun et al., 2025) are reduced-order surrogates for parametric systems.

3 BENCHMARK PROTOCOL

Terms. A method is *blind* when it sees only the 105 training samples. A *read-out* turns what a method learned into a graph (Table 1); an *edge score* is the statistic a read-out thresholds, and the *group contribution* of x_i to $\dot{x}_j$ is the size of the summed contribution of all library terms involving x_i, so that cancelling coefficients cancel. A *clamp* holds one variable constant, replacing its equation. An *oracle* knows the functional form and fits only its constants. An *equation family* groups systems whose equations map onto each other by fixing constants or rescaling states and time. A system is *sparse* when its graph has an absent edge.

Data. The nine methods were built and tuned on a *development* suite of seven systems, from a damped oscillator to Lorenz-63 (Lorenz, 1963) and five Kuramoto oscillators (Appendix D), 600–2500 samples long, with additive noise of 0, 3 and 10% of each state's standard deviation. The *test* suite is ODEBench (d'Ascoli et al., 2024): 63 systems (23 one-, 28 two-, 10 three- and 2 four-dimensional), generated with ODEBench's own code and defaults (LSODA, $t \in [0, 10]$, 150 samples, multiplicative noise $\tilde{x} = x(1 + \sigma\epsilon)$ at $\sigma \in \{0, 0.01, 0.05\}$, first sample exact); it shares Lorenz, Van der Pol and the damped harmonic oscillator with the development suite, which can only favour the blind methods. Every method is fitted to the first 70% ($n = 105$ samples) of ODEBench's first initial condition and to nothing else. Two further suites use ODEBench's format and code: a *hand-written* suite of 12 systems ($d = 3$–5; food webs, a repressilator, a toggle switch, coupled oscillators, linear cascades) written after a first ODEBench analysis, and a *random* suite of 60 systems ($d = 3$–5, 20 each) drawn by a fixed generator and seed (one or two parents per damped variable, through linear, bilinear, quadratic, sinusoidal or Hill terms) and kept iff they stay bounded, are not forecast by persistence without noise, have an absent and a present edge, and are predictable. Its generator, acceptance rule and analysis plan were fixed before any method was run on it (`prereg_random_suite.md`; self-attested, but regenerable from its seed and hash).

What was fixed when. Three tiers. (i) The nine methods, their settings, and the forecast, new-initial-condition and perturbation questions with their aggregation rules were frozen on the development suite

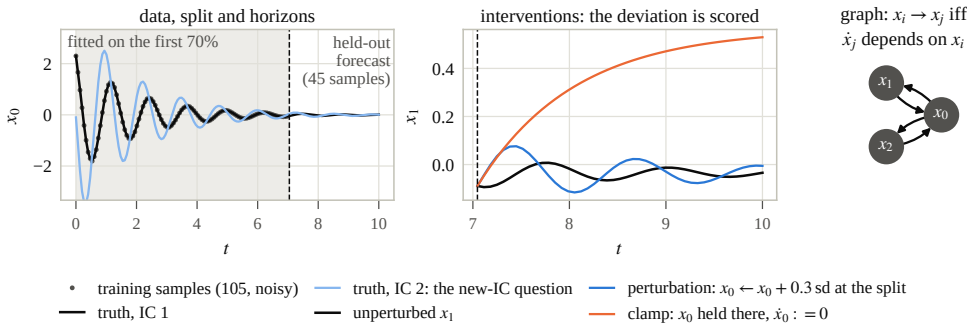


Figure 1: The five questions on ODEBench #57 (Rössler, stable fixed point). Left: every method is fitted to the first 70% of one noisy trajectory (105 samples, grey band) and asked for the held-out 30% and for the whole trajectory from a second initial condition. Middle: at the split, the state is perturbed once ($x_0 \leftarrow x_0 + 0.3\,\mathrm{sd}$) or x_0 is clamped at that value ($\dot{x}_0 := 0$); what is scored is the deviation from the unperturbed run over 45 samples, of every variable for the perturbation and of the other variables for the clamp. Right: the structural graph, $x_i \to x_j$ iff $\dot{x}_j$ depends on x_i; x_1 and x_2 do not depend on each other.

before ODEBench was run. (ii) Everything else was introduced after a first analysis of ODEBench results, with settings fixed without tuning on any suite and run once: the clamp, the AUROC read-out, the two-trajectory ablation, the oracles, the predictability rule, the clustered intervals, the hand-written suite, and every added method and variant (Appendix A); standardised SINDy, motivated by ODEBench #24, is also run on the development suite. (iii) The random suite is the only part of the evidence whose analysis plan was fixed before it was run.

Five questions (Figure 1). (1) The **held-out forecast** of the last 30% (45 samples, 2.95 time units) from the state at the split. (2) A **new initial condition**: the full trajectory from ODEBench's second one, as in d'Ascoli et al. (2024). (3) A **state perturbation**, $x_0 \leftarrow x_0 + 0.3\,\mathrm{sd}(x_0)$ (noise-free sd) once at the split t_s, propagated 45 steps; we score the *deviation* of every variable from the unperturbed run, a local version of (2) that changes a state, not a mechanism. (4) A **clamp do-intervention**: for every variable of every system with $d \geq 2$, replace its equation by $x_i := x_i(t_s) + 0.3\,\mathrm{sd}(x_i)$ held constant (Mooij et al., 2013; Peters et al., 2017) and score the other variables' deviation over 45 steps, pooled over the d clamps. Both start from the noise-free state at t_s, given to every method; the forecast starts from the noisy one. On the 30 fully coupled ODEBench systems the clamp tests the response to a changed mechanism; on the 10 sparse ones, reported separately, it can also penalise a spurious dependence, but mainly through the response's early shape, as a clamp moves every descendant of x_i and only 3 (clamped, responding) pairs have a true response of zero (37 and 120 in the new suites). (5) The **causal graph**, $x_i \to x_j$ ($i \neq j$) iff $\dot{x}_j$ depends on x_i ($\partial f_j / \partial x_i \not\equiv 0$), scored through each method's own read-out and through a threshold-free AUROC of its edge score.

Predictability. A system is left out of a question when its largest Lyapunov exponent λ_1 times the question's horizon is at least $\ln 10$, a heuristic for when an error in the state or the constants grows tenfold within the scored window (exponents: Table 20). It removes Lorenz (#56, $\lambda_1 = 0.91$), Rössler (#59, 0.36) and Chen–Lee (#61, 0.63) from the new-IC questions and only #56 from forecasts and perturbations; no clamped subsystem and no system of the other suites comes near the threshold. Left-out new-IC pairs are scored by valid prediction time (Pathak et al., 2018; Table 19).

References. The *trivial* reference forecasts by persistence, predicts no response (relative error exactly 1), and returns the complete graph. Two *oracles* are given the true functional form: which terms appear, the coefficients that are fixed numbers (those of 29% of all terms) and hence the true graph. They fit only its free constants (median two per system) to the same 105 samples, from the best of five starts drawn relative to ODEBench's constants, with the same sign; neither starts at the true values. The *derivative-fit* oracle regresses the smoothed finite-difference derivatives SINDy receives; the *trajectory-fit* oracle uses no derivative estimate, simulating every 10-sample window from its observed first state and fitting the observed samples. Starts are drawn within $\pm 50\%$ of ODEBench's

constants, and a second pair of oracles starts each constant anywhere within an order of magnitude ($\times 10^u$, $u \sim U[-1, 1]$) to test how much that prior contributes; the trajectory fit also starts from the derivative fit's solution, so for it that test is weaker.

Ablations and variants. The *two-trajectory ablation* ($\sigma = 0.01$) refits the methods that accept several trajectories on the first 70% of both initial conditions and scores them, and the single-trajectory fits, on IC 3, the midpoint of the two, and on IC 4, 50% beyond one of them along the same line (25% where 50% is inadmissible). The *time-unit protocol* re-expresses the frozen windows (11 and 15 samples) in the development suite's time units for five of the methods that use them (Table 12). The Koopman operators also answer the clamp through a second, lifted-space adapter (Appendix A). On the sparse ODEBench systems the clamp is repeated with interventions of 0.1 and 1.0 standard deviations, and applied at 35% of the trajectory instead of the split.

Interface. Each method returns a forecast, a rollout $\mathrm{traj}(\boldsymbol{x}_0, t, \mathrm{clamp})$ and, where it has them, a vector field, edges and edge scores; one scorer poses every question to that interface (Appendix A).

Metrics and uncertainty. For trajectories, $R^2 = 1 - \frac{1}{d}\sum_k \mathrm{MSE}_k/\mathrm{Var}_k$ against the noise-free trajectory, with Var_k over the whole trajectory so that a settled window does not inflate the error, and accuracy is the fraction of systems with $R^2 \geq 0.9$ (d'Ascoli et al., 2024). For responses we report the relative error $\|\hat{\Delta} - \Delta\|_F/\|\Delta\|_F$, $\hat{\Delta}$ being the method's intervened minus its own unintervened rollout from the same state, and its two components: the size ratio $r = \|\hat{\Delta}\|_F/\|\Delta\|_F$ (0: no response, 1: the right size) and the shape, the cosine similarity $\cos\theta$ of $\hat{\Delta}$ and Δ. As the squared error is $1 + r^2 - 2r\cos\theta$, a prediction beats no response iff $\cos\theta > r/2$: a response of the right size needs a cosine above 0.5. The field error is $\|\hat{f} - f\|_F/\|f\|_F$ on the noise-free states of the training window or of the new trajectory; a discrete-time model's field is its one-step change over Δt. A prediction that is non-finite anywhere in its window, or needs more than 10^5 right-hand-side evaluations, is a failure: $R^2 = -\infty$, error $+\infty$, and a miss in every rate. Rates carry 95% Wilson intervals over systems (Wilson, 1927), which ignore the family grouping below, and per-system win counts a two-sided sign test; nothing is corrected for multiple comparisons. ODEBench has 46 equation families (the three Lorenz variants form one), and the 10 sparse systems form 6. Medians, pooled scores, differences and rank correlations carry 95% bootstrap intervals (Efron, 1979) that resample families on ODEBench and systems on the new suites; those not in a main-text table are in Appendices E and F.

4 METHODS UNDER TEST

Selection. The nine methods of our development suite cover different assumptions: PCMCI+ assumes that lagged conditional independences (here, partial correlations) reveal the graph; SINDy assumes sparsity in a library; EDMD, Bayesian Koopman and the horseshoe operator assume linearity in a lifted space; GP-SINDy, our adaptation, assumes a Gaussian-process prior; the Neural ODE assumes no functional form; and DAPredDNN and SE-TFT extrapolate in time without an explicit vector field. For ODEBench we add the references of Section 3; ODEFormer (d'Ascoli et al., 2024), released with ODEBench, whose rates there are not directly comparable with ours, as every method here sees only 70% of the trajectory; a linear Granger test (Granger, 1969) as a graph-only method; a SINDy variant with a standardised library and a ridge penalty (Hoerl & Kennard, 1970) that tests our collinearity diagnosis (Section 5.1); and four methods aimed at the failures we diagnose: weak-form SINDy (Messenger & Bortz, 2021), which needs no pointwise derivative estimate; Ensemble-SINDy (Fasel et al., 2022), which bags the regression; C-NODE (Aliee et al., 2021), a Neural ODE regularised towards sparse dependences whose "variable intervention" is our clamp; and PySR (Cranmer, 2023), a genetic search not confined to a library. GP-SINDy (ours), DAPredDNN and SE-TFT (designed for parametric families) are reported in the appendix only, and no headline claim rests on them. We leave out language-model-guided symbolic regression (Shojaee et al., 2025a), as ODEBench's textbook systems may have been memorised (Shojaee et al., 2025b). Settings are in Appendix A.

Sparse regression and symbolic models. SINDy (Brunton et al., 2016) regresses Savitzky–Golay-smoothed finite-difference derivatives on Θ with sequentially thresholded least squares (threshold 0.05). The standardised variant scales every library column and each derivative to unit variance, thresholds the now dimensionless coefficients, and adds a ridge penalty of $0.01n$ fixed without tuning.

Table 1: What each method learns and how it answers the questions. All dictionary methods share one library $\Theta(\boldsymbol{x})$: monomials of total degree ≤ 3 and $\sin x_i, \cos x_i$ (6, 14, 26, 43 and 66 features for $d = 1, \ldots, 5$). "Probe": a read-out the method's authors do not propose; we add it and label it. For the clamp, discrete-time models iterate their one-step map in state space and overwrite x_i after each step, and ODE models set $\dot{x}_i = 0$.

Method	Learns	Rollouts and interventions	Graph read-out	Edge score
Oracles	true form, constants fitted	integrate ODE	(not scored)	–
PCMCI+	lag-1 parent sets	VAR(1) on parents	CI test, $\alpha = 0.01$	partial correlation
Granger	–	–	F-test, $p < 0.01$	F statistic
SINDy	sparse Ξ, $\dot{\boldsymbol{x}} = \Theta(\boldsymbol{x})\Xi$	integrate ODE	nonzero term	group contribution[§]
SINDy (std.+ridge)	same, standardised Θ, ridge	integrate ODE	nonzero term	group contribution[§]
Weak SINDy	same, weak-form regression	integrate ODE	nonzero term	group contribution[§]
Ensemble SINDy	same, median of 100 bootstrap fits	integrate ODE	nonzero median term	inclusion probability
ODEFormer	symbolic ODE (pretrained)	integrate ODE	variable in equation	Jacobian sensitivity
PySR	symbolic ODE (genetic search)	integrate ODE	variable in equation	Jacobian sensitivity
Koopman/EDMD	$\Theta(\boldsymbol{x}_{t+1}) \approx \Theta(\boldsymbol{x}_t)K$	iterate K	$\lvert K_{kj}\rvert > 10^{-3}$	group contribution[§]
Bayesian Koopman	matrix-normal posterior on K	60 paired draws	95% CI excludes 0	$\max \lvert\mu\rvert/\mathrm{sd}$
Horseshoe Koopman	horseshoe posterior mean K	iterate K (blind)	group test[¶]	post. mean contrib.
Neural ODE	MLP vector field	integrate ODE	mean $\lvert\partial f_j/\partial x_i\rvert$ (probe)	unthresholded
C-NODE	MLP, path-product penalty	integrate ODE	path-product entry	unthresholded

[§]RMS of the summed contribution of all terms involving x_i to output j, over standard deviation of output j. [¶]Terms involving x_i explain $> 3\%$ of output j with posterior probability > 0.95. The four appendix-only rows are in Table 9.

Weak SINDy integrates both sides against 100 compactly supported test functions, PySINDy's implementation with its defaults (de Silva et al., 2020), so no derivative is estimated pointwise; Ensemble SINDy refits on 100 bootstrap resamples and keeps the median coefficient. ODEFormer runs with its README settings, and PySR searches one expression per state over $+, -, \times, \div, \sin, \cos$ with its default budget on the same smoothed derivatives.

Koopman operators. EDMD (Williams et al., 2015a) fits a linear map between consecutive lifted states; Bayesian Koopman places a matrix-normal prior on K; the horseshoe variant fits each column of K by Gibbs sampling (Carvalho et al., 2010; Makalic & Schmidt, 2016), following Bayesian SINDy (Hirsh et al., 2022), reads its graph off a group test per variable, and is rolled out blindly. Its Kalman-filtered form from the development suite (Surana & Banaszuk, 2016; Netto & Mili, 2018), the Horseshoe-KF, assimilates the test observations, so it answers an easier question (appendix only).

Deep models. The Neural ODE (Chen et al., 2018) is a two-hidden-layer tanh MLP trained on random 15-step windows. C-NODE (Aliee et al., 2021) is the same network and trainer on standardised states plus the penalty $\lambda \,\|\, |W^3|\,|W^2|\,|W^1| \,\|_1$ on the product of absolute weight matrices, an upper bound on how strongly each output depends on each input; its authors read the graph off that matrix, and their variable intervention, $x_i := c$ with $f_i := 0$, is our clamp. We fix $\lambda = 0.01$ from their Figure 2 without tuning. Where no graph read-out exists we probe the learned map's Jacobian (Table 1), using each model's own vector field, so no ground truth enters.

5 Results

Table 2 summarises the twelve blind methods and two oracles on 189 system–noise pairs (Figure 4; appendix-only methods: Table 9). On ODEBench no method crashed or hit its time limit. SINDy diverges most often: 27% of its forecasts and 44% of its new-IC rollouts are non-finite, divergences of the fitted models rather than integrator artefacts (Appendix B).

5.1 Off the training orbit: what the oracles and a second trajectory show

Persistence scores $R^2 \geq 0.9$ on 40% of held-out windows, because many ODEBench trajectories have settled by the split, so the new initial condition, scored over 10 time units rather than 2.95, is the harder test. Fitted to the trajectory, the true form generalises to it on 98/88/63% of the 60 systems; fitted to the smoothed derivatives SINDy receives, on 85/78/63%. Without noise the derivative estimate alone costs eight systems, on six of which it misses the true derivative by 18–54%. At $\sigma = 0.05$ both oracles reach only 63%, so there noise limits even a known form. Starting the constants

Table 2: ODEBench summary; each cell is $\sigma = 0 \,/\, 0.01 \,/\, 0.05$. Forecast and new IC: % of the 62 and 60 predictable systems with $R^2 \geq 0.9$. Perturbation (62 systems) and clamp: median relative error of the predicted response (no response $= 1$), the clamp on the 30 fully coupled systems ("dense") and on the 10 with an absent edge ("sparse"). A diverged prediction enters every median as $+\infty$, so a cell reads ∞ exactly when at least half of its population diverged; every median's interval is in Table 10. Bold: the best blind value per column and noise level. Time: median seconds per system on one core. Graph scores are in Table 16; the appendix-only methods in Table 9.

	Forecast ↑ (62)	New IC ↑ (60)	Perturbation ↓ (62)	Clamp, dense ↓ (30)	Clamp, sparse ↓ (10)	Time
references						
Trivial	40/40/40	0/0/0	1.00/1.00/1.00	1.00/1.00/1.00	1.00/1.00/1.00	–
Oracle, trajectory fit	98/97/79	98/88/63	0.00/0.01/0.05	0.00/0.01/0.03	0.00/0.05/0.21	1.6
Oracle, derivative fit	90/89/79	85/78/63	0.00/0.01/0.05	0.00/0.02/0.03	0.06/0.06/0.18	0.1
blind methods						
PCMCI+	26/26/18	3/2/0	0.89/0.88/0.90	1.00/1.00/1.00	1.08/**1.00**/**1.00**	0.0
SINDy	61/47/45	30/17/8	0.72/0.91/1.30	0.92/1.01/3.01	∞/∞/∞	0.2
SINDy (std.+ridge)	82/71/45	37/33/10	0.27/0.37/0.73	0.32/0.32/0.64	0.93/1.49/∞	0.2
Weak SINDy	82/48/34	**62**/22/3	**0.02**/0.98/1.52	**0.01**/1.71/4.72	**0.57**/∞/∞	0.4
Ensemble SINDy	45/29/15	22/10/3	0.81/1.66/4.72	2.12/∞/∞	∞/∞/∞	1.0
ODEFormer	65/58/**61**	32/32/**33**	0.37/0.40/**0.32**	0.93/0.74/**0.54**	1.03/**1.00**/**1.00**	2.8
PySR	**87**/**79**/53	45/**37**/13	0.06/**0.29**/0.67	0.10/**0.24**/0.96	1.05/∞/1.82	160
Koopman/EDMD	74/50/39	22/12/5	1.01/1.07/1.00	0.34/1.13/1.39	∞/∞/∞	0.0
Bayesian Koopman	60/56/42	10/10/7	0.77/0.78/0.84	1.96/∞/∞	∞/∞/∞	0.6
Horseshoe Koopman	56/52/44	20/8/7	1.00/0.87/0.97	0.20/0.60/1.01	0.79/2.15/1.55	44
Neural ODE	52/52/50	18/18/13	0.73/0.73/0.70	0.77/0.78/0.78	0.99/**1.00**/**1.00**	59
C-NODE	56/58/50	3/3/3	0.68/0.69/0.69	0.65/0.68/0.84	1.00/**1.00**/**1.00**	64

anywhere within an order of magnitude instead of within $\pm 50\%$ leaves both oracles' rates unchanged, also on the 22 systems with a constant entering nonlinearly, where a start could matter (Table 11). Without noise the best blind method, Weak SINDy, generalises on 62%; with noise the best, PySR, reaches 37% at $\sigma = 0.01$, and ODEFormer's 33% is best at 0.05.

Given the form, then, 105 samples suffice to estimate the constants at low noise; the blind methods fit the orbit without identifying the form, although for 35 of the 60 systems it lies in their shared library. Without noise, the median field error on the training orbit is 1.0% for SINDy, 0.05% for Weak SINDy and 0.6% for PySR, and along the new trajectory 181%, 11% and 15% (oracle: 0.0% on both); of the blind fits with a field error below 10% on the orbit, 40% generalise (all noise levels pooled). With noise the fits also miss on the orbit (SINDy, Weak SINDy and EDMD by 9–15% at $\sigma = 0.01$, the oracle by 0.7%). A good forecast is nearly necessary but far from sufficient: of the blind (system, σ) pairs whose held-out forecast reaches $R^2 \geq 0.9$, 31% generalise to the new initial condition (39/31/20% at $\sigma = 0$/0.01/0.05), against 0.6% of those whose forecast fails. On the undamped oscillator (#24, Figure 7) x_0 and $\sin x_0$ are nearly collinear on the orbit (correlation 0.99998), and at $\sigma = 0.01$ least squares returns cancelling coefficients ($+677\,x_0$, $-678 \sin x_0$) that reproduce the training orbit's field to 1% and miss it by 59% along the new trajectory.

