In-context learning to predict critical transitions in dynamical systems

Yunus Sevinchan[*]
kausable
Heidelberg, Germany
yunus@kausable.ai

Juan Nathaniel[*]
Columbia University
New York, USA
jn2808@columbia.edu

Kai Ueltzhöffer[*]
kausable
Heidelberg, Germany
kai@kausable.ai

Carla Roesch[*]
University of Edinburgh
Scotland, UK
carla.roesch@ed.ac.uk

Vaios Laschos[*]
kausable
Heidelberg, Germany
vaios@kausable.ai

Tobias Weber
kausable
Heidelberg, Germany
tobias@kausable.ai

Hang Fan
Columbia University
New York, USA
hf2526@columbia.edu

Gregor Ramien
kausable
Heidelberg, Germany
gregor@kausable.ai

Johannes Haux
kausable
Heidelberg, Germany
johannes@kausable.ai

Pierre Gentine
Columbia University
New York, USA
pg2328@columbia.edu

Benjamin Herdeanu[*]
kausable
Heidelberg, Germany
benjamin@kausable.ai

Abstract

Critical transitions – abrupt, often irreversible changes in system dynamics – arise across human and natural systems, often with catastrophic consequences. Real-world observations of such shifts remain scarce, preventing the development of reliable early warning systems. Conventional statistical and spectral indicators, such as increasing variance, tend to fail under realistic conditions of limited data and correlated noise, whereas existing deep learning classifiers do not extrapolate beyond their training data distribution. In this work, we introduce TipPFN, an in-context learning (ICL) framework that uses a prior-data fitted network to infer a system's proximity to a critical transition. Trained on our novel synthetic data generator, which is based on canonical bifurcation scenarios coupled to diverse, randomized stochastic dynamics, TipPFN flexibly capitalizes on contexts of various sizes, complexity and dimensionalities. We demonstrate robust, state-of-the-art early detection of critical transitions in previously unseen tipping regimes, sim-to-real examples, and real-world observations in both ICL and zero-shot settings.

[*]Equal contribution

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Tipping points occur when variation in forcing or system parameters triggers an abrupt, potentially irreversible transition to a qualitatively different dynamical regime [1, 2]. Such transitions are central to climate [3], ecological [4], infrastructural [5], and collective biological and social systems [6, 7, 8], where local shifts can cascade through interacting components with far-reaching consequences. Classical theory has largely focused on bifurcation (b-tipping) classes of tipping mechanism. Here, the loss of stability of an attracting state produces critical slowing down (CSD) [9], where recovery from perturbations becomes progressively weaker [10, 11]. This has motivated widely used early warning signals (EWS), including increasing lag-1 autocorrelation (AR1) [12], variance [13], and skewness [14]. However, these indicators rely on restrictive assumptions, including near-equilibrium linearization, stationarity, sufficiently long records, and simple bifurcation structure [15, 16]. They may therefore fail for rate-induced (r-tipping) and noise-induced (n-tipping), where transitions can occur without pronounced CSD [7, 17, 18].

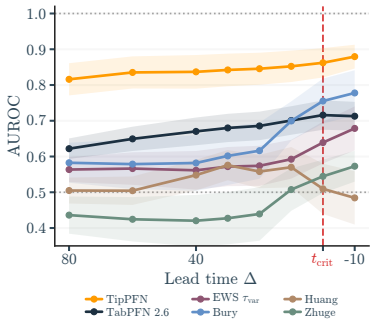


Figure 1: **TipPFN detects critical transitions** earlier and more accurately than classical early warning signals (EWS), state-of-the-art deep learning and ICL-based methods across the 14 systems of Table 1 except DaphniaExt, spanning climate, engineering, and biology. Lines show across-system mean and standard error.

Characterizing critical transitions requires forecasting an unseen future regime from observations in a qualitatively distinct initial regime, so that the data-generating dynamics shift between what is observed and what must be predicted [19, 20, 21]. This makes tipping prediction fundamentally different from standard regression tasks, since models must infer the onset and type of critical transitions from sparse, noisy data and partially observed trajectories, often without examples of the event itself. Throughout, we use *generalization* to mean empirical transfer to held-out tipping mechanisms and domains. Recent ML approaches have shown promise by learning features of canonical tipping systems directly from data [22, 23, 7, 24, 25]. However, many of these methods are trained on particular classes, limiting their ability to generalize across observations and different critical mechanisms.

A promising alternative has recently emerged in the form of prior-data fitted networks (PFNs), which are trained on large ensembles of synthetic datasets and learn to perform inference via in-context learning (ICL) [26]. Rather than fitting a model to a single dataset, PFNs are trained across millions of simulated tasks, enabling them to infer structure and make predictions directly from limited observations. Thus, PFNs have already transformed tabular ML through models such as TabPFN [27, 28, 29], which, despite being trained exclusively on synthetic data, achieve state-of-the-art performance on real-world benchmarks. More broadly, PFNs have demonstrated remarkable generalisation across diverse tasks, including time series forecasting [30], causal inference [31, 32], and robotic controls [33].

Our **key contributions** are (see Figure 2):

- **Model:** TipPFN, an in-context learning model that infers proximity to a critical transition (RDTC) from short, noisy, multivariate time series.
- **Training data and benchmark:** a training dataset and evaluation benchmark of stochastic dynamical systems across diverse tipping regimes and complexities, simulated with TipBox[2], released together with the trained model and code[3].
- **Generalization:** Robust performance across unseen tipping regimes, sim-to-real cases, and real-world systems in both ICL and zero-shot settings (see Fig. 1).

[2] `github.com/juannat7/tipbox`
[3] Code: `github.com/kausable/tippfn-neurips26`; model and data: `huggingface.co/kausable`.

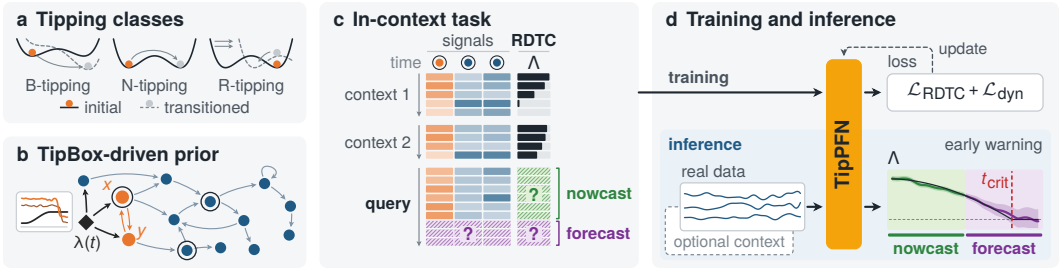


Figure 2: **Overview. (a)** Tipping classes (Sec. 2). **(b)** The TipBox-driven prior embeds canonical b-tipping drivers (x, y), forced by $\lambda(t)$, into randomized stochastic auxiliary dynamics, of which a random subset is observed. **(c)** A training task: context episodes with known RDTC Λ, and a query whose RDTC is masked and predicted for the observed rows (nowcast) and beyond them (forecast). **(d)** TipPFN is trained on such tasks and applied to real data, optionally with context; it transfers to other tipping classes and to several real-world systems.

2 Background

Critical transition in dynamical systems Critical transition, specifically tipping points in dynamical systems, can arise through multiple distinct mechanisms, broadly classified as bifurcation-induced (b-tipping), noise-induced (n-tipping), and rate-induced tipping (r-tipping) [2] (Fig. 2a). Consider a general nonautonomous stochastic differential equation (SDE) of the form:

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}, \lambda(t)) + \sigma\, \xi(t), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$ denotes the system state, $\lambda(t)$ is a time-dependent forcing parameter, $\sigma > 0$ controls the noise amplitude, and $\xi(t)$ represents Gaussian white noise.

Bifurcation-induced tipping In b-tipping, a transition occurs when a quasi-static equilibrium $\mathbf{x}^*(\lambda)$ loses stability at a critical parameter value λ_{crit}. Linearizing the dynamics around $\mathbf{x}^*(\lambda)$ yields the Jacobian $J(\lambda) = \partial f/\partial \mathbf{x}\big|_{\mathbf{x}^*(\lambda)}$, whose eigenvalues $\{\mu_i(\lambda)\}$ govern local stability. A bifurcation occurs when the real part of the leading eigenvalue crosses zero, i.e.,

$$\max_i \operatorname{Re}(\mu_i(\lambda_{\text{crit}})) = 0. \tag{2}$$

Rate-induced tipping In r-tipping, a transition occurs when the rate of change of the forcing, given by $\dot{\lambda}$, is sufficiently large such that the system fails to track the moving equilibrium $\mathbf{x}^*(\lambda(t))$, leading to a transition without any local loss of stability, i.e., the linearized system satisfy $\max_i \operatorname{Re}(\mu_i(\lambda)) < 0$. Thus, these tipping mechanisms cannot, in general, be inferred solely from equilibrium stability analysis commonly used to check b-tipping (see Appendix A for more details).

Noise-induced tipping In n-tipping, a transition occurs primarily through stochastic perturbations $\sigma\, \xi(t)$ that induce escape from the basin of attraction of a stable equilibrium. This process can occur in conjunction with b-tipping and r-tipping in SDEs, a setup which form the basis for this work.

Prior-data fitted networks PFNs are transformer-based neural networks trained on synthetically generated data drawn from a predefined prior, enabling approximate Bayesian inference via ICL [26]. In this framework, inference is amortized at the dataset level. The model learns a mapping from a dataset D to a posterior predictive distribution $p(y|x, D)$ by training on large collections of simulated tasks (D_i, x_i, y_i). Rather than fitting a model to each new dataset, PFNs directly infer predictions conditioned on the observed data, performing Bayesian inference in a single forward pass. The Bayesian interpretation of PFN predictions only holds on the support of the training prior.

Related work In addition to the classical EWS based on statistics (AR1, variance, skewness), there is an emergence of dynamics-based approaches extracting additional spectral structure of the systems, including tracking of dominant eigenvalues [34, 35, 36]. Recently, fully data-driven

ML approaches, including reservoir computing [25, 23], convolutional, and recurrent-based neural networks [22, 24], have been proposed to detect tipping points directly from time series data. These models can outperform both classical and spectral EWS when trained on simulated data, but are tailored to specific tipping regimes (e.g., b-tipping [22, 24] or r-tipping [18]). Closest to our approach, [22] also pre-trains on synthetic dynamical systems, but uses univariate inputs, a single categorical label per sequence (bifurcation type or no transition) as supervision, and no in-context learning. They also show limited generalization capabilities, which we further showcase throughout.

3 TipPFN: Predicting tipping behaviors with PFNs

TipPFN is designed to infer approaching critical transitions directly from observed trajectories. To this end, we train a model on synthetic dynamical systems in which a control parameter λ is gradually varied to produce trajectories that progress from stable regimes to towards a critical transition. By embedding controlled, stereotypical tipping systems within larger, randomized systems of nonlinear SDEs, the synthetic data provide a scalable source of diverse transition patterns and allow the model to learn transferable structure rather than system-specific rules. This is essential for our setting, where the goal is generalization to previously unseen systems.

Next, we define the prediction target, the relative distance to criticality. We then describe synthetic data generation, and finally the TipPFN training and inference setup; also see Fig. 3 and Appendix B.

Relative distance to criticality (RDTC) Direct supervision on critical events is often ill-posed, since the observed transition time can depend on noise, finite-time effects, and threshold choices. For the bifurcation-induced tipping systems used in training, however, proximity to criticality is well defined through the forcing parameter λ.

We therefore introduce the *relative distance to criticality* (RDTC) Λ as a continuous supervision target. Let $\tilde{\lambda} \in [0, 1]$ denote the normalized control parameter, where $\tilde{\lambda} = 0$ corresponds to a stable regime and $\tilde{\lambda}_{\text{crit}} = 1$ to the forcing value at which the underlying deterministic system undergoes a bifurcation. We define

$$\Lambda = 1 - \tilde{\lambda}. \tag{3}$$

For $\Lambda = 1$, the system is maximally far from the bifurcation within the normalized forcing range; for $0 < \Lambda < 1$, it remains in the sub-critical regime, reaching criticality at $\Lambda_{\text{crit}} = 0$. RDTC serves as the primary training target for synthetic bifurcation systems. Furthermore, it is intended to act as a transferable measure of proximity to criticality across otherwise distinct systems.

Prior structure TipPFN's prior $p(\psi)$ over generative processes ψ factorises over a canonical tipping system M, a random potentially cyclic interaction graph $\mathcal{G}$ and an auxiliary nonlinear SDE

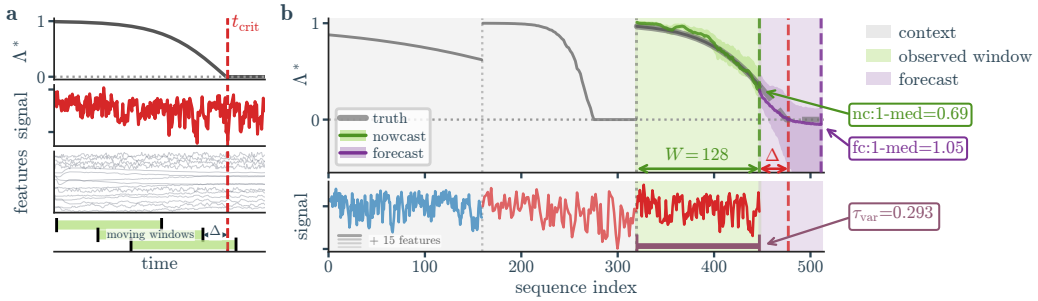


Figure 3: **Tipping Prediction with TipPFN. (a)** Example multi-variate time series from a critical episode and underlying RDTC Λ. Green bars mark candidate observation windows ending Δ time steps before the critical time t_{crit}. **(b)** Example query episode at $\Delta = 30$ and context composed from a critical (red) and non-critical (blue) episode. TipPFN and TabPFN are conditioned on the shaded context and observed window, including the signal and all available features, and then predict relative distance to criticality, Λ^*, for the nowcast and forecast region. These predictions form the basis of the tipping-risk scores used for early-warning assessment. Baseline scores (e.g. EWS τ_{var}) are computed only from the observed signal window.

system $\mathcal{Z}$. $\mathcal{G}$ specifies the interaction structure of $\mathcal{Z}$, and how it is driven by the state variables $z_M(t) = (x_M(t), y_M(t))$, the forcing parameter $\lambda(t)$ and the RDTC $\Lambda(t)$. This driving timeseries is generated by sampling from a flexible family of low-dimensional nonlinear dynamical systems designed to exhibit canonical tipping behavior [22] (see Appendix B.1 for details). We define the prior over the generative process $\psi = (M, \mathcal{G}, \mathcal{Z})$ as:

$$p(\psi) = p(M)\, p(\mathcal{G})\, p(\mathcal{Z} \mid M, \mathcal{G}), \tag{4}$$

where $M \sim \mathrm{Uniform}(\{\text{fold, Hopf, transcritical}\})$ determines the bifurcation class inducing qualitatively different signatures in the pre-tipping dynamics. $\mathcal{G}$ is generated by drawing from a distribution over directed, potentially cyclic graphs with power-law distributions over incoming and outgoing node degrees [37] to yield the interaction structure within the auxiliary dynamic nodes, followed by random attachment of four driver nodes representing M's state variables, its forcing parameter and the RDTC. Based on this dynamical causal structure, nonlinear SDE interaction terms are drawn from a distribution over multi-layer perceptrons. Additional linear stabilization terms and a global timescale parameter are added to the auxiliary nodes and also randomized. The hyperparameter distribution of the auxiliary system is chosen to keep it in a stable regime, preventing additional, spontaneous bifurcations due to intrinsic auxiliary system dynamics, which would confound the signal provided by the driving system.

Prior-data generation As illustrated in Fig. S1a, to fit the prior, we specify a sampling scheme over context-query pairs $\mathcal{D}$ of the form

$$p(\mathcal{D}) = \mathbb{E}_{\psi \sim p(\psi), \mathcal{S} \sim p(\mathcal{S}|\psi)} \big[p(\mathcal{D} \mid \mathcal{S}) \big], \tag{5}$$

which first samples a generative process $\psi = (M, \mathcal{G}, \mathcal{Z}) \sim p(\psi)$ and then a synthetic episode ensemble $\mathcal{S} = \{\mathcal{E}_k\}_{k=1}^K \sim p(\mathcal{S} \mid \psi)$ which fixes the canonical tipping system M, the interaction graph $\mathcal{G}$ and SDE system $\mathcal{Z}$. Each episode $\mathcal{E}_k = (z^k_{1:T}, u^k_{1:T}, \tilde{\lambda}^k_{1:T})$ is based on an independent noise realization and contains the trajectories of the driving system $z^k_{1:T}$ and the auxiliary nodes $u^k_{1:T}$ created by the independent forcing schedule $\tilde{\lambda}^k_{1:T}$ (Bezier, ramp, or constant; Table S2). Forcing schedules are constrained to produce trajectories spanning multiple dynamical regimes: *critical* (tipping), *approaching* towards or *receding* from criticality, *constant*, and *equilibrium* (non-critical); see Appendix B.7. From each sample $\mathcal{S} = \{\mathcal{E}_k\}_{k=1}^K$, one episode $\mathcal{E}^{\text{query}}$ is held out as the query and a context set $\mathcal{C} \subset \mathcal{E} \setminus \mathcal{E}^{\text{query}}$ of $|\mathcal{C}| \in \{0, .., 3\}$ context trajectories, with $|\mathcal{C}|$ drawn with probabilities 0.2, 0.3, 0.3 and 0.2, is sampled uniformly from the remaining $K - 1$ episodes. A subset of feature dimensions of the driving system z and the auxiliary system u is randomly selected to yield the observed features $\mathbf{o}$ with $\dim(\mathbf{o}) \in \{1 : 16\}$, rendering some the resulting prediction problems only partially observed. By resampling the time series with different resolutions, we realize variable-context training tasks in which query and context episodes share the same underlying system class and coupling structure, but differ in forcing and stochastic realization. This yields a query-context pair $\mathcal{D} = ((\Lambda_{1:T}, \mathbf{o}_{1:T}) \cup \mathcal{C})$. As a result, the model learns to use context when informative without depending on its presence. In total, we trained on ∼3.7M context-query pairs $\mathcal{D}$ (22k optimizer steps at batch size 168) from ∼270k ψ samples with $K = 4$ episodes and $M \in \mathbb{R}^2,\ \mathcal{Z} \in \mathbb{R}^{14}$.

Input representation Each training task is encoded as a tabular multi-episode sequence with episode and time identifiers, observed variables, target variables, and prediction masks. To avoid target leakage, variables are normalized using only observed values. Observed and target variables are centered and transformed with asinh scaling, which reduces the effect of large amplitude differences across heterogeneous signals. RDTC is transformed separately as $\Lambda^* = \tanh(5\Lambda)$, the nonlinearity providing finer resolution near criticality. In the query episode, RDTC is always fully masked, whereas the remaining target variables are only partially masked.

Architecture TipPFN is a transformer-based architecture with a similar structure to TabPFN [27, 28]; the main adaptations for tipping prediction lie in the task construction, masking scheme, and training objective described above. Transformer attention is structured, such that all tokens from the context set $\mathcal{C}$ can attend to all other tokens of $\mathcal{C}$. Tokens from the query trajectory can attend to all tokens from $\mathcal{C}$, themselves, and all tokens from the query trajectory with smaller time features, i.e., tokens representing preceding time steps (*causal attention*), as illustrated in Fig. S1b. This ensures that no future information leaks into the estimate at the end of the observed window, as required for early warning signals. More details on the architecture of PFNs is provided in Appendix B. The released model has ∼210M parameters; much smaller variants perform almost as well (Sec. 4.4).

Training TipPFN is trained as a TabPFN-style transformer on tabularized multi-episode trajectories. The model predicts quantile distributions for masked targets and is optimized with a weighted pinball loss. Along the temporal dimension, attention is causal, so each prediction can depend only on past and present observations. Context episodes are fully visible.

The parameters θ of a transformer model q_θ are optimised using the following training loss:

$$\mathcal{L} = \mathbb{E}_{\mathcal{D}=((\Lambda_{1:T},\mathbf{o}_{1:T})\cup\mathcal{C})\sim p(\mathcal{D})}\left[\sum_{t=1}^{T}\frac{1}{99}\sum_{j=1}^{99}\ell_{\alpha_j}\left(\hat{q}_{\theta,j}\left(\mathbf{o}_{1:\min(t,t_{\mathrm{nc}})},\,\mathcal{C}\right),\,\Lambda_t^*\right)\right]. \tag{6}$$

Here, $\hat{q}_{\theta,j}$ is the predicted quantile at level $\alpha_j \in \{0.01,\ldots,0.99\}$, ℓ_α the pinball loss (Appendix B.6), Λ_t^* the transformed RDTC at time step t in the query episode, and $\mathbf{o}_{1:\min(t,t_{\mathrm{nc}})}$ denotes query observations up to step t or a cut-off $t_{\mathrm{nc}} < t$, up to which observational data is available, to train nowcasting and forecasting capabilities. The RDTC prediction loss is augmented with a similar loss, scaled by a factor of 0.2 relative to the RDTC loss, which requires to predict a randomly selected subset of the observational features from the remaining set of at least 1 feature. This loss was added to encourage the model to learn a more general, task-agnostic representation of the nonlinear stochastic dynamics of the synthetic training data.

Posterior Predictive Distribution for RDTC By minimising the training loss given in Equation 6, the quantiles predicted by the transformer model $q_{\theta^{\mathrm{opt}}}$ approximate those of the true Bayesian posterior predictive distribution (PPD) [26]:

$$\begin{aligned} q_{\theta^{\mathrm{opt}}}\left(\Lambda_t \mid \mathbf{o}_{1:\min(t,t_{\mathrm{nc}})},\,\mathcal{C}\right) &\approx p(\Lambda_t \mid \mathbf{o}_{1:\min(t,t_{\mathrm{nc}})},\,\mathcal{C}) \\ &= \int_\Psi p(\Lambda_t \mid \mathbf{o}_{1:\min(t,t_{\mathrm{nc}})},\,\mathcal{C},\psi)\,p(\psi \mid \mathbf{o}_{1:\min(t,t_{\mathrm{nc}})},\,\mathcal{C})\;d\psi. \end{aligned}$$

Thus, TipPFN approximates this intractable integral in a single transformer forward pass by amortising inference over a large ensemble of synthetic prediction problems sampled from $p(\psi)$ during training, yielding a predictive distribution approximating Bayes-optimal uncertainty estimates. When no context episodes are available, the prediction relies entirely on the query trajectory and the prior $p(\psi)$. Increasing the available context sharpens the posterior $p(\psi \mid \mathbf{o}_{1:t},\,\mathcal{C})$ over generative processes compatible with the observed data, depending on the information content of the added context samples.

Inference TipPFN supports two inference modes depending on the operational setting: For *nowcasting*, it estimates the current RDTC at time t, conditioned on all observations up to that point:

$$q_{\theta^{\mathrm{opt}}}\left(\Lambda_t \mid \mathbf{o}_{1:t},\,\mathcal{C}\right), \tag{7}$$

where Λ_t denotes the RDTC at the current time step t. This mode is appropriate when the primary goal is to monitor the system's instantaneous distance to a critical point in real time, issuing an updated estimate as each new observation arrives.

In the *forecasting* setting, TipPFN predicts the future trajectory of RDTC, conditioned on observations up to the current time step:

$$q_{\theta^{\mathrm{opt}}}\left(\Lambda_{t:T} \mid \mathbf{o}_{1:t},\,\mathcal{C}\right), \tag{8}$$

where $\Lambda_{t:T}$ denotes the sequence of RDTC values from the current time step t to a future horizon T. This mode is appropriate when the goal is to anticipate if the system will approach a tipping point, enabling proactive intervention before the critical transition occurs.

4 Experiments

In this section, we evaluate the performance and predictive skill of TipPFN for tipping behavior across different mechanisms. We benchmark against a comprehensive set of competitive baselines on datasets ranging from synthetic systems to real-world observations. Evaluation is based on classification performance (critical vs. non critical) using receiver operating characteristic (ROC) curves [38, 39] and Area Under the ROC curve (AUROC) scores [40, 41], as well as lead-time analysis; more details in Appendix C.

Validation datasets We generated and collected cross-domain validation datasets covering 12 model families (canonical and semi-real) and 9 observational systems (sim-to-real & real-world); see Appendix C.1 for the full list and descriptions.

Canonical datasets are held-out realizations from the synthetic prior. *Semi-real* datasets are reduced-order models of real-world dynamics that are more complex than the canonical systems and include unseen tipping behavior, e.g., r-ipping. *Sim-to-real* datasets contain both simulated and observed trajectories, allowing us to test whether matched simulations can provide context for empirical transition prediction. *Real-world observation* datasets consist of empirical time series from systems without fully specified generative dynamics, including neurological seizure recordings, ocean circulation (AMOC), power-grid blackout, and cyanobacteria population dynamics; depending on the available data, they support either context-based transfer or strict zero-shot RDTC prediction.

Baselines We compare TipPFN against classical EWS, including AR1, variance, and skewness in rolling windows, summarize their temporal trends with Kendall-τ [42], and obtain ROC curves by thresholding these trend scores, following standard practice [9]. ML baselines include Bury [22], Huang [18], and Zhuge [24], operating on univariate time series. TabPFN2.6 [28] provides a state-of-the-art ICL baseline; more details are provided in Appendix C.2.

Querying procedure & scoring All methods are evaluated on matched moving query windows ending Δ time steps before the critical event. Uni-variate baselines were evaluated on the driving time series, $z_M(t)$. PFN-based models additionally receive context episodes, whereas classical and ML baselines use only the query window; see Fig. 3. The original window's time is re-indexed to start at 0 such that no positional information is leaked. At test time, no method has direct access to future observations, hidden system parameters, or ground-truth RDTC, except for Zhuge, which also observes the forcing parameter and therefore receives privileged query-episode information. On the synthetic systems, the auxiliary variables are in addition driven by the forcing and the RDTC (Sec. 3), so methods that observe them, TipPFN and TabPFN, also receive indirect information about the forcing level that the univariate baselines do not. Appendix C.4.1 reports results with auxiliary variables driven by the state variables only.

As the ML baselines require a window size $W \geq 500$ steps while TipPFN uses $W \leq 128$, we construct length-adapted baseline inputs by resampling, back-filling, or forward-filling the query window. For each model, we pick the best-performing variant when reporting results (Appendix C.5.2).

TipPFN and TabPFN receive the same context episodes and query window. For each query trajectory, they predict the (transformed) RDTC Λ^* over the observed window and future time points; see Fig. 3. TabPFN is cast as tabular regression by treating trajectory time points as rows and context episodes as in-context training examples. Predicted RDTC distributions are then converted into tipping-risk scores, e.g. $1-\mathrm{median}(\Lambda)$ or $\mathrm{CDF}(\Lambda < 0.05)$ at the final nowcast or forecast time point, which rank queries by predicted risk. For TabPFN and TipPFN, we report the final forecast-time $1-\mathrm{median}(\Lambda)$ score for all methods. As illustrated in Fig. 3b, this score is the parameter-free estimate of forecast remaining RDTC at the forecast horizon, and therefore aligns directly with the early-warning task. Note that the forecast is always $192 - W$ steps into the future, so shorter windows make a longer-horizon forecast, thus also increasing uncertainty. We report score sensitivity analyses in Table S26 and Appendix C.5.2.