The methods built to address estimation, and the one searching beyond the library, do not supply the form. Weak SINDy, which never estimates a derivative pointwise, is the best blind method on clean data (new IC 62% against SINDy's 30%) but falls to 22 and 3% with noise; Ensemble SINDy is worse than SINDy on every trajectory question; C-NODE, which adds the penalty and trains on standardised states, trades generalisation for sparser dependences (new IC 3% against the Neural ODE's 18/18/13%); and PySR, not confined to the library, stays far behind the oracle. Windows set in time units give no systematic gain (Table 12), and a standardised library with a ridge penalty raises SINDy's forecast accuracy by 21 $[6, 34]$ and 24 $[8, 38]$ points at $\sigma \leq 0.01$ without detectably helping new initial conditions, a gain that did not replicate on the development suite (Appendix B).

A second training trajectory raises the new-IC accuracy of every blind method that accepts several trajectories (eight of twelve) between the two training initial conditions (IC 3: $+12$ to $+32$ points, intervals excluding zero for all but Ensemble SINDy; Table 15). Beyond them (IC 4), SINDy and its standardised variant keep their gains ($+23$ and $+30$) and the Koopman operators keep less ($+12$ to $+18$), while the two neural ODEs' gains are not detectable, consistent with an inductive bias that extrapolates in the dictionary methods, not the networks.

5.2 INTERVENTIONS: PERTURBATIONS AND CLAMPS

Both oracles answer clamps almost exactly on the 30 fully coupled systems (median error at most 0.03). On the 10 sparse systems the trajectory-fit oracle does too (0.00/0.05/0.21); the derivative-fit oracle's medians are similar (0.06/0.06/0.18), with wide intervals from one family (Table 10). Given the form, the data suffice to estimate the response to a mechanism change on these systems; what the blind methods lack is the form. On the fully coupled systems several blind methods answer the clamp well (Table 2), and with noise the intervals of the Neural ODE (both levels), C-NODE and standardised SINDy ($\sigma = 0.01$) and ODEFormer ($\sigma = 0.05$) lie below 1 (Table 10). On the sparse systems the best blind median with noise is 1.00, the score of predicting no response, and no blind method beats it on more than 4 of the 10 systems at either noise level (sign test $p = 0.75$). Without noise only Weak SINDy's interval excludes 1 (0.57 $[0.06, 0.88]$), and not when systems rather than families are resampled (Appendix B). The contrast is confounded: 28 of the 30 fully coupled systems are two-dimensional, so a clamp leaves a one-dimensional ODE, while every sparse one has three or four variables and six are Lorenz or Rössler variants. With six families these intervals are descriptive, and the random suite (Section 5.4) carries the inference.

An error near 1 hides two different failures, which the response's size and shape separate (Table 10). ODEFormer's and the Neural ODE's responses are 0.57–1.09 times the true size but their shapes are 0.21–0.50 with intervals including 0: about the right size, the wrong shape, which leaves their median errors at 0.99–1.03. C-NODE's are 0.37–0.51 times the true size with shapes of 0.15–0.24, and PCMCI+'s (0.03 with noise) are close to no response at all. The responses that are right have both components: Weak SINDy at $\sigma = 0$ (size 0.88, shape 0.96 $[0.76, 1.00]$) and the horseshoe operator there (0.62 and 0.93). Under neither Koopman adapter does a Koopman row beat no response with noise (Table 13). The verdict survives clamps of 0.1 and 1.0 standard deviations instead of 0.3 (best blind median with noise 1.00 and 0.81 $[0.49, 1.46]$; oracle 0.24 and 0.12 at $\sigma = 0.05$; Table 14). Moving the clamp to 35% of the trajectory, inside the window the methods were fitted on, is the exception: C-NODE's median is 0.93 $[0.60, 0.99]$ at $\sigma = 0.01$ and 0.97 $[0.72, 0.99]$ at $\sigma = 0.05$, the only blind intervals below 1 with noise in any sparse-system setting we ran.

Across the 15 learned methods, clamp rankings agree with perturbation rankings (Spearman 0.46–0.81; Table 18) and forecast rankings (0.49–0.61), but within a system a good forecast is far from sufficient.

5.3 CAUSAL GRAPHS: F1 REWARDS THE COMPLETE GRAPH

The complete graph scores a mean F1 of 0.955, because every two-dimensional system and 2 of the 12 higher-dimensional ones are fully coupled, and no read-out exceeds it by more than 0.014 (Table 16). SINDy, EDMD, Ensemble SINDy and GP-SINDy return the complete graph on all 120 multi-dimensional system–noise pairs, and only 24 absent edges, in 10 systems from 6 families, can penalise one. ODEFormer is the only read-out with high specificity at every noise level (0.88–0.92, lower bounds 0.71–0.75) and high recall (0.79–0.84), and the only one whose balanced accuracy clears the complete graph's 0.5 at every level (0.78, lower bounds 0.58–0.61); PCMCI+ excludes nearly all absent edges but misses 72–90% of true ones, and the other specificities are too uncertain to rank.

Threshold-free AUROC separates read-out from model where the intervals allow it. ODEFormer's equation sensitivities rank edges above chance at every noise level (pooled 0.74–0.75, lower bounds 0.60–0.62), and Weak SINDy's group contributions do at $\sigma \leq 0.01$ (0.74 and 0.67, lower bounds 0.58 and 0.54) although its threshold keeps almost every edge. The Neural ODE's Jacobian does so at $\sigma \leq 0.01$ (pooled 0.68–0.70, lower bounds 0.53), so its near-complete graph is at least partly a thresholding failure; C-NODE's path product (0.51–0.59) and PySR's equations (above chance only at $\sigma = 0.01$) rank edges less well. SINDy's, Ensemble SINDy's and GP-SINDy's scores are indistinguishable from chance (0.49–0.66, 0.51–0.53 and 0.50–0.51, every interval reaching 0.5).

5.4 TWO SPARSE SUITES: ONE WRITTEN, ONE GENERATED UNDER A FIXED PLAN

The hand-written suite has 12 systems ($d = 3$–5, 71 absent edges) and the random suite 60 ($d = 3$–5, 411 absent edges; 41 of its 60 systems lie in the shared library), both answered with the same frozen settings (Appendix F). The random suite's analysis plan fixed four claims and their criteria before

Table 3: Random suite: the four claims fixed before any method was run on it, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$.

Claim	Criterion in the plan	Result	Verdict
1 Clamp	no blind upper bound below 1 with noise; oracle median below 0.25	no blind bound below 1; oracle 0.00/0.03/0.30	not confirmed
2 Form	oracle new-IC accuracy above every blind method's by at least 30 points	100/92/33% against 13/7/2%	confirmed
3 Graphs	complete-graph F1 below 0.7; some AUROC above chance; SINDy's and EDMD's at chance	F1 0.67; ten ($\sigma = 0$) and seven ($\sigma = 0.05$) read-outs; SINDy 0.537 $[0.503, 0.568]$ at $\sigma = 0$	not confirmed
4 Baselines	no added method's clamp upper bound below 1 with noise	none	confirmed

any method was run on it (`prereg_random_suite.md`). Two hold as written and two fail one criterion each (Table 3; scores: Figure 12 and Table 25).

Claim 1 fails on its oracle half. With noise no blind interval lies below the reference and the best blind median rounds to it (C-NODE, 0.999 and 1.001), but the oracle's 0.30 $[0.15, 0.38]$ at $\sigma = 0.05$ exceeds the planned 0.25: with noise even a known form leaves a visible error on these systems. Without noise, by contrast, two blind methods do beat no response here: the horseshoe operator (0.26 $[0.20, 0.42]$) and Weak SINDy (0.71 $[0.57, 0.88]$; Table 26). *Claim 2 holds*, at $\sigma = 0.05$ by one system: the oracle generalises on 20 of 60 and the best blind method on 1, where 30 points need a gap of 18. *Claim 3 fails on SINDy.* F1 separates read-outs only without noise (horseshoe group test 0.86, PySR 0.74, complete graph 0.67); at $\sigma = 0.05$ none beats the complete graph by more than 0.001. Balanced accuracy (complete graph: 0.5) still separates read-outs there (ODEFormer and PySR 0.65, lower bounds 0.61), and so does pooled AUROC: ten read-outs exclude chance without noise and seven at $\sigma = 0.05$ (Table 27), three of the ten only marginally, led by the horseshoe operator's posterior-mean contributions at 0.901 $[0.868, 0.929]$; but SINDy's sits just above chance without noise (0.537 $[0.503, 0.568]$), where the plan said it would not. Claim 4 holds.

The hand-written suite behaves alike (Appendix F): the oracle generalises on 100/100/75%, the best blind method on at most 17%, no blind clamp interval with noise falls below 1 (best median 0.96 $[0.79, 1.12]$), and seven read-outs beat its complete graph's F1 of 0.59 at every level.

6 Discussion

What to report. Beyond the recommendations of Section 1: state where an intervention is applied, as the only blind intervals below the reference with noise on sparse systems came from a clamp inside the fitted window; and do not read a forecast from the split as evidence of a mechanism, as persistence scores $R^2 \geq 0.9$ on 40% of ODEBench's held-out windows.

Limitations. Settings were tuned on the development suite (the nine original methods) or fixed without tuning (every added method), never on a test suite, and each method is fitted once with one seed and one noise draw; a tuned control would bound how much of the oracle gap tuning closes. One trajectory need not identify a vector field off its orbit (Stanhope et al., 2014), so we partly measure whether a method's inductive bias matches the system. The sparse evidence rests on 10 ODEBench systems (6 families), 12 written after seeing ODEBench results and 60 generated ones fixed in advance, all with three to five variables. Gray-box models (Rackauckas et al., 2020) and language-model-guided symbolic regression are not included, and every system is synthetic, unlike physical testbeds (Gamella et al., 2025; Appendix C).

7 Conclusion

Fitting one trajectory well usually did not mean generalising off it: of the blind fits that forecast ODEBench's held-out window with $R^2 \geq 0.9$, 31% generalised to a new initial condition, while an oracle given the form and the same samples generalised on 63–98%. With noise, no blind method answered a clamp on a system with an absent edge detectably better than predicting no response. F1 could not separate any method from the complete graph on ODEBench's dense graphs; balanced accuracy and a threshold-free AUROC could. Benchmarks should report form-aware oracles, mechanism-changing interventions and threshold-free graph scores alongside forecasts.

REPRODUCIBILITY STATEMENT

All code is in the supplementary material, with every per-system score and the predictions of the ODEBench and hand-written runs (those of the random suite exceed the upload limit; its scores are included). `benchmark/make_data.py` generates the ODEBench trajectories, the two extra initial conditions and the two sparse suites (`sparse_systems.py`; `random_systems.py`, whose rule and analysis plan are in `prereg_random_suite.md`) with ODEBench's own functions, and `benchmark/lyapunov.py` computes the exponents of the predictability rule. `benchmark/run_odeformer.py` and `benchmark/run_pysr.py` fit the two symbolic models in their own environments, and `benchmark/run_odebench.py` runs every other method, from its original source files where these are usable as released (Appendix A), and scores all of them. `benchmark/aggregate.py` and `benchmark/figures.py` regenerate every number, interval and figure, and every table but the three written by hand (Tables 1, 3 and 4), from the saved scores and predictions. The checks of Appendix B are the other `benchmark/check_*.py` scripts, and `check_paper_numbers.py` verifies every number quoted in the text. Section 3 and Appendix A give the complete protocol, including which analyses were added after a first ODEBench analysis. The ODEBench grid takes about 160 minutes, the hand-written suite about 80 and the random suite about 170 on 13 CPU cores, and the symbolic models are fitted separately in their own environments.

AI USE STATEMENT

Following the ICLR 2027 policy, we state for each research subtask whether a large language model (a coding assistant) was used. *Required disclosures, used:* it implemented the benchmark harness, the aggregation, plotting and validation scripts and the adapters that connect each released method to the harness (*implement methods*); as part of the harness, it wrote the scripts that regenerate ODEBench's trajectories with ODEBench's own code and store all three suites in the harness's format (*clean and reformat data sets*), and it generated the random suite by running the fixed generator of Section 3 (*generate synthetic data sets*); it proposed and gave feedback on parts of the methodology, including the oracle references, the AUROC read-out, the predictability rule and the random suite's acceptance rule (*design or provide feedback on research methodology or experiments*); it gave feedback on the conceptual framework, the five questions and the oracle bracket, which the authors designed (*help develop conceptual frameworks*); it refined the wording and criteria of hypotheses the authors proposed, including the four claims fixed for the random suite (*propose or refine hypotheses*); and it drafted interpretations of the results that the authors reviewed (*interpret results*). *Required disclosures, not used or not applicable:* it did not assist with translation; the paper states no theorems, so formulating mathematical claims and providing ingredients for or writing proofs do not apply; and there is no qualitative or thematic data analysis. *Recommended disclosures:* it created and edited software code, created the figures, drafted and edited parts of this paper, identified related literature, and formatted references. The methods' implementations are imported where possible, with four small pieces re-stated and two methods ported (Appendix A); the ported methods pass their own self-checks, the shared library is asserted equal to PySINDy's, and the graph scoring is checked against known adjacency matrices. Every number quoted in the text is checked by script against the result files, and every reference was matched against its Crossref, arXiv or proceedings record. The authors reviewed all LLM-assisted code and text and take responsibility for the content.

REFERENCES

Hananeh Aliee, Fabian J. Theis, and Niki Kilbertus. Beyond predictions in neural ODEs: Identification and interventions. *arXiv preprint arXiv:2106.12430*, 2021.

Alexis Bellot, Kim Branson, and Mihaela van der Schaar. Neural graphical modelling in continuous-time: consistency guarantees and algorithms. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=SshBkfeRF9L`.

Stephan Bongers, Tineke Blom, and Joris M. Mooij. Causal modeling of dynamical systems. *arXiv preprint arXiv:1803.08784*, 2018.

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

Ben Calderhead, Mark Girolami, and Neil D. Lawrence. Accelerating Bayesian inference over nonlinear differential equations with Gaussian processes. In *Advances in Neural Information Processing Systems*, volume 21, 2008.

Carlos M. Carvalho, Nicholas G. Polson, and James G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116 (45):22445–22451, 2019.

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

Yuxiao Cheng, Ziqian Wang, Tingxiong Xiao, Qin Zhong, Jinli Suo, and Kunlun He. CausalTime: Realistically generated time-series for benchmarking of causal discovery. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=iad1yyyGme.

Miles Cranmer. Interpretable machine learning for science with PySR and SymbolicRegression.jl. *arXiv preprint arXiv:2305.01582*, 2023.

Stéphane d'Ascoli, Sören Becker, Philippe Schwaller, Alexander Mathis, and Niki Kilbertus. ODEFormer: Symbolic regression of dynamical systems with transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=TzoHLiGVMo.

Brian M. de Silva, Kathleen Champion, Markus Quade, Jean-Christophe Loiseau, J. Nathan Kutz, and Steven L. Brunton. PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, 5(49):2104, 2020.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979.

Urban Fasel, J. Nathan Kutz, Bingni W. Brunton, and Steven L. Brunton. Ensemble-SINDy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control. *Proceedings of the Royal Society A*, 478(2260):20210904, 2022.

Juan L. Gamella, Jonas Peters, and Peter Bühlmann. Causal chambers as a real-world physical testbed for AI methodology. *Nature Machine Intelligence*, 7(1):107–118, 2025.

William Gilpin. Chaos as an interpretable benchmark for forecasting and data-driven modelling. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

Mingming Gong, Kun Zhang, Bernhard Schölkopf, Dacheng Tao, and Philipp Geiger. Discovering temporal causal relations from subsampled data. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1898–1906, 2015.

Clive W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

Niels Hansen and Alexander Sokol. Causal interpretation of stochastic differential equations. *Electronic Journal of Probability*, 19, 2014. doi: 10.1214/EJP.v19-2891.

Markus Heinonen, Cagatay Yildiz, Henrik Mannerström, Jukka Intosalmi, and Harri Lähdesmäki. Learning unknown ODE models with Gaussian processes. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1959–1968, 2018.

Benjamin Herdeanu, Juan Nathaniel, Carla Roesch, Jatan Buch, Gregor Ramien, Johannes Haux, and Pierre Gentine. CausalDynamics: A large-scale benchmark for structural discovery of dynamical causal models. In *Advances in Neural Information Processing Systems*, 2025.

Seth M. Hirsh, David A. Barajas-Solano, and J. Nathan Kutz. Sparsifying priors for Bayesian uncertainty quantification in model discovery. *Royal Society Open Science*, 9(2):211823, 2022.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Kadierdan Kaheman, J. Nathan Kutz, and Steven L. Brunton. SINDy-PI: a robust algorithm for parallel implicit sparse identification of nonlinear dynamics. *Proceedings of the Royal Society A*, 476(2242):20200279, 2020.

Pierre-Alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and François Charton. End-to-end symbolic regression with transformers. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Alan A. Kaptanoglu, Lanyue Zhang, Zachary G. Nicolaou, Urban Fasel, and Steven L. Brunton. Benchmarking sparse system identification with low-dimensional chaos. *Nonlinear Dynamics*, 111 (14):13143–13164, 2023.

Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the Koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020.

Bernard O. Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabrício Olivetti de França, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason H. Moore. Contemporary symbolic regression methods and their relative performance. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4): 1748–1764, 2021.

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

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9:4950, 2018.

Enes Makalic and Daniel F. Schmidt. A simple sampler for the horseshoe estimator. *IEEE Signal Processing Letters*, 23(1):179–182, 2016.

Daniel A. Messenger and David M. Bortz. Weak SINDy: Galerkin-based data-driven model selection. *Multiscale Modeling & Simulation*, 19(3):1474–1497, 2021.

Joris M. Mooij, Dominik Janzing, and Bernhard Schölkopf. From ordinary differential equations to structural causal models: the deterministic case. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, pp. 440–448, 2013.

Henry B. Moss, Lachlan Astfalck, Thomas Cowperthwaite, Colin Doumont, Sam Willis, Philipp Hennig, Christopher Nemeth, and Andrew Zammit-Mangion. Conditioning Gaussian processes on almost anything. *arXiv preprint arXiv:2605.21041*, 2026.

Marcos Netto and Lamine Mili. A robust data-driven Koopman Kalman filter for power systems dynamic state estimation. *IEEE Transactions on Power Systems*, 33(6):7228–7237, 2018.

Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical Review Letters*, 120(2):024102, 2018.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. MIT Press, 2017.

Brenden K. Petersen, Mikel Landajuela Larma, Terrell N. Mundhenk, Claudio Prata Santiago, Soo Kyung Kim, and Joanne Taery Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=m5Qsh0kBQG.

Niklas Pfister, Stefan Bauer, and Jonas Peters. Learning stable and predictive structures in kinetic systems. *Proceedings of the National Academy of Sciences*, 116(51):25405–25411, 2019.

Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

Paul K. Rubenstein, Stephan Bongers, Bernhard Schölkopf, and Joris M. Mooij. From deterministic ODEs to dynamic structural causal models. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*, pp. 114–123, 2018.

Jakob Runge. Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, volume 124 of *Proceedings of Machine Learning Research*, pp. 1388–1397, 2020.

Jakob Runge, Sebastian Bathiany, Erik Bollt, Gustau Camps-Valls, Dim Coumou, Ethan Deyle, et al. Inferring causation from time series in Earth system sciences. *Nature Communications*, 10:2553, 2019a.

Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances*, 5 (11):eaau4996, 2019b.

Adam Rupe, Derek DeSantis, Craig Bakker, Parvathi Kooloth, and Jian Lu. Causal discovery in nonlinear dynamical systems using Koopman operators. *arXiv preprint arXiv:2410.10103*, 2024.

Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K. Reddy. LLM-SR: Scientific equation discovery via programming with large language models. In *International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=m2nmp8P5in.

Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D. Doan, and Chandan K. Reddy. LLM-SRBench: A new benchmark for scientific equation discovery with large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55325–55359, 2025b.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Shelby Stanhope, Jonathan E. Rubin, and David Swigon. Identifiability of linear and linear-in-parameters dynamical systems from a single trajectory. *SIAM Journal on Applied Dynamical Systems*, 13(4):1792–1815, 2014.

Gideon Stein, Maha Shadaydeh, Jan Blunk, Niklas Penzel, and Joachim Denzler. CausalRivers – scaling up benchmarking of causal discovery for real-world time-series. In *International Conference on Learning Representations*, 2025.

George Sugihara, Robert May, Hao Ye, Chih-hao Hsieh, Ethan Deyle, Michael Fogarty, and Stephan Munch. Detecting causality in complex ecosystems. *Science*, 338(6106):496–500, 2012.

Shuwen Sun, Lihong Feng, and Peter Benner. Spatially-enhanced temporal fusion transformer: Interpretable multi-output prediction for parametric dynamical systems with time-varying inputs. *arXiv preprint arXiv:2505.00473*, 2025.

Shuwen Sun, Lihong Feng, and Peter Benner. Data-augmented predictive deep neural network: Enhancing the extrapolation capabilities of non-intrusive surrogate models. *Computer Methods in Applied Mechanics and Engineering*, 450:118604, 2026. doi: 10.1016/j.cma.2025.118604.

Amit Surana and Andrzej Banaszuk. Linear observer synthesis for nonlinear systems using Koopman operator framework. *IFAC-PapersOnLine*, 49(18):716–723, 2016.

Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily B. Fox. Neural Granger causality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(8):4267–4279, 2022.

Dennis Thumm, Billy Tim Anthony, and Ying Chen. DoTime: A synthetic benchmark generator for interventional and counterfactual time series. *arXiv preprint arXiv:2607.27263*, 2026.

Philippe Wenk, Gabriele Abbati, Michael A. Osborne, Bernhard Schölkopf, Andreas Krause, and Stefan Bauer. ODIN: ODE-informed regression for parameter and state inference in time-continuous dynamical systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6364–6371, 2020.

Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015a.

Matthew O. Williams, Clarence W. Rowley, and Ioannis G. Kevrekidis. A kernel-based method for data-driven Koopman spectral analysis. *Journal of Computational Dynamics*, 2(2):247–265, 2015b.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

A Hyperparameters and adapter choices

All settings of the nine development-suite methods are those of the development benchmark; none were changed for ODEBench or the two sparse suites. The rows added after a first ODEBench analysis (oracles, Granger, the SINDy variants, ODEFormer, blind horseshoe rollout, Weak and Ensemble SINDy, C-NODE, PySR) use settings fixed without tuning on any suite: generic defaults (Granger, ODEFormer's README, PySINDy's weak-form and ensemble defaults, PySR's defaults), the development suite's settings (blind horseshoe, C-NODE's trainer), or one fixed value (the SINDy ridge weight, C-NODE's λ, the oracles' starts and window length). Every source file named here is used as released. Four small pieces are re-stated in the harness because the originals are reachable only through scripts with hard-coded data paths: the Neural ODE trainer and the Bayesian-EDMD fit (`src/learn_spotlight.py`), the horseshoe operator fit (`src/horseshoe_koopman.py`) and the in-memory forms of the Neural-ODE and Bayesian-Koopman edge rules (`src/causal_graphs_all.py`); each is the same computation, generalised to a list of trajectories. Clipping the horseshoe operator's eigenvalues to the unit disc was considered for the blind rollout and dropped, because it also removes genuine growth such as the early phase of logistic dynamics.

Provenance of the external code. Every external component is a published, publicly archived artefact, used under its own license: ODEBench's generator and ODEFormer's weights from the ODEBench repository, the DAPredDNN code from its Zenodo archive (`10.5281/zenodo.13940888`), and the SE-TFT code from its Zenodo record (18772646). The kernel-DMD half of DAPredDNN is the authors' file used unmodified; its autoencoder and feed-forward network are ported to torch because the release is TensorFlow/Keras and this harness is torch-only. SE-TFT is ported layer by layer from the release and run at reduced width and epoch budget (above). Both ports pass assertion-based self-checks against the released algebra (Appendix B).

The appendix-only rows. GP-SINDy is our adaptation of the FlowGP sampler (Moss et al., 2026), which samples a GP conditioned on non-Gaussian information by integrating a guided probability-flow ODE and, in the original work, conditions on a *known* equation. The adaptation leaves the equation unknown and draws the trajectory on the observation grid jointly with SINDy-style coefficients Ξ; the data enter as a GP likelihood, and the ODE residual $\dot{\boldsymbol{x}} - \Theta(\boldsymbol{x})\Xi$ (on central differences) and a Laplace sparsity term enter as the non-Gaussian condition. Appendix B checks it against known coefficients. DAPredDNN (Sun et al., 2026) extrapolates an autoencoder's latent trajectory with kernel DMD (Williams et al., 2015b) and trains a network from time to latent state; SE-TFT (Sun et al., 2025) extends the temporal fusion transformer (Lim et al., 2021) to multiple outputs and reads couplings off its attention matrix. Both were designed for parametric families and run at reduced width here. The Horseshoe-KF is the Kalman filter of the development suite on the horseshoe operator; on the development suite, blind rollouts of the lightly shrunk operator diverged on 5 of 21 combinations, which is why it was deployed as a filter there.

Table 4: Per-method settings.

Method	Settings
Oracle, derivative fit	ODEBench's functional form with its constants as free parameters; `scipy` least squares (TRF, ≤ 200 evaluations per constant) on the residual of the smoothed derivatives SINDy uses, scaled per state; best of five starts drawn uniformly in $[0.5, 1.5]\times$ ODEBench's constants; integrated with LSODA. A variant started at the true constants gives identical accuracy rates and medians within 0.002.
Oracle, trajectory fit	Same form and starts, plus the derivative fit's solution; residual: every window of 10 samples starting at every second training sample, simulated from its observed first state by classical RK4 (10 substeps per sample) and compared with the observed samples, scaled per state (non-finite residuals clipped to 10^3); least squares with ≤ 30 evaluations per constant; lowest cost of the six fits.
Oracles, wide starts	As above, with each constant's five starts drawn as its value times 10^u, $u \sim U[-1, 1]$; every other setting unchanged.
Weak SINDy	PySINDy 2.1 `WeakPDELibrary` over the same library: $K = 100$ test functions $((t-a)(b-t))^4$ of half-width $T/20$ placed at random (seed 0), STLSQ threshold 0.05, 10 iterations; the library's columns are matched to the shared dictionary numerically and asserted a bijection.
Ensemble SINDy	PySINDy 2.1 `EnsembleOptimizer`: 100 bootstrap resamples of the (library, derivative) rows with replacement (seed 0), STLSQ threshold 0.05 on each, median coefficients; edge score: largest inclusion probability among the terms involving x_i in output j; same derivatives and rollout as SINDy.
C-NODE	The Neural ODE's network and trainer on states standardised to unit variance; loss $+ \lambda \| \, \lvert W^3 \rvert \lvert W^2 \rvert \lvert W^1 \rvert \, \|_1$ with $\lambda = 0.01$; graph: entries of the path-product matrix above 5% of its largest (the same relative rule as the Jacobian probe, which is also reported); vector field mapped back to raw units for every rollout.
Time-unit protocol	Savitzky–Golay window $\lceil 0.22/\Delta t \rceil$ rounded up to an odd number ≥ 5 (5 on ODEBench); Neural ODE training windows $\lceil 0.30/\Delta t \rceil$ samples (5 on ODEBench); nothing else changed.
PCMCI+	ParCorr test, $\tau_{\max} = 1$, $\alpha_{\mathrm{PC}} = 0.01$ (tigramite); input stride 1 (generic default); VAR(1) by OLS on discovered parents $\cup$ self-lag.
Granger	OLS VAR(1) with intercept; F-test for dropping x_i's lag from x_j's row, $p < 0.01$.
SINDy	Library Θ (Table 1); derivatives from a Savitzky–Golay filter (window 11, order 3) and central differences; STLSQ threshold 0.05, 10 iterations; rollout RK45, $\mathrm{rtol} = \mathrm{atol} = 10^{-6}$.
SINDy (std.+ridge)	As SINDy; library columns and derivatives standardised; STLSQ threshold 0.05 on standardised coefficients; ridge $0.01n$ (bias unpenalised, never thresholded). Std. only: the same without ridge. Ridge only: ridge $0.01n$ on the raw library at SINDy's threshold.
ODEFormer	Pretrained weights, beam size 50, beam temperature 0.1, candidates ranked by the model's own fit metric; separate environment (torch 2.4.1, sympy 1.11.1, numpy 1.26.4); top equation integrated with LSODA.
PySR	PySR 2.4.1 (SymbolicRegression.jl through juliacall) with its defaults: binary operators $+, -, \times, \div$, unary $\sin, \cos$ (added; the shared dictionary's functions), 100 iterations, 31 populations of 27, maximum size 30, `model_selection="best"`; targets: the Savitzky–Golay derivatives of SINDy, one expression per state; serial, deterministic, `random_state`=0, wall-clock cap 600 s per fit; selected equation integrated with LSODA.
Koopman/EDMD	Least-squares K on Θ; state read from the linear coordinates; edge threshold $\lvert K \rvert > 10^{-3}$.
Bayesian Koopman	Matrix-normal prior, $\tau^2 = 25$; plug-in residual covariance; 60 posterior draws; edge if $\lvert \mu \rvert > 1.96\,\mathrm{sd}$.
Horseshoe (KF, blind)	Standardised design; Gibbs sampler of Makalic & Schmidt (2016), 8000 draws after 1500 burn-in per output column; group test $\epsilon = 0.03$, confidence 0.95. Filter: $Q =$ residual covariance, $P_0 = Q$, $R = \mathrm{diag}(\sigma^2\, \overline{x_k^2})$ for multiplicative noise. Blind: posterior-mean K iterated like EDMD.
GP-SINDy	Per-state RBF GP, hyperparameters and noise by marginal likelihood; grid = training samples; $\sigma_{\text{phys}} = 0.10$, Laplace weight 0.02, coefficient prior sd 3 (derivative-sd units); 48 paths, 400 Euler steps on a log-SNR grid, $S = 5$; edge contribution threshold 0.08, $P(\text{edge}) > 0.5$.
Neural ODE	MLP $d \to 32 \to 32 \to d$, tanh; Adam 10^{-3}, 1500 steps on random 15-step windows; dopri5 (torchdiffeq); graph probe: mean $\lvert \partial f_j / \partial x_i \rvert$ over every fifth training sample, edge above 5% of the largest entry.
DAPredDNN	Dense AE $d \to 32 \to 16 \to d$ (swish); KDMD Gaussian kernel shape 10, rank threshold 0.997, first 10% dropped (authors' code); FFNN $1 \to 8 \to \ldots \to 128 \to d$; epochs 1000 / 10000 / 20000, batch 256, $\alpha = 0.1$; MinMax scaling on the training window only.
SE-TFT	$d_{\text{model}} = 64$ (paper: 160), 4 heads, dropout 0.2, $n_k = 9$ history, $n_\tau = 32$ horizon, ≤ 600 windows, quantile loss $\{0.1, 0.5, 0.9\}$, Adam 10^{-3}, clip 1.0, 500 epochs (paper: 5000); one shared target scaler as released; attention edge above the $1/d$ fair share.

Adapter choices. (i) Discrete-time models (VAR, the three Koopman operators, DAPredDNN's one-step map) answer the clamp by iterating their map in state space: they re-lift the state and overwrite x_i after every step, and their unclamped baseline uses the same scheme. ODE models

set $\dot{x}_i = 0$ from a clamped initial state, and GP-SINDy does so in every posterior draw. SE-TFT predicts one step at a time from a history ending at the clamped split state. (ii) DAPredDNN's new-IC and perturbation predictions use its autoencoder–KDMD branch, seeded at the given state, because its online network has no state input. (iii) SE-TFT needs a 9-step history, so its new-IC rollout is conditioned on the first 9 noisy samples of the second trajectory and scored from sample 10 onwards. It is not scored on the third initial condition, for which no observations exist. The same holds for the Horseshoe-KF. (iv) The library is evaluated with a vectorised implementation that matches PySINDy's (de Silva et al., 2020) output column by column, and this is asserted on every system. (v) Rollouts that need more than 10^5 right-hand-side evaluations are stopped and scored as failures (for torchdiffeq, $\lfloor 10^5/6 \rfloor$ steps), and each method has a 20–60 minute wall-clock limit per system. (vi) For read-outs that threshold a statistic (PCMCI+, Granger, Bayesian Koopman, the horseshoe group test, GP-SINDy, the Jacobian probes, C-NODE's path product, SE-TFT's attention, Ensemble SINDy's inclusion probability), the continuous edge score is that statistic before thresholding. SINDy, its standardised and weak-form variants and EDMD test individual coefficients, so for them we use the group contribution of Table 1, which sums collinear terms before measuring them. For ODEFormer and PySR the score is a Jacobian-based sensitivity of the equation. (vii) The second Koopman clamp adapter iterates the lifted vector $g \leftarrow gK$ as the forecast does and overwrites only the coordinate of x_i after every step; the other observables are never re-lifted, so terms such as $x_i x_j$ evolve under K rather than being recomputed from the clamped value. Refitting K with x_i as an exogenous input is not a third adapter: least squares, the matrix-normal posterior mean and the per-column Gibbs sampler all fit each output column separately, so the columns that predict the other variables are unchanged and the resulting rollout is the first adapter's.

B VALIDATION CHECKS

Port self-checks. Each ported method ships an assertion-based self-check, and all three pass unchanged. SE-TFT's block-wise causal mask equals the released TensorFlow algebra for several sizes; a small model trained on a clean damped oscillator rolls out blindly with NRMSE 0.17, and its one-step Jacobian recovers both couplings within 0.65–$1.07\times$ of the analytic values. DAPredDNN's online path and its autoencoder–KDMD branch extrapolate the same oscillator with NRMSE 0.18 and 0.19, and its Jacobian recovers both couplings within 0.98–$1.05\times$. The FlowGP sampler reproduces closed-form GP conditioning on a linear-Gaussian condition (maximum error 0.039 in the mean and 0.020 in the standard deviation). The shared library is asserted equal to PySINDy's on every benchmark system.

Graph scoring. On every sparse system, feeding the true adjacency through the benchmark's scoring code as an edge score gives AUROC 1, and its negation gives 0, so the threshold-free scores are orientation-correct.

SINDy divergences. Every SINDy prediction scored non-finite (51 forecasts and 83 new-IC rollouts) was re-integrated with LSODA, without a blow-up event or step limit and with a 2×10^6-evaluation budget. One forecast and three rollouts became finite, and none reached $R^2 \geq 0.9$, so the evaluation cap does not change any SINDy accuracy.

Standardised SINDy, and each of its two changes. The standardised variant was motivated by an ODEBench system, so we test it on the development suite, where the diagnosis was not made, and we split it into its two changes: a standardised library without ridge, and a ridge weight of $0.01n$ on the raw library (Table 5). Plain SINDy is refitted on the development suite exactly as in the development benchmark, and its forecast errors are asserted equal to the stored ones. There the combined variant does *not* transfer. Without noise it is worse on all three trajectory questions (median forecast NRMSE 0.42 against 0.02). At 3% noise it is worse on forecasts (0.76 against 0.31) and better on perturbations and clamps, where both variants predict worse than no response. At 10% forecasts and perturbations are similar, and plain SINDy's clamps diverge (median ∞ against 2.96). The degradation of the forecasts comes from the ridge: standardisation alone matches plain SINDy there (0.020 against 0.019), and the ridge alone raises the error to 0.34; on perturbations and clamps standardisation alone is also worse (0.60 against 0.31, and a diverged median against 0.44). On ODEBench, neither change alone reproduces the combined forecast gain at $\sigma = 0$ (Table 18), while the ridge alone gives most of the clamp improvement. The ODEBench gain is therefore consistent with the collinearity diagnosis,

but we cannot separate it from a simpler account: a ridge term changes which terms sequential thresholding keeps, which helps on some libraries and hurts on others.

Table 5: The four SINDy variants. Top: development suite (7 systems; 0/3/10% noise), medians over systems where a diverged prediction counts as $+\infty$, and mean F1. Bottom: ODEBench, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$, as in Table 2, with mean per-system AUROC.

development suite	Forecast NRMSE	Perturbation	Clamp	F1
SINDy	0.02/0.31/0.78	0.31/1.27/1.25	0.44/2.92/∞	1.00/0.95/0.94
SINDy (std. only)	0.02/0.30/0.76	0.60/1.36/1.39	∞/3.12/∞	1.00/0.96/0.91
SINDy (ridge only)	0.34/0.83/0.84	0.73/1.25/1.43	∞/2.50/∞	1.00/0.97/0.96
SINDy (std.+ridge)	0.42/0.76/0.76	0.77/1.11/1.15	1.58/1.35/2.96	0.99/0.97/0.94

ODEBench	Forecast (%)	New IC (%)	Perturbation	Clamp	AUROC
SINDy	61/47/45	30/17/8	0.72/0.91/1.30	0.92/1.01/3.01	0.39/0.50/0.52
SINDy (std. only)	63/55/44	35/17/10	0.36/1.02/1.74	1.03/0.88/6.16	0.39/0.49/0.47
SINDy (ridge only)	60/56/44	22/17/7	0.45/0.49/0.71	0.46/0.44/0.57	0.50/0.66/0.61
SINDy (std.+ridge)	82/71/45	37/33/10	0.27/0.37/0.73	0.32/0.32/0.64	0.67/0.62/0.68

Equation families. Systems are clustered by the rule of Section 3, which asks whether one equation becomes another by fixing constants or rescaling states and time. That rule involves judgement, so we recompute the intervals that carry a claim while resampling the 63 systems instead of the 46 families (`check_cluster_sensitivity.py`). Across the 21 claims checked – specificity and AUROC lower bounds, the sparse-clamp medians, the two-trajectory gains, suite transfer and the standardised-SINDy differences – every verdict but one is unchanged: Weak SINDy's noise-free sparse-clamp interval, $[0.06, 0.88]$ when the six families are resampled, reaches divergence ($[0.01, \infty]$) when the ten systems are. No other bound moves by more than 0.051 except the horseshoe operator's sparse-clamp lower bound, which moves by 0.14 and keeps its verdict, and family clustering gives the wider interval for 15 of the 21.

GP-SINDy: the FlowGP sampler with an unknown equation. GP-SINDy is ours, so we check it where the answer is known: the 38 ODEBench systems whose equation lies in the library, at $\sigma = 0$, fitted exactly as in the benchmark. Table 6 compares the posterior over raw-unit coefficients with the truth, with least squares restricted to the true support as a reference that knows the answer.

Table 6: GP-SINDy identification on 38 in-library, noise-free ODEBench systems. Coverage: fraction of true nonzero coefficients inside the 95% interval. Rel. error: $\|\bar{\Xi} - \Xi^\star\| / \|\Xi^\star\|$ on the true support. Spurious: largest $|\bar{\Xi}|$ off the support over the largest true coefficient. Field: relative error of the posterior-mean vector field on the training orbit. OLS: least squares on the true support.

	Coverage	Rel. error	Spurious	Field	OLS rel. error	OLS field
25th percentile	0.71	0.57	0.39	0.02	0.00	0.00
median	1.00	0.78	0.72	0.05	0.01	0.00
75th percentile	1.00	0.97	1.31	0.13	0.17	0.04

The adaptation learns the vector field on the orbit it saw: the median field error is 5%, and below 10% on 27 of the 38 systems. It does not identify the equation. The posterior mean misses the true coefficients by 78% (median), and off-support terms reach 72% of the largest true coefficient, although least squares on the true support recovers the same coefficients to 0.8%. The 95% intervals cover the truth because they are wide, not because they are centred on it. With x, $\sin x$ and x^3 nearly collinear on one orbit, many coefficient vectors satisfy the ODE-residual condition, and the Laplace term does not select the sparse one. The same large, cancelling coefficients make GP-SINDy's edge scores nearly flat: on the sparse systems, every variable's largest term contribution lies between about 3 and 6 derivative standard deviations, whether or not the edge exists. Its per-system AUROC below 0.5 is nonetheless systematic, not noise: on all three Lorenz variants the absent edge $x_2 \to x_0$ receives its highest or second-highest score of the six (per-system AUROC 0–0.2), and on the apoptosis model every absent edge outranks every true one (0). Pooled over edge pairs, where the single absent edges of the Lorenz systems carry less weight, its AUROC is at chance. The sampler itself is not at fault: the check above reproduces closed-form GP conditioning to 0.039. What fails is the joint inference over an unknown equation, which is the part we added.