4.1 Generalization across synthetic transition systems

We start with the canonical bifurcation setting, evaluating on held-out realizations from the fold, Hopf, and transcritical families used to define the training prior. TipPFN already performs strongly without additional context trajectories, with robust AUROC across all three systems and clear gains over classical early-warning baselines (Table 1, Fig. S14).

Importantly, the binary task is stricter than a simple critical-versus-noncritical split: negative examples include not only equilibrium trajectories, but also trajectories that remain subcritical while approaching or receding from criticality. Strong performance therefore indicates that TipPFN learns reusable dynamical signatures rather than memorizing individual trajectories.

Beyond canonical bifurcations We next test generalization. On semi-real reduced-order systems, TipPFN typically matches or exceeds the evaluated baselines, showing transfer from simple synthetic

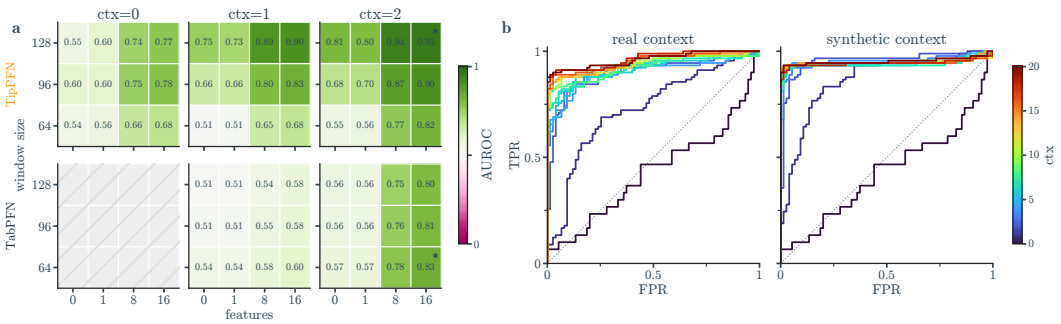


Figure 4: **Context matters. (a)** Multi-parameter sweep over number of context episodes, observed feature channels, and window size W for TipPFN and TabPFN. Color denotes the AUROC of the RDTC nowcast on the semi-real b-tipping systems of Table 1, pooled over systems and averaged over all eight lead times $\Delta \in [-10, 80]$; stars mark the best configurations. Note that TabPFN cannot be used in the zero-context setting. **(b)** TipPFN ROC on `DaphniaExt` for $\Delta = 2$, $W = 16$ and varying number of real or simulated context episodes.

bifurcations to more realistic unseen dynamics (Table 1; Fig. S14). A stronger test is r-tipping, which lies outside the training mechanisms altogether. Despite this mismatch, TipPFN remains competitive on Bautin, compost-bomb, saddle-node, and AMOC r-tipping systems.

These results use one fixed configuration per method, selected for strong aggregate performance across all systems. Under a more permissive best-per-system comparison, several baselines improve on individual tasks, but their gains are often system-specific (Table S12). TipPFN remains best-performing on many datasets, and where it is outperformed, it typically is close to the best baseline.

4.2 Leveraging context

A central motivation for TipPFN is that context provides system-specific information at inference time. This is particularly valuable for tipping prediction, where observations are often short, noisy, and only weakly informative on their own. Rather than reducing trajectories to one-dimensional

Table 1: **Average pre-tip AUROC by system and dataset.** Balanced AUROC for detecting critical episodes, averaged over positive lead times ($\Delta > 0$). Each column uses one fixed evaluation setting across all listed rows that yields highest AUROC across all these systems (see Table S25 for details). The last row leaves out the three canonical systems, which share their bifurcation types with the training data.

Dataset	TipPFN 0c	TipPFN 1c	TipPFN 2c	TabPFN 1c	TabPFN 2c	EWS τ_{var}	Bury [22] P_{tip}	Huang [18] P_{tip}	Zhuge [24] $\Delta^{\pm}_{\text{tip}}$
Canonical									
B-Fold	.876	.929	**.955**	.574	.786	.633	.697	.462	.522
B-Hopf	.861	.935	**.952**	.640	.820	.635	.842	.496	.665
B-Trans.	.864	.925	**.950**	.581	.807	.543	.722	.453	.458
Semi-real									
B-Harv.	.846	.958	**.974**	.685	.810	.914	.967	.873	.734
B-RM TC	.815	.951	**.963**	.494	.741	.604	.784	.604	.570
B-RM Hopf	.769	.829	**.904**	.499	.689	.557	.638	.484	.422
B-SEIRx	.582	.822	**.905**	.495	.685	.498	.468	.507	.381
B-AMOC	.853	.898	.940	.616	.825	.925	**.948**	.509	.679
R-Bautin	.308	.543	**.700**	.404	.489	.451	.308	.538	.180
R-SN	.644	.802	**.892**	.335	.569	.440	.354	.273	.080
R-Compost	**.767**	.623	.665	.472	.493	.481	.727	.678	.506
R-AMOC	.257	.782	**.913**	.521	.654	.270	.102	.712	.112
Real-world & Sim-to-Real									
SWEC-iEEG	.444	.512	.506	.489	.506	.524	**.541**	.488	–
TAC	.532	.508	.517	.530	.477	**.565**	.476	.522	–
DaphniaExt	.500	.717	**.832**	.511	.651	.540	.483	.404	–
All datasets	.661	.782	**.838**	.523	.667	.572	.604	.534	.443
All except canonical	.610	.745	**.809**	.504	.633	.564	.566	.549	.407

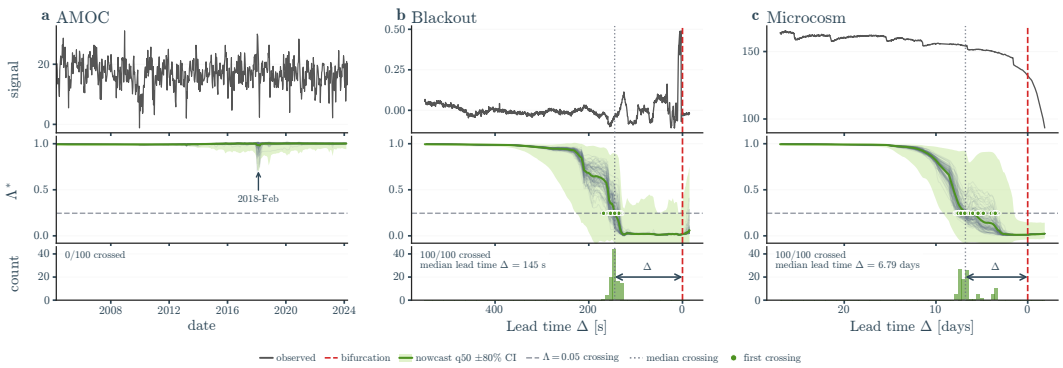


Figure 5: **Zero-shot TipPFN RDTC nowcasts** on three univariate real-world time series, without per-system fine-tuning. Rows: observed signal; TipPFN predictions of Λ^* (means of the predicted median and 10–90% quantiles over 100 ensemble members with randomly retained time points); distribution of the members' first crossings of $\Lambda^*_{\text{thrs}} = \tanh(5\Lambda_{\text{thrs}}) \approx 0.245$. Red dashed lines mark the bifurcation time from dominant-eigenvalue analysis [35], where available.

early-warning statistics, TipPFN conditions on full multivariate context episodes and can therefore infer which feature patterns distinguish tipping from non-tipping behavior.

Figure 4a shows that TipPFN benefits most from combining additional context, higher-dimensional observations, and sufficiently long windows. For both TipPFN and TabPFN, performance improves when context episodes are added even under a fixed input budget, suggesting that the gain comes not only from observing the query longer, but from comparing it against a broader range of system behavior. Increasing the number of observed features yields a further boost, highlighting the value of multivariate inputs.

4.3 Real-world transfer

We next evaluate TipPFN on sim-to-real and real-world datasets, where true RDTC labels are typically unavailable. We therefore construct surrogate RDTC targets, $\Lambda = \Lambda(t)$, from experimentally observed or independently estimated transition markers, and test whether TipPFN can generalize using limited empirical context or, where available, matched simulations.

On `DaphniaExt`, we analyse TipPFN on extremely sparse ($W = 16$) real observations and matched simulations [43]. As shown in Fig. 4b, simulated and observed context reach comparable performance but respond differently to context size. Adding observed episodes keeps improving performance and simulated context provides even better results, compensating for sparse observations. Furthermore, while TabPFN and TipPFN both perform well on predicting the extinction event, only TipPFN discerns the underlying bifurcation [44], see Figs. S9, S20 and Appendix C.4.6.

We further evaluate TipPFN on a noisy experimental system with a stochastic subcritical Hopf transition (`TAC` [45], surrogate RDTC from the known critical point) and on seizure onset in multichannel iEEG bandpower trajectories (`SWEC-iEEG`). In the standard protocol, all methods stay close to chance on both (Table 1). While the Hopf-specific Bury baseline performs strongly on `TAC`, TipPFN improves substantially with real and simulated context, especially when the supplied episodes are themselves critical (see Fig. S21, Appendix C.4.7). On `SWEC-iEEG`, no method separates pre-ictal from interictal segments (Appendix C.4.5).

Zero-shot prediction on real-world systems We also evaluate TipPFN in a strict zero-shot setting, without context trajectories. The relevant output is the predicted RDTC trajectory, a time-resolved nowcast of distance to criticality from a single observed time series. This allows critical-point timing to be estimated even when no matched empirical or simulated context is available.

Across six real-world observational time series (see Fig. 5, Appendix S22), TipPFN's RDTC predictions are informative: In the `AMOC` example, we detect an increase in RDTC variance coinciding with an abnormal reversal in AMOC slow-down [46]. In a `power-grid` time series preceding a blackout, TipPFN signals hidden instability 2 minutes before the failure becomes apparent. In the `microcosm`

series, it estimates a cyanobacteria population collapse approximately a week from the actual event. Because visible regime shifts may lag behind hidden loss of stability, zero-shot RDTC prediction can provide advance warning and create a window for mitigation or response.

4.4 Ablations and sensitivity analysis

To isolate what drives TipPFN's performance, we run ablations on 12 simulated systems, varying the training target, model size, and architecture (Appendix C.4.4).

TipPFN setup Performance gains come from RDTC as target combined with in-context learning, which improves AUROC on most systems (Sec. 4.2). Replacing RDTC with a single tipping label per episode (a classifier with the same architecture) or with a dense binary target reduces AUROC.

Baseline parity Prior and model scale do not account for the performance gap as the CNN-LSTM of [22], retrained on our prior, remains below TipPFN, and a model with 140k rather than 210M parameters still reaches 0.857 AUROC, against 0.893 for the full model (Table S19).

Data and context availability Training is also data-efficient, as a smaller variant reaches 90% of its final AUROC after 8% of one epoch, although this does not disentangle data volume from the number of optimisation steps. TipPFN is sensitive to the source of its context but robust to its noise levels. Context taken from a different draw of the same system lowers AUROC, whereas adding noise of one standard deviation to the context has less of an impact (Tables S20 and S21). TipPFN's advantage is also not a result of the shared 128-step window. Even at each baseline's native input length, TipPFN still leads on 6 of 8 canonical and b-tipping systems (Appendix C.4.3).

5 Conclusion

We introduce TipPFN, a transformer-based prior-data fitted network designed to predict the relative distance to criticality (RDTC) directly from observed trajectories. TipPFN is trained exclusively on a novel synthetic data-generation framework combining canonical tipping systems with randomized high-dimensional nonlinear stochastic dynamics, enabling prediction across diverse transition regimes. We evaluate TipPFN against state-of-the-art machine learning and dynamical systems baselines on more than 20 datasets spanning synthetic, sim-to-real, and real-world settings across multiple domains, noise levels, and tipping mechanisms. TipPFN outperforms existing approaches on most simulated systems while demonstrating strong generalization across previously unseen systems.

Unlike existing tipping-predicting methods, TipPFN can leverage additional context trajectories, auxiliary features, and variable observation windows through ICL, while remaining effective even in the absence of context information. The framework naturally integrates real observations, simulated trajectories, or hybrid combinations thereof, enabling prediction in settings where observational data are sparse or expensive [47, 48].

By inferring hidden proximity to a transition rather than relying on explicit early warning indicators, TipPFN provides a universal, system-agnostic estimate of criticality without requiring system-specific retraining or manual selection of diagnostic metrics. Finally, TipPFN consistently and accurately provides early prediction of tipping characteristics, enabling real-world deployment where early intervention is critical.

Limitations On our synthetic benchmarks, the auxiliary variables respond to both the state variables and the forcing, so part of TipPFN's advantage comes from observed variables that carry information about the forcing level. Such knowledge of the forcing, direct or indirect, is common in practice especially when detecting R-tipping, and future work will examine how it helps in real systems where the forcing is known.

Beyond this, we plan to combine ICL with near-real-time adaptation for online settings, for example through low-rank updates [49] or test-time training [50, 51], so that learned priors evolve as new observations arrive, and to address the spatial dependence of critical transitions in climate [3, 52] and ecological systems [14].

Acknowledgements and disclosures

JN, HF, PG acknowledge funding, computing, and storage resources from the NSF Science and Technology Center (STC) Learning the Earth with Artificial Intelligence and Physics (LEAP) (Award #2019625) and by NASA under award No 80NSSC25K0062. We thank Maximilian Hüttenrauch for his contributions to this work. Data from the RAPID AMOC observing project is funded by the Natural Environment Research Council, U.S. National Science Foundation (NSF) with support from NOAA. They are freely available from https://rapid.ac.uk/. YS, KU, VL, TW, GR, JH and BH are affiliated with kausable, a company developing machine-learning methods for complex dynamical systems. The authors declare no other competing interests.

References

[1] Sheng Fang, Ziyan Wang, Jürgen Kurths, and Jingfang Fan. Tipping points and cascading transitions: Methods, principles, and evidences. *arXiv preprint arXiv:2511.01168*, 2025.

[2] Peter Ashwin, Sebastian Wieczorek, Renato Vitolo, and Peter Cox. Tipping points in open systems: bifurcation, noise-induced and rate-dependent examples in the climate system. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 370(1962):1166–1184, 2012.

[3] Anastasia Romanou, David Rind, Jeff Jonas, Ron Miller, Maxwell Kelley, Gary Russell, Clara Orbe, Larissa Nazarenko, Rebecca Latto, and Gavin A Schmidt. Stochastic bifurcation of the north atlantic circulation under a midrange future climate scenario with the nasa-giss modele. *Journal of Climate*, 36(18):6141–6161, 2023.

[4] Annelies J Veraart, Elisabeth J Faassen, Vasilis Dakos, Egbert H Van Nes, Miquel Lürling, and Marten Scheffer. Recovery rates reflect distance to a tipping point in a living system. *Nature*, 481(7381):357–359, 2012.

[5] Western Electricity Coordinating Council. Western systems coordinating council disturbance report, 1996.

[6] Luis Gómez-Nava, Robert T. Lange, Pascal P. Klamser, Juliane Lukas, Lenin Arias-Rodriguez, David Bierbach, Jens Krause, Henning Sprekeler, and Pawel Romanczuk. Fish shoals resemble a stochastic excitable system driven by environmental perturbations. *Nature Physics*, 19(5):663–669, February 2023.

[7] Paul DL Ritchie, Hassan Alkhayuon, Peter M Cox, and Sebastian Wieczorek. Rate-induced tipping in natural and human systems. *Earth System Dynamics*, 14(3):669–683, 2023.

[8] Yunus Sevinchan, Petro Sarkanych, Abi Tenenbaum, Yurij Holovatch, and Pawel Romanczuk. Collective decision-making with heterogeneous biases: Role of network topology and susceptibility. *Physical Review Research*, 7(1), March 2025.

[9] Vasilis Dakos, Egbert H Van Nes, Paolo d'Odorico, and Marten Scheffer. Robustness of variance and autocorrelation as indicators of critical slowing down. *Ecology*, 93(2):264–271, 2012.

[10] C Wissel. A universal law of the characteristic return time near thresholds. *Oecologia*, 65(1):101–107, 1984.

[11] Egbert H Van Nes and Marten Scheffer. Slow recovery from perturbations as a generic indicator of a nearby catastrophic shift. *The American Naturalist*, 169(6):738–747, 2007.

[12] Vasilis Dakos, Marten Scheffer, Egbert H Van Nes, Victor Brovkin, Vladimir Petoukhov, and Hermann Held. Slowing down as an early warning signal for abrupt climate change. *Proceedings of the National Academy of Sciences*, 105(38):14308–14312, 2008.

[13] Stephen R Carpenter and William A Brock. Rising variance: a leading indicator of ecological transition. *Ecology letters*, 9(3):311–318, 2006.

[14] Vishwesha Guttal and Ciriyam Jayaprakash. Changing skewness: an early warning signal of regime shifts in ecosystems. *Ecology letters*, 11(5):450–460, 2008.

[15] Vasilis Dakos, Stephen R Carpenter, Egbert H van Nes, and Marten Scheffer. Resilience indicators: prospects and limitations for early warnings of regime shifts. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1659), 2015.

[16] Sonia Kéfi, Vasilis Dakos, Marten Scheffer, Egbert H Van Nes, and Max Rietkerk. Early warning signals also precede non-catastrophic transitions. *Oikos*, 122(5):641–648, 2013.

[17] Induja Pavithran and RI Sujith. Effect of rate of change of parameter on early warning signals for critical transitions. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(1), 2021.

[18] Yu Huang, Sebastian Bathiany, Peter Ashwin, and Niklas Boers. Deep learning for predicting rate-induced tipping. *Nature Machine Intelligence*, 6(12):1556–1565, 2024.

[19] Jiashuo Liu, Zheyan Shen, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey, 2023.

[20] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(4):4396–4415, 2022.

[21] Xin Wu, Fei Teng, Xingwang Li, Ji Zhang, Tianrui Li, and Qiang Duan. Out-of-distribution generalization in time series: A survey, 2025.

[22] Thomas M Bury, RI Sujith, Induja Pavithran, Marten Scheffer, Timothy M Lenton, Madhur Anand, and Chris T Bauch. Deep learning for early warning signals of tipping points. *Proceedings of the National Academy of Sciences*, 118(39):e2106140118, 2021.

[23] Xin Li, Qunxi Zhu, Chengli Zhao, Bolin Zhao, Xue Zhang, Xiaojun Duan, and Wei Lin. Ultra-early prediction of tipping points: Integrating dynamical measures with reservoir computing. *arXiv preprint arXiv:2603.14944*, 2026.

[24] Chengzuo Zhuge, Jiawei Li, and Wei Chen. Deep learning for predicting the occurrence of tipping points. *Royal Society Open Science*, 12(7), 2025.

[25] Shirin Panahi, Ling-Wei Kong, Mohammadamin Moradi, Zheng-Meng Zhai, Bryan Glaz, Mulugeta Haile, and Ying-Cheng Lai. Machine learning prediction of tipping in complex dynamical systems. *Physical Review Research*, 6(4):043194, 2024.

[26] Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022.

[27] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR)*, 2023.

[28] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

[29] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabiclv2: A better, faster, scalable, and open tabular foundation model, 2026.

[30] Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending tabpfn-v2 to time series forecasting. *Transactions on Machine Learning Research*, 2026.

[31] Jake Robertson, Arik Reuter, Siyuan Guo, Noah Hollmann, Frank Hutter, and Bernhard Schölkopf. Do-pfn: In-context learning for causal effect estimation. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[32] Vahid Balazadeh, Hamidreza Kamkari, Valentin Thomas, Benson Li, Junwei Ma, Jesse C. Cresswell, and Rahul G. Krishnan. Causalpfn: Amortized causal effect estimation via in-context learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[33] David Schiff, Ofir Lindenbaum, and Yonathan Efroni. Gradient free deep reinforcement learning with tabpfn, 2025.

[34] Yuta Miyauchi, Masahiro Ikeda, and Yoshinobu Kawahara. Generalized stochastic resilience for early warning signals based on koopman operator. *Nonlinear Dynamics*, 114(4):246, 2026.

[35] Florian Grziwotz, Chun-Wei Chang, Vasilis Dakos, Egbert H van Nes, Markus Schwarzländer, Oliver Kamps, Martin Heßler, Isao T Tokuda, Arndt Telschow, and Chih-hao Hsieh. Anticipating the occurrence and type of critical transitions. *Science Advances*, 9(1):eabq4558, 2023.

[36] Juan Nathaniel, Carla Roesch, Derek DeSantis, Parvathi Kooloth, Hang Fan, Valerio Lucarini, Anastasia Romanou, and Pierre Gentine. Koopman early warning signals for bifurcation and rate-induced tipping. *arXiv preprint arXiv:2608.14716*, 2026.

[37] Étienne Birmelé. A scale-free graph model based on bipartite graphs. *Discrete Applied Mathematics*, 157(10):2267–2284, 2009.

[38] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[39] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[40] WWTG Peterson, T Birdsall, and We Fox. The theory of signal detectability. *Transactions of the IRE professional group on information theory*, 4(4):171–212, 1954.

[41] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[42] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[43] W. E. Ricker. Stock and recruitment. *Journal of the Fisheries Research Board of Canada*, 11(5):559–623, May 1954.

[44] John M Drake and Blaine D Griffen. Early warning signals of extinction in deteriorating environments. *Nature*, 467(7314):456–459, 2010.

[45] Giacomo Bonciolini, Dominik Ebi, Edouard Boujo, and Nicolas Noiray. Experiments and modelling of rate-dependent transition delay in a stochastic subcritical bifurcation. *Royal Society open science*, 5(3), 2018.

[46] Sang-Ki Lee, Dongmin Kim, Fabian A Gomez, Hosmay Lopez, Denis L Volkov, Shenfu Dong, Rick Lumpkin, and Stephen Yeager. A pause in the weakening of the atlantic meridional overturning circulation since the early 2010s. *Nature Communications*, 15(1):10642, 2024.

[47] Juan Nathaniel, Jiangong Liu, and Pierre Gentine. Metaflux: Meta-learning global carbon fluxes from sparse spatiotemporal observations. *Scientific Data*, 10(1):440, 2023.

[48] Siyeon Kim, Juan Nathaniel, Zhewen Hou, Tian Zheng, and Pierre Gentine. Spatiotemporal upscaling of sparse air-sea pco2 data via physics-informed transfer learning. *Scientific data*, 11(1):1098, 2024.

[49] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[50] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020.

[51] Arnuv Tandon, Karan Dalal, Xinhao Li, Daniel Koceja, Marcel Rød, Sam Buchanan, Xiaolong Wang, Jure Leskovec, Sanmi Koyejo, Tatsunori Hashimoto, et al. End-to-end test-time training for long context. *arXiv preprint arXiv:2512.23675*, 2025.

[52] Juan Nathaniel, Yongquan Qu, Tung Nguyen, Sungduk Yu, Julius Busecke, Aditya Grover, and Pierre Gentine. Chaosbench: A multi-channel, physics-based benchmark for subseasonal-to-seasonal climate prediction. *Advances in Neural Information Processing Systems*, 37:43715–43729, 2024.

[53] Eusebius J Doedel, Alan R Champneys, Fabio Dercole, Thomas F Fairgrieve, Yuri A Kuznetsov, B Oldeman, RC Paffenroth, B Sandstede, XJ Wang, and CH Zhang. *AUTO-07P: Continuation and bifurcation software for ordinary differential equations*, 2007.

[54] 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 38 (NeurIPS 2025), Track on Datasets and Benchmarks*, 2025. arXiv preprint arXiv:2505.16620.

[55] Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/, 2024.

[56] Juan Nathaniel, Carla Roesch, Jatan Buch, Derek DeSantis, Adam Rupe, Kara D Lamb, and Pierre Gentine. Deep koopman operators for causal discovery. *Communications Physics*, 8(1):513, 2025.

[57] Yu Huang. YuHuang/DeepLearningR-tipping. Zenodo, https://doi.org/10.5281/zenodo.13939234, 2024.

[58] Robert M May. Thresholds and breakpoints in ecosystems with a multiplicity of stable states. *Nature*, 269(5628):471–477, 1977.

[59] Hassan Alkhayuon, Peter Ashwin, Laura C Jackson, Courtney Quinn, and Richard A Wood. Basin bifurcations, oscillatory instability and rate-induced thresholds for atlantic meridional overturning circulation in a global oceanic box model. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 475(2225), 2019.

[60] CM Luke and PM Cox. Soil carbon and climate change: from the jenkinson effect to the compost-bomb instability. *European journal of soil science*, 62(1):5–12, 2011.

[61] Sebastian Wieczorek, Peter Ashwin, Catherine M Luke, and Peter M Cox. Excitability in ramped systems: the compost-bomb instability. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2129):1243–1269, 2011.

[62] Alessio Burrello, Lukas Cavigelli, Kaspar Schindler, Luca Benini, and Abbas Rahimi. Laelaps: An energy-efficient seizure detection algorithm from long-term human iEEG recordings without false alarms. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 752–757, 2019.

[63] Viktor K. Jirsa, William C. Stacey, Pascale P. Quilichini, Anton I. Ivanov, and Christophe Bernard. On the nature of seizure dynamics. *Brain*, 137(8):2210–2230, 2014.

[64] Patrick Mergell, Hanspeter Herzel, Thomas Wittenberg, Monika Tigges, and Ulrich Eysholdt. Phonation onset: vocal fold modeling and high-speed glottography. *The Journal of the Acoustical Society of America*, 104(1):464–470, 1998.

[65] Preston R Murray and Scott L Thomson. Vibratory responses of synthetic, self-oscillating vocal fold models. *The Journal of the Acoustical Society of America*, 132(5):3428–3438, 2012.

[66] Ryo Shimamura and Isao T Tokuda. Effect of level difference between left and right vocal folds on phonation: Physical experiment and theoretical study. *Journal of the Acoustical Society of America*, 140(4_Supplement):3393–3394, 2016.

[67] Stephan Wagner, Janina Steinbeck, Philippe Fuchs, Sophie Lichtenauer, Marlene Elsässer, Jos HM Schippers, Thomas Nietzel, Cristina Ruberti, Olivier Van Aken, Andreas J Meyer, et al. Multiparametric real-time sensing of cytosolic physiology links hypoxia responses to mitochondrial electron transport. *New Phytologist*, 224(4):1668–1684, 2019.

[68] Ben I Moat, David Smeed, Darren Rayner, William E Johns, Ryan H Smith, Denis L Volkov, Shane Elipot, Tillys Petit, Jules B Kajtar, Molly O Baringer, et al. Atlantic meridional overturning circulation observed by the rapid-mocha-wbts (rapid-meridional overturning circulation and heatflux array-western boundary time series) array at 26n from 2004 to 2024 (v2024.1a). 2024.

[69] Michael L Rosenzweig and Robert H MacArthur. Graphical representation and stability conditions of predator-prey interactions. *The American Naturalist*, 97(895):209–223, 1963.

[70] Tamer Oraby, Vivek Thampi, and Chris T Bauch. The influence of social norms on the dynamics of vaccinating behaviour for paediatric infectious diseases. *Proceedings of the Royal Society B: Biological Sciences*, 281(1780), 2014.

[71] A Demetri Pananos, Thomas M Bury, Clara Wang, Justin Schonfeld, Sharada P Mohanty, Brendan Nyhan, Marcel Salathé, and Chris T Bauch. Critical dynamics in population vaccinating behavior. *Proceedings of the National Academy of Sciences*, 114(52):13762–13767, 2017.

[72] Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1993.