C LIMITATIONS IN FULL

Each system gives a method one trajectory of 105 noisy, uniformly sampled observations, so irregular sampling is untested, and one trajectory need not identify a nonlinear vector field off its orbit (Stanhope et al., 2014): part of what the new-IC and clamp questions measure is whether a method's inductive bias matches the system. *Tuning and seeds.* Settings were tuned only on the development suite, or not at all, so that no test suite tunes anything; the time-unit protocol tests one conversion of them, not a re-tuning. A per-system selection on a held-out part of the training window, or an optimistic upper bound tuned on the test questions, for SINDy's threshold, Weak SINDy's test-function width and the Neural ODE's size, would show how much of the gap to the oracle tuning closes; Weak SINDy's collapse with noise may be a matter of its default test functions. Each method is fitted once, with one seed and one noise draw, so seed variance is not reported, and the family bootstrap captures variation between systems, not within them. *Sparse systems.* The evidence that can expose a spurious dependence comes from 10 ODEBench systems in 6 families, 12 systems we wrote after seeing ODEBench results, and 60 generated systems. Six families are too few for percentile-bootstrap intervals to be reliable, so the ODEBench sparse intervals are descriptive. Only the random suite was fixed in advance (self-attested); its acceptance rule rejected 210 of 451 candidates as settled, so it favours damped systems that keep moving until the split, and its four- and five-dimensional systems ask a dictionary method for 43–66 coefficients from 105 samples while the oracle fits at most 21 constants, so part of the failure there is sample complexity. Every sparse system has three to five variables, while 28 of the 30 fully coupled ODEBench systems are two-dimensional, so the contrast between them does not separate sparsity from dimension; a fully coupled control suite of the same dimensions would. A clamp moves every descendant of the clamped variable, so where an absent edge is bridged by an indirect path it exposes a spurious dependence only through the response's early shape; the pairs with no directed path, whose true response is zero, number 3 on ODEBench, 37 in the hand-written and 120 in the random suite, and a direct test on them is left to future work. *References.* The oracles know more than the constants' values: every fixed coefficient, the true graph, and each constant's sign, from starts drawn relative to the truth, so the gap to them measures what the form, graph and signs are worth together; least squares on the true support alone would separate these. Discrete-time read-outs (PCMCI+, Granger, the Koopman operators) estimate the graph of the Δt-step map, which includes indirect paths at order Δt^2 (Gong et al., 2015), and are scored against the ODE's graph, so part of their false positives are true dependences of that map; a generator read-out (Klus et al., 2020), or scoring them against the union of the graph and its two-step paths, would remove that mismatch. *Scope.* Method classes that bring prior knowledge of forms are not included: language-model-guided symbolic regression (Shojaee et al., 2025a), for which ODEBench's textbook systems risk memorisation (Shojaee et al., 2025b), rational libraries (Kaheman et al., 2020) and gray-box models that learn part of a known form (Rackauckas et al., 2020), the natural midpoint between the blind methods and the oracle; nor is dedicated continuous-time graph learning (Bellot et al., 2022), of which C-NODE is the closest included relative. Every system is synthetic; physical testbeds with interventions (Gamella et al., 2025) would test the protocol on real data. The predictability threshold is a convention: with $\ln 100$, which keeps Rössler, no new-IC rate moves by more than a point. Equation families are grouped by a stated rule involving judgement, and resampling systems instead changes the verdict of one of 21 interval-based claims (Appendix B); Wilson intervals for rates ignore that grouping, and nothing is corrected for multiple comparisons. We test one noise model, one perturbation, applied to x_0 only, and one intervention type, posed at the noise-free state; deleting a coupling term or changing a parameter can only be posed to methods with an explicit equation. The development suite shares Lorenz, Van der Pol (#37, and #40, which ODEBench labels Duffing) and the damped harmonic oscillator (#25) with ODEBench, and its Duffing oscillator has the terms of ODEBench's damped double well (#32), which can only favour the blind methods. Everything beyond the three frozen questions was designed after a first look at ODEBench results, and standardised SINDy did not replicate on the development suite; the random suite is the only part of the evidence fixed in advance, and two of its four planned claims fail one criterion each.

D DEVELOPMENT SUITE RESULTS

The development suite is where the nine methods were built and tuned. Figure 2 and Table 7 give its results, which enter the transfer analysis of Table 8: development-suite rankings transfer to

ODEBench for forecasting ($\rho = 0.69$ $[0.12, 0.84]$) but not for perturbations (-0.19 $[-0.83, 0.43]$). Its graphs are also dense: the complete graph scores F1 0.93 there.

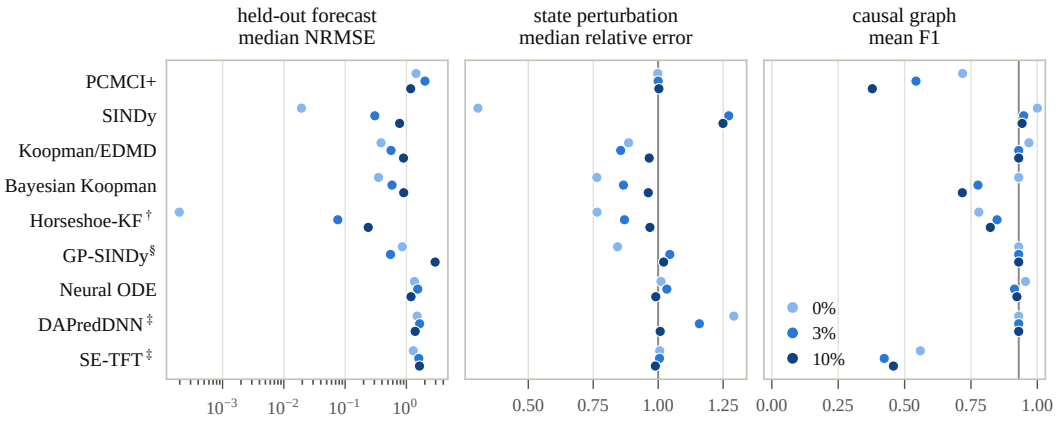


Figure 2: Development suite (7 systems, additive noise of 0, 3 and 10% of each state's standard deviation). Medians over systems for NRMSE and perturbation error, means for F1; a diverged prediction counts as $+\infty$. Grey lines: no response (1.0) and the complete graph (F1 0.93). †Assimilated forecasts.

Table 7: Development suite at 3% noise, per system. "–" marks a diverged prediction. NRMSE divides by each state's standard deviation over the scored window. The Horseshoe-KF's perturbation response uses the Gaussian-prior operator, so it matches Bayesian Koopman's to within 0.01.

System	PCMCI+	SINDy	EDMD	B. Koop.	HS-KF†	GP-SINDy	NODE	DAPred	SE-TFT
held-out forecast NRMSE									
Damped SHO	0.30	0.31	0.30	0.31	0.34	0.31	0.53	2.77	1.60
Duffing	3.33	0.16	0.12	0.14	0.04	0.17	0.30	0.08	0.03
Van der Pol	1.25	0.04	0.56	0.58	0.04	0.47	0.99	0.45	0.69
Lorenz-63	2.01	1.20	1.37	1.32	0.05	1.58	2.01	1.65	1.73
Double pendulum	3.21	–	1.66	1.71	0.08	4.29	1.66	2.08	1.66
Double pend. (damped)	1.84	0.10	0.20	0.23	0.08	1.99	1.53	1.31	0.59
Kuramoto (N=5)	5.03	1.06	0.94	1.17	0.16	0.55	2.36	2.22	4.55
perturbation response, relative error									
Damped SHO	0.06	0.05	0.13	0.14	0.14	0.02	0.49	1.68	0.78
Duffing	1.01	1.33	0.86	0.87	0.87	0.58	1.04	0.83	0.94
Van der Pol	1.00	0.04	0.77	0.77	0.78	0.63	1.11	0.67	1.02
Lorenz-63	1.00	1.27	1.00	1.00	1.00	1.04	1.03	1.34	1.19
Double pendulum	1.00	–	1.01	1.01	1.01	2.14	1.01	1.16	1.01
Double pend. (damped)	1.03	0.33	0.79	0.79	0.80	4.63	1.96	4.23	1.01
Kuramoto (N=5)	0.93	2.14	1.24	1.26	1.25	1.10	0.95	1.01	0.98
causal-graph F1									
Damped SHO	1.00	1.00	1.00	0.00	1.00	1.00	1.00	1.00	0.00
Duffing	0.67	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.67
Van der Pol	0.67	1.00	1.00	1.00	0.67	1.00	1.00	1.00	0.00
Lorenz-63	0.57	0.91	0.91	0.91	0.91	0.91	0.67	0.91	0.57
Double pendulum	0.40	0.89	0.80	0.84	0.93	0.80	0.84	0.80	0.71
Double pend. (damped)	0.40	0.84	0.80	0.93	0.86	0.80	0.89	0.80	0.50
Kuramoto (N=5)	0.10	1.00	1.00	0.75	0.57	1.00	1.00	1.00	0.52

Table 8: Spearman correlation between the rankings of the nine development-suite methods on the development suite and on ODEBench, with 95% intervals resampling development systems and ODEBench equation families, and the bootstrap probability that $\rho > 0$. ODEBench rows follow the predictability rule. Horseshoe-KF is excluded from the perturbation rows because its development-suite response came from the Gaussian-prior operator. With nine (eight) methods the intervals are wide by construction.

Task	n methods	ρ	95% CI	$P(\rho{>}0)$
forecast	9	$+0.69$	$[+0.12, +0.84]$	1.00
forecast→new IC	9	$+0.67$	$[+0.10, +0.77]$	0.99
perturbation (NRMSE)	8	-0.19	$[-0.83, +0.43]$	0.22
perturbation (rel. error)	8	-0.33	$[-0.76, +0.36]$	0.22
graph F1	9	$+0.78$	$[+0.61, +0.96]$	1.00

E ADDITIONAL ODEBENCH RESULTS

Table 9 gives the appendix-only rows with the main table's columns, Table 10 every median of Table 2 with its family-bootstrap interval and the two components of the sparse-system clamp response, Table 11 the oracles from the three start distributions (identical rates also on the 22 predictable systems with a constant entering nonlinearly: trajectory fit 100/82/50%, derivative fit 82/68/55%), Table 12 the time-unit protocol, Table 13 the second Koopman clamp adapter (under which the Bayesian operator's diverging sparse-clamp medians become finite and the horseshoe operator's 0.79 becomes an error above 10^3), Table 14 the intervention-size and clamp-start runs, and Table 15 the two-trajectory ablation. Figure 5 and Table 16 give the graph read-outs, Table 17 splits the trajectory questions by dimension and library membership, Table 18 gives rank correlations between questions (single methods separate the questions: DAPredDNN predicts perturbations with error 0.62–0.65 at $\sigma \leq 0.01$ but answers the dense-system clamp at 1.00 and 0.92, and EDMD the opposite at $\sigma = 0$, 1.01 and 0.34) and the paired SINDy comparisons, and Table 19 valid prediction times on the systems the predictability rule leaves out: the trajectory-fit oracle follows a new initial condition for 10/4.7/1.8 of the 10 time units (median over the three systems) and no learned model for more than 2.3 (median; Weak SINDy follows #56 for 6.8 without noise). Figure 6 shows every sparse ODEBench system's clamp error, Figure 7 example trajectories, Figure 8 one graph recovery and Figure 9 every system's trajectory scores; the systems are listed in Table 20.

Uncertainty and cost. GP-SINDy's 95% forecast band covers 93–100% of the noise-free test trajectory (median over systems), including where its mean forecast fails. Bayesian Koopman's band covers 3% at $\sigma = 0$, where its plug-in residual covariance is nearly zero, and the Horseshoe-KF's 36%, because its observation noise is set to the true σ of 0; these are configuration choices. SE-TFT's P10–P90 band covers 11–24% against a nominal 80%. Cost does not buy generalisation (Figure 3): the best new-IC accuracy at each noise level comes from Weak SINDy (0.4 s, 62%), PySR (160 s, 37%) and ODEFormer (3 s, 33%), while the two neural ODEs take a minute to reach 18 and 3%.

Table 9: Appendix-only rows, with the columns of Table 2 (forecast: 62 systems; new IC: 60; perturbation: 62; clamp: the 30 fully coupled and the 10 sparse ODEBench systems). [§]Our adaptation of the FlowGP sampler to an unknown equation. [‡]Designed for parametric families. [†]Assimilated: sees every observation, so its forecast and new-IC scores answer an easier question and are not comparable with the blind rows; ⋆it shares the blind horseshoe operator's fit.

	Forecast ↑	New IC ↑	Perturbation ↓	Clamp, dense ↓	Clamp, sparse ↓	Time
our adaptation of a sampler to unknown equations						
GP-SINDy[§]	52/52/48	13/13/10	0.99/0.97/0.83	0.99/0.96/0.97	1.20/1.08/2.40	18
surrogates designed for parametric families						
DAPredDNN[‡]	66/60/35	28/20/3	0.62/0.65/0.94	1.00/0.92/1.10	0.98/1.23/0.97	109
SE-TFT[‡]	21/23/21	3/3/3	1.00/1.00/1.00	1.01/1.13/1.12	1.00/1.01/1.01	118
assimilated filter: sees every observation, answers the trajectory questions only						
Horseshoe-KF[†]	95/95/90	68/68/65	–	–	–	⋆

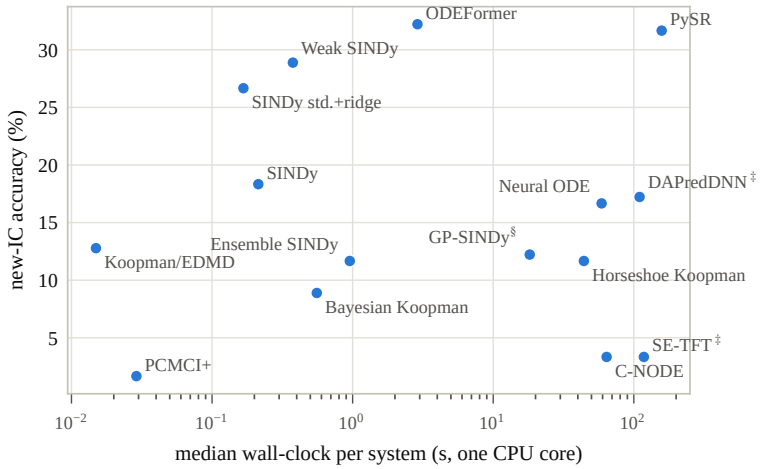


Figure 3: Cost does not buy generalisation on ODEBench: median wall-clock time per system (one CPU core, log scale) against new-IC accuracy, both averaged over the three noise levels. §Our adaptation. ‡Designed for parametric families; both are appendix-only rows.

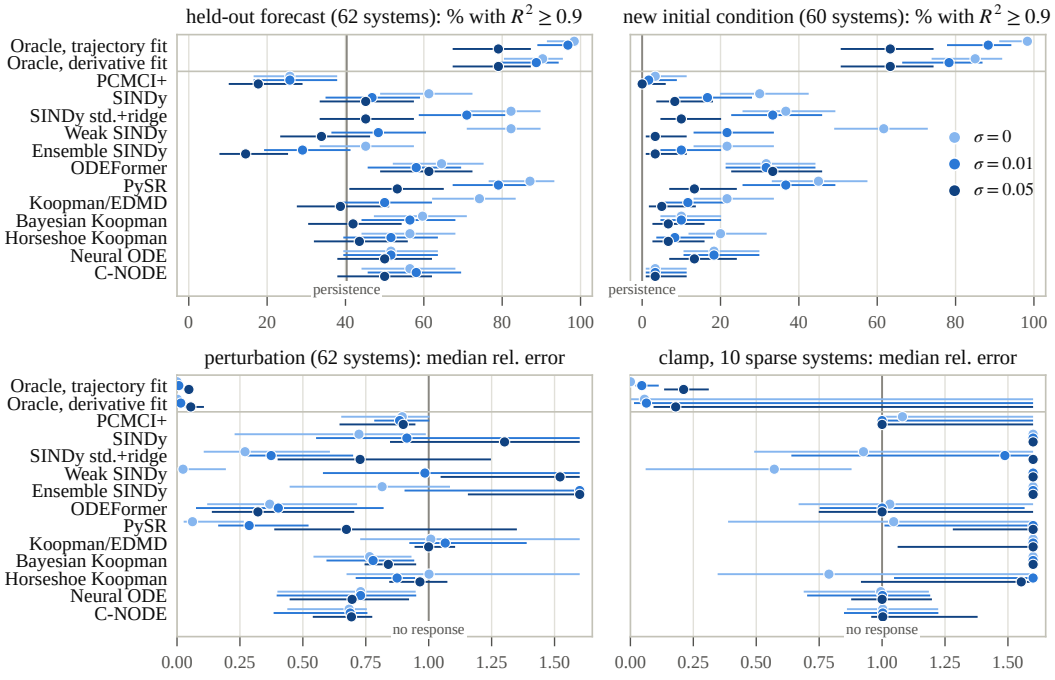


Figure 4: ODEBench. One row per method, one dot per noise level ($\sigma = 0$ light, 0.01, 0.05 dark); whiskers are 95% intervals (Wilson for rates, family bootstrap for medians), grey lines the trivial references, and response errors are clipped at 1.6. The two oracles are above the rule. The clamp panel shows the 10 ODEBench systems with an absent edge; the 30 fully coupled ones are in Table 2.

Table 10: ODEBench median response errors with 95% intervals resampling equation families (perturbation: 62 systems; clamp: 30 dense and 10 sparse systems), and the two components of the sparse-system clamp response among finite predictions: the size ratio $|\hat{\Delta}|/|\Delta|$ (0: no predicted response, 1: the right size) and the shape (cosine similarity; 1: the right shape at any size, 0: unrelated), the latter with its interval. With fewer than three finite predictions the shape is given with its n and no interval; "–" marks a shape that is undefined (no finite or no nonzero predicted response).

Method	σ	Perturbation	Clamp, dense	Clamp, sparse	Size ratio	Shape
Trivial	0	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.00	–
	0.01	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.00	–
	0.05	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.00	–
Oracle, trajectory fit	0	0.00 [0.00, 0.00]	0.00 [0.00, 0.00]	0.00 [0.00, 0.00]	1.00	1.00 [1.00, 1.00]
	0.01	0.01 [0.00, 0.01]	0.01 [0.00, 0.01]	0.05 [0.02, 0.11]	1.00	1.00 [1.00, 1.00]
	0.05	0.05 [0.03, 0.07]	0.03 [0.01, 0.06]	0.21 [0.13, 0.31]	0.97	0.99 [0.97, 1.00]
Oracle, derivative fit	0	0.00 [0.00, 0.01]	0.00 [0.00, 0.02]	0.06 [0.00, 3.81]	1.00	1.00 [1.00, 1.00]
	0.01	0.01 [0.01, 0.03]	0.02 [0.01, 0.03]	0.06 [0.01, 3.51]	1.01	1.00 [1.00, 1.00]
	0.05	0.05 [0.03, 0.11]	0.03 [0.02, 0.08]	0.18 [0.09, 2.62]	0.99	1.00 [0.99, 1.00]
PCMCI+	0	0.89 [0.65, 1.00]	1.00 [1.00, 1.00]	1.08 [1.00, 19.25]	0.36	0.02 [-0.09, 0.08]
	0.01	0.88 [0.78, 1.00]	1.00 [1.00, 1.00]	1.00 [1.00, 1.74]	0.03	0.01 [0.00, 0.09]
	0.05	0.90 [0.65, 0.95]	1.00 [1.00, 1.00]	1.00 [1.00, 2.48]	0.03	0.04 [0.00, 0.09]
SINDy	0	0.72 [0.23, 0.99]	0.92 [0.35, 9.31]	∞ [∞, ∞]	–	–
	0.01	0.91 [0.55, ∞]	1.01 [0.33, 27.09]	∞ [∞, ∞]	2.76	0.40 (n=2)
	0.05	1.30 [0.85, 58.41]	3.01 [0.56, ∞]	∞ [∞, ∞]	49.42	-0.12 (n=1)
SINDy (std.+ridge)	0	0.27 [0.11, 0.61]	0.32 [0.20, 0.66]	0.93 [0.49, ∞]	0.75	0.89 [-0.05, 0.99]
	0.01	0.37 [0.28, 0.70]	0.32 [0.19, 0.49]	1.49 [0.64, ∞]	0.92	0.84 [-0.25, 0.97]
	0.05	0.73 [0.40, 1.25]	0.64 [0.25, 1.16]	∞ [2.21, ∞]	1.95	0.29 [-0.50, 1.00]
Weak SINDy	0	0.02 [0.01, 0.19]	0.01 [0.00, 0.19]	0.57 [0.06, 0.88]	0.88	0.96 [0.76, 1.00]
	0.01	0.98 [0.58, 3.62]	1.71 [0.85, ∞]	∞ [∞, ∞]	0.96	0.97 (n=1)
	0.05	1.52 [1.05, 2.86]	4.72 [1.21, ∞]	∞ [2.74, ∞]	2.03	0.30 [0.02, 0.63]
Ensemble SINDy	0	0.81 [0.45, 1.08]	2.12 [1.09, 207]	∞ [1.70, ∞]	0.93	-0.06 [-0.21, 0.28]
	0.01	1.66 [0.90, ∞]	∞ [13.21, ∞]	∞ [∞, ∞]	5.29	-0.09 (n=1)
	0.05	4.72 [1.16, ∞]	∞ [∞, ∞]	∞ [∞, ∞]	–	–
ODEFormer	0	0.37 [0.12, 0.72]	0.93 [0.39, 2.59]	1.03 [0.67, 57.90]	1.09	0.44 [-0.21, 0.85]
	0.01	0.40 [0.08, 0.82]	0.74 [0.42, 1.74]	1.00 [0.75, 1.57]	0.74	0.21 [-0.01, 0.85]
	0.05	0.32 [0.14, 0.70]	0.54 [0.31, 0.91]	1.00 [0.75, 1.81]	0.73	0.50 [-0.00, 0.91]
PySR	0	0.06 [0.03, 0.27]	0.10 [0.04, 0.36]	1.05 [0.39, 2.33]	1.03	0.64 [0.06, 0.97]
	0.01	0.29 [0.16, 0.52]	0.24 [0.09, 1.01]	∞ [1.01, ∞]	1.13	0.56 [0.47, 1.00]
	0.05	0.67 [0.39, 1.35]	0.96 [0.39, 2.02]	1.82 [1.28, ∞]	1.38	0.39 [-0.09, 1.00]
Koopman/EDMD	0	1.01 [0.73, 2.87]	0.34 [0.19, 0.83]	∞ [∞, ∞]	0.56	0.95 (n=2)
	0.01	1.07 [0.92, 1.39]	1.13 [0.76, 1.95]	∞ [∞, ∞]	0.25	0.68 (n=1)
	0.05	1.00 [0.94, 1.10]	1.39 [1.02, ∞]	∞ [1.06, ∞]	0.61	0.36 [-0.53, 0.51]
Bayesian Koopman	0	0.77 [0.54, 0.93]	1.96 [0.85, ∞]	∞ [5.74, ∞]	5.07	0.11 [-0.42, 0.97]
	0.01	0.78 [0.59, 0.94]	∞ [0.96, ∞]	∞ [2.24, ∞]	0.40	0.30 [-0.73, 0.97]
	0.05	0.84 [0.75, 0.95]	∞ [∞, ∞]	∞ [∞, ∞]	0.61	0.98 (n=1)
Horseshoe Koopman	0	1.00 [0.67, 2.85]	0.20 [0.13, 0.35]	0.79 [0.35, ∞]	0.62	0.93 [0.17, 0.99]
	0.01	0.87 [0.71, 0.95]	0.60 [0.26, 1.09]	2.15 [1.05, ∞]	0.60	0.63 [-0.41, 0.80]
	0.05	0.97 [0.84, 1.07]	1.01 [0.60, 1.46]	1.55 [0.92, ∞]	0.60	0.50 [-0.48, 0.59]
Neural ODE	0	0.73 [0.40, 0.95]	0.77 [0.31, 1.00]	0.99 [0.69, 1.19]	0.74	0.31 [-0.09, 0.88]
	0.01	0.73 [0.40, 0.95]	0.78 [0.31, 0.99]	1.00 [0.70, 1.19]	0.73	0.31 [-0.11, 0.87]
	0.05	0.70 [0.45, 0.92]	0.78 [0.36, 0.98]	1.00 [0.88, 1.20]	0.57	0.34 [-0.08, 0.67]
C-NODE	0	0.68 [0.44, 0.75]	0.65 [0.38, 0.99]	1.00 [0.86, 1.22]	0.51	0.24 [0.03, 0.54]
	0.01	0.69 [0.38, 0.76]	0.68 [0.39, 0.99]	1.00 [0.85, 1.22]	0.49	0.23 [0.03, 0.55]
	0.05	0.69 [0.54, 0.78]	0.84 [0.36, 1.00]	1.00 [0.96, 1.38]	0.37	0.15 [-0.11, 0.37]
GP-SINDy[§]	0	0.99 [0.43, 1.24]	0.99 [0.71, 2.22]	1.20 [1.00, 20.14]	1.38	0.25 [0.05, 0.60]
	0.01	0.97 [0.56, 1.72]	0.96 [0.62, 3.00]	1.08 [1.00, 16.03]	1.32	0.16 [-0.03, 0.63]
	0.05	0.83 [0.50, 1.07]	0.97 [0.69, 2.00]	2.40 [0.94, 7.51]	2.46	0.16 [0.01, 0.51]
DAPredDNN[‡]	0	0.62 [0.19, 0.85]	1.00 [0.76, 1.47]	0.98 [0.93, 3.68]	0.87	0.33 [-0.11, 0.55]
	0.01	0.65 [0.53, 0.88]	0.92 [0.67, 1.10]	1.23 [0.92, 3.26]	0.86	0.35 [-0.22, 0.59]
	0.05	0.94 [0.87, 0.98]	1.10 [0.88, 1.38]	0.97 [0.92, 1.55]	0.65	0.40 [-0.30, 0.46]
SE-TFT[‡]	0	1.00 [1.00, 1.01]	1.01 [0.95, 1.53]	1.00 [0.99, 1.09]	0.11	0.07 [-0.04, 0.19]
	0.01	1.00 [1.00, 1.00]	1.13 [0.96, 1.92]	1.01 [0.99, 1.20]	0.19	-0.01 [-0.51, 0.11]
	0.05	1.00 [1.00, 1.00]	1.12 [1.00, 1.51]	1.01 [0.99, 1.72]	0.15	0.01 [-0.15, 0.19]

Table 11: What the oracles' starts contribute. Each oracle from starts within $\pm 50\%$ of ODEBench's constants (the reference of the main text), from starts anywhere within an order of magnitude of them on either side, and, for the derivative fit, started at the true constants. Columns as in Table 2, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$.

Starts	Forecast ↑	New IC ↑	Perturbation ↓	Clamp, dense ↓	Clamp, sparse ↓
trajectory fit					
starts within $\pm 50\%$	98/97/79	98/88/63	0.00/0.01/0.05	0.00/0.01/0.03	0.00/0.05/0.21
starts within one order of magnitude	98/97/79	98/88/63	0.00/0.01/0.05	0.00/0.01/0.03	0.00/0.05/0.26
derivative fit					
starts within $\pm 50\%$	90/89/79	85/78/63	0.00/0.01/0.05	0.00/0.02/0.03	0.06/0.06/0.18
starts within one order of magnitude	90/89/79	85/78/63	0.00/0.01/0.05	0.00/0.02/0.03	0.06/0.06/0.18
started at the true constants	90/89/79	85/78/63	0.00/0.01/0.05	0.00/0.02/0.03	0.06/0.06/0.18

Table 12: The frozen settings (11-sample Savitzky–Golay window, 15-step Neural ODE windows) against the same settings in the development suite's time units (0.22 and 0.30 time units, 5 samples each on ODEBench's grid). Columns as in Table 2, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$.

Method	Windows	Forecast ↑	New IC ↑	Perturbation ↓	Clamp, dense ↓	Clamp, sparse ↓
SINDy	frozen	61/47/45	30/17/8	0.72/0.91/1.30	0.92/1.01/3.01	$\infty/\infty/\infty$
	time units	66/50/34	30/15/5	0.53/1.00/1.18	0.66/3.14/∞	$\infty/\infty/\infty$
SINDy (std.+ridge)	frozen	82/71/45	37/33/10	0.27/0.37/0.73	0.32/0.32/0.64	0.93/1.49/∞
	time units	84/74/45	37/30/17	0.24/0.32/0.70	0.26/0.52/0.92	0.99/7.51/23.08
Ensemble SINDy	frozen	45/29/15	22/10/3	0.81/1.66/4.72	2.12/∞/∞	$\infty/\infty/\infty$
	time units	52/24/13	28/5/3	0.65/1.31/11.58	2.02/∞/∞	$\infty/\infty/\infty$
Oracle, derivative fit	frozen	90/89/79	85/78/63	0.00/0.01/0.05	0.00/0.02/0.03	0.06/0.06/0.18
	time units	92/90/77	88/83/58	0.00/0.01/0.06	0.00/0.01/0.03	0.04/0.06/0.17
Neural ODE	frozen	52/52/50	18/18/13	0.73/0.73/0.70	0.77/0.78/0.78	0.99/1.00/1.00
	time units	53/48/42	20/15/8	0.71/0.72/0.76	0.79/0.84/0.88	1.03/1.03/1.06

Table 13: The three Koopman operators' median clamp errors under the two adapters, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$: re-lifting the state and overwriting x_i after every step (the adapter of every table), and iterating in the lifted space and overwriting only x_i's coordinate. On ODEBench the sparse systems are shown; the other suites pool all systems.

Operator	Suite	Re-lift adapter	Lifted-space adapter
Koopman/EDMD	ODEBench, sparse systems	$\infty/\infty/\infty$	$>10^3/\infty/>10^3$
	hand-written suite	$\infty/\infty/\infty$	$>10^3/>10^3/>10^3$
	random suite	$\infty/\infty/\infty$	$>10^3/>10^3/>10^3$
Bayesian Koopman	ODEBench, sparse systems	$\infty/\infty/\infty$	2.12/1.91/3.38
	hand-written suite	$\infty/\infty/\infty$	1.58/1.55/10.64
	random suite	$\infty/\infty/\infty$	$1.41/1.22/>10^3$
Horseshoe Koopman	ODEBench, sparse systems	0.79/2.15/1.55	$>10^3$/43.08/1.08
	hand-written suite	0.51/1.13/1.24	6.85/1.45/2.36
	random suite	0.26/1.22/1.30	18.37/2.31/1.34

Table 14: Clamp sensitivity on the 10 sparse ODEBench systems: median relative clamp error, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$, for interventions of 0.1, 0.3 (the main text) and 1.0 standard deviations applied at the split, and for 0.3 standard deviations applied at 35% of the trajectory, inside the training window. The perturbation size changes with the clamp size in the first three columns. PCMCI+, SINDy, Weak SINDy and Bayesian Koopman are not shown: in every setting here their medians with noise are 1.00 (PCMCI+) or diverge. The intervals of the rows shown are in `results/clamp_sensitivity.csv`.

Method	0.1 sd, split	0.3 sd, split	1.0 sd, split	0.3 sd, at 35%
Oracle, trajectory fit	0.00/0.06/0.24	0.00/0.05/0.21	0.00/0.02/0.12	0.00/0.03/0.15
SINDy (std.+ridge)	1.08/1.27/∞	0.93/1.49/∞	1.17/12.54/∞	1.12/4.42/4.06
Ensemble SINDy	$\infty/\infty/\infty$	$\infty/\infty/\infty$	$\infty/\infty/\infty$	$\infty/\infty/\infty$
ODEFormer	1.41/1.13/1.18	1.03/1.00/1.00	0.76/0.81/0.96	1.03/1.06/1.00
PySR	1.01/∞/1.90	1.05/∞/1.82	0.77/∞/∞	0.87/0.85/1.15
Koopman/EDMD	$\infty/\infty/\infty$	$\infty/\infty/\infty$	$\infty/\infty/\infty$	$\infty/\infty/\infty$
Horseshoe Koopman	0.87/1.95/1.56	0.79/2.15/1.55	0.66/3.82/1.55	0.83/1.54/1.12
Neural ODE	1.00/1.02/1.00	0.99/1.00/1.00	1.00/1.00/1.00	0.99/0.99/1.00
C-NODE	1.00/1.00/1.05	1.00/1.00/1.00	0.97/0.97/0.99	0.93/0.93/0.97

Table 15: Two-trajectory ablation at $\sigma = 0.01$: % with $R^2 \geq 0.9$ when fitted on IC 1 alone or on IC 1 and 2, tested at their midpoint (IC 3, 60 systems) and 50% beyond one of them (25% where 50% is inadmissible; IC 4, 57 systems; #48, #53 and #62 have no admissible fourth initial condition under ODEBench's constraints). Gains in points with 95% family-bootstrap intervals.

	IC 3 (interpolation)			IC 4 (extrapolation)		
Method	1 IC	2 ICs	gain	1 IC	2 ICs	gain
Oracle, trajectory fit	92	98	+7 [+1, +14]	93	95	+2 [-3, +9]
Oracle, derivative fit	80	87	+7 [-3, +18]	79	82	+4 [-7, +13]
SINDy	27	55	+28 [+14, +41]	16	39	+23 [+10, +40]
SINDy (std.+ridge)	40	68	+28 [+17, +41]	25	54	+30 [+18, +45]
Ensemble SINDy	18	30	+12 [-2, +23]	11	19	+9 [-2, +21]
Koopman/EDMD	22	53	+32 [+18, +43]	9	26	+18 [+8, +27]
Bayesian Koopman	23	48	+25 [+15, +36]	9	26	+18 [+8, +27]
Horseshoe Koopman	25	50	+25 [+15, +37]	11	23	+12 [+4, +23]
Neural ODE	22	38	+17 [+8, +26]	14	18	+4 [-3, +12]
C-NODE	20	35	+15 [+5, +27]	12	12	+0 [-10, +12]

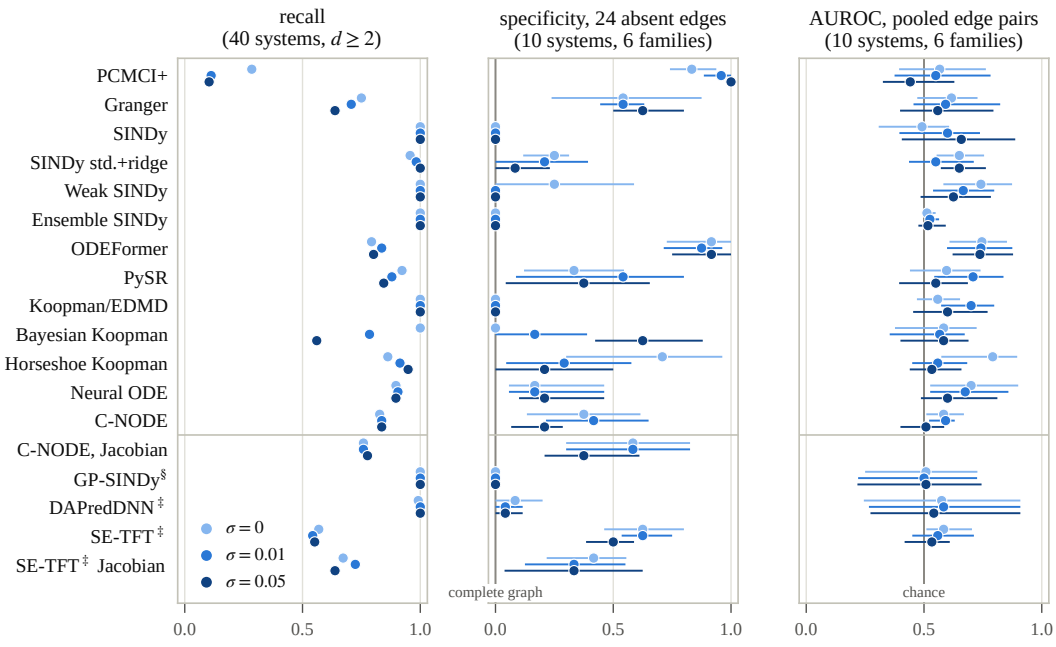


Figure 5: Graph read-outs on ODEBench. Recall pools the 40 systems with $d \geq 2$. Specificity pools the 24 absent edges of the 10 sparse systems. AUROC: the fraction of within-system (true, absent) edge pairs that each method's own edge statistic ranks correctly. Intervals resample the 6 equation families these systems form. The rows below the rule are appendix-only methods.

Table 16: Graph read-outs on ODEBench, $\sigma = 0 \,/\, 0.01 \,/\, 0.05$. F1 averages the 40 systems with $d \geq 2$, and recall pools their edges. Specificity pools the 24 absent edges of the 10 sparse systems. AUROC: per-system mean over those systems, and pooled, the fraction of all within-system (true, absent) edge pairs ranked correctly; systems with one absent edge can only reach per-system values in steps of 0.2–0.25. Balanced accuracy: (recall + specificity)/2, pooled over the 10 sparse systems; the complete graph scores 0.5 by construction. Intervals are in Figure 5 and `results/odebench_summary.csv`. Exact: sparse graphs recovered exactly, out of 10.

	F1	Recall	Specificity	Bal. acc.	AUROC, mean	AUROC, pooled	Exact
Trivial	0.96/0.96/0.96	1.00/1.00/1.00	0.00/0.00/0.00	0.50/0.50/0.50	0.50/0.50/0.50	0.50/0.50/0.50	0/0/0
PCMCI+	0.34/0.13/0.11	0.28/0.11/0.10	0.83/0.96/1.00	0.58/0.56/0.59	0.69/0.60/0.53	0.57/0.55/0.44	1/0/0
Granger	0.83/0.81/0.73	0.75/0.71/0.64	0.54/0.54/0.62	0.60/0.59/0.61	0.61/0.69/0.62	0.62/0.59/0.56	0/0/1
SINDy	0.96/0.96/0.96	1.00/1.00/1.00	0.00/0.00/0.00	0.50/0.50/0.50	0.39/0.50/0.52	0.49/0.60/0.66	0/0/0
SINDy (std.+ridge)	0.95/0.96/0.96	0.96/0.98/1.00	0.25/0.21/0.08	0.57/0.58/0.54	0.67/0.62/0.68	0.65/0.55/0.65	0/0/0
Weak SINDy	0.97/0.96/0.96	1.00/1.00/1.00	0.25/0.00/0.00	0.62/0.50/0.50	0.78/0.71/0.71	0.74/0.67/0.62	3/0/0
Ensemble SINDy	0.96/0.96/0.96	1.00/1.00/1.00	0.00/0.00/0.00	0.50/0.50/0.50	0.53/0.55/0.55	0.51/0.53/0.52	0/0/0
ODEFormer	0.90/0.92/0.90	0.79/0.84/0.80	0.92/0.88/0.92	0.78/0.78/0.78	0.77/0.77/0.79	0.75/0.74/0.74	2/2/2
PySR	0.94/0.92/0.89	0.92/0.88/0.84	0.33/0.54/0.38	0.58/0.67/0.57	0.60/0.70/0.56	0.60/0.71/0.55	1/1/0
Koopman/EDMD	0.96/0.96/0.96	1.00/1.00/1.00	0.00/0.00/0.00	0.50/0.50/0.50	0.57/0.66/0.58	0.56/0.70/0.60	0/0/0
Bayesian Koopman	0.96/0.80/0.64	1.00/0.78/0.56	0.00/0.17/0.62	0.50/0.48/0.54	0.47/0.43/0.57	0.58/0.57/0.58	0/0/0
Horseshoe (KF and blind)	0.90/0.89/0.91	0.86/0.91/0.95	0.71/0.29/0.21	0.73/0.62/0.59	0.67/0.59/0.56	0.79/0.56/0.53	1/1/1
Neural ODE	0.92/0.93/0.92	0.90/0.91/0.90	0.17/0.17/0.21	0.53/0.54/0.55	0.77/0.76/0.73	0.70/0.68/0.60	1/1/2
C-NODE	0.89/0.90/0.88	0.83/0.84/0.84	0.38/0.42/0.21	0.54/0.57/0.49	0.61/0.58/0.50	0.58/0.59/0.51	0/1/0
C-NODE, Jacobian probe	0.87/0.87/0.87	0.76/0.76/0.78	0.58/0.58/0.38	0.58/0.58/0.50	0.70/0.71/0.70	–	0/0/0
GP-SINDy[§]	0.96/0.96/0.96	1.00/1.00/1.00	0.00/0.00/0.00	0.50/0.50/0.50	0.36/0.34/0.33	0.51/0.50/0.51	0/0/0
DAPredDNN[‡]	0.95/0.96/0.96	0.99/1.00/1.00	0.08/0.04/0.04	0.54/0.52/0.52	0.77/0.77/0.73	0.57/0.58/0.54	1/0/0
SE-TFT[‡]	0.68/0.67/0.67	0.57/0.54/0.55	0.62/0.62/0.50	0.61/0.58/0.56	0.64/0.64/0.47	0.58/0.56/0.53	0/0/0
SE-TFT[‡], Jacobian probe	0.76/0.79/0.71	0.67/0.72/0.64	0.42/0.33/0.33	0.50/0.50/0.50	0.41/0.50/0.52	–	0/0/0

Table 17: Accuracy (% of system–noise pairs with $R^2 \geq 0.9$), pooled over noise levels, by state dimension and by whether the true equation lies in the shared library (35 of the 60 new-IC systems). Forecast columns leave out #56, new-IC columns #56, #59 and #61 (23, 28 and 9 systems by dimension). DAPredDNN's forecast uses its online path. The Horseshoe-KF generalises better outside the library than inside it, because it assimilates, and because 12 of the out-of-library systems are one-dimensional. [†]Assimilated. [‡]Designed for parametric families. [§]Our adaptation.

	Forecast			New IC			New IC	
	$d{=}1$	$d{=}2$	$d{\geq}3$	$d{=}1$	$d{=}2$	$d{\geq}3$	in lib.	out
Trivial	61	29	27	0	0	0	0	0
Oracle, trajectory fit	91	96	79	81	90	67	82	85
Oracle, derivative fit	91	90	64	78	83	44	73	79
PCMCI+	28	25	9	3	1	0	2	1
SINDy	61	54	24	32	12	4	19	17
SINDy (std.+ridge)	72	68	48	35	26	7	30	23
Weak SINDy	64	56	33	35	29	15	31	25
Ensemble SINDy	48	24	6	23	6	0	10	15
ODEFormer	84	50	42	49	29	0	35	28
PySR	83	74	52	43	30	7	32	31
Koopman/EDMD	62	52	42	25	7	0	16	8
Bayesian Koopman	65	46	42	10	10	4	13	3
Horseshoe Koopman	70	39	39	16	11	4	14	8
Neural ODE	51	61	27	16	23	0	19	13
C-NODE	64	50	48	9	0	0	6	0
GP-SINDy[§]	84	40	6	28	4	0	10	15
DAPredDNN[‡]	72	43	42	36	7	0	17	17
SE-TFT[‡]	33	15	12	9	0	0	6	0
Horseshoe-KF[†]	99	98	73	88	65	19	63	73

Table 18: Top: Spearman correlation, across the 15 learned methods, between their perturbation, clamp and forecast scores on the 39 systems with $d \geq 2$ kept by the predictability rule, with 95% family-bootstrap intervals. Below: each SINDy variant minus plain SINDy on ODEBench (accuracy differences in percentage points, and the median clamp error on the 30 fully coupled systems, the population of Table 2), with the same intervals; $\pm\infty$ where at least half of a resample's clamps diverge.