[73] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Appendix contents

A Additional background **16**

B Prior-Data Fitted Networks **19**

B.1 Driver variables 19
B.2 Training Dataset Forcing Schedule 20
B.3 Augmenting SDE System 20
B.4 Task Construction for TipPFN 21
B.5 TipPFN Model Architecture 23
B.6 Training Configuration 24
B.7 Linear Forcing schedules 24
B.8 Compute Resources 26
B.9 Released assets 26

C Experiment details **28**

C.1 Datasets 28
C.2 Baselines 39
C.3 Metrics 40
C.4 Further Results 43
C.5 AUROC Uncertainty 58

A Additional background

Critical transitions. As discussed in the main text, tipping phenomena can be broadly classified into three distinct types depending on the mechanisms driving state transitions, namely bifurcation-induced tipping (b-tipping), noise-induced tipping (n-tipping), and rate-induced tipping (r-tipping) [2]. Refer to Fig. 2a for illustration on the different mechanisms underlying each one of them.

Bifurcation-induced tipping and critical slowing down. In b-tipping, critical transitions arise from local bifurcations, where an equilibrium $\mathbf{x}^*(\lambda)$ loses stability as parameters vary. Linearizing the dynamics around $\mathbf{x}^*(\lambda)$ yields

$$\frac{d\boldsymbol{\epsilon}}{dt} = J(\lambda)\,\boldsymbol{\epsilon}, \tag{S1}$$

where $\boldsymbol{\epsilon} = \mathbf{x} - \mathbf{x}^*(\lambda)$ and the Jacobian matrix is

$$J_{ij}(\lambda) = \left.\frac{\partial f_i(\mathbf{x}, \lambda)}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}^*(\lambda)}. \tag{S2}$$

A bifurcation occurs when the linear stability of the equilibrium changes, typically because one or more eigenvalues $\mu_i(\lambda)$ of $J(\lambda)$ cross the stability boundary in the complex plane, i.e.,

$$\max_i \mathrm{Re}(\mu_i(\lambda_c)) = 0. \tag{S3}$$

The manner in which eigenvalues approach this boundary constrains the bifurcation type: a real eigenvalue crossing zero is associated with steady-state bifurcations (e.g., fold, transcritical), whereas a complex-conjugate pair crossing the imaginary axis is associated with a Hopf bifurcation. In discrete-time systems, the corresponding stability boundary is the unit circle.

As the system approaches such a bifurcation, perturbations decay increasingly slowly, a phenomenon known CSD [1, 12]. In particular, as $\lambda \to \lambda_{\text{crit}}$, the leading eigenvalue satisfies $\text{Re}(\mu_i(\lambda)) \to 0$, implying a vanishing local contraction rate and increasingly slow recovery from perturbations.

Rate-induced tipping. In contrast to b-tipping, r-tipping arises from the inability of the system to track a moving attracting state under time-dependent forcing [2]. Consider again the nonautonomous system with control parameter $\lambda(t)$. For each fixed λ, assume the associated frozen system admits a locally stable equilibrium $\mathbf{x}^*(\lambda)$ satisfying

$$f(\mathbf{x}^*(\lambda), \lambda) = 0, \qquad \max_i \text{Re}(\mu_i(\lambda)) < 0. \tag{S4}$$

Defining the tracking error

$$\boldsymbol{\epsilon}(t) := \mathbf{x}(t) - \mathbf{x}^*(\lambda(t)), \tag{S5}$$

and linearizing around $\mathbf{x}^*(\lambda(t))$ yields

$$\frac{d\boldsymbol{\epsilon}}{dt} = J(\lambda(t))\,\boldsymbol{\epsilon} + \underbrace{\frac{d}{dt}\mathbf{x}^*(\lambda(t))}_{\text{branch drift}} + \text{higher-order and stochastic terms.} \tag{S6}$$

The additional term $\frac{d}{dt}\mathbf{x}^*(\lambda(t))$ represents the motion of the equilibrium induced by the changing control parameter. R-tipping occurs when this control-induced drift is sufficiently large relative to the local contraction governed by $J(\lambda(t))$, so that the trajectory can no longer track the moving equilibrium, even though the instantaneous dynamics remain locally stable. Unlike in b-tipping, no eigenvalue needs to cross the stability boundary, and therefore classical indicators based on CSD may fail to provide reliable early warning signals.

Noise-induced tipping. In n-tipping, transitions occur through stochastic perturbations that induce escape from the basin of attraction of a stable equilibrium, even when $\text{Re}(\mu_i(\lambda)) < 0$. This mechanism can act independently or interact with both b-tipping and r-tipping in stochastic systems.

Early warning signals. CSD manifests in observable time series statistics [9, 13]. For a discrete trajectory $\{x_t\}_{t=1}^T$, the *lag-1 autocorrelation* (AR1) is defined as

$$\text{AR1} = \frac{\sum_{t=2}^{T}(x_t - \bar{x})(x_{t-1} - \bar{x})}{\sum_{t=1}^{T}(x_t - \bar{x})^2}, \tag{S7}$$

where $\bar{x} := \frac{1}{T}\sum_{t=1}^{T} x_t$ denotes the empirical mean over the time window.

To connect AR1 and variance to local stability, consider the scalar linearized stochastic dynamics

$$x_{t+1} - \bar{x} = a(x_t - \bar{x}) + \sigma\xi_t, \tag{S8}$$

where $\{\xi_t\}$ is an i.i.d. zero-mean stochastic process and $|a| < 1$ ensures stability. Under stationarity and independence of the innovations $\sigma\xi_t$, $\text{AR1} \approx a$. If the system arises from a continuous-time linearization with leading eigenvalue $\mu < 0$ and sampling interval Δt, then $a = e^{\mu\Delta t}$, so $\text{AR1} \to 1$ as $\mu \to 0^-$.

Similarly, the stationary variance satisfies

$$\text{VAR}(x) = \frac{\sigma^2}{1 - a^2}, \tag{S9}$$

which increases as $a \to 1$.

A detailed derivation of the relationship between AR1, variance, and the underlying spectral properties of the linearized dynamics is provided in e.g., [35, 34].

These signatures, rising AR1 and variance, form the basis of classical EWS for b-tipping. However, these indicators rely on local linearization near equilibrium and are therefore intrinsically tied to eigenvalue dynamics. In n-tipping, transitions occur due to stochastic escape from a basin of attraction even when $\text{Re}(\mu_i) < 0$, while in r-tipping, rapid parameter changes prevent the system from tracking its quasi-static equilibrium without any eigenvalue crossing. In both cases, critical slowing down may be weak or absent.

While bifurcation theory provides a principled framework for characterizing tipping behaviour, its practical application is limited. Identifying tipping points requires access to the governing equations, parameters, and equilibrium states needed to compute the Jacobian and its eigenvalues. In real-world systems, these quantities are typically unknown or only partially observed, motivating data-driven approaches that infer tipping behaviour directly from observed trajectories without requiring explicit knowledge of the underlying dynamical system.

B Prior-Data Fitted Networks

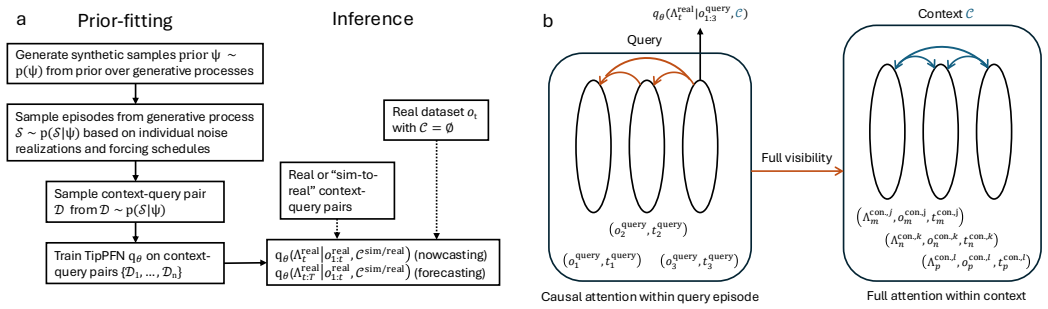


Figure S1: (a) Prior-data fitting and inference and (b) attention structure of TipPFN. Schematics based on [26].

B.1 Driver variables

Following [22], the driver M is generated from randomly constructed two-dimensional dynamical systems of the form

$$\dot{x} = \sum_{i=1}^{10} a_i p_i(x, y), \qquad \dot{y} = \sum_{i=1}^{10} b_i p_i(x, y), \tag{S10}$$

where $(x, y) \in \mathbb{R}^2$ and $\{p_i(x, y)\}_{i=1}^{10}$ denotes the set of all monomials up to third order:

$$p(x, y) = (1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3). \tag{S11}$$

The coefficients a_i, b_i are independently sampled from $\mathcal{N}(0, 1)$, after which a random subset (50%) is set to zero to induce sparsity. To encourage bounded trajectories, coefficients associated with cubic terms are constrained to be negative. We then perform integration for 10^4 timesteps and 10^{-2} discretization to select trajectories that converge to an equilibrium, defined as the final 10 points with difference of less than 10^{-8}. We then use AUTO-07P program [53] to identify bifurcation types and their corresponding critical values along the identified equilibrium branch as each nonzero parameter is varied within the interval $[-5, 5]$. Table S1 lists the relevant driver-generation hyperparameters.

Table S1: **Hyperparameters of the synthetic polynomial driver.**

Quantity	Value or distribution
Polynomial degree	All monomials up to degree 3; 10 coefficients per equation, 20 total.
Coefficient prior	$a_j, b_j \sim \mathcal{N}(0, 1)$ before sparsification.
Coefficient sparsity	Exactly 50% of the 20 coefficients are set to zero uniformly at random.
Boundedness bias	Cubic coefficients x^3, x^2y, xy^2, y^3 in both equations are replaced by their negative absolute values.
Initial condition for equilibrium screening	$z_0 \sim \mathcal{N}(0, 4I_2)$.
Equilibrium screening integration	Forward Euler with $\Delta t = 0.01$ for up to 100 time units.
Convergence criterion	Norm of the range of the final 10 states below 10^{-8} and maximum trajectory amplitude below 10^3.
Stability criterion	All Jacobian eigenvalues at the equilibrium have negative real part.
Recovery rate	$r_M = \lvert \max_i \operatorname{Re} \mu_i \rvert$ at the stable equilibrium.
Model retry budget	Up to 100 random polynomial models per TipBox attempt.
Bifurcation retry budget	Up to 50 TipBox attempts per raw sample; AUTO output suppressed.

For a selected bifurcation point, let p_0 be the initial coefficient value and p_{crit} the AUTO-07P bifurcation value. The schedule $\tilde{\lambda}_k(t)$ is applied as

$$\lambda_{M,k}(t) = p_0 + \tilde{\lambda}_k(t)\big(p_{\text{crit}} - p_0\big). \tag{S12}$$

The RDTC target is

$$\Lambda_k(t) = 1 - \tilde{\lambda}_k(t). \tag{S13}$$

Thus $\Lambda = 0$ marks the bifurcation point, $\Lambda > 0$ is subcritical, and $\Lambda < 0$ means the schedule has passed the detected critical value.

Driver SDE simulation Given M and a forcing schedule, TipBox simulates the stochastic driver with Euler–Maruyama at integration step $\Delta t = 0.01$ and output sampling interval 1.0. A 100-time-unit burn-in is run at the initial parameter before the recorded trajectory starts. The additive driver noise scale is

$$\sigma_M = \sqrt{2r_M}\,\sigma_{\text{tilde}}\,\xi, \qquad \sigma_{\text{tilde}} = 0.01, \quad \xi \sim \text{Triangular}(0.75, 1.0, 1.25), \tag{S14}$$

and Brownian increments are scaled by $\sigma_M\sqrt{\Delta t}$.

B.2 Training Dataset Forcing Schedule

For the training dataset, forcing schedules are sampled independently per episode, while the bifurcation system M is shared across the $K = 4$ episodes of a raw sample. Table S2 summarizes the full schedule prior. All schedules start at $\tilde{\lambda} = 0$. Examples of bezier-type schedules are shown in Fig. S2.

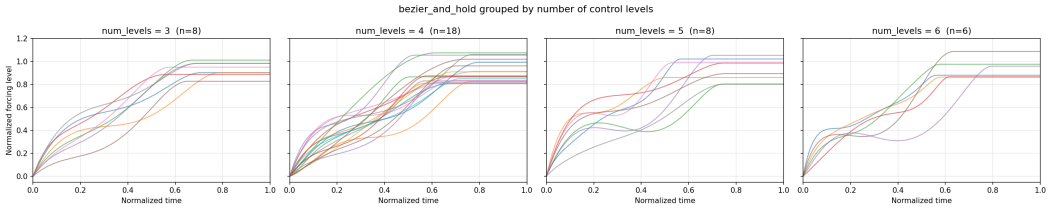


Figure S2: Bezier-type forcing schedules used in generation of the training dataset.

Note that this partially non-linear and beyond-criticality forcing is unlike that used in validation, see Section B.7. During training, it is causally permissible to also show non-linear forcings, through which the model may learn to recognise a wider range of dynamical system responses to changes in forcing.

B.3 Augmenting SDE System

The two-dimensional driver is embedded into a higher-dimensional stochastic system so that TipPFN does not only see canonical normal-form trajectories.

The auxiliary variables $u_k(t) = (u_{1,k}(t), \ldots, u_{14,k}(t))$ solve an Ito SDE with diagonal diffusion,

$$du_{i,k}(t) = \left[(1-\eta_i) f_i(r_{i,k}(t)) - \gamma_i u_{i,k}(t)\right] dt + \sigma_i\, dW_{i,k}(t), \tag{S15}$$

where $r_{i,k}(t)$ concatenates the parent auxiliary variables and external-input channels selected by $\mathcal{G}$ for variable i. The functions f_i are sampled MLPs, one per dynamic variable. The same graph, MLPs, and SDE parameters are shared across the episodes of a raw sample, while initial states, Brownian noise, TipBox noise, and forcing schedules vary by episode.

Directed graph sampler The graph $\mathcal{G}$, which is structuring the interactions within the variables of the auxiliary system [54], is generated by a scale-free graph model based on bi-partite graphs [37]: For each raw sample, the code first samples a directed bipartite graph between variable nodes V and interaction nodes H, with $|V| = 14$ and $|H| = 8$. For every variable i, the desired outgoing and incoming interaction degrees D_i^+ and D_i^- are sampled independently from the truncated power law

$$\Pr(D = d) = \frac{d^{-2}}{\sum_{\ell=1}^{8} \ell^{-2}}, \qquad d \in \{1, \ldots, 8\}.$$

Table S2: **Forcing schedule prior**. Final levels below 1 create subcritical approaches, level 1 reaches the bifurcation, and levels above 1 pass the bifurcation.

Schedule	Probability	Parameters	Definition and effect

`bezier_and_hold`	0.4	`start=0`
		`smooth_hold=true`
		$h \sim U(0.6, 0.8)$
		$L = 1$ w.p. $1/2$; else $L \sim U(0.6, 1.2)$
		$n_c \in \{3, 4, 5, 6\}$ with ratios $2:3:2:1$
		control range $[0.1L, 0.9L]$
			Bezier ramp from 0 to final level L by fraction h, followed by a hold at L. Interior control-point values are clipped to $[0, L]$; duplicating the final control point makes the join to the hold phase approximately smooth.

`ramp_and_hold`	0.4	$h \sim U(0.6, 0.8)$ $L = 1$ w.p. $1/2$; else $L \sim U(0.6, 1.2)$	Linear ramp from 0 to L until fraction h, followed by a constant hold at L.
`constant`	0.2	$\tilde{\lambda}(t) = 0$	Stationary no-forcing baseline at the initial stable equilibrium, with $\Lambda(t) = 1$.

Conditional on these sampled degrees, each candidate edge $i \to h$ is included with probability $D_i^+/8$, and each candidate edge $h \to i$ with probability $D_i^-/8$, independently across interaction nodes. The bipartite graph is then projected to a directed variable graph: an edge $i \to j$ is added whenever there exists an interaction node h such that $i \to h$ and $h \to j$. Multiple such paths collapse to one directed edge, and self-loops are retained. After this projection, four external input channels are added in the training set and in all synthetic evaluation sets: the two state variables of the canonical driver M, its forcing parameter λ and the RDTC $\Lambda = 1 - \tilde{\lambda}$. The auxiliary variables therefore carry a noisy, filtered readout of the forcing level; Appendix C.4.1 reports results on evaluation sets in which they are driven by the state variables only. Each external-input node connects to each auxiliary variable independently with probability $p_{\text{inp}} = 0.8$. MLP hyperparameters are given in table S3

Table S3: **Parameter hyperparameters for the auxiliary SDE.** These are sampled once for the shared SDE system inside a raw sample.

Quantity in Eq. S15	Value or distribution
MLP depth	2 linear layers
Hidden width	32 hidden units
Activation	`tanh`
Weight and bias initialization scale	For each MLP, $s_w = 1.0 + \text{HalfCauchy}(1.0)$ resampled until $s_w \leq 1.5$; weights and biases use $\mathcal{N}(0, s_w^2)$.
Hidden-layer sparsity	For each MLP, $s_p = 0.5 - \text{HalfCauchy}(0.5)$ resampled until $s_p \geq 0.1$; hence $s_p \in [0.1, 0.5]$.
Sparsity application	Hidden-layer weights only; retained weights are scaled by $(1 - s_p)^{-1/2}$. With 2 layers, there is no interior hidden-to-hidden layer, so this setting is effectively inactive for variable MLPs.
Preactivation noise	0.0.
Self-loop coefficient	$\gamma_i = 20\,\lvert\mathcal{N}(0, 1)\rvert$.
Friction coefficient	$\eta_i = 0.5\,\lvert\mathcal{N}(0, 1)\rvert$; the code multiplies the MLP flow by $1 - \eta_i$.
Diffusion coefficient	$\sigma_i = 0.05\,\lvert\mathcal{N}(0, 1)\rvert$ with independent diagonal noise.

B.4 Task Construction for TipPFN

Each raw sample is transformed into a masked tabular time-series task.

Row selection by resampling A subset of time steps is selected based on stratified jitter, to yield query time-series with lengths between 92 and 256 time steps, constrained to a total context

size of 320 time steps or less, to reserve 192 of the 512 step budget for the query trajectory and potential forecasting time steps, with indices $1:T^*$ referring to the resampled time steps. Using $\Lambda_{1:T^*} = 1 - \tilde{\lambda}_{1:T^*}$, this yields the query-context pair $\mathcal{D} = ((\Lambda_{1:T^*}, \mathbf{o}_{1:T^*}) \cup \mathcal{C})$ with prior distribution $p(\mathcal{D})$ (see Fig. 3), hyperparameters are given in Table S4.

Table S4: **Episode partitioning and temporal subsampling.**

Quantity	Value or distribution
Maximum sequence length	512 rows.
Query episodes	Always 1 episode.
Context episodes	$N_{\text{ctx}} \in \{0, 1, 2, 3\}$ with probabilities $(0.2, 0.3, 0.3, 0.2)$.
Episode selection	Deterministic-random selection; query and context episodes do not overlap.
Context length budget	$N_{\text{ctx}} = 0 : [0, 0]$, $1 : [92, 256]$, $2 : [184, 320]$, $3 : [276, 320]$ total rows.
Context per-episode bounds	Minimum 92 rows, maximum 256 rows.
Query length budget	Target 192 rows, with per-episode bounds $[1, 256]$.
Temporal subsampling	Stratified jitter with endpoints retained.
Overflow handling	Shrink context; use available episodes if insufficient; error on empty query.
Identifier normalization	Episode ID and time are min-max normalized to $[-1, 1]$ over valid rows.

Column selection The dynamic variables are partitioned into target columns, which the model may be asked to predict, and observation columns. RDTC is always the first target column. Additional targets are sampled from the auxiliary time series $\mathbf{u}$, and the drivers x and y. Feature columns are sampled from the remaining var.

The number of target columns is sampled from $\{1, \ldots, 16\}$ with probabilities proportional to

$$\rho_{\text{act}} = (1.00, 0.95, 0.90, 0.86, 0.82, 0.78, 0.74, 0.70, 0.67, 0.64, 0.61, 0.58, 0.55, 0.52, 0.49, 0.46).$$

The number of feature columns is sampled from $\{1, \ldots, 20\}$ with probabilities proportional to

$$\rho_{\text{feat}} = (1.00, 0.97, 0.94, 0.91, 0.88, 0.85, 0.82, 0.79, 0.76, 0.73, 0.70, 0.67, 0.64, 0.61, 0.58, 0.55, 0.52, 0.49, 0.46, 0.43).$$

Masking RDTC is always masked for every query row, making $\Lambda(t)$ the primary hidden target. The additional query task is sampled with a 50/50 mixture. In the `none` task, no additional query mask is applied; these views train nowcasting of current distance to criticality from currently observed signals. In the `forecast` task, a cutoff fraction $c \sim U(0.2, 0.4)$ is sampled and further selected action columns are masked after that cutoff.

Normalization and missing values Normalization is applied after partitioning and masking, using only information available under inference-time constraints. Raw RDTC is transformed to the model target

$$\Lambda^* = \tanh(5\Lambda). \tag{S16}$$

All other columns are median-centered and transformed with an adaptive inverse-hyperbolic-sine map,

$$x \mapsto \operatorname{asinh}\left(\frac{x - \operatorname{median}(x)}{s}\right), \qquad s = \max(0.01,\ \operatorname{IQR}(x)). \tag{S17}$$

The median and IQR are computed from the observed rows only, excluding masked prediction targets and excluding RDTC from the statistics. They are computed once per task over the whole observed window, so only the prediction at the last observed row, at which all risk scores are read, and the forecast rows are free of later observations; earlier points of a nowcast trajectory are normalized using later observations.

Training targets The model predicts 99 quantiles for every masked action value and is optimized with pinball loss. For distributional decoding and validation metrics, quantile outputs are sorted to enforce non-crossing quantiles; CDF scores are 0 or 1 outside the predicted quantile range. RDTC targets receive unit loss weight, while auxiliary signal reconstruction targets receive weight 0.2. This makes distance-to-criticality prediction the primary objective while using reconstruction of observed dynamics as an auxiliary regularizer.

B.5 TipPFN Model Architecture

The trained model is a transformer architecture. A task is represented as a padded table with row index $r = 1, \ldots, R$ and column index split into action columns and feature columns. Each row has two identifier values, episode id and time, denoted $\iota_r = (e_r, t_r)$. Let $y_{r,a}$ denote the value in action slot $a \in \{1, \ldots, A\}$ and $x_{r,f}$ the value in feature slot $f \in \{1, \ldots, F\}$. The first action slot is reserved for RDTC, $y_{r,1} = \Lambda_r^*$ after the tanh transform in Equation S16. A Boolean mask $m_{r,a}$ indicates which action values are prediction targets.

The model uses separate scalar encoders for action and feature values,

$$E_y : \mathbb{R} \to \mathbb{R}^{d_{\text{model}}}, \qquad E_x : \mathbb{R} \to \mathbb{R}^{d_{\text{model}}},$$

and fixed random positional buffers for rows, action columns, and identifier/feature columns. For active, unmasked slots the initial tokens are therefore

$$h_{r,a}^{(0)} = E_y(y_{r,a}) + p_r^{\text{row}} + p_a^{\text{act}}, \qquad h_{r,A+g}^{(0)} = E_x(v_{r,g}) + p_r^{\text{row}} + p_g^{\text{feat}},$$

where $a = 1, \ldots, A$, $g = 1, \ldots, F + 2$, and $v_r = (e_r, t_r, x_{r,1}, \ldots, x_{r,F})$. Masked action tokens are replaced by a zero vector before entering the transformer.

Table S5: **Resolved TipPFN architecture hyperparameters.**

Quantity	Value
Maximum rows R	512.
Identifier columns	2: episode id and time.
Maximum feature columns F	20 payload features, plus the 2 identifier columns in the feature stream.
Maximum action columns A	16.
Total token columns per row	$16 + 2 + 20 = 38$.
Hidden width d_{model}	1280.
Transformer blocks	8 row-column blocks.
Attention heads	8 heads; head dimension 160.
MLP width	Ratio 4; hidden width 5120.
Dropout	0.1 in attention and MLP residual branches during training.
Output dimension	99 per action token.
Quantile levels	$\alpha_j = j/100$, $j = 1, \ldots, 99$.
Row attention mode	Causal.
Causal time tolerance	10^{-6}.
Learned parameters	209,959,779, excluding fixed positional buffers.
Fixed positional buffers	704,000 scalar entries: 512×1280 row, 22×1280 id/feature-column, and 16×1280 action-column buffers.

Each row-column block applies column attention, row attention, and an MLP with pre-normalization and residual connections:

$$\begin{aligned} H &\leftarrow H + \text{Dropout}\big(\text{Attn}_{\text{col}}(\text{LN}(H))\big), \\ H &\leftarrow H + \text{Dropout}\big(\text{Attn}_{\text{row}}(\text{LN}(H))\big), \\ H &\leftarrow H + \text{Dropout}\big(\text{MLP}(\text{LN}(H))\big). \end{aligned}$$

Column attention is applied independently within each row across the active action, identifier, and feature columns. Row attention is applied independently for each column across the temporal/episode rows. Attention uses fused scaled dot-product attention with bias-free query, key, value, and output projections. The MLP is a two-layer feed-forward network with GELU activation using the `tanh` approximation.

Causal row mask Let c_i indicate that row i belongs to a context episode. In causal mode, the set of row keys visible to query row i is

$$\mathcal{A}(i) = \begin{cases} \{j : c_j = 1\} \cup \{i\}, & c_i = 1, \\ \{j : c_j = 1\} \cup \{j : e_j = e_i,\ t_j < t_i - 10^{-6}\} \cup \{i\}, & c_i = 0. \end{cases} \tag{S18}$$

Padded keys are removed from $\mathcal{A}(i)$. Thus context rows can exchange information bidirectionally within the context set, while query rows can attend to all context rows and only to their own past and present query rows.

B.6 Training Configuration

Training uses the synthetic task distribution described above.

For an output vector $\hat{q}_{r,a,1:99}$ and target $y_{r,a}$, the pinball loss at quantile level α is

$$\ell_\alpha(\hat{q}, y) = \begin{cases} \alpha(y - \hat{q}), & y \geq \hat{q}, \\ (\alpha - 1)(y - \hat{q}), & y < \hat{q}. \end{cases}$$

The per-entry loss is the mean of ℓ_{α_j} over the 99 quantile levels. Let $\mathcal{M}_{\mathrm{rdtc}}$ be the masked RDTC entries, and let $\mathcal{M}_{\mathrm{sig}}$ be all other masked target entries. The training loss is

$$\mathcal{L} = 1.0\ \mathrm{mean}_{(r,a)\in\mathcal{M}_{\mathrm{rdtc}}}\, \ell(\hat{q}_{r,a}, y_{r,a}) + 0.2\ \mathrm{mean}_{(r,a)\in\mathcal{M}_{\mathrm{sig}}}\, \ell(\hat{q}_{r,a}, y_{r,a}),$$

with empty groups skipped.

Training hyperparameters are given in table S6.

Table S6: **Resolved optimization and trainer hyperparameters.**

Quantity	Value
Per-device batch size	42 transformed views.
Global batch size	168 transformed views across 4 DDP workers (4 NVIDIA H200 GPUs, one node).
Data-loader workers	1 worker, `persistent_workers=false`, `pin_memory=true`.
Precision	`bf16-mixed`.
Optimizer	Muon [55] for 2D trainable tensors, AdamW for remaining tensors.
Learning rate	0.01.
Weight decay	0.01 for decay groups; no decay for biases, 1D norms, and parameters matching no-decay keywords.
Muon parameters	Momentum 0.95; Nesterov enabled; Newton-Schulz steps 5; $\epsilon = 10^{-7}$.
AdamW parameters	$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-7}$.
Scheduler	Linear warmup followed by linear decay.
Warmup	1% of 200,000 steps, i.e. 2,000 optimizer steps.
Minimum LR scale	0.1, giving final LR 0.001 from the initial LR 0.01.
Maximum optimizer steps	200,000.
Released checkpoint	Step 22,000 (learning rate $0.91\times$ peak).
Gradient clipping	Global norm clipping at 1.0.
Random seeding	Lightning `seed_everything=true`.

B.7 Linear Forcing schedules

In the following we provide more details on the forcing schedule to generate the various tipping scenarios used for validation datasets, see AppendixC.1. Note that these differ from the non-linear forcing schedules used in the training dataset.

For each episode in the synthetic and semi-real b-type systems, the a forcing schedule was randomly selected. To gain a balanced distribution of positive (critical) and negative (non-critical) episodes, an episode was grouped as *critical* with 50% probability. If not critical, one of the four non-critical groups described in Table S7 was uniform randomly selected.

The goal of these forcing schedules was to provide a realistic evaluation task: instead of just distinguishing between critical and equilibrium trajectories, the evaluation task includes distinguishing

non-critically forced b-type systems (*approaching* criticality, but not reaching it) from actual critical forcings. We employ exclusively linear forcing schedules for all experiments; changing the forcing at an unobserved future time would be causally intractable for a prediction model [56].

As figures S3 and S4 show, these normalized forcing schedules $\Lambda(t) = 1 - \tilde{\lambda}(t)$ also varied in the initial starting point and the time of reaching criticality. To avoid strong perturbation, the tipping system was warmed-up by slowly and smoothly varying the control parameter from equilibrium to the parameter that is the start of the forcing schedule (not shown).

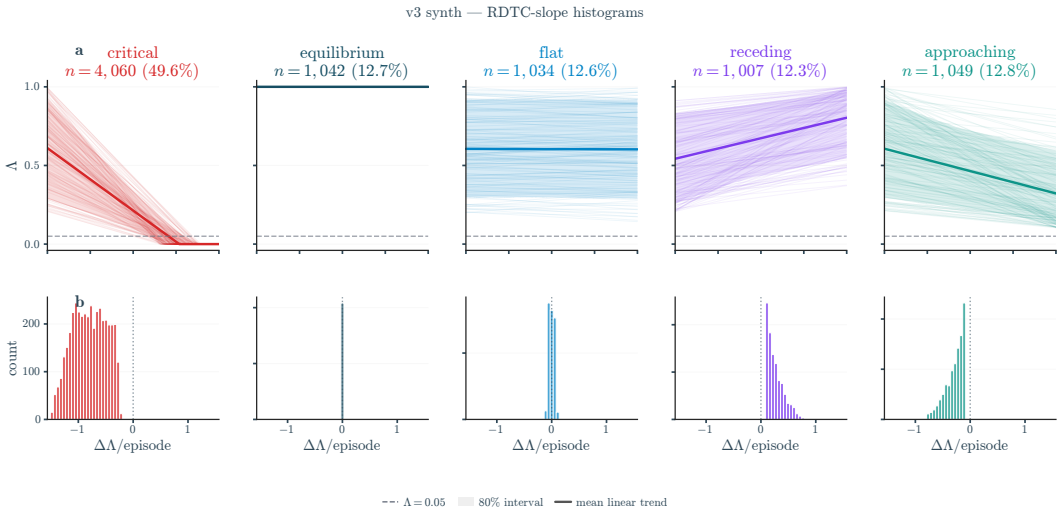


Figure S3: Forcing schedules $\Lambda(t)$ and overall group sizes (number of episodes) and composition for the canonical evaluation datasets (b_fold, b_hopf, b_transcritical). Refer to Table S14 for the actually used episode counts out of this larger dataset.

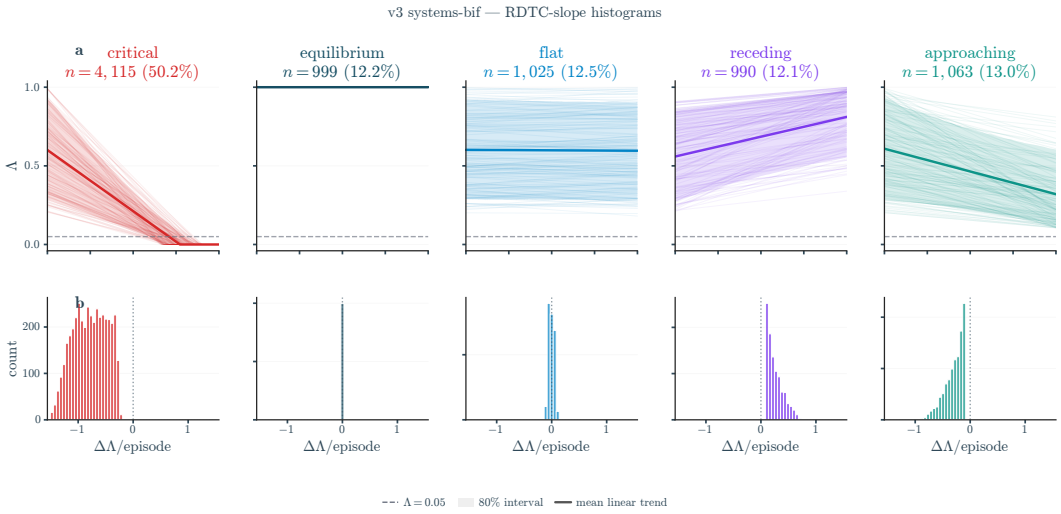


Figure S4: Forcing schedules $\Lambda(t)$ and overall group sizes (number of episodes) and composition for the semi-real bifurcation systems (see Table S8). Refer to Table S14 for the actually used episode counts out of this larger dataset.

R-tipping The notion of a *normalized* forcing schedule does not easily translate to r-type tipping systems, which require to surpass not a certain forcing value but a critical change in forcing $\dot{\lambda}$. In particular, there is no straight-forward equivalent of the *approaching* or *receding* groups. To still create some variability in schedules, we split the non-critical group into *equilbibrium* and *flat* groups, where the flat group has a higher but still non-critical maximum forcing rate.

Table S7: Trajectory classes defined by the evolution of the normalized control parameter $\tilde{\lambda}(t)$ and their corresponding $\Lambda(t)$ behavior.

Trajectory type	**Definition of $\tilde{\lambda}(t)$**	**$\Lambda(t)$ behavior**
Critical (tipping)	$\tilde{\lambda}(t) = \begin{cases} \tilde{\lambda}_0 + \alpha t, & t < t_c, \\ 1, & t \geq t_c, \end{cases} \quad \alpha > 0$	$\Lambda(t) \to 0$. Critical value $\tilde{\lambda} = 1$ is reached at t_c.
Approaching (non-critical)	$\tilde{\lambda}(t) = \tilde{\lambda}_0 + \alpha t, \quad \tilde{\lambda}(t) < 1 - \epsilon \; \forall t, \quad \epsilon > 0$	$\Lambda(t) \to \epsilon$. The trajectory remains bounded away from criticality.
Receding (non-critical)	$\tilde{\lambda}(t) = \tilde{\lambda}_0 - \alpha t, \quad \tilde{\lambda}_0 < 1 \; \forall t$	$\Lambda(t) \to 1$. The trajectory moves away from the critical point.
Flat (non-critical)	$\tilde{\lambda}(t) = \tilde{\lambda}_0 + \delta(t), \quad \lvert\delta(t)\rvert \ll 1$	$\Lambda(t) \approx \text{const.}$
Equilibrium	$\tilde{\lambda}(t) = 0 \quad \forall t$	$\Lambda(t) \approx 1$. System remains far from criticality.

B.8 Compute Resources

All experiments were conducted on rented GPU servers (IONOS cloud) and a single dedicated CPU server (Hetzner, Germany). We report hardware specifications, per-stage compute, and total project compute below.

- **Dataset generation.** The synthetic training dataset was generated on a single CPU server (AMD EPYC 9454P, 48 cores / 96 threads, 125 GB RAM) over approximately 8 days at $\sim 80\%$ CPU utilisation ($\sim 7{,}400$ CPU core-hours).
- **Model training (reported).** The checkpoint used in all experiments was trained on 4$\times$ NVIDIA H200 NVL GPUs (141 GB HBM3e each, 1 TB system RAM) for approximately 47 hours (~ 189 H200 GPU-hours) at near-full utilisation ($\sim 100\%$ SM occupancy, ~ 493 W average draw per device).
- **Evaluation (reported).** All evaluation runs used NVIDIA H200 NVL GPUs (1–4 per run). The paper-bound evaluation runs (7 datasets, including ablations and step-size variants) consumed approximately ~ 150 GPU-hours in total. Individual runs ranged from ~ 1 h (lean / fast configurations) to ~ 20 h (full SWEC and TAC sweeps on 4$\times$ H200), with TabPFN and Bury baselines requiring considerable CPU and GPU time.
- **Total project compute.** The full research project, including preliminary and failed experiments not reported in the paper, consumed substantially more compute than the final experiments. Over the course of development, the training project logged $\sim 1{,}400$ GPU-hours across 44 runs (spanning smaller NVIDIA RTX 6000 Ada, RTX A6000, RTX Pro class GPUs). For evaluation, we logged ~ 750 GPU-hours across 347 runs (all on H200). Including dataset generation, we estimate the total project compute at $\sim 2{,}150$ GPU-hours plus $\sim 7{,}400$ CPU core-hours.

B.9 Released assets

We release the following assets.

- **Code** for data loading, training, evaluation and all baselines: `https://github.com/kausable/tippfn-neurips26` (Apache-2.0).
- **TipPFN checkpoint** (210M parameters, step 22,000): `https://huggingface.co/kausable/tippfn-neurips26` (Apache-2.0); **training data**: `https://huggingface.co/datasets/kausable/tippfn-neurips26-train` (CC BY 4.0).
- **Evaluation sets** for the canonical, semi-real, TAC and DaphniaExt rows of Table 1: `https://huggingface.co/datasets/kausable/tippfn-neurips26-eval`. Simulated data are CC BY 4.0; the TAC and DaphniaExt recordings are derived from Dryad deposits

under CC0 (https://doi.org/10.5061/dryad.4cj4k, https://doi.org/10.5061/dryad.q3p64).

- **Baseline checkpoints** trained by us: the Bury et al. [22] classifiers retrained on our training data (https://huggingface.co/kausable/tippfn-neurips26-bury-retrained, Apache-2.0), and the Huang et al. [18] classifier, trained by Juan Nathaniel with the authors' code and data [57] (https://huggingface.co/kausable/tippfn-neurips26-huang, CC BY 4.0).

The following assets are not redistributed. The SWEC-iEEG recordings may be used for research and education only and may not be redistributed; the code repository provides a guide and a preprocessing script that rebuild our evaluation set from the recordings (http://ieeg-swez.ethz.ch/). The TabPFN 2.6 weights are licensed by Prior Labs and are obtained through the `tabpfn` package. The original classifiers of Bury et al. and Zhuge et al. are downloaded from the authors' repositories (https://github.com/ThomasMBury/ewstools, https://github.com/zhugchzo/dl_occurrence_tipping).

C Experiment details

C.1 Datasets

Summary Table S8 summarizes the datasets we simulated with TipBox[4] as well as additional benchmark datasets for evaluation on real-world examples. Sample tipping and non-tipping trajectories, along with their specific instantiation of the bifurcating parameter or forcing rate schedules are shown in Figures S5 and S6.

Table S8: Datasets used for evaluating tipping point prediction across b-tipping (B), r-tipping (R), and n-tipping (N), spanning canonical, semi-real, sim-to-real, and real-world systems.

Dataset	System	Description	Type	References
Canonical	`b_fold`	Fold (saddle-node) bifurcation	B	[22]
Canonical	`b_hopf`	Hopf bifurcation	B	[22]
Canonical	`b_transcritical`	Transcritical bifurcation	B	[22]
Semi-real	`b_harvesting`	May's harvesting model	B	[58]
Semi-real	`b_rosenzweig_macarthur_tc`	Consumer-resource (transcritical)	B	[22]
Semi-real	`b_rosenzweig_macarthur_hopf`	Consumer-resource (Hopf)	B	[22]
Semi-real	`b_seirx_tc`	SEIRx disease-vaccination	B	[22]
Semi-real	`b_amoc`	AMOC 2-box model	B	[7, 59]
Semi-real	`r_bautin`	Bautin normal form	R	[7]
Semi-real	`r_compost_bomb`	Compost bomb instability	R	[60, 61]
Semi-real	`r_saddle_node`	Saddle-node normal form	R	[2]
Semi-real	`r_amoc`	AMOC 2-box model	R	[7, 59]
Sim-to-real	`TAC`	Thermoacoustic instability experiments	B	[45]
Sim-to-real	`DaphniaExt`	Ecological extinction events	N	[44]
Real	`SWEC-iEEG`	Epileptic seizure onset (iEEG)	B	[62, 63]
Real	`microcosm`	Cyanobacteria under light stress	B/N	[4]
Real	`voice`	Phonation onset under increasing flow (Hopf)	B	[64, 65, 66]
Real	`cellular_atp`	Cellular energy status under hypoxia	B/N	[67]
Real	`greenhouse_earth`	Calcium carbonate in the end of greenhouse Earth	B	[12]
Real	`blackout_frequency`	Bus voltage frequency before power grid failure	N	[5]
Real	`AMOC`	Observed variability of the meridional overturning circulation (MOC)	R/B	[68]

[4] https://github.com/juannat7/tipbox

b_fold The fold (saddle-node) bifurcation is described by the normal form

$$\frac{dx}{dt} = \mu - x^2, \tag{S19}$$

where two equilibria collide and annihilate at $\mu = 0$, leading to an abrupt transition. This bifurcation is associated with catastrophic shifts, where the system loses stability and jumps to a distant attractor.

b_hopf The Hopf bifurcation involves a two-dimensional system of the form

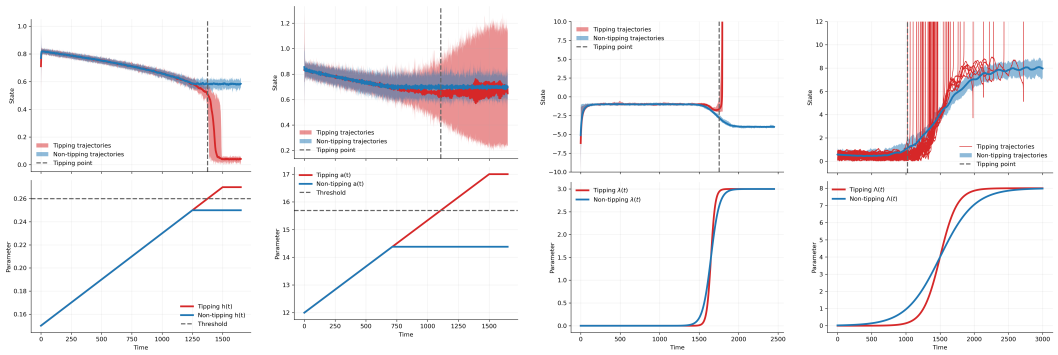
$$\begin{aligned} \frac{dx}{dt} &= \mu x - y - x(x^2 + y^2), \\ \frac{dy}{dt} &= x + \mu y - y(x^2 + y^2), \end{aligned} \tag{S20}$$

where a pair of complex conjugate eigenvalues crosses the imaginary axis at $\mu = 0$, leading to the emergence of a stable or unstable limit cycle. This transition results in oscillatory dynamics rather than a shift between steady states.

b_transcritical The transcritical bifurcation is given by

$$\frac{dx}{dt} = \mu x - x^2, \tag{S21}$$

in which two equilibria exist for all parameter values and exchange stability at $\mu = 0$. In contrast to the fold case, the transition is continuous and does not involve the disappearance of equilibria, but rather a reorganization of stability.



(a) May's harvesting (b) Rosenzweig-MacArthur (c) Saddle-node (d) Bautin

Figure S5: Representative B-tipping (**a**, **b**) and R-tipping (**c**, **d**) examples. In each panel, the upper plot shows examples of tipping (red) and non-tipping (blue) trajectories, and the lower plot shows the corresponding parameter schedule. Note that the validation datasets use a more diverse set of forcing schedules, see Figures S3 and S4, which vary the initial value and slope (for b-type systems only) as well as the timing of the critical transition (for all systems, including r-type).

b_harvesting We consider May's single-species harvesting model [58] describing the dynamics of a biomass variable $x(t) \in \mathbb{R}_+$. The stochastic dynamics are given by

$$\frac{dx}{dt} = rx\left(1 - \frac{x}{k}\right) - h(t)\frac{x^2}{s^2 + x^2} + \sigma_x\,\xi(t), \tag{S22}$$

where r is the intrinsic growth rate, k is the carrying capacity, s controls the saturation scale of harvesting, $h(t)$ is the harvesting rate, and $\xi(t)$ denotes Gaussian white noise with amplitude σ_x.

The logistic growth term promotes population persistence, while the nonlinear harvesting term induces a destabilizing feedback. As the harvesting rate h increases, the system undergoes a fold (saddle-node) bifurcation at a critical value $h^\star$, beyond which the stable positive-biomass equilibrium disappears, leading to an abrupt collapse of the population.

We use parameters $r = 1$, $k = 1$, $s = 0.1$, and $\sigma_x = 0.01$, for which the deterministic system exhibits a fold bifurcation at $h^\star \approx 0.26$. Tipping trajectories are generated by ramping the harvesting rate from $h = 0.15$ to $h = 0.27$, crossing the bifurcation threshold, while non-tipping trajectories follow the same initial ramp but are capped at $h = 0.25$, remaining below the critical point. Tipping is identified as the collapse of the biomass toward the low-density state.

b_rosenzweig_macarthur_tc and b_rosenzweig_macarthur_hopf We consider the Rosenzweig–MacArthur consumer–resource model [69, 22] with state $\mathbf{x}(t) = (x(t), y(t))^\top \in \mathbb{R}^2_+$, where $x(t)$ denotes the resource population and $y(t)$ the consumer population. The stochastic dynamics are given by

$$\frac{dx}{dt} = rx\left(1 - \frac{x}{k}\right) - \frac{a(t)\,xy}{1 + a(t)hx} + \sigma_x\,\xi_x(t), \tag{S23}$$

$$\frac{dy}{dt} = \frac{e\,a(t)\,xy}{1 + a(t)hx} - my + \sigma_y\,\xi_y(t), \tag{S24}$$

where r is the intrinsic growth rate of the resource, k its carrying capacity, $a(t)$ the consumer attack rate, e the conversion efficiency, h the handling time, and m the consumer mortality rate. The noise terms $\xi_x(t)$ and $\xi_y(t)$ are independent Gaussian white noise processes with amplitudes σ_x and σ_y.

As the attack rate a increases, the system exhibits two bifurcations that structure its dynamical regimes. At $a \approx 5.60$, a transcritical bifurcation marks the consumer invasion threshold: for $a < 5.60$, the system converges to a consumer-free equilibrium $(x^*, 0)$, while for $a > 5.60$, a stable coexistence equilibrium (x^*, y^*) emerges. As a increases further, the coexistence equilibrium remains stable until a Hopf bifurcation at $a^\star \approx 15.69$, where a pair of complex conjugate eigenvalues crosses the imaginary axis, leading to the onset of predator–prey cycles. For $a > a^\star$, the system exhibits stable limit cycle behaviour.

Initial conditions are sampled near the equilibrium corresponding to the initial parameter value. In the transcritical regime, trajectories are initialized near the resource-only equilibrium, with $x(0)$ close to carrying capacity and a small but positive consumer density $y(0) > 0$, since $y = 0$ is always an invariant equilibrium and would otherwise prevent consumer invasion even after crossing the transcritical threshold. In the Hopf regime, initial conditions are sampled near the stable coexistence equilibrium at the ramp start, ensuring trajectories begin on the equilibrium branch that later destabilizes.

We use parameters $r = 4$, $k = 1.7$, $e = 0.5$, $h = 0.15$, and $m = 2$. For transcritical tipping, we generate trajectories by ramping a from below to above $a \approx 5.60$, enabling the transition from a consumer-free to a coexistence state. For Hopf tipping, trajectories are generated by ramping a from 12 to 17, thereby crossing the Hopf threshold, while non-tipping trajectories are capped below the respective bifurcation points. Tipping is identified either as successful consumer invasion (transcritical) or as the transition from equilibrium dynamics to persistent oscillations (Hopf).

b_seirx_tc We consider a stochastic SEIRx model [70, 71, 22] capturing the coupled dynamics of infectious disease transmission and vaccination behaviour. The state is given by $\mathbf{x}(t) = (S(t), E(t), I(t), x(t))^\top$, where S, E, and I denote the susceptible, exposed, and infectious populations, respectively, and $x(t)$ represents the fraction of individuals with provaccine sentiment. The recovered population is given by $R(t) = N - S - E - I$. The dynamics are

$$\frac{dS}{dt} = \mu N(1 - x) - \beta\frac{SI}{N} - \mu S + \sigma_S\,\xi_S(t), \tag{S25}$$

$$\frac{dE}{dt} = \beta\frac{SI}{N} - (\epsilon + \mu)E + \sigma_E\,\xi_E(t), \tag{S26}$$

$$\frac{dI}{dt} = \epsilon E - (\gamma + \mu)I + \sigma_I\,\xi_I(t), \tag{S27}$$

$$\frac{dx}{dt} = \kappa x(1 - x)\big(-\omega(t) + I + \delta(2x - 1)\big) + \sigma_x\,\xi_x(t), \tag{S28}$$

where μ is the birth/death rate, β the transmission rate, ϵ the exposed-to-infectious rate, γ the recovery rate, κ the social learning rate, δ the strength of injunctive social norms, and $\omega(t)$ the perceived risk

of vaccination relative to infection. The noise terms $\xi_S, \xi_E, \xi_I, \xi_x$ are independent Gaussian white noise processes.

As the perceived vaccination risk ω increases, the system undergoes a transcritical bifurcation at $\omega = \delta$, corresponding to a loss of stability of the disease-free, high-vaccination equilibrium. Below the threshold ($\omega < \delta$), the system remains in a regime with high vaccination uptake ($x \approx 1$) and low infection prevalence. Above the threshold ($\omega > \delta$), vaccination sentiment declines, leading to the emergence of endemic infection. This transition from disease-free to endemic dynamics constitutes bifurcation-induced tipping.

We use parameters $N = 100{,}000$, $\mu = 0.02/52$, $\beta = 10.5$, $\epsilon = 0.7$, $\gamma = 0.7$, $\kappa = 0.007$, and $\delta = 50$, corresponding to a typical pediatric infectious disease. Noise amplitudes are $\sigma_S = \sigma_E = \sigma_I = 5$ and $\sigma_x = 5 \times 10^{-4}$. Tipping trajectories are generated by ramping ω from 0 to 100, thereby crossing the bifurcation threshold, while non-tipping trajectories remain below $\omega = \delta$. Tipping is identified as a collapse in vaccination sentiment ($x(t)$ decreasing) accompanied by a resurgence of infection ($I(t)$ increasing).

Non-tipping r-tipping episodes Non-tipping counterparts of r-tipping episodes can be constructed in several ways, for example with a slower forcing rate from the start of the episode, as in the sets used here, or with forcing that follows the tipping episodes up to a later, slower ramp. With the first construction, the two classes may already differ before the forcing ramp; Appendix C.4.1 reports results for the second.

r_saddle_node We consider a stochastic saddle-node normal form with scalar state $x(t) \in \mathbb{R}$ and time-dependent forcing $\lambda(t)$. The dynamics are given by

$$\frac{dx}{dt} = (x + \lambda(t))^2 - 1 + \sqrt{2\sigma_x}\,\xi(t), \tag{S29}$$

where σ_x controls the noise magnitude and $\xi(t)$ denotes Gaussian white noise. The external forcing evolves according to

$$\lambda(t) = \frac{\lambda_{\max}}{2}\left[\tanh\left(\frac{\lambda_{\max}\epsilon t}{2}\right) + 1\right], \tag{S30}$$

where $\lambda_{\max}$ sets the forcing amplitude and ϵ determines the rate of change.

For each fixed λ, the frozen system exhibits the characteristic saddle-node structure with a stable and unstable equilibrium that collide at a critical parameter value. For sufficiently slow forcing (ϵ small), trajectories track the attracting equilibrium branch. However, when the forcing rate exceeds a critical threshold, the system fails to follow this quasi-static equilibrium and undergoes a rapid transition, even though the instantaneous system remains locally stable. This behaviour constitutes rate-induced tipping.

We use $\lambda_{\max} = 3$ and compare a tipping regime with $\epsilon = 1.25$ to a non-tipping regime with $\epsilon = 0.625$, with noise level $\sigma_x = 0.008$. For analysis, we consider the scalar observable $x(t)$ and define tipping as the event $x(t) \geq 0$.

r_bautin We consider a stochastic Bautin system with complex state $z(t) = x(t) + iy(t)$ and time-dependent forcing $\Lambda(t)$. The dynamics follow a shifted Bautin normal form

$$\frac{dz}{dt} = (a + i\omega)\big(z - \Lambda(t)\big) - b\,|z - \Lambda(t)|^2\big(z - \Lambda(t)\big) + |z - \Lambda(t)|^4\big(z - \Lambda(t)\big) + \sigma_z\,\xi_z(t), \tag{S31}$$

where a controls linear growth, ω is the angular frequency, b is the cubic coefficient, σ_z sets the noise amplitude, and $\xi_z(t)$ is complex-valued white noise. The external forcing evolves according to

$$\Lambda(t) = \frac{\Lambda_{\max}}{2}\left[\tanh\left(\frac{\Lambda_{\max} r t}{2}\right) + 1\right], \tag{S32}$$

where $\Lambda_{\max}$ determines the forcing magnitude and r controls the rate of change.

The forcing $\Lambda(t)$ translates the underlying Bautin system through state space. For sufficiently large r, the system fails to track the quasi-static attractor and undergoes rate-induced tipping, even though the instantaneous dynamics remain locally stable. We use parameters $a = 0.1$, $\omega = 3$, $b = 1$, $\sigma_z = 0.2$, and $\Lambda_{\max} = 8$. Tipping runs are generated with $r = 0.10$, while non-tipping runs use $r = 0.05$.

For analysis, we consider the scalar observable

$$\rho(t) = \sqrt{x(t)^2 + y(t)^2}, \tag{S33}$$

and define tipping as the event $\rho(t) \geq 10$.

r_compost_bomb We consider a stochastic compost-bomb model describing thermally driven instability in a coupled temperature–carbon system [60, 61]. The dynamics are given by

$$\mu \frac{dT_s}{dt} = -\lambda(T_s - T_a(t)) + AC_s r_0 e^{(\alpha(T_s - T_{\text{ref}}))} + D_3\,\xi(t), \tag{S34}$$

$$\frac{dC_s}{dt} = \Pi - C_s r_0 e^{(\alpha(T_s - T_{\text{ref}}))}, \tag{S35}$$

where $T_s(t)$ denotes soil temperature and $C_s(t)$ the soil carbon content. The external forcing is given by a linearly increasing atmospheric temperature

$$T_a(t) = vt, \tag{S36}$$

where v controls the rate of forcing. The nonlinear term

$$r(T_s) = r_0 e^{(\alpha(T_s - T_{\text{ref}}))}$$

represents temperature-dependent microbial respiration, giving rise to a positive feedback loop in which increasing temperature accelerates respiration, releasing heat and further increasing temperature.