	$\sigma = 0$	$\sigma = 0.01$	$\sigma = 0.05$
perturbation vs clamp	$+0.46$ $[+0.26, +0.73]$	$+0.81$ $[+0.40, +0.89]$	$+0.76$ $[+0.45, +0.84]$
forecast vs clamp	$+0.61$ $[+0.35, +0.75]$	$+0.55$ $[+0.16, +0.71]$	$+0.49$ $[+0.09, +0.65]$
standardised + ridge minus plain SINDy			
forecast acc. (points)	$+21$ $[+6, +34]$	$+24$ $[+8, +38]$	$+0$ $[-13, +13]$
new-IC acc. (points)	$+7$ $[-12, +22]$	$+17$ $[+0, +30]$	$+2$ $[-11, +13]$
median clamp error	-0.60 $[-8.93, -0.07]$	-0.69 $[-26.83, -0.04]$	-2.37 $[-\infty, -0.17]$
standardised only minus plain SINDy			
forecast acc. (points)	$+2$ $[-14, +15]$	$+8$ $[+2, +15]$	-2 $[-8, +4]$
new-IC acc. (points)	$+5$ $[-9, +18]$	$+0$ $[-6, +4]$	$+2$ $[+0, +4]$
median clamp error	$+0.11$ $[-0.03, +4.20]$	-0.13 $[-0.52, +31.13]$	$+3.15$ $[-9.06, +\infty]$
ridge only minus plain SINDy			
forecast acc. (points)	-2 $[-13, +9]$	$+10$ $[-5, +25]$	-2 $[-9, +7]$
new-IC acc. (points)	-8 $[-19, +4]$	$+0$ $[-12, +12]$	-2 $[-11, +6]$
median clamp error	-0.46 $[-8.80, +0.10]$	-0.57 $[-26.53, +0.18]$	-2.44 $[-\infty, -0.05]$

Table 19: Valid prediction time on the three systems the predictability rule removes from the new-IC questions (#56/#59/#61): time units from the first predicted sample until the normalised error $\left(\frac{1}{d}\sum_k((\hat{x}_k - x_k)/\mathrm{sd}_k)^2\right)^{1/2}$ first exceeds 0.5, out of 10; samples are 0.067 apart. A non-finite sample counts as an exceedance; *f* marks a rollout the integrator stopped, which has no finite sample and scores 0. Unlike in the other tables, the slash separates the three systems, and the columns are noise levels. Repeated cells are not copies: the derivative-fit oracle's rollouts, for instance, differ between noise levels but leave the tolerance at the same sample.

	$\sigma = 0$	$\sigma = 0.01$	$\sigma = 0.05$
Trivial	0.1/0.1/0.1	0.1/0.1/0.1	0.1/0.1/0.1
Oracle, trajectory fit	10.0/10.0/10.0	3.7/4.7/4.8	0.9/2.2/1.8
Oracle, derivative fit	0.6/0.3/0.1	0.6/0.3/0.1	0.6/0.3/0.1
PCMCI+	0.1/0.4/0.1	0.1/0.4/0.1	0.1/0.4/0.1
SINDy	*f*/*f*/*f*	0.1/*f*/*f*	*f*/*f*/*f*
SINDy (std.+ridge)	*f*/0.5/0.1	*f*/0.5/0.1	*f*/0.3/0.1
Weak SINDy	6.8/2.3/1.7	0.7/*f*/4.7	0.1/*f*/*f*
Ensemble SINDy	0.1/*f*/*f*	0.1/*f*/*f*	*f*/*f*/*f*
ODEFormer	*f*/*f*/*f*	0.5/*f*/0.4	*f*/*f*/0.3
PySR	*f*/*f*/*f*	*f*/*f*/*f*	0.1/*f*/0.1
Koopman/EDMD	0.1/0.1/0.1	0.2/0.1/0.1	0.1/0.1/0.1
Bayesian Koopman	0.1/0.1/0.1	0.2/0.1/0.1	0.1/0.1/0.1
Horseshoe Koopman	0.1/0.1/0.1	0.1/0.1/0.2	0.1/0.1/0.2
Neural ODE	0.1/0.1/0.1	0.1/0.1/0.1	0.1/0.1/0.1
C-NODE	0.1/0.1/0.1	0.1/0.1/0.1	0.1/0.1/0.1
GP-SINDy[§]	0.1/0.1/0.2	0.1/0.1/0.2	0.1/0.1/0.1
DAPredDNN[‡]	0.6/0.9/0.2	0.6/0.9/0.3	0.6/0.9/0.2
SE-TFT[‡]	0.0/0.1/0.0	0.1/0.1/0.0	0.0/0.1/0.0
Horseshoe-KF[†]	0.1/0.1/0.1	0.2/0.1/0.4	0.1/0.1/0.3

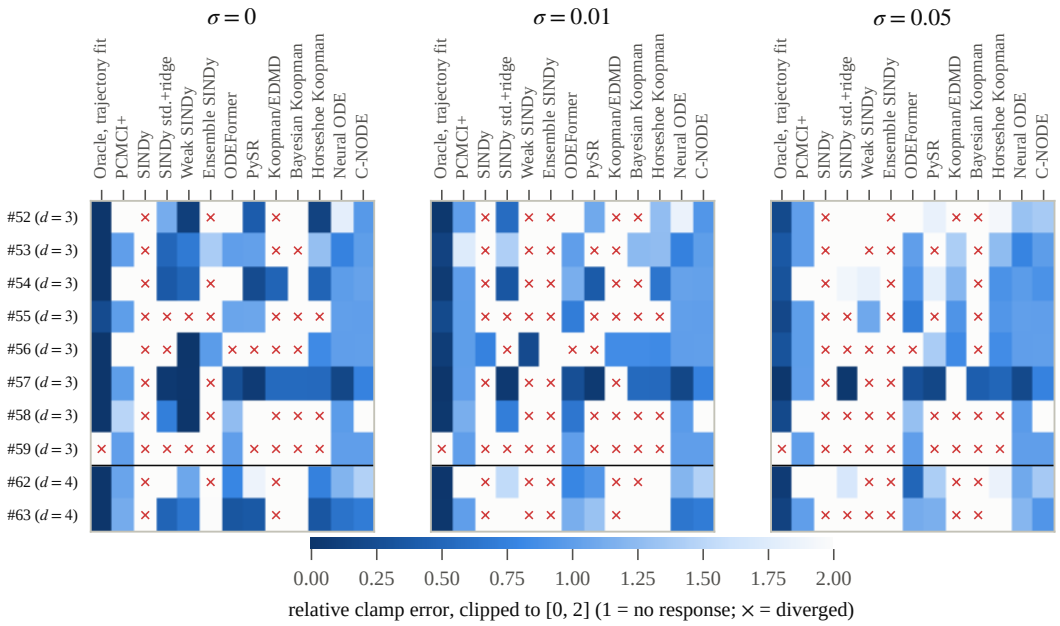


Figure 6: Per-system clamp error on the 10 sparse ODEBench systems (rows, sorted by dimension) for the trajectory-fit oracle and the blind methods, at the three noise levels.

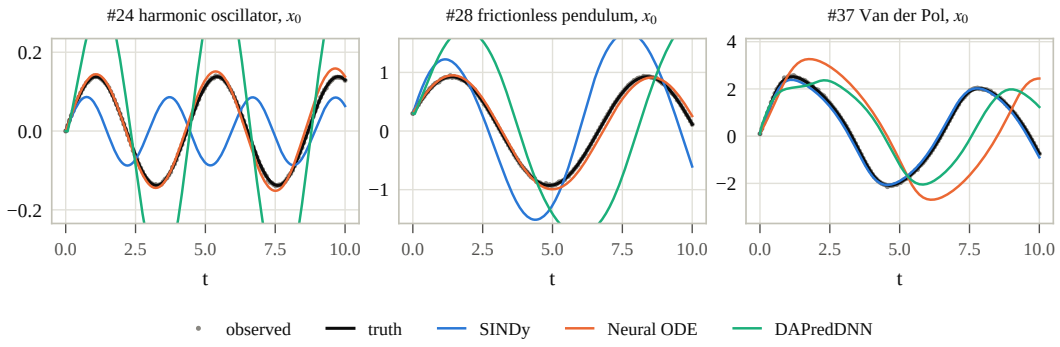


Figure 7: Second initial condition, $\sigma = 0.01$. All three models forecast the held-out window of the first trajectory with $R^2 \geq 0.99$ on all three systems, and none follows all three new trajectories.

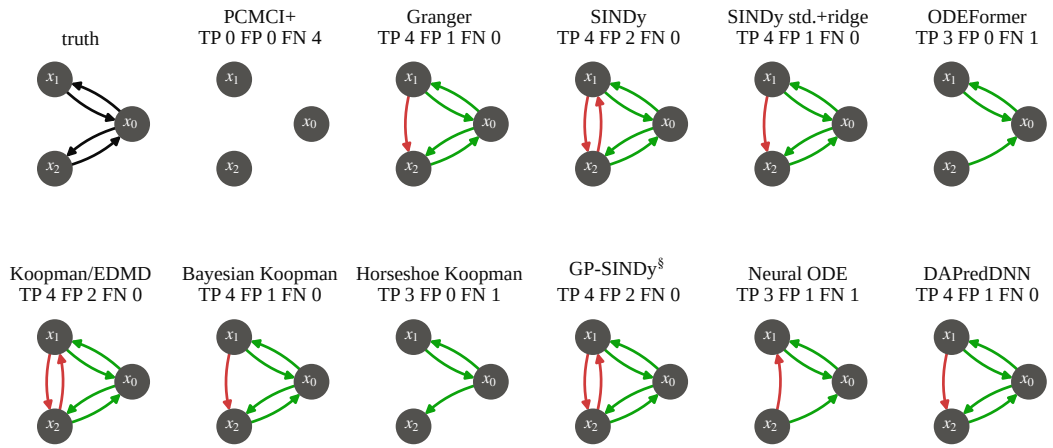


Figure 8: Graph read-outs on the Rössler system with a stable fixed point (ODEBench #57, $\sigma = 0.01$). The true graph has 4 edges, and x_1 and x_2 do not influence each other. Green: correct edge; red: spurious edge. ODEFormer and the horseshoe test leave both absent edges out and miss one true edge. The dictionary read-outs add one or both absent edges, and PCMCI+ returns no cross edge. Weak and Ensemble SINDy, PySR and C-NODE are not shown.

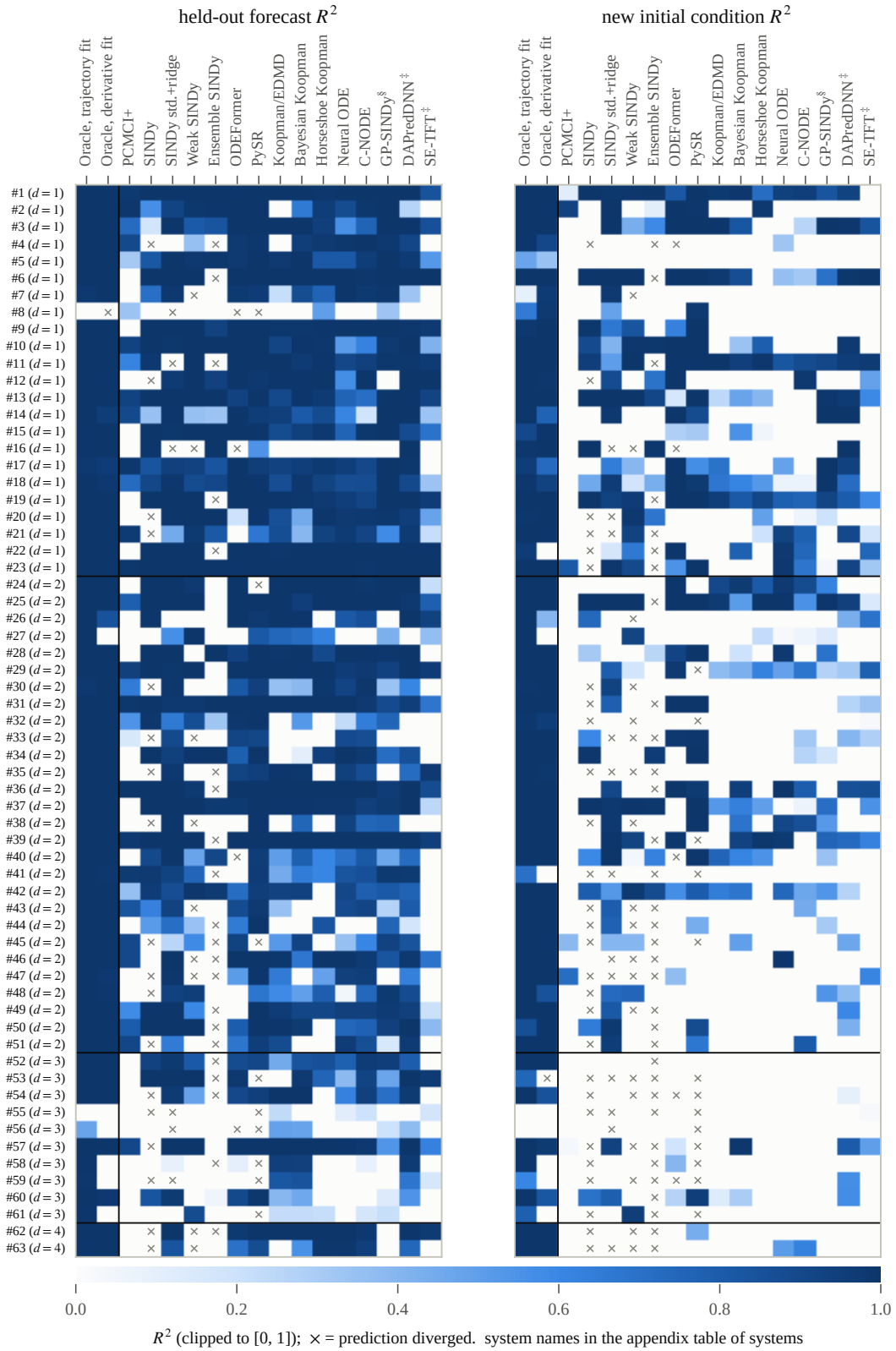


Figure 9: Per-system R^2 at $\sigma = 0.01$, clipped to $[0, 1]$; $\times$ marks a diverged prediction. Systems are sorted by dimension (horizontal rules) and ODEBench id; names are in Table 20.

Table 20: The 63 ODEBench systems: dimension, description, whether the true equation lies in the shared library, whether its causal graph has an absent edge, and the largest Lyapunov exponent (for $d \geq 3$); *left out of the new-IC questions. Merged equation families: #3, 6, 12, 14, 15, 17–19 (logistic growth with harvesting or predation terms); #9–10; #11, 16; #24–25; #28, 43, 44; #37, 40; #54–56 (Lorenz); #57–59 (Rössler); #40, which ODEBench labels Duffing, has Van der Pol's equation (#37). Every other system is its own family.

Id	d	Description	In library	Sparse	λ_1
1	1	RC-circuit (charging capacitor)	yes		
2	1	Population growth (naive)	yes		
3	1	Population growth with carrying capacity	yes		
4	1	RC-circuit with non-linear resistor (charging capacitor)	no		
5	1	Velocity of a falling object with air resistance	yes		
6	1	Autocatalysis with one fixed abundant chemical	yes		
7	1	Gompertz law for tumor growth	no		
8	1	Logistic equation with Allee effect	yes		
9	1	Language death model for two languages	yes		
10	1	Refined language death model for two languages	no		
11	1	Naive critical slowing down (statistical mechanics)	yes		
12	1	Photons in a laser (simple)	yes		
13	1	Overdamped bead on a rotating hoop	no		
14	1	Budworm outbreak model with predation	no		
15	1	Budworm outbreak with predation (dimensionless)	no		
16	1	Landau equation (typical time scale tau = 1)	no		
17	1	Logistic equation with harvesting/fishing	yes		
18	1	Improved logistic equation with harvesting/fishing	no		
19	1	Improved logistic equation with harvesting/fishing (dimensionless)	no		
20	1	Autocatalytic gene switching (dimensionless)	no		
21	1	Dimensionally reduced SIR infection model for dead people (dimensionless)	no		
22	1	Hysteretic activation of a protein expression (positive feedback, basal promoter expression)	no		
23	1	Overdamped pendulum with constant driving torque/fireflies/Josephson junction (dimensionless)	yes		
24	2	Harmonic oscillator without damping	yes		
25	2	Harmonic oscillator with damping	yes		
26	2	Lotka-Volterra competition model (Strogatz version with sheeps and rabbits)	yes		
27	2	Lotka-Volterra simple (as on Wikipedia)	yes		
28	2	Pendulum without friction	yes		
29	2	Dipole fixed point	yes		
30	2	RNA molecules catalyzing each others replication	yes		
31	2	SIR infection model only for healthy and sick	yes		
32	2	Damped double well oscillator	yes		
33	2	Glider (dimensionless)	no		
34	2	Frictionless bead on a rotating hoop (dimensionless)	no		
35	2	Rotational dynamics of an object in a shear flow	no		
36	2	Pendulum with non-linear damping, no driving (dimensionless)	no		
37	2	Van der Pol oscillator (standard form)	yes		
38	2	Van der Pol oscillator (simplified form from Strogatz)	yes		
39	2	Glycolytic oscillator, e.g., ADP and F6P in yeast (dimensionless)	yes		
40	2	Duffing equation (weakly non-linear oscillation)	yes		
41	2	Cell cycle model by Tyson for interaction between protein cdc2 and cyclin (dimensionless)	yes		
42	2	Reduced model for chlorine dioxide-iodine-malonic acid rection (dimensionless)	no		
43	2	Driven pendulum with linear damping / Josephson junction (dimensionless)	yes		
44	2	Driven pendulum with quadratic damping (dimensionless)	no		
45	2	Isothermal autocatalytic reaction model by Gray and Scott 1985 (dimensionless)	yes		
46	2	Interacting bar magnets	no		
47	2	Binocular rivalry model (no oscillations)	no		
48	2	Bacterial respiration model for nutrients and oxygen levels	no		
49	2	Brusselator: hypothetical chemical oscillation model (dimensionless)	yes		
50	2	Chemical oscillator model by Schnackenberg 1979 (dimensionless)	yes		
51	2	Oscillator death model by Ermentrout and Kopell 1990	no		
52	3	Maxwell-Bloch equations (laser dynamics)	yes	yes	-0.18
53	3	Model for apoptosis (cell death)	no	yes	+0.00
54	3	Lorenz equations in well-behaved periodic regime	yes	yes	-0.21
55	3	Lorenz equations in complex periodic regime	yes	yes	+0.00
56	3	Lorenz equations standard parameters (chaotic)	yes	yes	+0.91*
57	3	Rössler attractor (stable fixed point)	yes	yes	-0.52
58	3	Rössler attractor (periodic)	yes	yes	-0.00
59	3	Rössler attractor (chaotic)	yes	yes	+0.36*
60	3	Aizawa attractor (chaotic)	no		-0.00
61	3	Chen-Lee attractor; system for gyro motion with feedback control of rigid body (chaotic)	yes		+0.63*
62	4	Binocular rivalry model with adaptation (oscillations)	no	yes	-0.93
63	4	SEIR infection model (proportions)	yes	yes	+0.00

F THE TWO SPARSE SUITES

Hand-written suite. Figure 10 and Table 21 give its scores, Tables 22 and 23 their intervals, Figure 11 every system's clamp error, and Table 24 its systems.

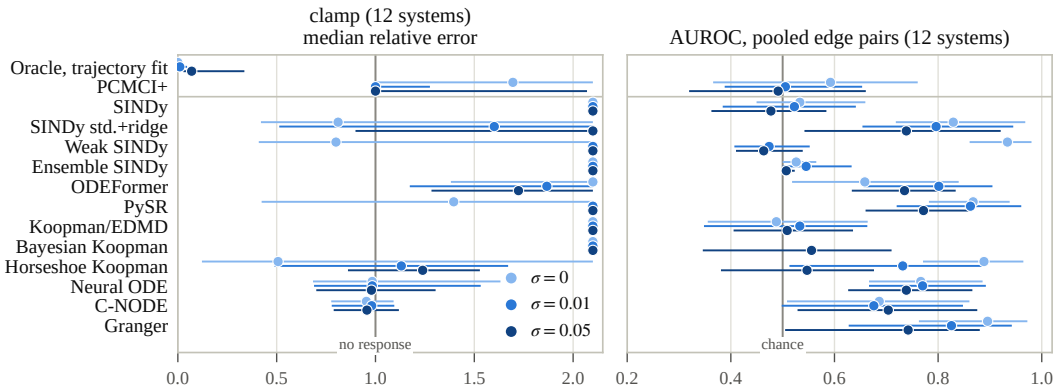


Figure 10: Hand-written suite (12 systems, $d = 3$–5). Whiskers are 95% bootstrap intervals over its systems; clamp errors are clipped at 2.1 for display, so the dictionary methods that diverge sit at the right edge. Granger answers the graph question only.

Table 21: Hand-written suite (12 systems, $d = 3$–5, 71 absent edges), $\sigma = 0 \,/\, 0.01 \,/\, 0.05$. Forecast and new IC: % with $R^2 \geq 0.9$. Clamp: median relative error (no response = 1). F1 averages systems, specificity pools the absent edges, balanced accuracy is (pooled recall + specificity)/2, 0.5 for the complete graph, and AUROC pools the within-system (true, absent) edge pairs. Intervals, and the appendix-only rows, are in Tables 22 and 23; "–" marks a score undefined because a fit failed on one system.