For sufficiently small forcing rates v, the system tracks a quasi-static equilibrium corresponding to stable temperature–carbon states. However, when v exceeds a critical threshold, the system fails to track this equilibrium branch and undergoes rapid thermal runaway, despite remaining locally stable at each instantaneous forcing level. This behaviour constitutes rate-induced tipping.

We use parameters $\mu = 2.5 \times 10^6$, $\lambda = 5.049 \times 10^6$, $A = 3.9 \times 10^7$, $\Pi = 1.055$, $r_0 = 0.01$, and $\alpha = \log(2.5)/10$, with noise magnitude $D_3 = 50$. Tipping and non-tipping regimes are generated by varying the forcing rate v. Tipping is identified when $T_s > 30$, indicating entry into the runaway regime.

b_amoc and r_amoc We consider a reduced Atlantic Meridional Overturning Circulation (AMOC) box model derived from the five-box formulation in [7, 59]. By treating the Southern Ocean and bottom water salinities as slow variables and enforcing salt conservation, the system reduces to a two-dimensional stochastic dynamical system with state $\mathbf{x}(t) = (S_N(t), S_T(t))^\top \in \mathbb{R}^2$, where S_N and S_T denote North Atlantic and tropical Atlantic salinity, respectively. The AMOC strength is diagnosed as

$$Q = \frac{\lambda\left[\alpha(T_S - T_0) + \beta(S_N - S_S)\right]}{1 + \lambda\alpha\mu}, \tag{S37}$$

and serves as the observable for tipping detection.

The salinity dynamics are piecewise-defined depending on the sign of Q. For $Q \geq 0$,

$$V_N \frac{dS_N}{dt} = Q(S_T - S_N) + K_N(S_T - S_N) - F_N(H(t))S_0 + \sigma_N \xi_N(t), \tag{S38}$$

$$V_T \frac{dS_T}{dt} = Q\big(\gamma S_S + (1-\gamma)S_{IP} - S_T\big) + K_S(S_S - S_T) \tag{S39}$$

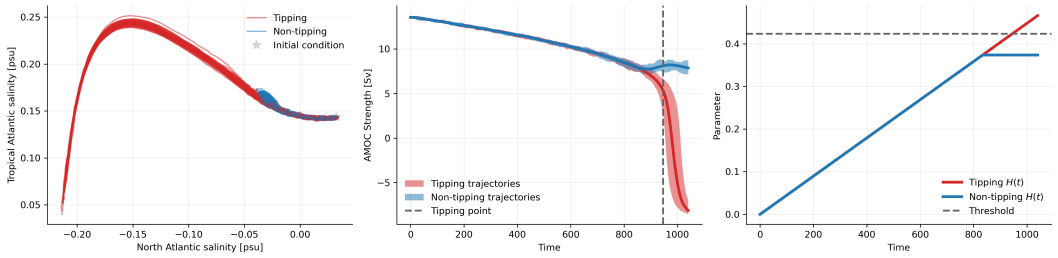
$$+ K_N(S_N - S_T) - F_T(H(t))S_0 + \sigma_T \xi_T(t), \tag{S40}$$

while for $Q < 0$,

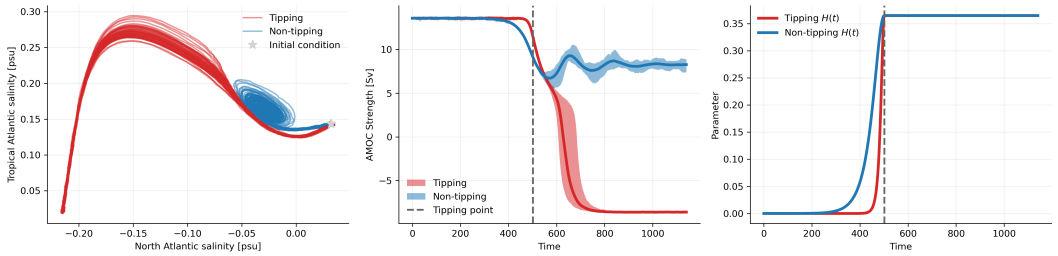
$$V_N \frac{dS_N}{dt} = |Q|(S_B - S_N) + K_N(S_T - S_N) - F_N(H(t))S_0 + \sigma_N \xi_N(t), \tag{S41}$$

$$V_T \frac{dS_T}{dt} = |Q|(S_N - S_T) + K_S(S_S - S_T) + K_N(S_N - S_T) - F_T(H(t))S_0 + \sigma_T \xi_T(t). \tag{S42}$$

Here, F_N and F_T denote freshwater fluxes induced by a hosing forcing $H(t)$, and $\xi_N(t), \xi_T(t)$ are independent Gaussian white noise processes with amplitudes $\sigma_N = \sigma_T = 1.0$.



(a) AMOC box-model (B-tipping)



(b) AMOC box-model (R-tipping)

Figure S6: Examples of idealized AMOC box-model trajectories. (**a**) B-tipping and (**b**) R-tipping. Each panel shows **left**: phase-space evolution in (S_T, S_N), **center**: AMOC strength Q, and **right**: the corresponding hosing or control schedule. Note that for B-tipping, we use a more diverse set of forcing schedules than shown here, see Figures S3 and S4, which vary the initial value and slope as well as the critical time.

For quasi-static forcing, the system exhibits b-tipping (fold bifurcation) at a critical freshwater forcing $H^\star \approx 0.4236$, corresponding to the collapse of the overturning circulation [59]. Tipping trajectories are generated by linearly ramping H from 0.0 to 0.4736, crossing the bifurcation threshold, while non-tipping trajectories follow the same ramp but are capped at $H = 0.3736$, remaining on the stable branch.

To isolate r-tipping, we use a transient forcing protocol

$$H(r,t) = \begin{cases} H_0 + \Delta H \,\mathrm{sech}\big(r(t - t_{\mathrm{crit}})\big), & t < t_{\mathrm{crit}}, \\ H_0 + \Delta H, & t \geq t_{\mathrm{crit}}, \end{cases} \tag{S43}$$

such that tipping and non-tipping trajectories differ only through the forcing rate r. For sufficiently large r, the system fails to track the quasi-static equilibrium and undergoes collapse even though the instantaneous system remains locally stable. We use $r = 0.017$ for tipping and $r = 0.005$ for non-tipping, with $t_{\mathrm{crit}} = 500$.

Tipping is identified as a rapid decline in AMOC strength $Q(t)$, indicating a transition from the strong to the weak circulation state.

SWEC-iEEG We analyze continuous intracranial electroencephalography (iEEG) recordings from the SWEC-ETHZ database [62], which comprises 116 expert-annotated seizures across 2,656 hours of multi-channel iEEG from 18 patients with pharmacoresistant epilepsy. Signals were recorded intracranially via strip, grid, and depth electrodes, band-pass filtered between 0.5 and 120 Hz, and digitized at 512 or 1024 Hz depending on the patient. Seizure onset and end times were determined by a board-certified EEG epileptologist. Seizure onset can be interpreted as a bifurcation-induced tipping event: in the Epileptor framework [63], the transition from interictal to ictal dynamics corresponds to a saddle-node bifurcation driven by a slow permittivity variable that modulates neural excitability. The system remains near a stable resting-state equilibrium during interictal periods, and seizure onset occurs when this equilibrium is annihilated through the bifurcation, triggering a rapid transition to high-amplitude ictal oscillations. For evaluation, we extract fixed-length segments preceding each annotated seizure onset as critical trajectories, paired with multiple interictal segments, at least 1h away from any seizure, as non-critical controls.

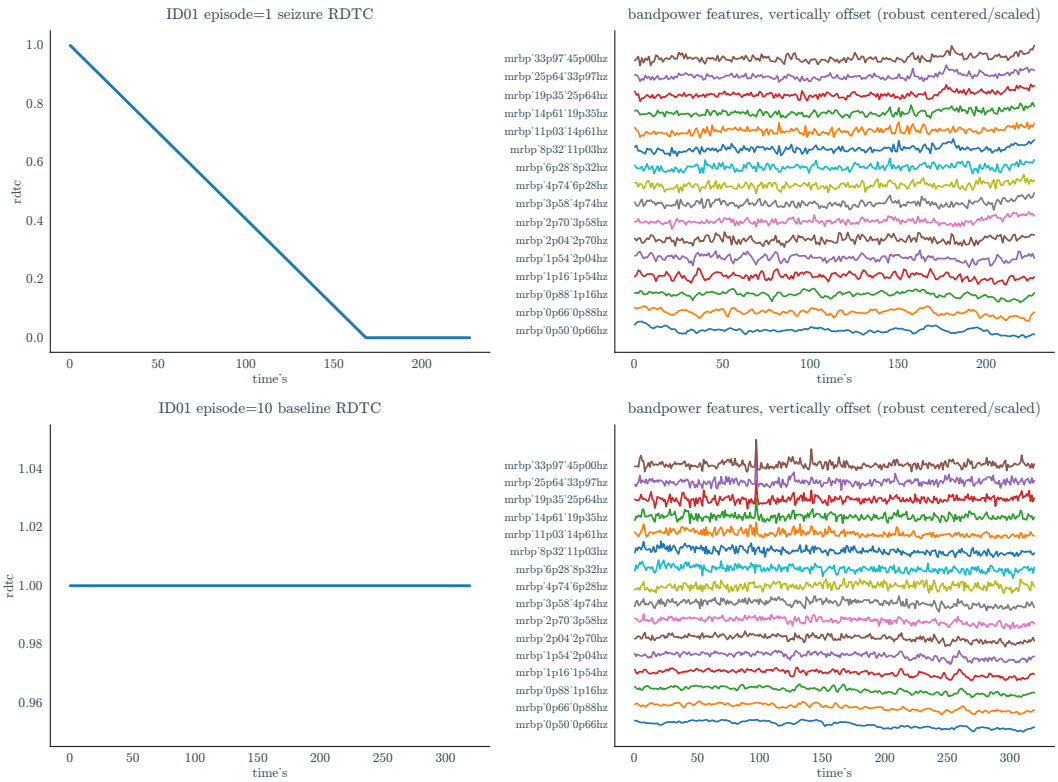


Figure S7: Representative SWEC-iEEG traces around seizure onset and baseline. Right-hand side plot shows vertically stacked sub-rows of the $K{=}16$ multi-resolution log-power features, one band per row, with seizure onset at 168 s. The expert-annotated seizure time is used for the surrogate RDTC definition (top row).

We do not feed raw iEEG into our framework, but a multi-resolution band-power representation that summarises spectral content on a common 1 s grid, see Fig. S7. For each patient we down-sample to $C{=}16$ evenly-spaced electrodes and compute log-power in $K{=}16$ contiguous frequency bands whose edges are geometrically spaced from 0.5 to 45 Hz ($\mathrm{geomspace}(0.5, 45, 17)$). Each band's power is estimated with Welch's method on a causal-trailing window of $\max(2\,\mathrm{s},\ 6/f_{\mathrm{lo}})$ seconds, where f_{lo} is the band's lower edge (so low-frequency bands integrate longer windows for adequate spectral resolution); we cap `nperseg` at 1024 samples and zero-pad the FFT to 2048 to fix the frequency-grid spacing across bands. Band power is then obtained by trapezoidal integration of the PSD over the band edges, $\log_{10}$-transformed, and aggregated across the 16 channels via median. The resulting $K{=}16$-dimensional time series, sampled at 1 Hz, is the input to all methods on SWEC-iEEG.

This dataset provides a challenging real-world benchmark with high dimensionality (36–100 electrodes per patient or representative bandpowers), substantial inter-patient variability, and clinically defined transition times.

TAC We consider a stochastic thermoacoustic system exhibiting a subcritical Hopf bifurcation, following the experimental and modelling study of Bonciolini et al. [45]. The system describes the evolution of acoustic pressure amplitude $A(t)$ in a combustion chamber, where thermoacoustic feedback can lead to large-amplitude oscillations. Near the bifurcation, the dynamics can be approximated by a stochastic normal form

$$\frac{dA}{dt} = \mu(t)A - \alpha A^3 + \beta A^5 + \sigma\xi(t), \tag{S44}$$

where $A(t)$ denotes the oscillation amplitude, $\mu(t)$ is a time-dependent control parameter, $\alpha, \beta > 0$ determine the nonlinear saturation, and $\xi(t)$ is Gaussian white noise. The subcritical nature of the bifurcation implies bistability between a low-amplitude (stable) state and a high-amplitude limit cycle.

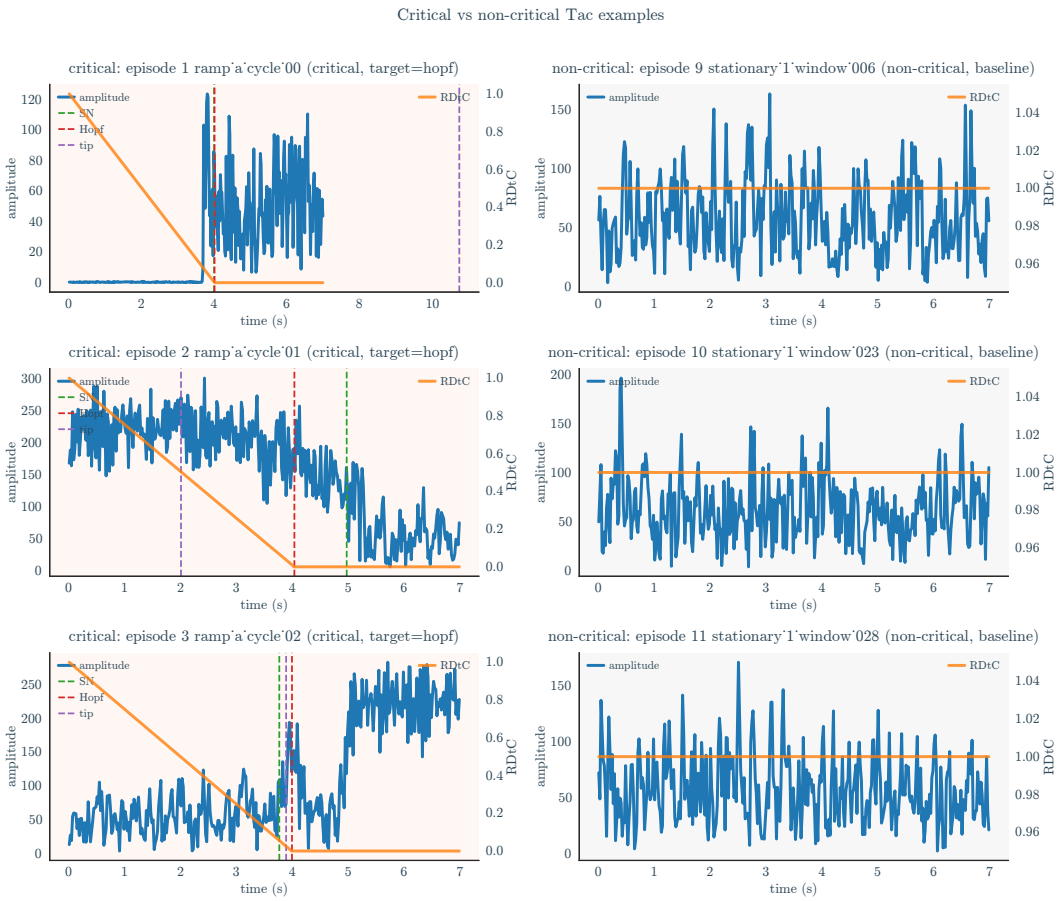


Figure S8: Examples of critical and non-critical `TAC` datasets trajectories. Dashed lines indicate the varios transitions, of which we use the Hopf bifurcation as surrogate RDTC target.

The control parameter $\mu(t)$ is ramped in time, corresponding to changes in operating conditions of the combustor. In the quasi-static case, the system undergoes a bifurcation-induced transition when the stable equilibrium disappears. However, for finite ramping rates, the system exhibits a rate-dependent delay: the transition to large-amplitude oscillations occurs beyond the quasi-static bifurcation point, with the delay increasing as the rate of change increases [45]. This phenomenon leads to dynamic hysteresis when the parameter is ramped forward and backward.

For the TipPFN benchmark, the query trajectories are real experimental episodes extracted from the Bonciolini pressure recordings, rather than simulated test trajectories. We use the Mic1 pressure channel sampled at 2 kHz, extract a 100–200 Hz band-pass Hilbert amplitude envelope, and construct a balanced real-query pool of ramp-up tipping episodes and low-state stationary baseline episodes; see Fig. S8 for example trajectories. The additional three microphone channels are processed analogously and provided as features. We then provide real and simulated context data for up to four context episodes. RDTC is defined with respect to the fitted Hopf crossing; the actual forcing $\mu(t)$ is not provided to the models. This setting combines bifurcation structure, finite-rate forcing, and stochastic transition variability, making it a stringent sim-to-real test of whether TipPFN can use limited context to forecast criticality in noisy laboratory trajectories.

DaphniaExt We use experimental observations of population collapse from controlled microcosm experiments reported in [44]. The system consists of replicate populations of *Daphnia magna* subjected to gradually deteriorating environmental conditions through sustained reductions in food availability. The resulting population dynamics can be interpreted through a minimal stochastic

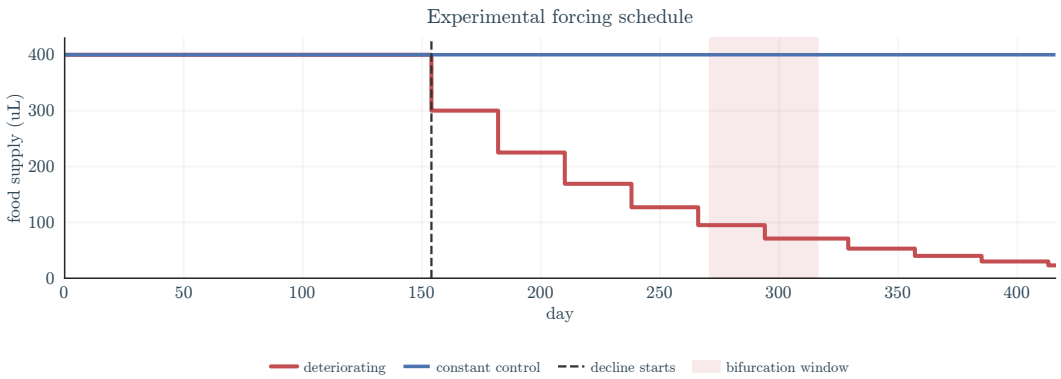


Figure S9: Experimental forcing schedule and estimated bifurcation range [44] for the `DaphniaExt` dataset.

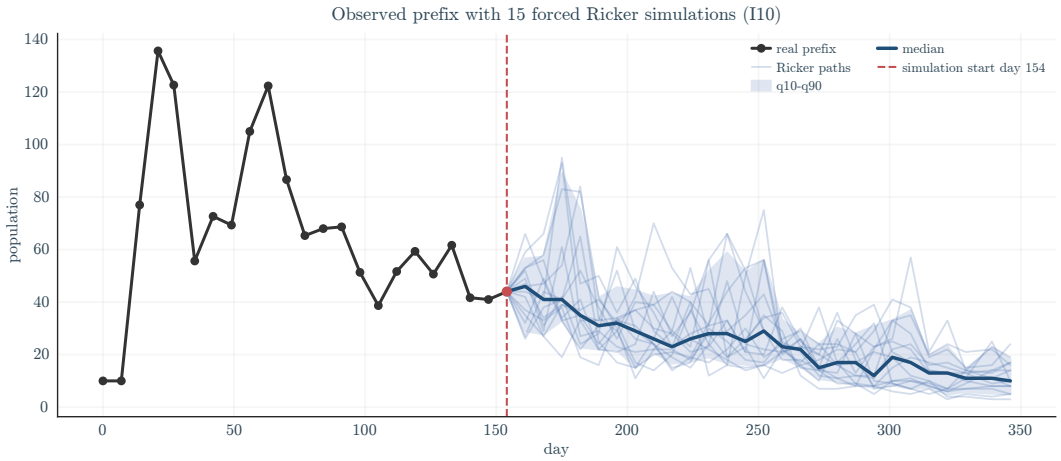


Figure S10: Initial empirical `DaphniaExt` observations, extended with simulated stochastic growth model trajectories which are used for synthetic context trajectories.

growth model with time-dependent parameters,

$$\frac{dN}{dt} = r(t)\,N + \sigma\xi(t), \tag{S45}$$

where $N(t)$ denotes population size, $r(t)$ is a time-varying growth rate reflecting environmental deterioration, and $\xi(t)$ is Gaussian white noise [43].

As environmental conditions worsen, $r(t)$ decreases and crosses zero, corresponding to a transcritical bifurcation in which the stable positive population state loses stability and extinction becomes inevitable. The observed dynamics exhibit a transition from stationary fluctuations around a stable equilibrium to a persistent decline toward extinction. Tipping is identified as the onset of irreversible population decline.

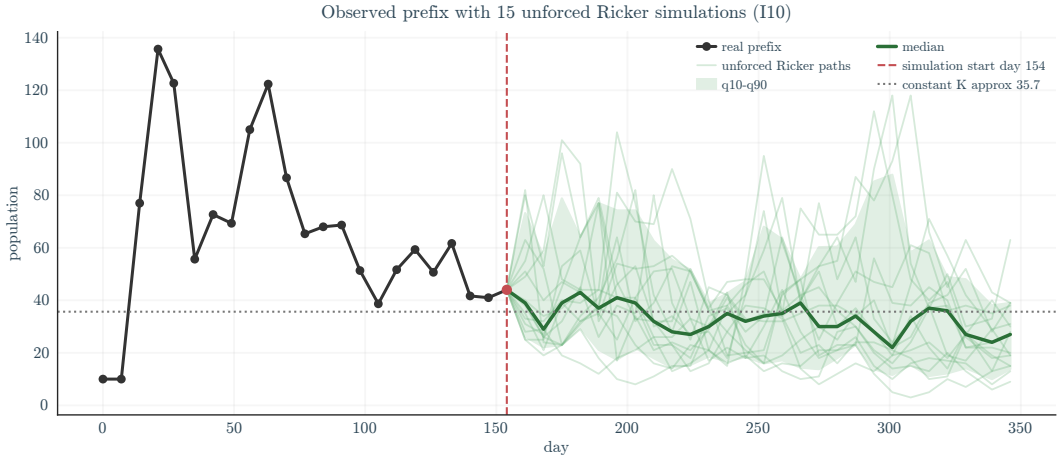


Figure S11: Like above, but without forcing, corresponding to a non-critical trajectory.

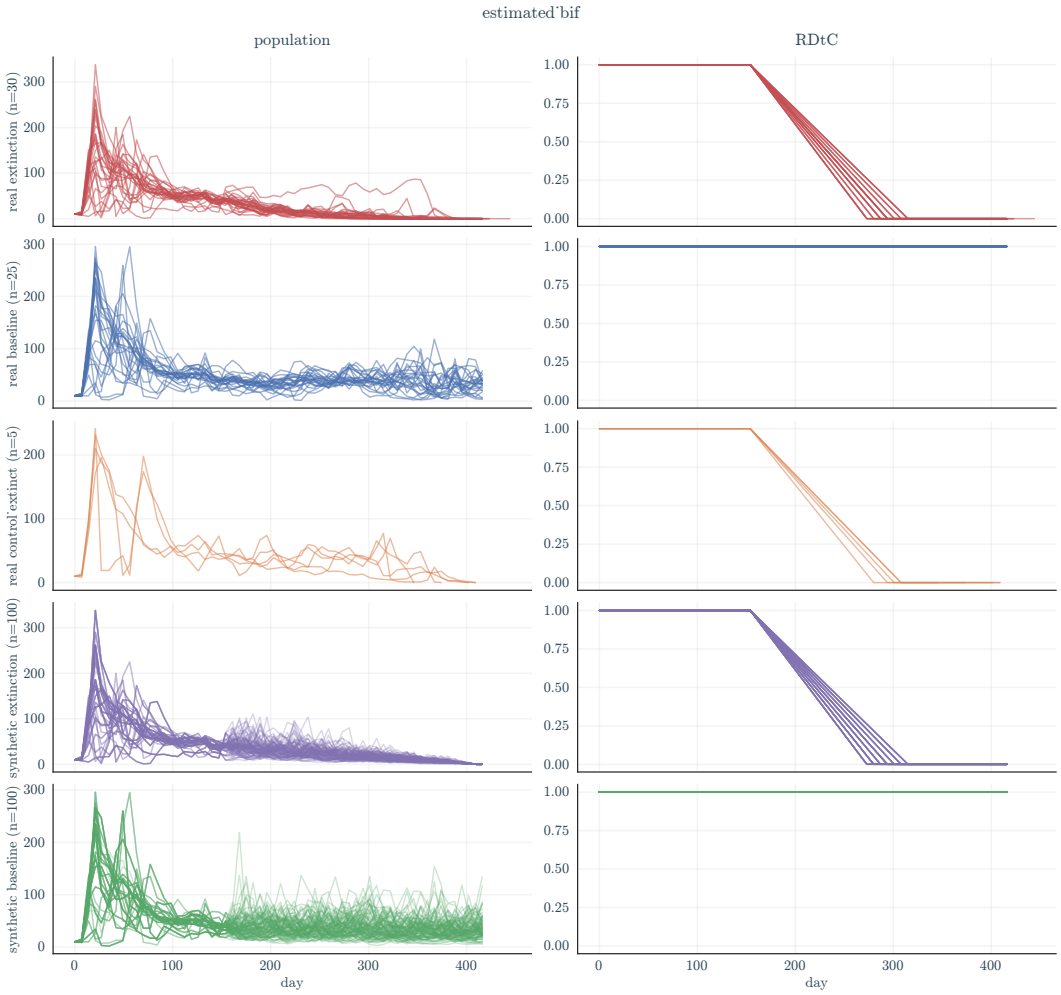


Figure S12: Real and synthetic `DaphniaExt` episodes with corresponding heuristic RDTC ramps, anchored at a random point within the estimated bifurcation range, see Fig. S9.

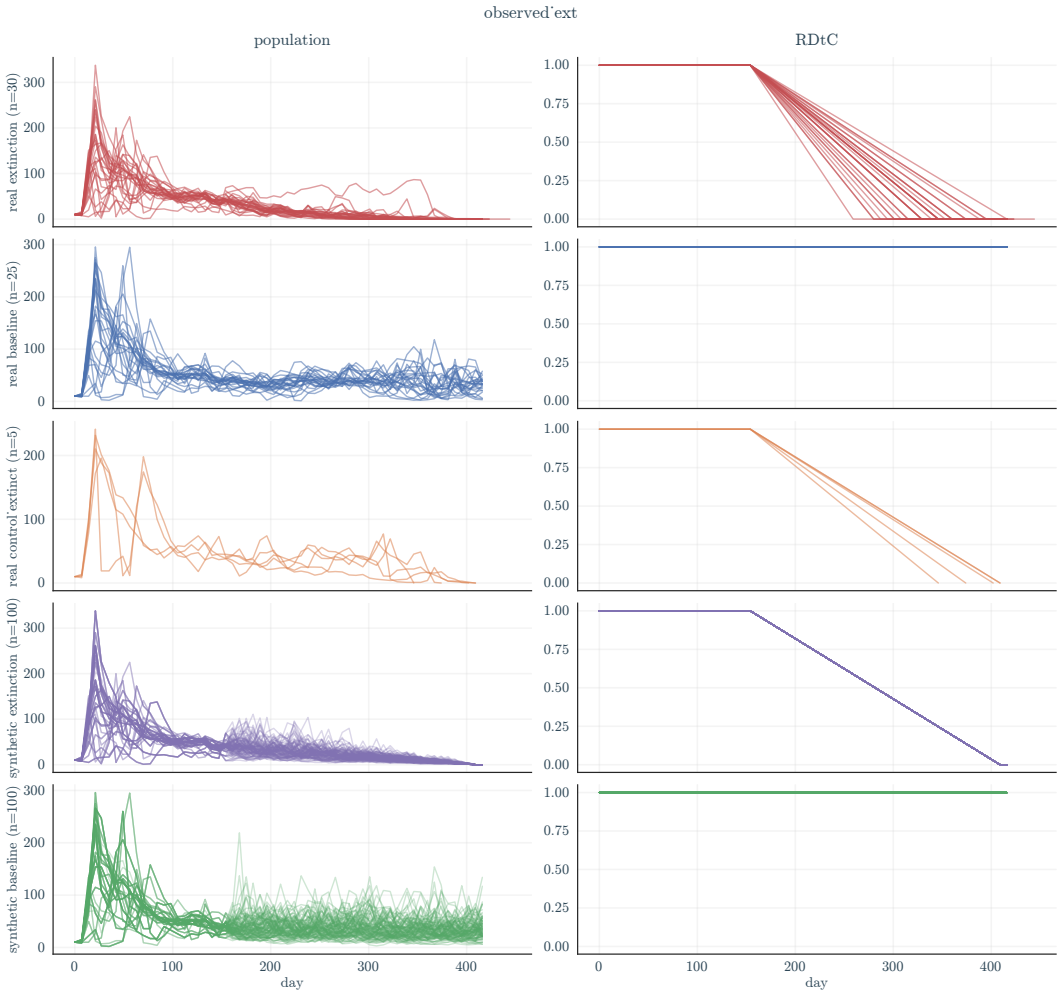


Figure S13: Like above, but with the $\Lambda = 0$ point set to the actual extinction.

microcosm We analyze a cyanobacteria population in chemostats subjected to dilution events under gradually increasing light levels [4], using the data compiled by [35]. Population density is inferred from the light attenuation coefficient, computed from continuous measurements of outgoing light intensity. The full dataset comprises 7,784 observations over 28.86 days at a sampling interval of 5 minutes. The time series is divided into six segments separated by dilution events, which are external perturbations and not part of the intrinsic population dynamics. To ensure consistency, we restrict analysis to recovery phases between dilution events. Specifically, for each segment, we use the final 250 time points (approximately one day) prior to the next dilution event, yielding a total of 1,500 data points across all segments.

voice We analyze experimental time series of phonation onset, corresponding to the emergence of vocal fold oscillations, which can be interpreted as a Hopf bifurcation [64, 65]. The data are obtained from a physical replica of human vocal folds (EPI model) with a multilayer body–cover structure that reproduces realistic tissue mechanics. Oscillations are induced by gradually increasing airflow, while subglottal pressure is recorded using a pressure transducer. The available time series, from the data compiled by [35], consists of 7,501 data points over 0.375 s with a sampling interval of 5×10^{-5} s. Tipping corresponds to the transition from a stable, non-oscillatory state to sustained oscillations as the flow rate crosses a critical threshold. We use the measured pressure signal as the observable for tipping detection.

cellular_atp We analyze experimental time series of cytosolic ATP (i.e., the readily available energy pool) dynamics in living plant cells, measured via a genetically encoded FRET sensor sensitive to $MgATP^{2-}$ [67], using the data compiled by [35]. The data capture the response of leaf tissue under gradually increasing hypoxia, induced by sealing samples in a dark environment where respiration depletes available oxygen. ATP levels remain relatively stable during oxygen decline and then undergo a sudden collapse, indicating a critical transition in cellular energy state. The dataset consists of 271 observations over 3 minutes with a sampling interval of 0.011 minutes. We use the fluorescence ratio signal as the observable, with tipping identified as the abrupt drop in ATP concentration.

greenhouse_earth We analyze a paleoclimate time series of calcium carbonate ($CaCO_3$) associated with the end of greenhouse Earth, marking the transition from an ice-free state to the formation of polar ice caps [12], using the data compiled by [35]. The data exhibit increasing autocorrelation prior to the climate shift, consistent with critical slowing down. The dataset consists of 462 observations spanning 5.9 million years with a sampling interval of 0.013 million years. Tipping is identified as the transition to a glaciated climate state.

blackout_frequency We analyze bus voltage frequency data preceding the Western Interconnect blackout of August 1996, measured within the Bonneville Power Administration network [5], using the data compiled by [35]. The time series captures system dynamics leading up to grid separation and exhibits critical fluctuations prior to the blackout, consistent with EWS of instability. The dataset consists of 11,301 observations over 565 seconds with a sampling interval of 0.05 s. Tipping is identified as the transition to system-wide failure, i.e., blackout.

AMOC The RAPID-MOCHA-WBTS dataset [68] provides continuous measurements of the AMOC at 26.5°N, derived from current velocity, temperature, salinity, and pressure observations collected by a trans-basin mooring array across the Atlantic and cable measurements across the Florida Straits. These variables are used to compute oceanic volume and heat transports in both depth and density space, yielding a 12-hourly, 10-day low-pass filtered transport time series spanning April 2nd 2004 to March 27th 2024.

C.2 Baselines

In addition to the classical indicators such as AR1 and variance, we provide additional details on the comprehensive set of DL-based baselines included in this work.

TabPFN (v2.6) We include TabPFN [27, 28] as a state-of-the-art ICL baseline. TabPFN is a Prior-Data Fitted Network (PFN), a transformer model trained on large amounts of synthetically generated tabular data to approximate Bayesian inference via in-context learning. In our setting,

we adapt TabPFN to time series by representing each trajectory segment as a tabular sample with engineered features (e.g., recent observations or summary statistics), and evaluate its ability to predict RDTC. We evaluate TabPFN's performance for an increasing number of context episodes (1, 2, 3). Zero-context datasets are skipped for TabPFN because the row-wise regressor requires observed target rows.

The context information is transformed into a tabular regression problem suitable for TabPFN in the following way: Let the processed single-sample batch have row index $r = 1, \ldots, R$, identifier columns $\iota_r = (e_r, t_r)$, selected feature vector x_r, action vector $y_r = (y_{r,1}, \ldots, y_{r,A})$, action mask $m_r = (m_{r,1}, \ldots, m_{r,A})$, context indicator c_r, and valid-row indicator v_r. The first action slot is transformed RDTC, $y_{r,1} = \Lambda_r^*$. The row-wise TabPFN feature vector is

$$X_r^{\mathrm{Tab}} = [\iota_r,\ x_r,\ y_{r,2:A},\ m_{r,2:A},\ c_r,\ v_r]\,, \qquad Y_r^{\mathrm{Tab}} = y_{r,1}. \tag{S46}$$

The supervised fit set and prediction set are

$$\mathcal{T} = \{r : v_r = 1,\ m_{r,1} = 0,\ Y_r^{\mathrm{Tab}} \text{ finite}\}, \qquad \mathcal{Q} = \{r : v_r = 1,\ c_r = 0,\ m_{r,1} = 1\}. \tag{S47}$$

Non-finite entries in X^{Tab} are passed as missing values. TabPFN is fit separately for each query episode on $\mathcal{T}$ and predicts only at $\mathcal{Q}$. The model is evaluated using default parameters. It returns logits $\ell_{q,j}$ over histogram bins and bin borders $b_0, \ldots, b_B$. With $p_{q,j} = \mathrm{softmax}(\ell_q)_j$, $C_{q,j} = \sum_{k=1}^{j} p_{q,k}$, and $C_{q,0} = 0$, the inverse-CDF quantile is

$$\hat{Q}_q(\alpha) = b_{j-1} + \frac{\alpha - C_{q,j-1}}{\max(p_{q,j}, 10^{-12})}(b_j - b_{j-1}), \qquad j = \min\{k : C_{q,k} \geq \alpha\}. \tag{S48}$$

Bury We include the deep learning approach of [22], which uses a CNN-LSTM architecture trained on simulated time series from canonical dynamical systems (Hopf, transcritial and fold). The model predicts the probability of an upcoming tipping point directly from observed trajectories in a single forward pass, without relying on handcrafted indicators. As it is trained on bifurcation normal forms, the method is primarily designed for b-tipping and has previously not been generalize to r- or n-induced transitions.

Zhuge We include the deep learning approach of [24], which builds on [22] using a CNN-LSTM architecture trained on simulated time series from canonical bifurcation normal forms. The model learns a direct mapping from observed trajectories (and control parameters) to the predicted tipping point, and can handle irregularly sampled data. As it is trained on normal-form dynamics, the method (like [22]) is primarily designed for B-tipping and relies on shared features of local bifurcation structure.

Huang We include the deep learning approach of [18], which develops a CNN-based classifier for predicting R-tipping under time-varying forcing and stochastic perturbations. Unlike Bury and Zhuge, this method is explicitly designed for r-tipping and predicts the probability that an observed trajectory will tip, rather than relying on CSD indicators. The original study trains and evaluates separate models on three prototypical R-tipping systems: a saddle-node system, a Bautin system, and a compost-bomb model. It further considers extrapolation to out-of-sample forcing rates, showing that models trained at one forcing rate can generalize to unseen rates in some systems, while performance is system-dependent. In our experiments, we use this method in a stronger cross-system setting: the model is trained only on the saddle-node r-tipping system and then evaluated on other R-tipping systems. This differs from the original setup, where separate system-specific training was used. We adopt this protocol because the publicly released code and data [57] primarily provide the saddle-node training workflow, limiting direct replication of the full multi-system training procedure for our benchmark.

Comparison of the learned methods Table S9 summarizes how the learned methods differ in training data, task and inference setting.

C.3 Metrics

We describe the metrics used in this work. For tipping point detection, we consider a binary classification setting where each sample is labeled as either tipping ($y = 1$) or non-tipping ($y = 0$).

Table S9: **Comparison of the learned methods.** *Inference setting* describes the use in this work; in Table 1 all methods receive the same query window of at most 128 steps, which Bury, Zhuge and Huang bring to their native input length. Parameter counts are learned parameters per network, read from the checkpoints.

	Bury [22]	Zhuge [24]	Huang [18]	TabPFN 2.6 [28]	TipPFN
Driver construction	Random 2-D polynomial systems, bifurcation parameter increased towards a fold, Hopf or transcritical point; null runs	Bifurcation normal forms with a varying control parameter	Saddle-node system with forcing ramped at different rates	Generic synthetic tabular prior; no time series	Random 2-D polynomial systems as in Bury (fold, Hopf, transcritical); Bezier, ramp or constant forcing (App. B.1, B.2)
System embedding	None	None	None	–	14-variable random SDE network driven by the driver through external inputs (App. B.3)
Observation model	One state variable, detrended	One state variable and the control parameter, irregular sampling	One state variable	Table rows of per-time-step features (Eq. (S46))	Random subset of 1–16 driver and auxiliary variables, resampled in time
Task	Classification	Regression	Binary classification	In-context regression	In-context distributional regression
Target	Upcoming bifurcation type or none	Control value at the transition	Tipping vs. no tipping	Transformed RDTC	Quantiles of transformed RDTC at every time step
Inference setting	Univariate, 500 steps; no context; mean of 6 networks	Univariate plus forcing parameter (privileged), 500 steps; no context; mean of 10 networks	Univariate, 680 steps; no context; one network	Fitted per query episode on 1–2 context episodes	Up to 16 features, $\leq$128 steps; 0–2 context episodes; one forward pass
Parameters	23,334	41,721	915,074	12.9M	210.7M

Let $\hat{s} : \mathcal{D} \to \mathbb{R}$ denote a scoring function that assigns a confidence score to each observed trajectory, where higher values indicate a higher likelihood of an impending tipping event. A prediction $\hat{y} = 1$ is made whenever $\hat{s} > \tau$ for a threshold τ.

For each threshold τ, we define the true positive rate (TPR) and false positive rate (FPR) as

$$\mathrm{TPR}(\tau) = \frac{|\{i : y_i = 1,\ \hat{s}_i > \tau\}|}{|\{i : y_i = 1\}|}, \quad \mathrm{FPR}(\tau) = \frac{|\{i : y_i = 0,\ \hat{s}_i > \tau\}|}{|\{i : y_i = 0\}|}. \tag{S49}$$

The receiver operating characteristic (ROC) curve [39, 38] is obtained by plotting $\mathrm{TPR}(\tau)$ against $\mathrm{FPR}(\tau)$ as the threshold τ varies over all possible values. The ROC curve characterizes the trade-off between correctly detecting tipping events and incorrectly classifying non-tipping trajectories.

The Area Under the ROC Curve (AUROC) [40, 41] is defined as

$$\mathrm{AUROC} = \int_0^1 \mathrm{TPR}\big(\mathrm{FPR}^{-1}(u)\big)\, du, \tag{S50}$$

which corresponds to the probability that a randomly chosen tipping trajectory is assigned a higher score than a randomly chosen non-tipping trajectory. An AUROC of 0.5 indicates random performance, while 1.0 corresponds to perfect discrimination.

C.3.1 Scores

Table S10 gives an overview of computed scores for each model and baseline.

Table S10: Score definitions across all methods. TipPFN and TabPFN expose six heads per scope: two distributional summaries of the predicted RDtC quantile distribution (`1-mean`, `1-median`) and four CDF probabilities at thresholds $\tau \in \{0.05, 0.1, 0.2, 0.3\}$. The model emits RDtC in $\tanh$-bounded coordinates; raw thresholds τ in physical RDtC units are mapped by $\tilde{\tau} = \tanh(\tau/0.2)$. Both scopes use the last query position in their respective scope (forecast horizon aggregation if a horizon row is present). Baseline score names follow `<method>:<short_mode>:<scope>:<head>` where `<scope>` is `w`N for a fixed query window of length N or `whist` for the whole available history; `<short_mode>` $\in \{$`pad`, `backfill`, `resample`$\}$ controls how short signals are extended to the model's input length. EWS does not consume `short_mode`; Zhuge additionally carries a `<control_source>` $\in \{$`param`, `rdtc`$\}$ slot. Higher score $\Rightarrow$ closer to tipping for every entry except where noted.

Score name	Definition	Notes
TipPFN, TabPFN — nowcast scope (last nowcast query position)		
`<m>:nc:1-median`	$1 - \mathrm{median}(Q_{\mathrm{nc}})$	Q_{nc}: predicted RDtC quantile distribution at the last nowcast query
`<m>:nc:1-mean`	$1 - \mathbb{E}[Q_{\mathrm{nc}}]$	Trapezoidal integration of the predicted quantiles over $p \in [0.01, 0.99]$, without the tail mass
`<m>:nc:P(rdtc<`τ`)`	$\Pr(Q_{\mathrm{nc}} < \tilde{\tau})$	$\tilde{\tau} = \tanh(\tau/0.2)$, $\tau \in \{0.05, 0.1, 0.2, 0.3\}$
TipPFN, TabPFN — forecast scope (forecast query position(s))		
`<m>:fc:1-median`	$1 - \mathrm{median}(Q_{\mathrm{fc}})$	Last forecast position; multi-horizon runs aggregate per-horizon scores
`<m>:fc:1-mean`	$1 - \mathbb{E}[Q_{\mathrm{fc}}]$	As above; same scope rules as `fc:1-median`
`<m>:fc:P(rdtc<`τ`)`	$\Pr(Q_{\mathrm{fc}} < \tilde{\tau})$	$\tilde{\tau}$, τ as in the nowcast row
EWS[12, 9] — rolling-window indicator with Kendall-τ trend test		
`dews:w`N`:var_tau`	$\tau\big(\mathrm{Var}_w(x_t)\big)$	Kendall-τ of rolling variance over window of length N
`dews:w`N`:ar1_tau`	$\tau\big(\mathrm{AR}(1)_w(x_t)\big)$	Kendall-τ of rolling lag-1 autocorrelation
`dews:w`N`:acf_tau`	$\tau\big(\mathrm{ACF}_w(x_t)\big)$	Kendall-τ of rolling autocorrelation function (lag 1, window)
`dews:w`N`:skw_tau`	$\tau\big(\mathrm{Skew}_w(x_t)\big)$	Kendall-τ of rolling skewness
`dews:w`N`:lambd_tau`	$\tau\big(\hat{\lambda}_w(x_t)\big)$	Kendall-τ of rolling local return rate $\hat{\lambda}$
Bury [22] — CNN classifier over {fold, hopf, transcritical, null}		
`bury:<sm>:w`N`:prob_tip`	$\Pr(\mathrm{tip}) = 1 - \Pr(\mathrm{null})$	Sum of fold + hopf + transcritical class probabilities
`bury:<sm>:w`N`:prob_fold`	$\Pr(\mathrm{fold})$	Per-class softmax output
`bury:<sm>:w`N`:prob_hopf`	$\Pr(\mathrm{hopf})$	Per-class softmax output
`bury:<sm>:w`N`:prob_transcritical`	$\Pr(\mathrm{transcritical})$	Per-class softmax output
Huang [18] — CNN classifier (tip vs. no-tip)		
`huang:<sm>:w`N`:prob_tip`	$\Pr(\mathrm{tip})$	Single-output sigmoid probability of an upcoming tipping event
Zhuge [24] — neural regressor for the tipping bifurcation parameter		
`zhuge:<cs>:<sm>:w`N`:signed_tip_margin`	$\Delta^{\pm}_{\mathrm{tip}} = -\hat{p}_{\mathrm{rem}}/\lvert p_{\Delta}\rvert$	Normalized signed margin to the bifurcation parameter; identical for `<cs>` $\in \{$`param`, `rdtc`$\}$

C.4 Further Results

In this section, we provide additional information to the conducted experiments and their results.

C.4.1 Auxiliary variables driven by the state variables only

In the training set and in the synthetic evaluation sets, the 14 auxiliary variables are driven by the state variables x, y of the canonical driver, by its forcing parameter λ and by the RDTC Λ (Appendix B.3). They therefore carry a noisy, filtered readout of the forcing level: a linear regression on the auxiliary variables of one sample recovers Λ with a median R^2 of .93 on the semi-real b-tipping set and .96 on the canonical set, compared with .18 and .57 from x and y. Methods that observe the auxiliary variables, TipPFN and TabPFN, thus receive information about the forcing that the univariate baselines do not, as Zhuge does through its forcing input.

To quantify the effect, we regenerated the three synthetic evaluation sets with the auxiliary variables driven by x and y only, keeping systems, parameters, seeds and forcing schedules, and repeated the evaluation of Table 1 with the released model (Table S11). Averaged over all datasets, TipPFN drops from .661/.782/.838 to .586/.606/.644 (0/1/2 context episodes) and TabPFN from .523/.667 to .500/.535, while the baselines, which observe only x, change by at most .03. TipPFN with two context episodes remains the best column overall and on the three canonical systems, but its lead over the best baseline shrinks from .23 to .04, and baselines score higher on all five semi-real b-tipping systems and on the four r-tipping systems. In the regenerated r-tipping set, the non-critical episodes also follow the critical forcing profile until their ramp onset, unlike the schedules of Appendix B.7, so these rows differ from Table 1 in both the auxiliary drive and the forcing; with non-critical windows placed at the same position relative to their ramp onset, the averages over all datasets are .638/.614/.651 for TipPFN and .644 for Bury. Because the released model was trained on data with the same wiring, these numbers measure how much it relies on the forcing information in the auxiliary variables, not the accuracy a model trained without it would reach.

Table S11: **AUROC with auxiliary variables driven by the state variables only.** Balanced AUROC averaged over $\Delta > 0$ on the synthetic evaluation sets regenerated with the auxiliary variables driven by x and y only, with the protocol, model and column selection of Table 1 (EWS now selects $W = 128$ instead of $W = 64$). The real-world rows are unchanged (EWS at $W = 128$: SWEC-iEEG .537, TAC .568) and enter the *All datasets* row. The r-tipping rows use the regenerated forcing design described in the text. Best value per row in bold.

System	TipPFN 0c	TipPFN 1c	TipPFN 2c	TabPFN 1c	TabPFN 2c	EWS τ_{var}	Bury [22] P_{tip}	Huang [18] P_{tip}	Zhuge [24] $\Delta^{\pm}_{\text{tip}}$
Canonical									
B-Fold	.862	.863	**.902**	.509	.569	.655	.704	.452	.524
B-Hopf	.823	.861	**.934**	.568	.685	.769	.818	.484	.645
B-Trans.	.709	.716	**.744**	.508	.599	.639	.721	.461	.453
Semi-real									
B-Harv.	.837	.862	.922	.619	.667	**.991**	.967	.874	.734
B-RM TC	.630	.550	.644	.411	.434	.732	**.789**	.615	.573
B-RM Hopf	.570	.533	.565	.414	.406	.616	**.641**	.486	.422
B-SEIRx	.476	.500	.508	.506	.506	**.540**	.473	.508	.377
B-AMOC	.869	.893	.934	.619	.829	.935	**.948**	.509	.679
R-Bautin	.334	.416	.410	.453	.411	.339	.458	**.580**	.208
R-SN	.299	.382	.368	.442	.393	.393	.291	**.615**	.100
R-Compost	.542	.424	.448	.514	.517	.477	**.573**	.433	.360
R-AMOC	.231	.407	.413	.416	.389	.251	.249	**.589**	.147
All datasets	.586	.606	**.644**	.500	.535	.599	.609	.535	.435
All datasets, Table 1	.670	.779	**.838**	.522	.666	.572	.604	.534	.443

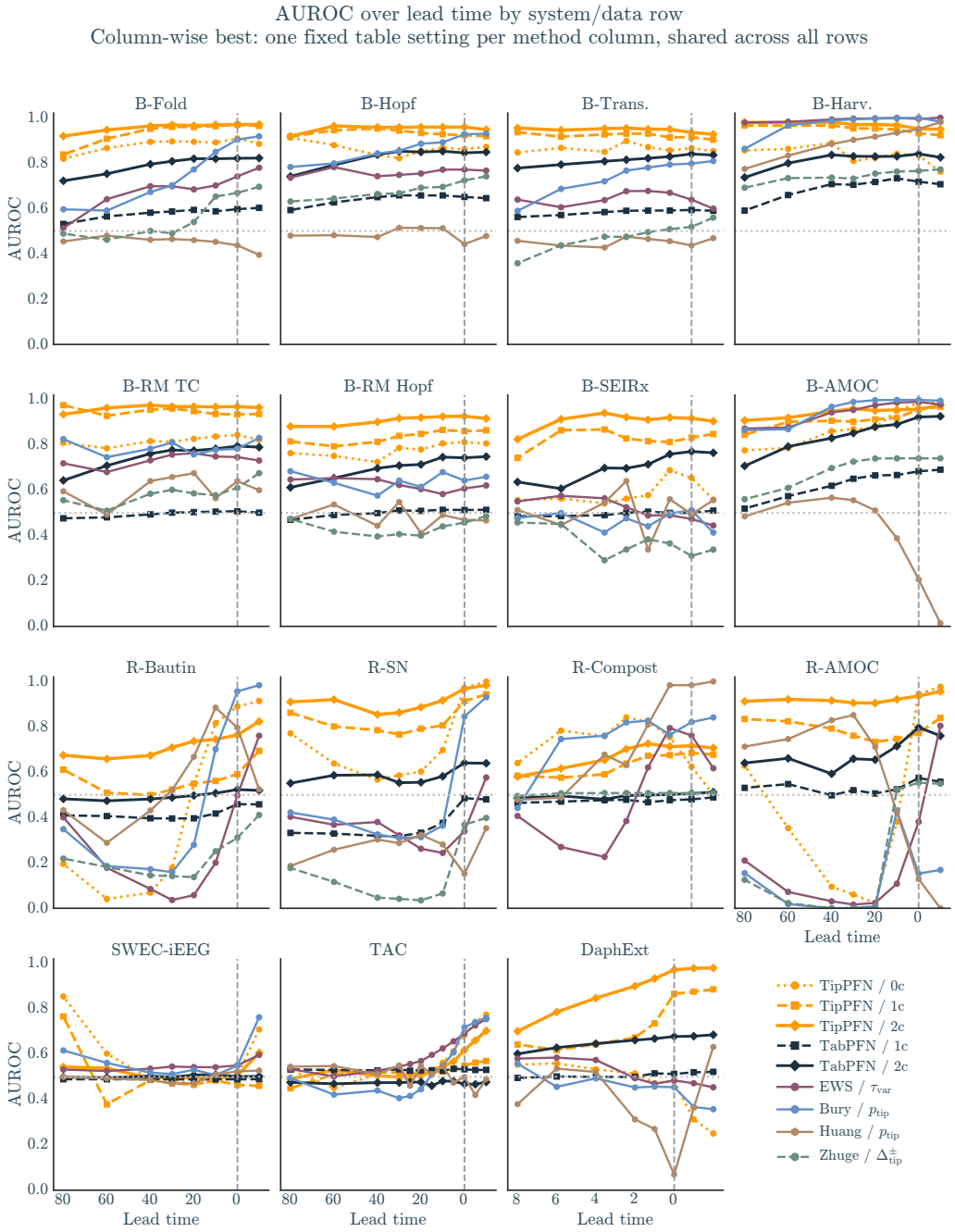


Figure S14: **AUROC over lead time for all systems**, selecting each model's score variant that performed best across all these systems (same as in Table 1 for $\Delta > 0$.