	Forecast ↑	New IC ↑	Clamp ↓	F1 ↑	Specificity ↑	Bal. acc. ↑	AUROC ↑
references							
Trivial	17/25/8	0/0/0	1.00/1.00/1.00	0.59/0.59/0.59	0.00/0.00/0.00	0.50/0.50/0.50	0.50/0.50/0.50
Oracle, trajectory fit	100/100/92	100/100/75	0.00/0.01/0.07	–	–	–	–
blind methods							
PCMCI+	0/0/0	0/0/0	1.70/1.00/1.00	0.46/0.18/0.04	0.87/0.99/0.94	0.64/0.56/0.48	0.59/0.51/0.49
SINDy	25/8/8	0/0/0	∞/∞/∞	0.60/0.59/0.59	0.04/0.00/0.00	0.52/0.50/0.50	0.53/0.52/0.48
SINDy (std.+ridge)	67/33/25	8/0/0	0.81/1.60/3.75	0.66/0.65/0.61	0.31/0.30/0.08	0.64/0.64/0.54	0.83/0.80/0.74
Weak SINDy	75/0/0	17/0/0	0.80/∞/∞	0.63/0.59/0.59	0.14/0.00/0.00	0.57/0.50/0.50	0.93/0.47/0.46
Ensemble SINDy	0/0/0	0/0/0	∞/∞/∞	0.60/0.59/0.59	0.04/0.00/0.00	0.52/0.50/0.50	0.53/0.55/0.51
ODEFormer	17/25/25	0/0/0	3.74/1.87/1.72	0.67/0.70/0.68	0.63/0.69/0.66	0.68/0.75/0.71	0.66/0.80/0.74
PySR	42/25/8	8/17/8	1.40/∞/12.08	0.81/0.80/0.67	0.72/0.79/0.59	0.84/0.84/0.71	0.87/0.86/0.77
Koopman/EDMD	50/33/17	0/0/0	∞/∞/∞	0.59/0.59/0.59	0.00/0.00/0.00	0.50/0.50/0.50	0.49/0.53/0.51
Bayesian Koopman	58/58/25	0/8/0	∞/∞/∞	–/–/0.26	–/–/0.68	–/–/0.53	–/–/0.56
Horseshoe Koopman	58/50/8	17/0/0	0.51/1.13/1.24	0.77/0.71/0.64	0.80/0.61/0.32	0.86/0.74/0.62	0.89/0.73/0.55
Neural ODE	42/42/25	0/0/8	0.98/0.98/0.98	0.60/0.60/0.62	0.17/0.17/0.14	0.56/0.56/0.57	0.77/0.77/0.74
C-NODE	42/42/33	0/0/0	0.95/0.98/0.96	0.64/0.63/0.64	0.32/0.31/0.34	0.62/0.61/0.61	0.69/0.68/0.70
Granger	–	–	–	0.65/0.66/0.61	0.35/0.72/0.72	0.63/0.73/0.68	0.90/0.83/0.74

Random suite. Table 25 gives its scores, Tables 26 and 27 their intervals, Figure 13 every system's clamp error, Table 28 its systems. The 60 equations themselves, with their constants, are in `benchmark/random_systems.json` of the supplementary material, regenerated exactly by `random_systems.py` from seed 2027; Section 3 states the generator that draws them.

Table 22: Hand-written suite, trajectory questions: median relative error with 95% bootstrap intervals over its 12 systems, and the size ratio and shape of the predicted clamp response among finite predictions (below three finite predictions, the shape is given with its n and no interval); "$> 10^3$" marks a response larger than the true one by three orders of magnitude.

Method	σ	Perturbation	Clamp	Size ratio	Shape
Oracle, trajectory fit	0	0.00 [0.00, 0.00]	0.00 [0.00, 0.00]	1.00	1.00 [1.00, 1.00]
	0.01	0.01 [0.00, 0.02]	0.01 [0.00, 0.05]	1.00	1.00 [1.00, 1.00]
	0.05	0.07 [0.02, 0.12]	0.07 [0.03, 0.34]	1.00	1.00 [0.96, 1.00]
Oracle, derivative fit	0	0.01 [0.00, 0.05]	0.01 [0.00, 0.03]	1.00	1.00 [1.00, 1.00]
	0.01	0.01 [0.01, 0.05]	0.03 [0.01, 0.05]	1.00	1.00 [1.00, 1.00]
	0.05	0.06 [0.03, 0.17]	0.09 [0.04, 0.27]	0.99	1.00 [1.00, 1.00]
PCMCI+	0	1.09 [0.78, 1.26]	1.70 [1.01, 13.92]	1.44	0.04 [-0.14, 0.51]
	0.01	1.03 [0.85, 1.16]	1.00 [1.00, 1.28]	0.00	0.00 [0.00, 0.00]
	0.05	0.93 [0.85, 1.05]	1.00 [1.00, 2.07]	0.00	0.00 [-0.00, 0.00]
SINDy	0	∞ [1.21, ∞]	∞ [∞, ∞]	0.69	0.96 (n=1)
	0.01	∞ [4.09, ∞]	∞ [∞, ∞]	3.45	-0.30 (n=2)
	0.05	∞ [5.68, ∞]	∞ [∞, ∞]	3.74	0.08 (n=1)
SINDy (std.+ridge)	0	0.71 [0.25, 1.37]	0.81 [0.42, 9.62]	0.96	0.86 [0.20, 0.91]
	0.01	1.16 [0.48, 3.38]	1.60 [0.51, ∞]	1.19	0.62 [0.03, 0.91]
	0.05	1.12 [0.77, 9.70]	3.75 [0.90, ∞]	1.10	0.63 [0.06, 0.74]
Weak SINDy	0	0.45 [0.20, 0.83]	0.80 [0.41, ∞]	0.98	0.86 [0.75, 0.98]
	0.01	∞ [∞, ∞]	∞ [∞, ∞]	25.79	0.30 (n=2)
	0.05	∞ [40.42, ∞]	∞ [∞, ∞]	36.22	-0.06 (n=2)
Ensemble SINDy	0	∞ [2.00, ∞]	∞ [∞, ∞]	1.00	0.21 [-0.41, 0.49]
	0.01	∞ [∞, ∞]	∞ [∞, ∞]	21.29	0.28 (n=1)
	0.05	∞ [1.17, ∞]	∞ [∞, ∞]	–	–
ODEFormer	0	1.38 [0.83, ∞]	3.74 [1.38, ∞]	1.53	0.32 [-0.39, 0.77]
	0.01	1.02 [0.83, 1.32]	1.87 [1.17, 4.81]	1.54	0.36 [-0.06, 0.79]
	0.05	1.01 [0.81, 7.95]	1.72 [1.28, 6.63]	1.48	0.40 [-0.15, 0.78]
PySR	0	0.78 [0.32, ∞]	1.40 [0.42, ∞]	0.83	0.89 [0.77, 0.99]
	0.01	1.43 [0.72, ∞]	∞ [4.27, ∞]	1.44	0.75 [-0.61, 1.00]
	0.05	1.55 [0.91, 7.70]	12.08 [2.72, ∞]	3.18	0.31 [0.04, 0.70]
Koopman/EDMD	0	425 [6.93, $> 10^3$]	∞ [∞, ∞]	–	–
	0.01	1.73 [1.08, $> 10^3$]	∞ [∞, ∞]	$> 10^3$	0.23 (n=2)
	0.05	1.82 [1.06, 93.25]	∞ [∞, ∞]	3.51	-0.18 (n=1)
Bayesian Koopman	0	0.97 [0.81, 2.54]	∞ [∞, ∞]	0.78	0.74 (n=2)
	0.01	0.83 [0.77, 1.96]	∞ [∞, ∞]	$> 10^3$	0.43 (n=2)
	0.05	0.94 [0.77, 1.11]	∞ [∞, ∞]	–	–
Horseshoe Koopman	0	6.96 [0.30, $> 10^3$]	0.51 [0.12, ∞]	0.93	0.99 [0.73, 1.00]
	0.01	1.22 [0.82, 3.60]	1.13 [0.49, 1.67]	1.28	0.70 [0.23, 0.96]
	0.05	0.97 [0.86, 5.21]	1.24 [0.86, 1.53]	1.08	0.39 [-0.06, 0.61]
Neural ODE	0	0.90 [0.61, 1.32]	0.98 [0.68, 1.63]	0.94	0.50 [0.28, 0.81]
	0.01	0.91 [0.60, 1.32]	0.98 [0.69, 1.53]	0.93	0.51 [0.29, 0.81]
	0.05	0.82 [0.56, 1.16]	0.98 [0.70, 1.30]	0.84	0.50 [0.28, 0.79]
C-NODE	0	0.89 [0.71, 0.96]	0.95 [0.78, 1.09]	0.94	0.56 [0.30, 0.71]
	0.01	0.91 [0.71, 0.97]	0.98 [0.78, 1.10]	0.93	0.56 [0.29, 0.71]
	0.05	0.80 [0.68, 0.95]	0.96 [0.79, 1.12]	0.71	0.53 [0.30, 0.68]
GP-SINDy[§]	0	3.66 [2.39, 7.73]	2.53 [1.69, 7.55]	2.55	0.24 [0.08, 0.46]
	0.01	5.64 [1.96, 9.97]	4.83 [2.09, 15.88]	4.97	0.24 [-0.04, 0.58]
	0.05	4.23 [2.18, 5.05]	3.17 [1.80, 8.26]	3.05	0.15 [-0.02, 0.34]
DAPredDNN[‡]	0	0.89 [0.83, 0.94]	1.91 [0.99, 4.48]	1.84	0.36 [-0.25, 0.60]
	0.01	0.93 [0.82, 1.03]	1.69 [1.06, 2.19]	1.64	0.14 [-0.15, 0.56]
	0.05	0.99 [0.88, 1.25]	1.48 [1.00, 2.67]	1.58	0.11 [-0.13, 0.47]
SE-TFT[‡]	0	1.00 [0.97, 1.14]	1.70 [1.03, 3.99]	1.28	-0.07 [-0.33, 0.09]
	0.01	1.00 [1.00, 1.47]	1.73 [1.00, 3.18]	1.24	-0.08 [-0.27, 0.08]
	0.05	1.00 [0.98, 1.18]	1.91 [1.02, 5.47]	1.44	-0.13 [-0.31, 0.24]

Table 23: Hand-written suite, graph question, with 95% bootstrap intervals over its 12 systems. F1 averages systems, specificity pools the 71 absent edges, and AUROC is per system and pooled over within-system (true, absent) edge pairs. "–" marks a score that is undefined because a fit failed on one system, which leaves that system unscored.

Method	σ	F1	Specificity	AUROC, per system	AUROC, pooled
PCMCI+	0	0.465 [0.311, 0.605]	0.87 [0.75, 0.96]	0.59 [0.42, 0.74]	0.592 [0.366, 0.761]
	0.01	0.179 [0.042, 0.333]	0.99 [0.96, 1.00]	0.55 [0.41, 0.68]	0.505 [0.388, 0.653]
	0.05	0.042 [0.000, 0.125]	0.94 [0.89, 1.00]	0.54 [0.38, 0.69]	0.491 [0.319, 0.660]
SINDy	0	0.598 [0.530, 0.669]	0.04 [0.00, 0.13]	0.53 [0.40, 0.67]	0.533 [0.450, 0.660]
	0.01	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.51 [0.38, 0.64]	0.523 [0.384, 0.641]
	0.05	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.48 [0.32, 0.63]	0.477 [0.362, 0.584]
SINDy (std.+ridge)	0	0.658 [0.590, 0.724]	0.31 [0.18, 0.40]	0.86 [0.76, 0.96]	0.829 [0.718, 0.968]
	0.01	0.650 [0.573, 0.721]	0.30 [0.12, 0.44]	0.81 [0.68, 0.93]	0.796 [0.654, 0.945]
	0.05	0.609 [0.535, 0.678]	0.08 [0.00, 0.19]	0.72 [0.53, 0.88]	0.739 [0.542, 0.921]
Weak SINDy	0	0.632 [0.539, 0.730]	0.14 [0.00, 0.36]	0.90 [0.78, 0.99]	0.934 [0.861, 0.980]
	0.01	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.49 [0.41, 0.56]	0.474 [0.407, 0.552]
	0.05	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.47 [0.34, 0.60]	0.463 [0.410, 0.539]
Ensemble SINDy	0	0.598 [0.530, 0.669]	0.04 [0.00, 0.13]	0.52 [0.50, 0.55]	0.526 [0.500, 0.565]
	0.01	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.53 [0.50, 0.59]	0.545 [0.500, 0.633]
	0.05	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.50 [0.50, 0.51]	0.507 [0.500, 0.523]
ODEFormer	0	0.670 [0.526, 0.804]	0.63 [0.50, 0.80]	0.72 [0.62, 0.83]	0.659 [0.518, 0.840]
	0.01	0.698 [0.587, 0.804]	0.69 [0.62, 0.77]	0.78 [0.69, 0.87]	0.801 [0.656, 0.905]
	0.05	0.680 [0.580, 0.784]	0.66 [0.54, 0.78]	0.76 [0.67, 0.85]	0.735 [0.633, 0.834]
PySR	0	0.808 [0.720, 0.892]	0.72 [0.57, 0.83]	0.87 [0.79, 0.95]	0.868 [0.782, 0.938]
	0.01	0.796 [0.698, 0.885]	0.79 [0.67, 0.88]	0.81 [0.68, 0.92]	0.862 [0.720, 0.960]
	0.05	0.666 [0.559, 0.761]	0.59 [0.43, 0.70]	0.76 [0.65, 0.86]	0.772 [0.660, 0.858]
Koopman/EDMD	0	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.52 [0.36, 0.68]	0.488 [0.355, 0.664]
	0.01	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.48 [0.32, 0.62]	0.533 [0.348, 0.663]
	0.05	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.53 [0.40, 0.66]	0.509 [0.405, 0.636]
Bayesian Koopman	0	– [–, –]	– [–, –]	– [–, –]	– [–, –]
	0.01	– [–, –]	– [–, –]	– [–, –]	– [–, –]
	0.05	0.264 [0.095, 0.438]	0.68 [0.59, 0.77]	0.45 [0.34, 0.58]	0.556 [0.346, 0.710]
Horseshoe Koopman	0	0.768 [0.587, 0.921]	0.80 [0.60, 0.94]	0.83 [0.69, 0.94]	0.889 [0.771, 0.964]
	0.01	0.708 [0.562, 0.834]	0.61 [0.35, 0.82]	0.62 [0.45, 0.79]	0.732 [0.513, 0.886]
	0.05	0.638 [0.550, 0.722]	0.32 [0.11, 0.58]	0.48 [0.32, 0.65]	0.547 [0.381, 0.676]
Neural ODE	0	0.600 [0.536, 0.662]	0.17 [0.06, 0.28]	0.72 [0.61, 0.84]	0.767 [0.667, 0.886]
	0.01	0.600 [0.536, 0.662]	0.17 [0.06, 0.28]	0.73 [0.61, 0.85]	0.770 [0.667, 0.892]
	0.05	0.621 [0.545, 0.688]	0.14 [0.04, 0.26]	0.68 [0.54, 0.82]	0.739 [0.626, 0.866]
C-NODE	0	0.644 [0.567, 0.712]	0.32 [0.17, 0.51]	0.69 [0.55, 0.83]	0.686 [0.508, 0.860]
	0.01	0.635 [0.566, 0.694]	0.31 [0.16, 0.49]	0.68 [0.54, 0.80]	0.676 [0.498, 0.848]
	0.05	0.639 [0.550, 0.715]	0.34 [0.21, 0.50]	0.73 [0.61, 0.85]	0.704 [0.529, 0.876]
Granger	0	0.650 [0.557, 0.740]	0.35 [0.18, 0.59]	0.83 [0.65, 0.96]	0.895 [0.763, 0.972]
	0.01	0.656 [0.522, 0.789]	0.72 [0.54, 0.87]	0.74 [0.58, 0.90]	0.826 [0.627, 0.942]
	0.05	0.605 [0.411, 0.777]	0.72 [0.52, 0.89]	0.65 [0.47, 0.83]	0.742 [0.505, 0.880]
GP-SINDy[§]	0	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.48 [0.34, 0.60]	0.537 [0.422, 0.610]
	0.01	– [–, –]	– [–, –]	– [–, –]	– [–, –]
	0.05	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.45 [0.31, 0.59]	0.502 [0.365, 0.587]
DAPredDNN[‡]	0	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.58 [0.46, 0.70]	0.620 [0.492, 0.728]
	0.01	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.57 [0.43, 0.73]	0.592 [0.440, 0.740]
	0.05	0.590 [0.515, 0.664]	0.00 [0.00, 0.00]	0.46 [0.31, 0.61]	0.425 [0.335, 0.517]
SE-TFT[‡]	0	0.365 [0.284, 0.438]	0.65 [0.55, 0.73]	0.50 [0.42, 0.59]	0.505 [0.415, 0.578]
	0.01	0.329 [0.226, 0.429]	0.61 [0.49, 0.69]	0.45 [0.32, 0.59]	0.488 [0.390, 0.563]
	0.05	0.338 [0.233, 0.436]	0.62 [0.51, 0.71]	0.44 [0.32, 0.55]	0.502 [0.373, 0.589]

Table 24: The hand-written suite: dimension, whether the equation lies in the shared library, number of absent edges, and largest Lyapunov exponent. Equations are in `benchmark/sparse_systems.py`.

Id	d	Description	In library	Absent	λ_1
101	3	Lotka-Volterra food chain (resource, consumer, predator)	yes	2	-0.22
102	3	Two prey sharing one predator, no prey competition	yes	2	-0.25
103	3	Repressilator, protein-only (three genes in a repression ring)	no	3	-0.00
104	3	Genetic toggle switch with a reporter gene	no	3	-0.17
105	4	FitzHugh-Nagumo neurons, one-way coupling (drive and response)	yes	7	-0.00
106	4	Damped linear oscillator driving a two-stage linear cascade	yes	8	-0.10
107	5	Linear five-stage ring with negative feedback	yes	15	-0.03
108	4	Van der Pol oscillator driving a detuned Van der Pol oscillator	yes	7	+0.00
109	3	Michaelis-Menten metabolic cascade	no	4	-0.17
110	4	Kuramoto star: a pacemaker phase entraining three oscillators	no	9	+0.00
111	3	Brusselator driving a first-order linear filter	yes	3	-0.00
112	4	Predator-prey and a damped pendulum, uncoupled	yes	8	+0.01

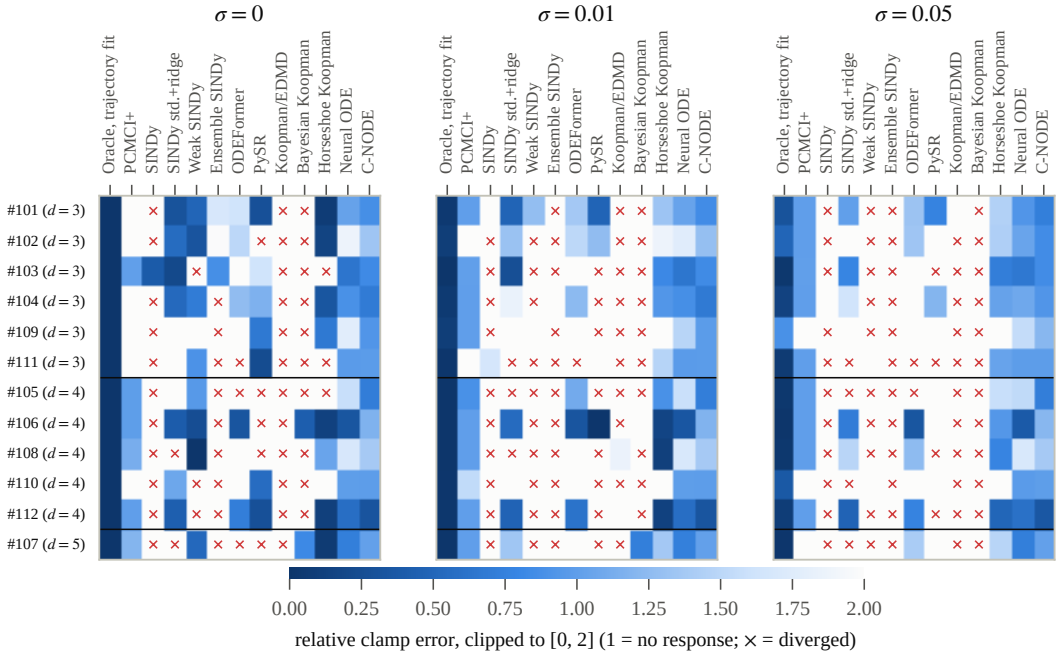


Figure 11: Per-system clamp error on the hand-written suite.

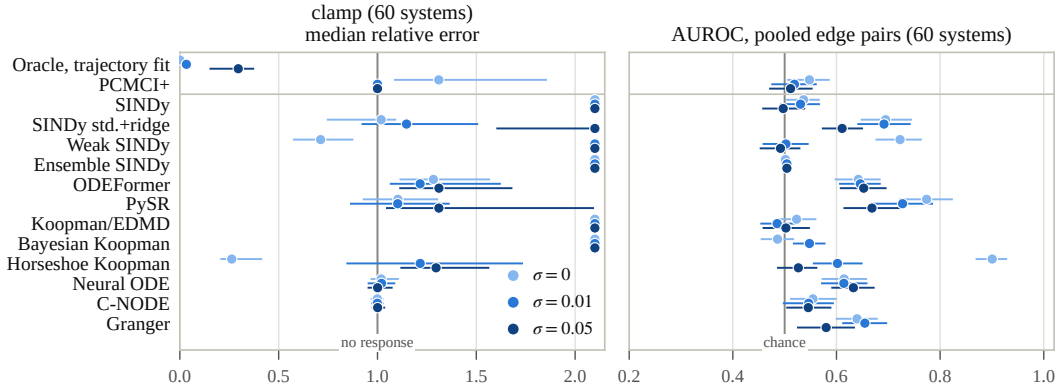


Figure 12: Random suite (60 systems, $d = 3$–5). Whiskers are 95% bootstrap intervals over its systems; clamp errors are clipped at 2.1 for display, so methods that diverge sit at the right edge. Granger answers the graph question only.