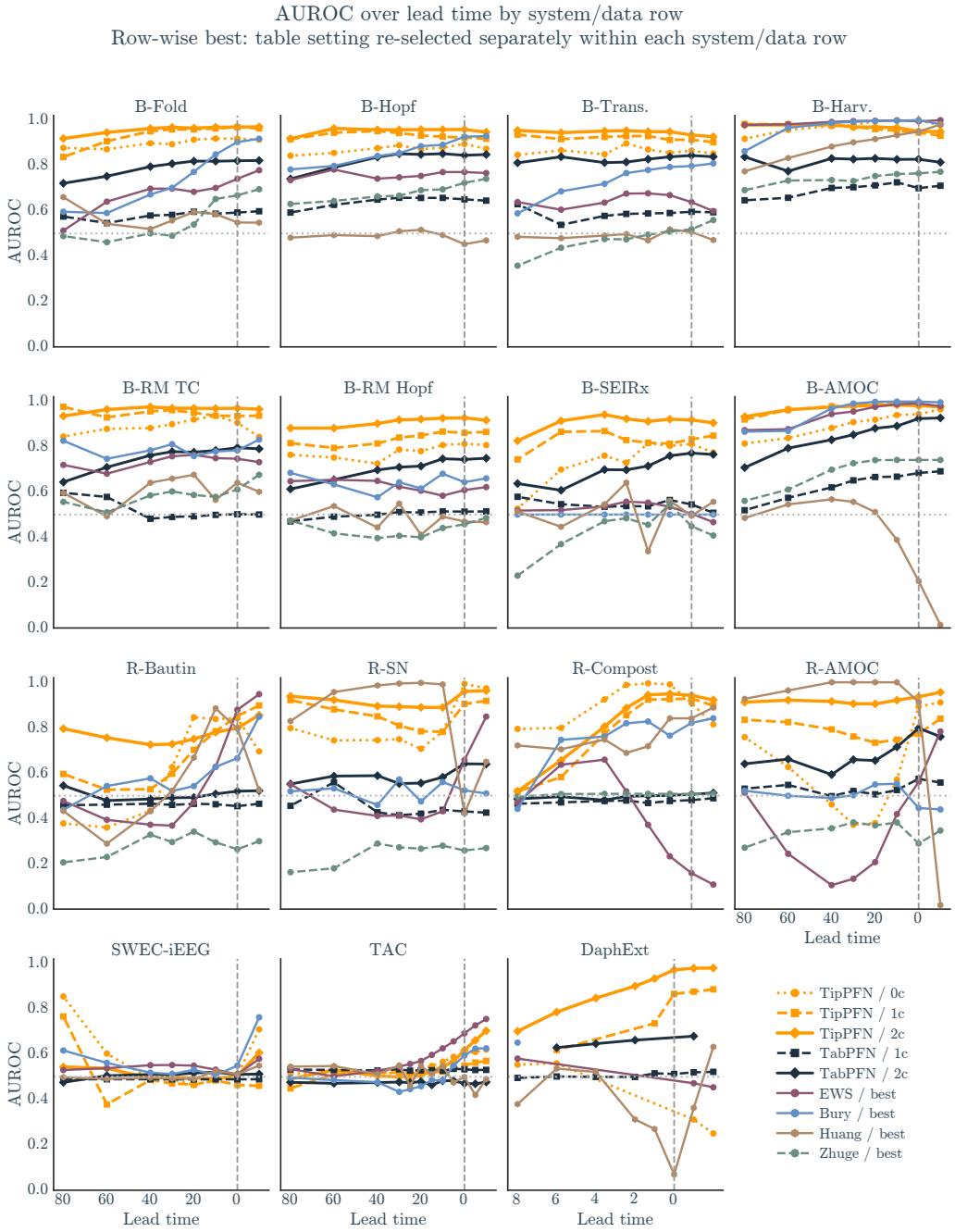


Figure S15: **AUROC over lead time for all systems**, selecting each model's score variant that performed best on that particular system (corresponding to Table S12) for $\Delta > 0$.

C.4.2 Held-out configuration selection

Each column of Table 1 uses the configuration (query-window length and, for the ML baselines, the length adaptation) with the highest mean AUROC over all rows, and Table S12 selects a configuration, including the TipPFN and TabPFN score head, separately for each row. Because the configuration is selected on the data on which it is reported, the reported values can be optimistic. To estimate this effect, we repeat the selection on held-out episodes. For each of $R = 200$ repeats, we split the evaluation episodes of each system or dataset, stratified by class, into a 20% selection fold and an 80% report fold. An episode is one source trajectory and is assigned to one fold as a whole, so all its lead times and query windows stay on the same side. Episodes of one synthetic sample share the system draw and serve as each other's context, so the folds are not independent at the level of system draws. The SWEC-iEEG row pools all patients, so its episodes are split over all patients rather than per patient. On the selection fold, each column (column-wise) or each cell (row-wise) selects its configuration with the rule of the published table; the selected configuration is then evaluated on the report fold with the balanced AUROC averaged over $\Delta > 0$. All configurations are scored from the existing per-query scores; no model is re-run. Table S13 reports means over the 12 non-canonical rows (9 for Zhuge, which has no real-world rows); the canonical systems share their bifurcation types with the training prior and are left out.

With column-wise selection, the held-out means differ from the post hoc values by at most .02, and TipPFN with two context episodes is unchanged (.809). Including the canonical systems, its held-out mean is .838 [.832, .844], equal to Table 1. Row-wise selection is more optimistic, by up to .045 for TabPFN and .017 for TipPFN with two context episodes, because each cell chooses among several configurations on few episodes. In both variants, TipPFN with two context episodes remains the best method, .18 (column-wise) and .19 (row-wise) above the best other method (TabPFN with two context episodes and Bury, respectively).

The selected configuration is stable for TipPFN with one and two context episodes (128-step window in 191 and 196 of 200 repeats). Without context, the selection folds pick the 96- or 64-step window in 195 of 200 repeats, whereas all data give the 128-step window. The held-out mean over the non-canonical rows is then higher (.641 against .610).

C.4.3 Native-window baselines

Table 1 restricts every method to query windows of at most 128 steps, although the ML baselines were designed for longer inputs. For a comparison at each method's native window, we re-simulated the canonical and semi-real b-tipping validation systems with 1024 steps per episode. The baselines run at full resolution with their native windows (Bury and Zhuge $W = 500$, Huang $W = 680$, TabPFN $W = 768$ with two context episodes; EWS uses the variance trend over the available history), while TipPFN observes every fourth step with $W = 128$, a comparable span of 512 steps. Lead times are matched in time: Δ TipPFN steps correspond to 4Δ full-resolution steps. TipPFN with two context episodes leads on 6 of 8 systems (Table S15); Bury and EWS score higher on B-Harvesting, and Bury on B-AMOC. The TipPFN runs use all 16 input variables, and the re-simulated sets keep the auxiliary-variable wiring of Table 1, so this lead is subject to the same caveat (Appendix C.4.1). R-tipping systems are not included: with the construction of their non-tipping episodes used in these sets (Appendix C.1), the two classes may already differ before the forcing ramp; Appendix C.4.1 reports r-tipping results for the second construction.

Real-world data On SWEC-iEEG and TAC, all methods read the same time grid (1 s and 0.02 s), so lead times need no conversion, and all methods are scored on the same critical episodes and lead times. For SWEC-iEEG, we use a version of the rate-corrected set with an 872 s look-back before each of the 116 seizures and 529 interictal segments, so every native window fits at all lead times $\Delta \in \{10, \ldots, 80\}$ s (three seeds; 345–348 critical and 1,581 non-critical windows per lead time); its TipPFN values therefore differ from Table 1. The univariate baselines are averaged over three band-power channels. The TAC episodes of Table 1 hold only about 200 steps before the transition. We re-extracted the critical cycles from the raw recordings with a longer segment; the usable history before a cycle ends at the oscillating state of the previous cycle (median 670 steps). Of the 80 cycles of Table 1, eight at the start of a recording and one with a contaminated history are dropped, and the remaining 71 have at least 500 clean steps, so Bury, Zhuge and EWS run at $W = 500$ for all lead times $\Delta \in \{5, \ldots, 80\}$ steps (five seeds; 355 critical and 400 stationary windows per lead time).

Table S12: **Row-wise best pre-tip AUROC by system and dataset.** Balanced AUROC for critical-vs-all tipping detection, averaged over positive lead times ($\Delta > 0$). Each cell re-selects the best score/configuration within that system/data row and method column. The small line below each AUROC is the picked configuration: `w<N>` for query-window length, `res` = *resample*, `bf` = *back-fill*, `pad` = *pad*; for TipPFN/TabPFN it also lists the selected score head (e.g. `fc:1-median` or `fc:P(rdtc<0.05)`). Feature budget is fixed at 16 within the TipPFN/TabPFN columns; baseline score heads are those of Table S25, and all score heads are defined in Table S10.
For the `TAC` dataset, higher performance can be achieved by selective context composition, see Fig. S21.

System/data	TipPFN 0c	TipPFN 1c	TipPFN 2c	TabPFN 1c	TabPFN 2c	EWS best	Bury [22] best	Huang [18] best	Zhuge [24] best
Canonical									
B-Fold	.935 w96 / fc:P(rdtc<0.2)	.929 w128 / fc:1-median	**.955** w128 / fc:1-median	.584 w64 / fc:P(rdtc<0.3)	.786 w64 / fc:1-median	.655 w128	.697 res / w128	.577 pad / w128	.522 res / w128
B-Hopf	.893 w128 / fc:P(rdtc<0.1)	.935 w128 / fc:1-median	**.952** w128 / fc:P(rdtc<0.05)	.640 w64 / fc:1-median	.820 w64 / fc:1-median	.755 w128	.842 res / w128	.497 res / w96	.665 res / w128
B-Trans.	.874 w128 / fc:P(rdtc<0.3)	.925 w128 / fc:1-median	**.950** w128 / fc:1-median	.601 w128 / fc:P(rdtc<0.3)	.824 w128 / fc:1-median	.650 w128	.722 res / w128	.490 bf / w64	.458 res / w128
Semi-real									
B-Harv.	.970 w128 / fc:1-median	.971 w128 / fc:1-median	.980 w128 / fc:1-mean	.691 w96 / fc:1-median	.847 w96 / fc:P(rdtc<0.3)	**.991** w128	.967 res / w128	.873 res / w64	.734 res / w128
B-RM TC	.901 w96 / fc:P(rdtc<0.2)	.952 w128 / fc:P(rdtc<0.05)	**.963** w128 / fc:P(rdtc<0.1)	.569 w128 / fc:P(rdtc<0.3)	.741 w64 / fc:1-median	.733 w128	.784 res / w128	.604 res / w64	.570 res / w128
B-RM Hopf	.770 w128 / fc:P(rdtc<0.3)	.845 w128 / fc:P(rdtc<0.1)	**.912** w128 / fc:P(rdtc<0.05)	.529 w128 / fc:P(rdtc<0.3)	.689 w64 / fc:1-median	.627 w128	.638 res / w128	.484 res / w64	.422 res / w128
B-SEIRx	.723 w128 / fc:P(rdtc<0.3)	.831 w128 / fc:1-mean	**.907** w128 / fc:1-mean	.583 w64 / fc:P(rdtc<0.05)	.699 w64 / fc:P(rdtc<0.05)	.536 w96	.500 pad / w128	.507 res / w64	.426 bf / w96
B-AMOC	.906 w96 / fc:P(rdtc<0.3)	.969 w128 / fc:1-median	**.972** w128 / fc:P(rdtc<0.05)	.632 w64 / fc:P(rdtc<0.3)	.829 w64 / fc:P(rdtc<0.3)	.935 w128	.948 res / w128	.509 res / w64	.679 res / w128
R-Bautin	.580 w64 / fc:1-median	.643 w64 / fc:P(rdtc<0.1)	**.776** w64 / fc:P(rdtc<0.05)	.468 w64 / fc:P(rdtc<0.05)	.501 w96 / fc:P(rdtc<0.3)	.451 w64	.541 res / w64	.538 res / w64	.283 bf / w96
R-SN	.754 w64 / fc:1-median	.864 w96 / fc:P(rdtc<0.3)	.931 w128 / fc:P(rdtc<0.3)	.501 w64 / fc:P(rdtc<0.3)	.592 w64 / fc:P(rdtc<0.3)	.440 w64	.520 bf / w96	**.959** pad / w128	.242 bf / w96
R-Compost	**.927** w128 / fc:P(rdtc<0.05)	.763 w128 / fc:1-median	.793 w128 / fc:1-median	.500 w64 / fc:P(rdtc<0.3)	.505 w64 / fc:P(rdtc<0.3)	.481 w64	.727 res / w128	.737 res / w128	.506 res / w128
R-AMOC	.584 w64 / fc:P(rdtc<0.3)	.798 w128 / fc:P(rdtc<0.3)	.924 w128 / fc:P(rdtc<0.05)	.616 w64 / fc:P(rdtc<0.3)	.699 w64 / fc:P(rdtc<0.3)	.270 w64	.519 bf / w96	**.982** bf / w128	.350 bf / w96
Real-world & Sim-to-Real									
SWEC-iEEG	**.580** w128 / fc:P(rdtc<0.2)	.512 w128 / fc:1-mean	.563 w128 / fc:P(rdtc<0.3)	.489 w64 / fc:1-median	.516 w128 / fc:P(rdtc<0.3)	.542 w96	.541 res / w128	.499 res / w128	–
TAC	.536 w96 / fc:P(rdtc<0.05)	.526 w128 / fc:P(rdtc<0.3)	.554 w128 / fc:P(rdtc<0.3)	.533 w64 / fc:P(rdtc<0.1)	.476 w128 / fc:P(rdtc<0.1)	**.568** w128	.479 res / w64	.522 res / w64	–
DaphniaExt	.556 w16 / fc:1-median	.736 w16 / fc:P(rdtc<0.05)	**.833** w16 / fc:1-median	.514 w16 / fc:P(rdtc<0.1)	.672 w8 / fc:1-mean	.580 w12	.651 pad / w12	.404 res / w8	–
All datasets	.766	.813	**.864**	.563	.680	.614	.672	.612	.488

Table S13: **Held-out configuration selection.** Mean balanced AUROC ($\Delta > 0$) over the 12 non-canonical rows (Zhuge: 9 rows). *Post hoc*: configuration selected on all episodes, per column as in Table 1 or per cell as in Table S12. *Held-out*: configuration selected on a 20% episode fold and evaluated on the other 80%; mean over $R = 200$ splits, 5th–95th percentile in brackets.

	Column-wise		Row-wise	
Method	Post hoc	Held-out [p5, p95]	Post hoc	Held-out [p5, p95]
TipPFN 0c	.621	.641 [.631, .649]	.732	.707 [.684, .730]
TipPFN 1c	.741	.740 [.732, .749]	.784	.764 [.750, .777]
TipPFN 2c	**.810**	**.810** [.803, .816]	**.842**	**.825** [.811, .837]
TabPFN 1c	.503	.499 [.482, .515]	.552	.507 [.487, .528]
TabPFN 2c	.631	.627 [.609, .644]	.647	.605 [.583, .628]
EWS τ_{var}	.564	.552 [.537, .569]	.596	.588 [.576, .598]
Bury [22]	.566	.568 [.562, .576]	.651	.635 [.622, .647]
Huang [18]	.549	.549 [.541, .559]	.635	.627 [.612, .649]
Zhuge [24]	.407	.406 [.393, .420]	.468	.451 [.431, .469]

Huang's $W = 680$ fits in 9–33 cycles for $\Delta \leq 25$ and in none for $\Delta \geq 40$; we average it over $\Delta \leq 25$ and report TipPFN on the same items. No cycle has 768 clean steps, so TabPFN runs at $W = 500$ with two context episodes; its context pool also contains 18 cycles that are not in the pool of Table 1. TipPFN is scored on the same 71 cycles. Zhuge takes the control-parameter trajectory as a second input, which the real-world data do not provide. The RDTC and the TAC fuel flow cannot serve as this input, because they ramp only in critical episodes and would reveal the label; we feed elapsed time, which is identical for both classes. DaphniaExt is not included, as its windows have at most 16 steps.

Table S14: Source-trajectory support for the AUROC evaluation. For each system or empirical benchmark row, we report the number of unique source samples and query episodes represented in the result tables, split by real versus synthetic source origin and by critical versus non-critical labels. Counts are deduplicated over lead-time offsets and model-configuration sweeps. Canonical and semi-real systems are synthetic simulation benchmarks by construction.

System/data	samples	episodes	real n_{pos}	real n_{neg}	synth. n_{pos}	synth. n_{neg}	n_{neg} split
Canonical							
B-Fold	41	164	0	0	80	84	19 equilibrium, 15 flat 31 receding, 19 approaching
B-Hopf	46	184	0	0	93	91	18 equilibrium, 21 flat 29 receding, 23 approaching
B-Trans.	41	164	0	0	82	82	28 equilibrium, 17 flat 17 receding, 20 approaching
Semi-real							
B-Harv.	20	80	0	0	40	40	9 equilibrium, 13 flat 7 receding, 11 approaching
B-RM TC	31	124	0	0	62	62	15 equilibrium, 18 flat 16 receding, 13 approaching
B-RM Hopf	33	132	0	0	65	67	17 equilibrium, 16 flat 15 receding, 19 approaching
B-SEIRx	17	68	0	0	35	33	11 equilibrium, 7 flat 5 receding, 10 approaching
B-AMOC	27	108	0	0	54	54	14 equilibrium, 12 flat 13 receding, 15 approaching
R-Bautin	34	136	0	0	66	70	38 equilibrium, 32 flat
R-SN	35	140	0	0	71	69	41 equilibrium, 28 flat
R-Compost	31	124	0	0	63	61	30 equilibrium, 31 flat
R-AMOC	28	112	0	0	56	56	25 equilibrium, 31 flat
Others							
SWEC-iEEG	18	807	116	691	0	0	691 baseline
TAC	2	320	80	80	80	80	80 real baseline, 80 synth. baseline
DaphniaExt	2	110	30	25	30	25	25 real baseline, 25 synth. baseline
Summary							
Total	406	2,773	226	796	877	874	796 real baseline, 105 synth. baseline, 265 synth. equilibrium 241 synth. flat, 133 synth. receding, 130 synth. approaching

No method separates the classes at native windows either (Table S15, bottom). On SWEC-iEEG, all means lie between .455 and .519. On TAC, they lie between .474 and .542, except for Zhuge with the time input (.369, below .5 at every lead time; we do not flip its sign, as that would be fitted on the test data) and Huang on its $\Delta \leq 25$ subset (.574; TipPFN without context reaches .578 on the same items).

Table S15: **Native-window comparison** on re-simulated canonical and semi-real b-tipping systems (1024 steps per episode) and on two real-world datasets. Balanced AUROC averaged over matched lead times. Synthetic rows: $\Delta \in \{10, \ldots, 80\}$ TipPFN steps, i.e. 4Δ full-resolution steps; baselines use their native window W at full resolution, TipPFN uses $W = 128$ on every fourth step. Huang and TabPFN have no feasible critical windows at the longest lead times and are averaged over $\Delta \leq 40$ and $\Delta \leq 30$ (TipPFN 2c mean on these subsets: .959); TabPFN cells rest on 1–12 critical episodes at their largest lead time. Real-world rows: all methods use the recorded time grid, the baselines with the window in the column header and TipPFN with $W = 128$; $\Delta \in \{10, \ldots, 80\}$ s (SWEC-iEEG) and $\Delta \in \{5, \ldots, 80\}$ steps (TAC); Zhuge receives elapsed time as its control input. [†]Huang's output saturates on B-AMOC (all scores below 10^{-14}), so this value is degenerate. [‡]Only $\Delta \leq 25$ (9–33 cycles per lead time); TipPFN 0c/1c/2c on the same items: .578/.521/.559. [§]Run at $W = 500$, since no TAC cycle has 768 clean steps; the context pool also contains 18 cycles outside the pool of Table 1. Best value per synthetic row in bold.

System	Bury W=500	EWS τ_{var} history	Huang W=680	Zhuge W=500	TabPFN 2c W=768	TipPFN 0c	TipPFN 1c	TipPFN 2c
Canonical								
B-Fold	.805	.748	.508	.515	.830	.843	.931	**.949**
B-Hopf	.864	.807	.501	.578	.820	.864	.939	**.973**
B-Trans.	.740	.664	.522	.513	.857	.845	.928	**.946**
Semi-real								
B-Harv.	**.994**	.982	.471	.656	.798	.869	.940	.968
B-RM TC	.846	.879	.505	.409	.900	.787	.952	**.981**
B-RM Hopf	.550	.732	.482	.216	.861	.752	.849	**.895**
B-SEIRx	.511	.559	.503	.394	.801	.596	.831	**.939**
B-AMOC	**.989**	.935	.004[†]	.684	.791	.950	.963	.965
Mean (8 systems)	.787	.788	.437	.496	.832	.813	.917	**.952**
Real-world								
SWEC-iEEG	.459	.509	.473	.499	.513	.455	.501	.519
TAC	.542	.532	.574[‡]	.369	.474[§]	.513	.510	.513

C.4.4 Ablations and sensitivity analysis

This section details the ablations summarized in Sec. 4.4. All of them use the evaluation sets of Table 1, and both arms of every comparison are evaluated on the same data; SWEC-iEEG is not included. Unless stated otherwise, the score is the forecast $1 - \text{median}(\Lambda)$ with $W = 128$ and 16 observed features, averaged over $\Delta > 0$ as in Table 1; family values are unweighted means over systems, and *simulated* denotes the 3 canonical and 9 semi-real systems. The two target ablations need evaluation options that the code behind Table 1 does not have; they were run with later versions of our evaluation code, with the RDTC reference re-evaluated in the same version, so their RDTC family means differ from Table 1 by up to .03.

Training target We compare RDTC with two other training targets, each used to train a model with the architecture of TipPFN on the same prior. The *single-label classifier* predicts one label per episode, whether it eventually tips, read at the final forecast step and trained with a cross-entropy loss. Its training contexts end before the transition (non-tipping contexts at a pseudo-anchor drawn from the transition times), and we truncate the evaluation contexts in the same way; without truncation its context results are lower. The *dense binary target* keeps the loss, masking and context interface of TipPFN and replaces the graded RDTC at every time step by the episode outcome (± 0.95 for tipping and non-tipping episodes), in the query and in the context episodes. It is the closer control: only the information about the distance to the transition is removed, whereas the classifier also changes the loss, the output and the context format.

On identical query rows and context episodes (Table S16), the classifier is close to RDTC without context ($-.031$ on the simulated systems; it is ahead on the canonical systems), but context lowers its AUROC below its own zero-context value, so RDTC leads by .213 and .245 with one and two context episodes. The classifier checkpoint (28k steps, selected on validation data) was trained longer than the released model (22k steps); after 22k steps the classifier reaches .583 without context, against .724 for RDTC. The dense binary target (Table S17) is within .02 of RDTC without context on the

simulated systems, benefits from context, and stays below RDTC by .144–.167 with two context episodes; the graded target is better in every cell except r-tipping without context at $d = 96$. The paper-size binary model was trained for fewer steps (9k) than the released model (22k), which favors RDTC; the $d = 96$ pair (32k binary and 37k RDTC steps) shows the same pattern. On TAC, all four models of Table S17 are near chance (.40–.56); TAC and DaphniaExt are not included in the table. Dense supervision is therefore needed to learn from one or two context episodes, and the graded RDTC target adds to it.

Table S16: **RDTC versus a single-label classifier.** AUROC averaged over $\Delta > 0$ for the released model (RDTC) and a classifier with the same architecture trained on one eventual-tipping label per episode, on identical query rows and context episodes. Classifier contexts are truncated before the transition, as in its training. The last row gives the paired difference with a 95% cluster-bootstrap interval (2,000 replicates over source samples).

Systems	Target	0c	1c	2c
Canonical (3)	RDTC	.872	.931	.950
	Single label	.887	.832	.842
Semi-real (9)	RDTC	.675	.824	.899
	Single label	.628	.573	.608
Simulated (12)	RDTC	.724	.851	.911
	Single label	.693	.638	.666
	Difference	$+.031$ [$+.019, +.042$]	$+.213$ [$+.197, +.229$]	$+.245$ [$+.229, +.262$]

Table S17: **Graded RDTC versus a dense binary target.** AUROC averaged over $\Delta > 0$ for models trained on the graded RDTC (as TipPFN) or on the episode outcome at every time step (binary), at the size of the released model ($d = 1280$) and at model dimension 96. The binary models see the outcome encoding in their context episodes, as in training. Training steps: $d = 1280$ RDTC 22k (released model), binary 9k; $d = 96$ RDTC 37k, binary 32k.

		$d = 1280$			$d = 96$		
Systems	Target	0c	1c	2c	0c	1c	2c
Canonical (3)	RDTC	.876	.927	.951	.867	.899	.923
	Binary	.838	.856	.859	.666	.749	.826
Semi-real b-tipping (5)	RDTC	.790	.898	.939	.696	.866	.910
	Binary	.783	.785	.840	.673	.774	.799
Semi-real r-tipping (4)	RDTC	.544	.701	.812	.488	.657	.795
	Binary	.527	.444	.501	.616	.536	.575
Simulated (12)	RDTC	.730	.839	.899	.670	.805	.875
	Binary	.712	.689	.732	.652	.688	.731

Bury et al. retrained on the TipPFN prior To separate the effect of the training prior from that of the target, task construction and architecture, we retrained the CNN-LSTM classifier of [22] on TipPFN's training data. As in [22], it receives the univariate driver signal x of each training episode, detrended with a LOWESS filter (span 0.2), as a sequence of length 256, and is trained for a four-way classification into fold, Hopf, transcritical and null, where null includes forced but subcritical episodes; the score is one minus the predicted null probability. We kept the original architecture (one convolutional and two LSTM layers, about 23k parameters per network) and trained an ensemble of 20 networks for 300 epochs, ten for each of the two padding variants of [22], with 15% of the samples held out each for validation and testing. Evaluation follows Table 1; with the original weights, the same pipeline reproduces the Bury column of Table 1 within .012 on 13 of 14 rows (R-Compost: $+.060$).

Compared with the original networks, the retrained ensemble is better on B-SEIRx, the r-tipping systems except R-AMOC, TAC and DaphniaExt, and about .1 worse on the canonical systems, B-Harv. and B-AMOC (Table S18). TipPFN with two context episodes is above it on 12 of the 14 rows (all except R-Compost and TAC), and zero-context TipPFN on 9. On the simulated systems,

TipPFN's auxiliary variables also carry information about the forcing (Appendix C.4.1); observing only x, as the retrained model does, zero-context TipPFN is above it on 7 of the 12 simulated systems, mainly canonical and b-tipping ones. The training prior alone therefore does not explain TipPFN's performance.

Table S18: **Bury et al. retrained on the TipPFN prior.** AUROC averaged over $\Delta > 0$. *Change* is the difference to the original pretrained model (Bury column of Table 1). TipPFN columns give the number of context episodes (c) and observed auxiliary features (f); with 0f TipPFN observes only x, as the retrained model does. For TAC and DaphniaExt, the TipPFN columns are those of Table 1.

	Bury [22], retrained		TipPFN		
System/data	AUROC	Change	0c/0f	0c/16f	2c/16f
Canonical					
B-Fold	.587	−.109	.818	.876	.955
B-Hopf	.731	−.111	.816	.861	.952
B-Trans.	.620	−.102	.747	.864	.950
Semi-real					
B-Harv.	.863	−.104	.970	.846	.974
B-RM TC	.714	−.069	.791	.815	.963
B-RM Hopf	.585	−.054	.582	.769	.904
B-SEIRx	.520	+.052	.539	.582	.905
B-AMOC	.858	−.090	.783	.853	.940
R-Bautin	.508	+.200	.086	.308	.700
R-SN	.381	+.026	.348	.644	.892
R-Compost	.877	+.150	.686	.767	.665
R-AMOC	.094	−.009	.171	.257	.913
Real-world & Sim-to-Real					
TAC	.534	+.057	–	.521	.519
DaphniaExt	.514	+.030	–	.527	.833

Model size Table S19 compares the released model ($d = 1280$) with variants of model dimension 256, 96, 32 and 16, trained on the same prior; the smaller variants were trained on a subset of 200,000 training draws. Averaged over zero to two context episodes on the 12 simulated systems, the $d = 96$ and $d = 32$ variants are .034 below the released model and $d = 16$ is .078 below; with two context episodes, $d = 32$ reaches .857 against .893 (Sec. 4.4). All $d = 256$ runs with learning rates of $6.25 \cdot 10^{-4}$ and above diverged during training; we report the checkpoint with the lowest validation loss before divergence (17k steps), which is below $d = 96$ and $d = 32$ and reflects this instability rather than a capacity limit. On DaphniaExt, the small models are better than the released model with two context episodes and worse with one; with a single dataset, this is not evidence of a trend. Scale is therefore not required for most of the performance.