Table 25: Random suite (60 systems, $d = 3$–5, 411 absent edges), $\sigma = 0 \,/\, 0.01 \,/\, 0.05$; columns as in Table 21.

	Forecast ↑	New IC ↑	Clamp ↓	F1 ↑	Specificity ↑	Bal. acc. ↑	AUROC ↑
references							
Trivial	0/0/8	0/0/0	1.00/1.00/1.00	0.67/0.67/0.67	0.00/0.00/0.00	0.50/0.50/0.50	0.50/0.50/0.50
Oracle, trajectory fit	100/100/78	100/92/33	0.00/0.03/0.30	–	–	–	–
blind methods							
PCMCI+	3/2/2	0/0/0	1.31/1.00/1.00	0.39/0.13/0.10	0.82/0.98/0.98	0.56/0.53/0.52	0.55/0.52/0.51
SINDy	8/5/5	0/0/0	∞/∞/∞	0.67/0.67/0.67	0.00/0.00/0.00	0.50/0.50/0.50	0.54/0.53/0.50
SINDy (std.+ridge)	53/37/13	0/0/0	1.02/1.15/2.26	0.68/0.68/0.67	0.15/0.08/0.01	0.55/0.53/0.50	0.69/0.69/0.61
Weak SINDy	77/2/2	13/0/0	0.71/∞/∞	0.69/0.67/0.67	0.04/0.00/0.00	0.52/0.50/0.50	0.72/0.50/0.49
Ensemble SINDy	0/0/0	0/0/0	∞/∞/∞	0.67/0.67/0.67	0.00/0.00/0.00	0.50/0.50/0.50	0.50/0.50/0.50
ODEFormer	12/13/12	0/0/0	1.28/1.22/1.31	0.59/0.62/0.61	0.76/0.77/0.76	0.64/0.66/0.65	0.64/0.65/0.65
PySR	57/48/23	7/7/2	1.10/1.10/1.31	0.74/0.71/0.66	0.56/0.61/0.58	0.70/0.69/0.65	0.77/0.73/0.67
Koopman/EDMD	45/10/7	0/0/0	∞/∞/∞	0.67/0.67/0.67	0.00/0.00/0.00	0.50/0.50/0.50	0.52/0.49/0.50
Bayesian Koopman	35/32/20	0/0/2	∞/∞/∞	0.67/0.57/–	0.00/0.34/–	0.50/0.53/–	0.49/0.55/–
Horseshoe Koopman	58/27/8	0/0/0	0.26/1.22/1.30	0.86/0.69/0.66	0.88/0.43/0.19	0.87/0.63/0.54	0.90/0.60/0.53
Neural ODE	30/30/27	0/0/0	1.02/1.02/1.00	0.65/0.65/0.67	0.18/0.18/0.18	0.53/0.53/0.54	0.61/0.61/0.63
C-NODE	22/20/22	0/0/0	1.00/1.00/1.00	0.58/0.57/0.58	0.37/0.37/0.36	0.52/0.52/0.52	0.55/0.55/0.55
Granger	–	–	–	0.66/0.64/0.49	0.21/0.61/0.72	0.54/0.64/0.58	0.64/0.65/0.58

Table 26: Random suite, trajectory questions: median relative error with 95% bootstrap intervals over its 60 systems, and the size ratio and shape of the predicted clamp response among finite predictions (below three finite predictions, the shape is given with its n and no interval).

Method	σ	Perturbation	Clamp	Size ratio	Shape
Oracle, trajectory fit	0	0.00 [0.00, 0.00]	0.00 [0.00, 0.00]	1.00	1.00 [1.00, 1.00]
	0.01	0.02 [0.01, 0.03]	0.03 [0.03, 0.06]	1.00	1.00 [1.00, 1.00]
	0.05	0.14 [0.09, 0.21]	0.30 [0.15, 0.38]	0.95	0.97 [0.96, 0.99]
Oracle, derivative fit	0	0.02 [0.01, 0.04]	0.02 [0.01, 0.06]	1.00	1.00 [1.00, 1.00]
	0.01	0.05 [0.03, 0.09]	0.08 [0.06, 0.11]	0.99	1.00 [1.00, 1.00]
	0.05	0.19 [0.10, 0.28]	0.28 [0.18, 0.37]	0.97	0.98 [0.96, 0.99]
PCMCI+	0	1.01 [0.92, 1.23]	1.31 [1.08, 1.86]	0.80	0.05 [0.00, 0.14]
	0.01	1.00 [0.94, 1.07]	1.00 [1.00, 1.00]	0.00	0.00 [0.00, 0.00]
	0.05	0.94 [0.85, 0.99]	1.00 [1.00, 1.00]	0.00	0.00 [0.00, 0.00]
SINDy	0	∞ [∞, ∞]	∞ [∞, ∞]	1.24	0.20 [-0.37, 0.46]
	0.01	∞ [∞, ∞]	∞ [∞, ∞]	1.98	0.42 (n=2)
	0.05	∞ [∞, ∞]	∞ [∞, ∞]	1.63	0.04 [-0.27, 0.63]
SINDy (std.+ridge)	0	0.90 [0.70, 1.34]	1.02 [0.74, 1.09]	0.96	0.73 [0.51, 0.79]
	0.01	1.00 [0.85, 1.70]	1.15 [0.92, 1.51]	0.92	0.64 [0.36, 0.76]
	0.05	1.54 [1.27, 2.93]	2.26 [1.60, 6.14]	1.37	0.27 [0.09, 0.48]
Weak SINDy	0	0.49 [0.34, 0.65]	0.71 [0.57, 0.88]	1.00	0.89 [0.80, 0.94]
	0.01	∞ [∞, ∞]	∞ [∞, ∞]	1.83	-0.05 (n=2)
	0.05	∞ [∞, ∞]	∞ [∞, ∞]	3.06	0.39 [0.03, 0.71]
Ensemble SINDy	0	∞ [∞, ∞]	∞ [∞, ∞]	3.07	-0.25 [-0.44, 0.26]
	0.01	∞ [∞, ∞]	∞ [∞, ∞]	2.03	-0.04 (n=2)
	0.05	∞ [∞, ∞]	∞ [∞, ∞]	4.33	-0.24 [-0.52, 0.21]
ODEFormer	0	0.88 [0.78, 1.32]	1.28 [1.11, 1.57]	1.01	0.32 [0.16, 0.46]
	0.01	0.97 [0.84, 1.36]	1.22 [1.06, 1.62]	0.99	0.32 [0.15, 0.47]
	0.05	0.97 [0.83, 1.27]	1.31 [1.11, 1.68]	1.00	0.27 [0.10, 0.38]
PySR	0	0.96 [0.65, 1.33]	1.10 [0.93, 1.31]	0.99	0.51 [0.34, 0.71]
	0.01	0.88 [0.65, 1.11]	1.10 [0.86, 1.37]	1.10	0.65 [0.44, 0.73]
	0.05	1.33 [1.07, 2.16]	1.31 [1.04, 2.10]	1.08	0.45 [0.24, 0.62]
Koopman/EDMD	0	261 [22.25, $> 10^3$]	∞ [∞, ∞]	1.16	0.83 [0.07, 0.97]
	0.01	13.07 [4.52, 41.97]	∞ [∞, ∞]	1.84	0.47 [-0.17, 0.96]
	0.05	3.23 [2.06, 12.32]	∞ [∞, ∞]	1.35	0.71 [0.36, 0.92]
Bayesian Koopman	0	1.24 [0.97, 2.43]	∞ [∞, ∞]	0.93	0.44 [0.05, 0.58]
	0.01	1.00 [0.83, 1.38]	∞ [∞, ∞]	0.60	0.57 [0.30, 0.71]
	0.05	1.07 [0.92, 1.39]	∞ [∞, ∞]	0.75	0.62 [0.32, 0.72]
Horseshoe Koopman	0	7.36 [2.99, 23.43]	0.26 [0.20, 0.42]	0.97	0.98 [0.93, 0.99]
	0.01	3.07 [1.31, 9.77]	1.22 [0.84, 1.74]	0.99	0.68 [0.56, 0.78]
	0.05	1.37 [0.99, 6.44]	1.30 [1.12, 1.57]	1.00	0.34 [0.26, 0.48]
Neural ODE	0	0.96 [0.84, 1.06]	1.02 [0.96, 1.11]	0.61	0.37 [0.18, 0.52]
	0.01	0.96 [0.84, 1.05]	1.02 [0.95, 1.09]	0.60	0.36 [0.18, 0.53]
	0.05	0.93 [0.83, 1.00]	1.00 [0.95, 1.08]	0.56	0.34 [0.18, 0.50]
C-NODE	0	1.10 [0.93, 1.29]	1.00 [0.97, 1.03]	0.45	0.23 [0.13, 0.39]
	0.01	1.10 [0.93, 1.30]	1.00 [0.97, 1.03]	0.45	0.24 [0.15, 0.39]
	0.05	1.00 [0.91, 1.15]	1.00 [0.97, 1.04]	0.41	0.21 [0.11, 0.35]

Table 27: Random suite, graph question, with 95% bootstrap intervals over its 60 systems; specificity pools the 411 absent edges. F1 and pooled AUROC are printed to three decimals because several lower bounds lie within 0.005 of chance: PCMCI+ (0.505), SINDy (0.503) and C-NODE (0.510) clear 0.5 without noise, and Ensemble SINDy (0.501) and C-NODE (0.503) at $\sigma = 0.05$, while Ensemble SINDy's lower bound without noise is 0.500 and does not.

Method	σ	F1	Specificity	AUROC, per system	AUROC, pooled
PCMCI+	0	0.393 [0.332, 0.455]	0.82 [0.77, 0.86]	0.51 [0.45, 0.56]	0.548 [0.505, 0.587]
	0.01	0.126 [0.079, 0.174]	0.98 [0.96, 0.99]	0.55 [0.50, 0.59]	0.519 [0.474, 0.562]
	0.05	0.096 [0.056, 0.138]	0.98 [0.96, 0.99]	0.51 [0.47, 0.55]	0.511 [0.470, 0.554]
SINDy	0	0.673 [0.639, 0.706]	0.00 [0.00, 0.01]	0.54 [0.50, 0.59]	0.537 [0.503, 0.568]
	0.01	0.671 [0.637, 0.705]	0.00 [0.00, 0.01]	0.55 [0.50, 0.60]	0.531 [0.491, 0.568]
	0.05	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.52 [0.47, 0.59]	0.497 [0.457, 0.540]
SINDy (std.+ridge)	0	0.684 [0.650, 0.719]	0.15 [0.10, 0.19]	0.73 [0.68, 0.79]	0.695 [0.647, 0.746]
	0.01	0.676 [0.644, 0.710]	0.08 [0.05, 0.11]	0.72 [0.66, 0.78]	0.692 [0.640, 0.743]
	0.05	0.671 [0.638, 0.704]	0.01 [0.00, 0.02]	0.63 [0.58, 0.67]	0.611 [0.572, 0.651]
Weak SINDy	0	0.694 [0.654, 0.737]	0.04 [0.02, 0.07]	0.78 [0.72, 0.82]	0.723 [0.675, 0.765]
	0.01	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.50 [0.45, 0.54]	0.502 [0.458, 0.546]
	0.05	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.53 [0.48, 0.57]	0.492 [0.452, 0.530]
Ensemble SINDy	0	0.673 [0.639, 0.706]	0.00 [0.00, 0.01]	0.51 [0.50, 0.52]	0.502 [0.500, 0.505]
	0.01	0.670 [0.637, 0.704]	0.00 [0.00, 0.01]	0.52 [0.49, 0.54]	0.504 [0.500, 0.510]
	0.05	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.52 [0.50, 0.55]	0.504 [0.501, 0.510]
ODEFormer	0	0.592 [0.536, 0.649]	0.76 [0.71, 0.80]	0.66 [0.61, 0.71]	0.642 [0.597, 0.685]
	0.01	0.615 [0.564, 0.666]	0.77 [0.72, 0.81]	0.68 [0.63, 0.72]	0.646 [0.605, 0.685]
	0.05	0.607 [0.557, 0.656]	0.76 [0.71, 0.80]	0.67 [0.63, 0.71]	0.653 [0.606, 0.697]
PySR	0	0.740 [0.699, 0.779]	0.56 [0.51, 0.61]	0.80 [0.75, 0.84]	0.774 [0.721, 0.825]
	0.01	0.712 [0.661, 0.760]	0.61 [0.56, 0.67]	0.73 [0.67, 0.79]	0.728 [0.669, 0.786]
	0.05	0.662 [0.614, 0.706]	0.58 [0.54, 0.63]	0.64 [0.58, 0.70]	0.668 [0.613, 0.725]
Koopman/EDMD	0	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.52 [0.47, 0.56]	0.523 [0.483, 0.561]
	0.01	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.51 [0.47, 0.56]	0.486 [0.453, 0.521]
	0.05	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.56 [0.50, 0.61]	0.502 [0.457, 0.549]
Bayesian Koopman	0	0.670 [0.637, 0.703]	0.00 [0.00, 0.00]	0.45 [0.40, 0.50]	0.486 [0.454, 0.518]
	0.01	0.565 [0.513, 0.618]	0.34 [0.25, 0.42]	0.52 [0.47, 0.57]	0.548 [0.516, 0.579]
	0.05	– [–, –]	– [–, –]	– [–, –]	– [–, –]
Horseshoe Koopman	0	0.861 [0.832, 0.892]	0.88 [0.83, 0.92]	0.86 [0.81, 0.91]	0.901 [0.868, 0.929]
	0.01	0.690 [0.640, 0.741]	0.43 [0.36, 0.51]	0.61 [0.55, 0.66]	0.602 [0.554, 0.650]
	0.05	0.657 [0.621, 0.696]	0.19 [0.13, 0.26]	0.51 [0.46, 0.55]	0.526 [0.485, 0.563]
Neural ODE	0	0.653 [0.614, 0.695]	0.18 [0.14, 0.23]	0.61 [0.56, 0.66]	0.615 [0.571, 0.659]
	0.01	0.653 [0.614, 0.695]	0.18 [0.14, 0.23]	0.61 [0.57, 0.66]	0.614 [0.570, 0.660]
	0.05	0.666 [0.628, 0.709]	0.18 [0.14, 0.24]	0.64 [0.59, 0.68]	0.633 [0.590, 0.674]
C-NODE	0	0.577 [0.530, 0.626]	0.37 [0.31, 0.43]	0.56 [0.50, 0.61]	0.555 [0.510, 0.601]
	0.01	0.575 [0.527, 0.623]	0.37 [0.31, 0.43]	0.55 [0.50, 0.61]	0.546 [0.497, 0.595]
	0.05	0.581 [0.535, 0.627]	0.36 [0.30, 0.41]	0.56 [0.50, 0.62]	0.546 [0.503, 0.590]
Granger	0	0.659 [0.621, 0.698]	0.21 [0.15, 0.28]	0.65 [0.60, 0.69]	0.640 [0.599, 0.680]
	0.01	0.643 [0.594, 0.691]	0.61 [0.56, 0.65]	0.69 [0.64, 0.74]	0.655 [0.611, 0.698]
	0.05	0.488 [0.430, 0.550]	0.72 [0.67, 0.77]	0.57 [0.52, 0.63]	0.580 [0.524, 0.636]

Table 28: The random suite: id, dimension, whether the equation lies in the shared library, number of absent edges, and largest Lyapunov exponent; three systems per line.

Id	d	Lib.	Abs.	λ_1	Id	d	Lib.	Abs.	λ_1	Id	d	Lib.	Abs.	λ_1
201	3	yes	2	-0.28	202	3	yes	3	-0.17	203	3	yes	1	-0.33
204	3	no	2	-0.12	205	3	yes	1	+0.00	206	3	yes	2	-0.81
207	3	yes	2	-0.15	208	3	yes	3	-0.11	209	3	no	2	-0.10
210	3	yes	2	-0.65	211	3	no	2	-0.17	212	3	yes	2	+0.01
213	3	yes	1	-0.14	214	3	yes	3	+0.00	215	3	yes	2	-0.26
216	3	no	2	-0.00	217	3	yes	3	-0.29	218	3	yes	1	-0.10
219	3	yes	1	-0.24	220	3	no	1	-0.20	221	4	yes	4	-0.23
222	4	yes	5	-0.37	223	4	yes	6	+0.13	224	4	no	6	-0.18
225	4	yes	5	-0.30	226	4	no	7	-0.00	227	4	yes	6	-0.33
228	4	yes	7	-0.23	229	4	yes	5	-0.19	230	4	no	6	-0.17
231	4	no	5	-0.23	232	4	yes	7	+0.00	233	4	no	7	-0.01
234	4	yes	5	-0.05	235	4	yes	6	-0.61	236	4	yes	5	-0.40
237	4	yes	6	-0.28	238	4	yes	6	-0.06	239	4	no	6	-0.00
240	4	yes	6	-0.22	241	5	yes	11	+0.00	242	5	no	10	-0.05
243	5	no	13	-0.06	244	5	no	13	-0.67	245	5	yes	14	-0.00
246	5	yes	13	-0.19	247	5	yes	14	+0.00	248	5	yes	13	-0.06
249	5	yes	13	-0.32	250	5	no	14	-0.18	251	5	no	14	-0.49
252	5	yes	12	-0.29	253	5	yes	14	-0.15	254	5	yes	13	-1.21
255	5	yes	12	-0.15	256	5	no	14	-0.12	257	5	yes	13	-0.26
258	5	no	11	-1.47	259	5	yes	13	-0.78	260	5	no	13	-0.11

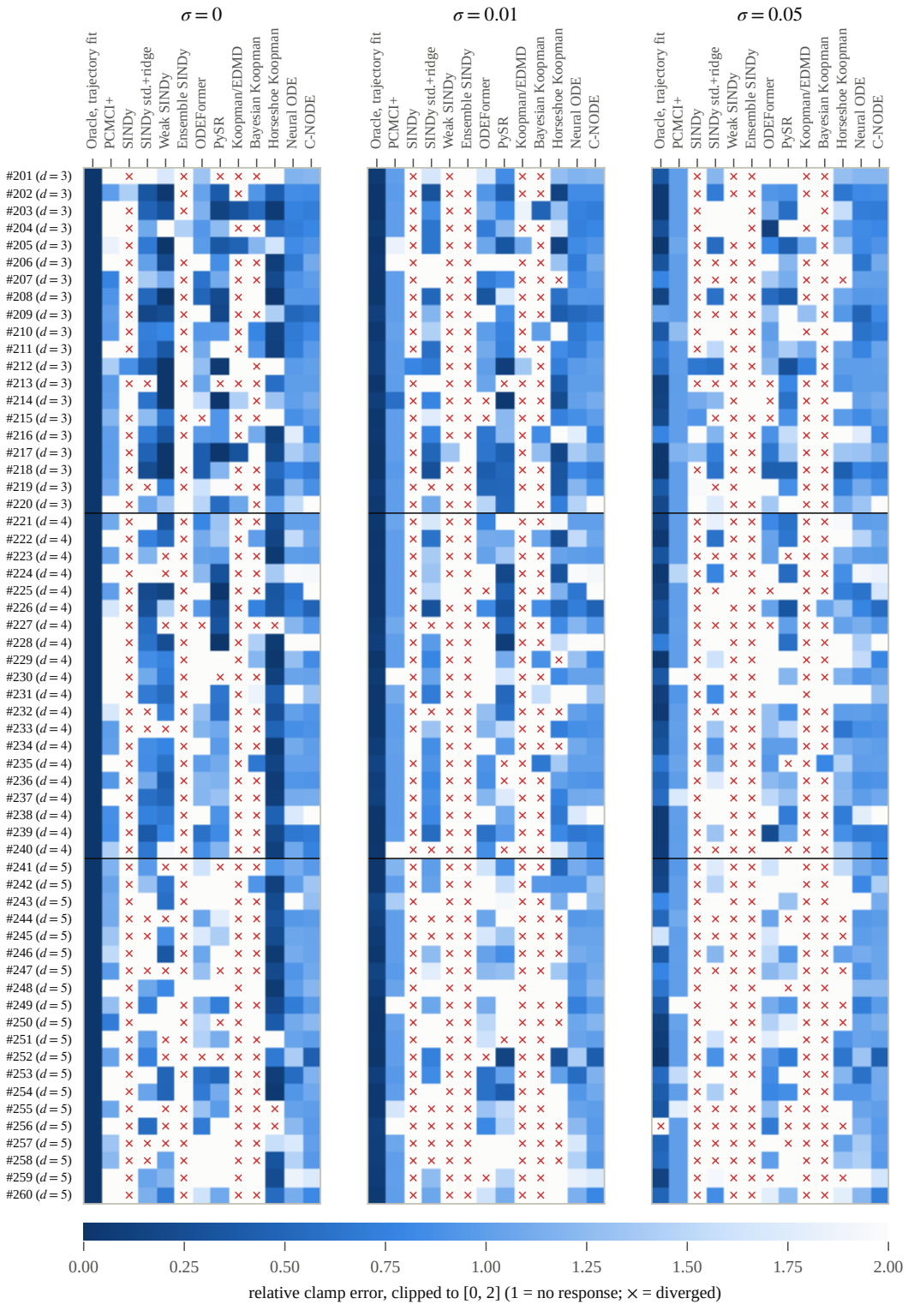


Figure 13: Per-system clamp error on the random suite (rows sorted by dimension).