Context noise and mismatch We perturb the context episodes in two ways (Table S20). For context noise, we add zero-mean Gaussian noise with standard deviation σ times the standard deviation of each context channel to the observed context signals; time stamps and RDTC are unchanged, and the query is not perturbed. The unperturbed runs reproduce Table 1 within .02 in 44 of 45 cells (R-AMOC without context: .233 instead of .257). For mismatched context, each query receives all its context episodes from one other sample, drawn at random per query either from the same system as the query, with other parameters and auxiliary dynamics, or from a different system; within that sample, the episodes are selected as for matched context. AUROC falls monotonically with noise on the simulated systems, mildly up to $\sigma = .5$ and clearly at $\sigma = 1$ (two context episodes: from .893 to .802); DaphniaExt degrades in the same way (.829 to .730), and TAC stays at chance. Mismatched context lowers AUROC on the simulated systems below the zero-context value of .700, both from the same system (.552 and .565 with one and two context episodes) and from a different system (.528 and .532). With two same-system context episodes, the paired difference is $-.135$ to zero context and $-.328$ to matched context (95% cluster-bootstrap intervals over query samples, 2,000 replicates: $[-.164, -.102]$ and $[-.356, -.299]$). R-Compost is the only system where mismatched context scores about as high as matched context (same system: .684 and .662, against .620 and .669); there, zero-context TipPFN (.772) is also better than matched context, as in Table 1. TAC

Table S19: **Model size.** AUROC averaged over $\Delta > 0$ for the released model ($d = 1280$) and variants of smaller model dimension d, for 0, 1 and 2 context episodes (c). The $d = 256$ checkpoint precedes a training divergence (see text). *Mean* averages the simulated rows over 0–2 context episodes.

		$d = 1280$	$d = 256$	$d = 96$	$d = 32$	$d = 16$
Parameters		210.7M	8.44M	1.25M	137k	35.9k
Training steps		22k	17k	37k	83k	39k
Canonical (3)	0c	.865	.880	.861	.874	.865
	1c	.928	.873	.905	.906	.856
	2c	.951	.899	.919	.939	.892
Semi-real (9)	0c	.645	.616	.608	.634	.610
	1c	.800	.718	.758	.725	.669
	2c	.874	.804	.834	.830	.770
Simulated (12)	0c	.700	.682	.671	.694	.674
	1c	.832	.757	.795	.770	.716
	2c	.893	.828	.855	.857	.800
	Mean	.808	.756	.774	.774	.730
TAC	0c	.521	.573	.509	.542	.497
	1c	.512	.520	.524	.516	.507
	2c	.519	.541	.534	.561	.556
DaphniaExt	0c	.529	.355	.583	.437	.373
	1c	.661	.572	.533	.591	.596
	2c	.829	.822	.897	.895	.894

and DaphniaExt offer no other sample to draw mismatched context from. TipPFN's context gains therefore require context from the same generating draw ψ as the query; context from another draw of the same system does not transfer.

Table S20: **Context noise and mismatch.** AUROC averaged over $\Delta > 0$ with 1 or 2 context episodes (c). Noise: zero-mean Gaussian noise with σ times the standard deviation of each context channel added to the context signals. Mismatched: all context episodes from one other sample, drawn at random per query, of the same system (other parameters and auxiliary dynamics) or of a different system, without noise; not defined for TAC and DaphniaExt.

		Context noise σ				Mismatched	
Systems		0	.25	.5	1	Same system	Other system
Canonical (3)	1c	.928	.922	.915	.875	.584	.576
	2c	.951	.942	.936	.916	.615	.586
Semi-real (9)	1c	.800	.775	.758	.718	.541	.512
	2c	.874	.853	.823	.764	.549	.514
Simulated (12)	1c	.832	.812	.797	.757	.552	.528
	2c	.893	.875	.851	.802	.565	.532
TAC	1c	.512	.504	.490	.479	–	–
	2c	.519	.514	.510	.504	–	–
DaphniaExt	1c	.661	.627	.598	.526	–	–
	2c	.829	.813	.791	.730	–	–

Sample efficiency (indicative) To estimate how much training data TipPFN needs, we evaluated intermediate checkpoints of the $d = 96$ variant, which was trained on 200,000 distinct training draws per epoch with 16 samples per step (Table S21). With two context episodes, it reaches 90% of its final AUROC (37k steps, about three epochs) after 8% of one epoch on the canonical and b-tipping systems and on DaphniaExt, and after 32% on the r-tipping systems; TAC stays near chance at every checkpoint. The ratio is taken on the raw AUROC scale, which flatters weak cells, and the curves are not monotone. A checkpoint saved after a fraction of the data has also received fewer optimizer steps,

so this analysis does not separate data volume from optimization progress. We therefore report it as an indicative estimate, not as a controlled dataset-size ablation, which would require training separate models to convergence on subsets of the prior.

Table S21: **Sample efficiency (indicative).** AUROC with two context episodes, averaged over $\Delta > 0$, for checkpoints of the $d = 96$ variant after the given fraction of one epoch (200,000 training draws); *final* is the evaluated end of training (37k steps, 2.96 epochs). The last column gives the first checkpoint at which AUROC reaches 90% of the final value.

Systems	4%	8%	16%	32%	96%	Final	90% at
Canonical (3)	.792	.894	.923	.913	.922	.919	8%
Semi-real b-tipping (5)	.649	.827	.884	.897	.907	.896	8%
Semi-real r-tipping (4)	.402	.468	.575	.691	.683	.757	32%
DaphniaExt	.463	.772	.873	.819	.821	.836	8%
TAC	.512	.495	.548	.537	.541	.535	–

C.4.5 SWEC-iEEG

Seizure onset prediction is a challenging and clinically relevant test case for transition forecasting, as it requires anticipating a rapid shift from interictal to ictal dynamics. In SWEC-iEEG, the model operates on multichannel iEEG bandpower trajectories. We interpret seizure onset as the critical event and evaluate prediction on preictal segments relative to interictal control segments from the same patient, drawn from periods at least one hour away from any seizure episode. No method separates pre-ictal from interictal segments: in Table 1, all columns lie between .49 and .56, and at lead times of 10–40 s between .46 and .55 (Fig. S14). The higher TipPFN zero-context value comes from $\Delta = 60$ and 80 s, where the recordings before onset are too short for a full window, so critical query windows hold 89–109 observed steps against 128 for interictal ones.

With neurological EEG patterns being highly patient-specific, one interesting use case for TipPFN is to evaluate individual seizure risk by including only same-patient observations into the context. Fig. S16 shows the resulting patient-specific AUROC scores. They scatter around chance for every method (medians .47–.55); with 2–23 seizures per patient, individual values are noisy.

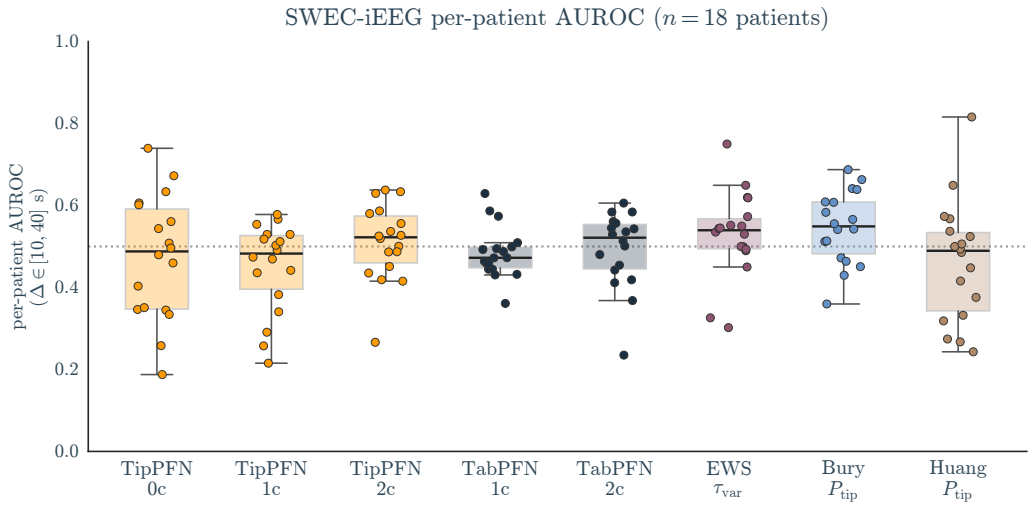


Figure S16: **SWEC per-patient AUROC distribution**. Each dot is one patient's AUROC computed within that patient ($n = 18$) and averaged over $\Delta = 10$–40 s, at the setting of each column in Table 1. For the uni-variate baseline models, this includes a per-signal-band selection: the bands `mrbp_0p50_0p66hz` (delta), `mrbp_3p58_4p74hz` (theta), and `mrbp_19p35_25p64hz` (beta) are scored separately and the best band is used, matching the choice in Table 1) and Fig. S14.

C.4.6 Daphnia Extinction

The controlled *Daphnia magna* extinction experiment of Drake and Griffen [44] is a valuable real-world transfer benchmark because it combines empirical observations with experimentally controlled deterioration, a well-characterized transcritical transition, and allows to match a Ricker-map simulation [43] to generate simulated context.

To construct context, we consider two surrogate RDTC choices: either setting $\Lambda = 0$ at the observed extinction event, or sampling it from the bifurcation-time interval estimated in [44]. Both observed and simulated contexts can yield comparable predictive performance, suggesting that matched simulations can act as a practical surrogate when repeated real trajectories are limited. Simulated context tends to peak already with fewer episodes, see Fig. S17 and S18.

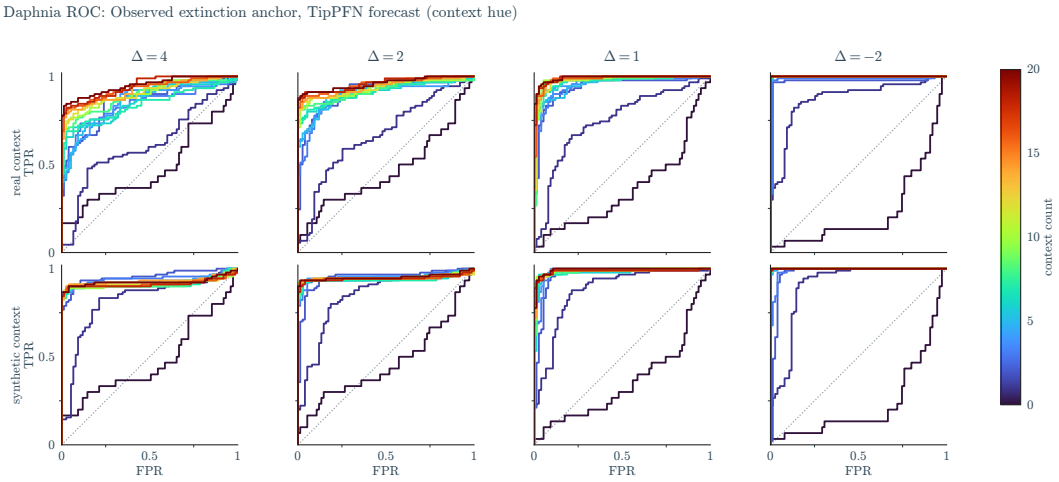


Figure S17: `DaphniaExt` dataset ROC curves for $W = 16$, different Δ and increasing context episodes, using real observations as context (top row) or purely simulated context (bottom row). Surrogate Λ was condition on $\Lambda = 0$ at the extinction time of the respective population.

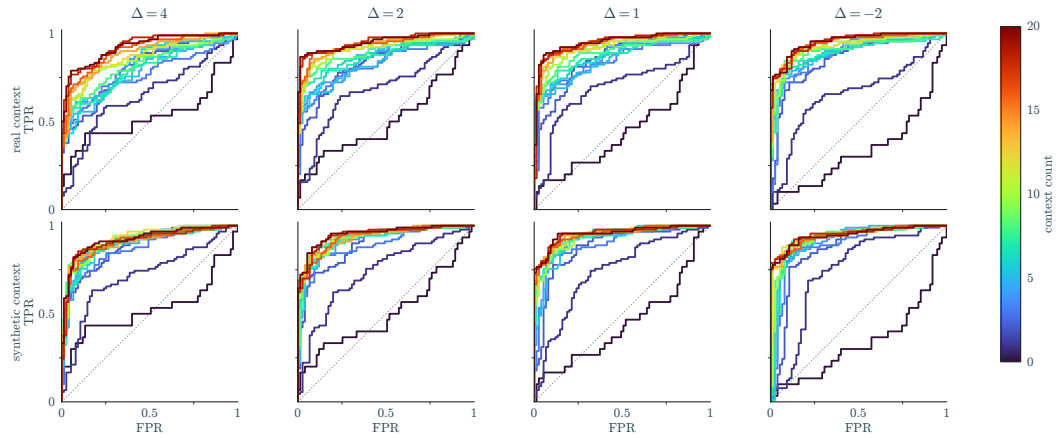


Figure S18: Like above, but surrogate Λ was conditioned on $\Lambda = 0$ at a random time within the bifurcation-time interval $[271, 316]$ days estimated in [44], aiming to estimate the unterlying transition.

The two Λ target constructions also reveal a qualitative difference between PFN-style models: TabPFN scores en par with TipPFN when it comes to distinguishing extinction; however, the arguably more difficult detection is that of the underlying bifurcation, which TipPFN discerns more clearly than other baselines, see Fig. S20.

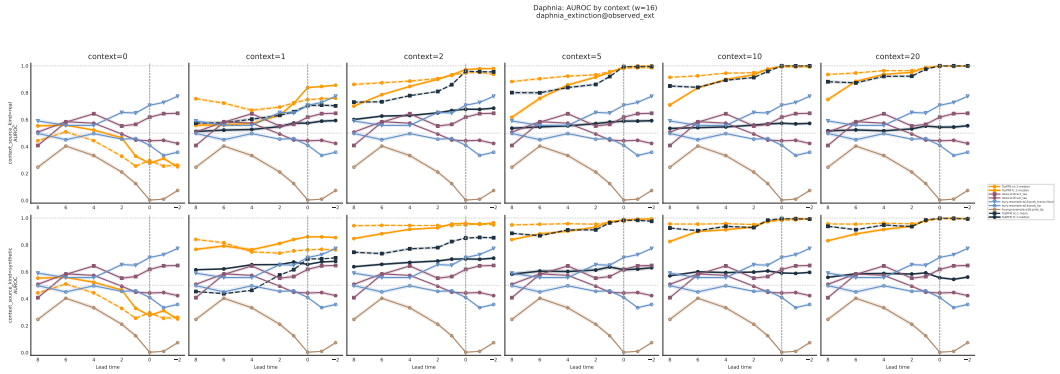


Figure S19: `DaphniaExt` dataset AUROC curves for $W = 16$, different context sizes, using real observations as context (top row) or purely simulated context (bottom row). Surrogate Λ was conditioned on $\Lambda = 0$ at the extinction time of the respective population.

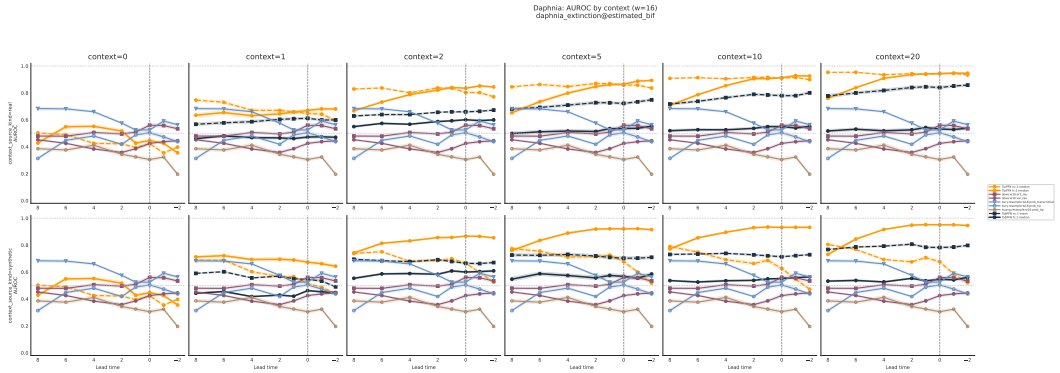


Figure S20: Like above, but with surrogate Λ on the estimated bifurcation, the systems hidden critical transition.

C.4.7 TAC

The thermo-acoustic combustor dataset provides a particularly noisy real-world benchmark built around a stochastic subcritical Hopf transition [45]. We formulate the task as prediction of the Hopf bifurcation and assign surrogate Λ labels by anchoring a linear ramp at the known critical point. In the basic setting, the Hopf-specific Bury model performs very well, consistent with its close match to the target mechanism. TipPFN performance is higher for an increased context size of 4; it additionally increases when composing the context with exclusively critical context episodes, presumably because they are the more informative and distinguishing signal. Fig. S21 shows the effect on AUROC when filling the context with only non-critical, only critical, or random episodes, causing TipPFN to outperform Bury.

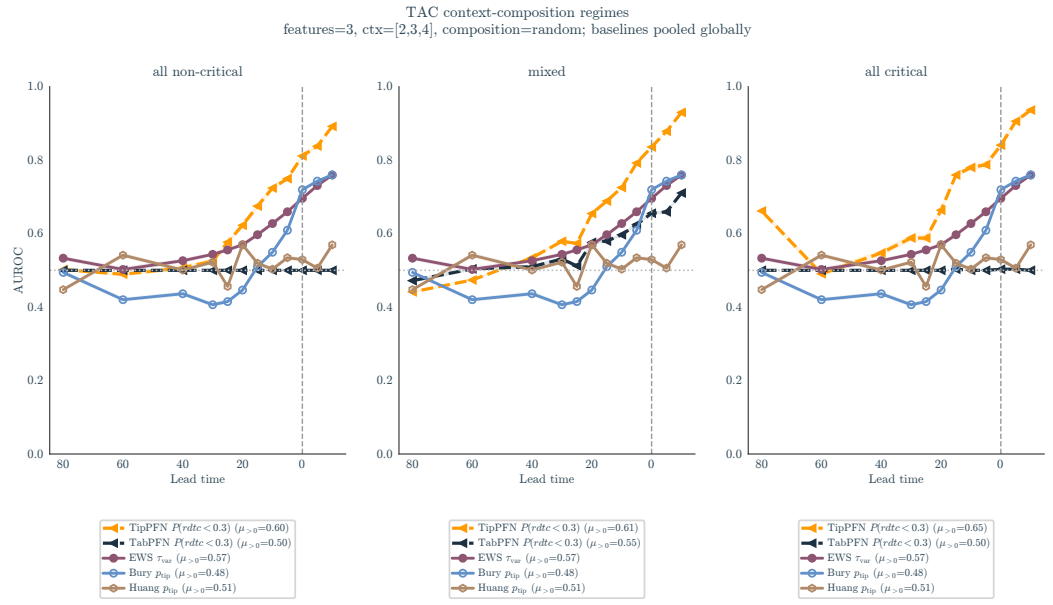


Figure S21: TAC dataset AUROC scores over lead time for contexts of different composition.

C.4.8 Zero-shot datasets

In addition to Fig. 5, we study three further zero-shot time series, depicted in Fig. S22. While we find early warning ($\Delta > 0$) for the mitochondria and voice dataset, the RDTC prediction for the paleoclimate (greenhouse_earth) dataset only crossed the threshold after the critical transition.

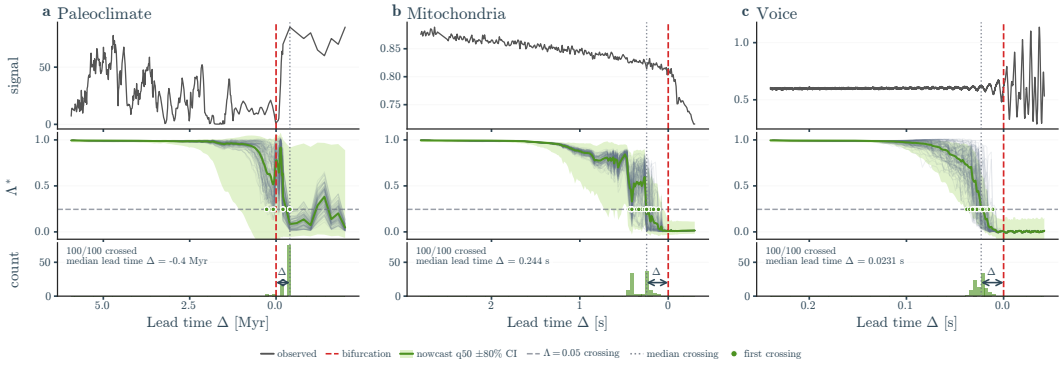


Figure S22: **Zero-shot TipPFN RDTC nowcasts** on three additional uni-variate real-world time series (see Fig. 5). Note the crossing threshold $\Lambda^* = \tanh(5\Lambda) \approx 0.245$.

C.4.9 Proximity of real-world inputs to the training prior

PFN predictions approximate the posterior predictive only for inputs supported by the training prior. We therefore measure how close the real-world inputs of TAC and DaphniaExt lie to inputs seen in training. For every evaluation query we take one segment, the last $L \in \{8, 16, 32, 64, 128\}$ steps of the model-visible query window after input normalization, with all input channels. The reference consists of 2,195,904 prior segments of the same length and number of channels, eight per training draw (all 274,488 draws), constructed like training queries: resampled in time, with randomly selected observed variables, and normalized in the same way as the evaluation input. The distance between two segments is the root-mean-square difference over all entries; RDTC does not enter it. A segment counts as covered if the distance to its 20th-nearest prior segment is at most the 95th percentile of

the same quantity for 19,668 held-out prior segments (one per draw from 12 of the 167 blocks of training draws, searched against the other 155 blocks), calibrated separately for each L and number of channels. Table S22 reports the covered fraction with 95% intervals from 2,000 bootstrap replicates over source recordings (TAC) or segments (DaphniaExt).

Except for short TAC segments ($L \leq 16$: 51% and 77%), at least 94% of the segments are covered, and all TAC segments with $L \geq 64$; by construction, 95% of held-out prior segments are covered. Restricting the neighbors to prior segments whose RDTC is close to that of the query (band width 0.025–0.1) does not lower any of these fractions. The analysis measures proximity in the space of normalized observations: it shows that the real-world inputs resemble inputs TipPFN was trained on, not that the evaluation data follow the prior distribution or that the physical systems belong to the model class of the prior.

Table S22: **Coverage of real-world evaluation segments by the training prior.** Percentage of segments of length L whose distance to their 20th-nearest prior segment is at most the 95th percentile for held-out prior segments (Appendix C.4.9), with 95% bootstrap interval; no interval where all segments are covered. DaphniaExt windows have at most 16 steps; its two rows place $\Lambda = 0$ at the observed extinction or within the estimated bifurcation interval.

Dataset	Segments	$L = 8$	$L = 16$	$L = 32$	$L = 64$	$L = 128$
TAC	160	51.3 [42.5, 64.6]	76.9 [66.7, 81.7]	94.4 [92.4, 99.0]	100.0	100.0
DaphniaExt, extinction	55	94.5 [87.3, 100.0]	94.5 [89.0, 100.0]	–	–	–
DaphniaExt, bifurcation	55	100.0	100.0	–	–	–

C.5 AUROC Uncertainty

Table S23: Per-system standard errors for the macro AUROC over $\Delta > 0$ for the values reported in Tables 1 and S12. *Total samples* is $\sum_c (n_{\text{pos}}^{(c)} + n_{\text{neg}}^{(c)})$ across all contributing (Δ, system) points *pooled* $n_{\min}$ is $\sum_c \min(n_{\text{pos}}^{(c)}, n_{\text{neg}}^{(c)})$, the effective balanced-sample count entering equation (S51). SE_{HM} is reported at $A = 0.8$; values at $A = 0.5$ are $\sim 25\%$ larger. SE_{BS} is the propagated balanced-subsample uncertainty from equation (S52); for TAC it is effectively zero because n_{pos}=n_{neg} at every contributing point and no subsampling occurs.

System / dataset	total samples	pooled $n_{\min}$	SE_{HM} at A=0.8	SE_{BS}
Canonical				
B-Fold	911	407	0.020	0.0009
B-Hopf	1,019	465	0.019	0.0008
B-Trans.	910	418	0.020	0.0005
Semi-real				
B-Harv.	434	194	0.029	0.0008
B-RM TC	672	300	0.023	0.0021
B-RM Hopf	737	335	0.022	0.0013
B-SEIRx	370	164	0.031	0.0022
B-AMOC	597	273	0.024	0.0019
R-Bautin	738	318	0.022	0.0042
R-SN	768	346	0.022	0.0017
R-Compost	682	308	0.023	0.0023
R-AMOC	608	272	0.024	0.0029
Others				
SWEC-iEEG	14,526	2,088	0.009	0.0013
TAC	21,600	10,800	0.004	~ 0
DaphniaExt	4,400	2,000	0.009	0.0006
Summary				
All systems	49,070	18,688	0.003	0.0005

For each AUROC value reported, we compute a balanced AUROC by drawing $K = 10$ class-balanced subsamples without replacement from the available critical and non-critical query windows, evaluating `sklearn.metrics.roc_auc_score` on each, and reporting the mean. Each subsample contains $n_{\min} = \min(n_{\text{pos}}, n_{\text{neg}})$ examples per class.

We attach a sample-size standard error using a simplified form of the Hanley–McNeil approximation,

$$\text{SE}_{\text{HM}} = \sqrt{\frac{A\,(1-A)}{n_{\min}^{\text{pooled}}}}, \tag{S51}$$

where $n_{\min}^{\text{pooled}} = \sum_c \min(n_{\text{pos}}^{(c)}, n_{\text{neg}}^{(c)})$ sums per-point minority-class counts across all lead times $\Delta > 0$ contributing to the reported macro AUROC. Equation (S51) ignores the correlation between lead times and between overlapping windows of the same episode, so it understates the uncertainty; the cluster bootstrap in Appendix C.5.1 resamples whole source samples and accounts for this dependence. AUROC differences below $2\,\text{SE}_{\text{HM}}$ should be regarded as not statistically resolved.

We additionally report the propagated standard error of the macro AUROC from the balanced-subsampling procedure. For each contributing (Δ, system) point we have an empirical std $\sigma_{\text{BS}}^{(c)}$ across the K=10 subsamples; pooling these across the K_{macro} points contributing to a system's macro AUROC gives

$$\text{SE}_{\text{BS}} = \frac{1}{K_{\text{macro}}} \sqrt{\frac{1}{K} \sum_{c=1}^{K_{\text{macro}}} \left(\sigma_{\text{BS}}^{(c)}\right)^2}. \tag{S52}$$

This quantity captures sensitivity of the AUROC estimator to which $n_{\min}$ examples are chosen for a given imbalance ratio at each point. It does not capture sampling variation in the underlying episode

population, and is therefore consistently 5–30$\times$ smaller than $\mathrm{SE}_{\mathrm{HM}}$ in our data. We report it as a separate diagnostic rather than combining it with $\mathrm{SE}_{\mathrm{HM}}$, since the two estimate related but distinct variances and the second-order contribution would not move the displayed precision.

Per-system pooled $n_{\min}$, $\mathrm{SE}_{\mathrm{HM}}$, $\mathrm{SE}_{\mathrm{BS}}$, and the underlying total sample count are shown in Table S23.

Selection bias from best-of-K picking. For columns whose value is the best across a family of configurations (e.g. any column that selects across window size, feature-budget, or score-head combinations within a method), the reported AUROC is the maximum over K correlated noisy estimates and is therefore positively biased relative to the true expected AUROC at the chosen setting. Under the null hypothesis that all candidate settings have equal expected AUROC, the expected maximum exceeds the per-cell mean by approximately $\sigma \cdot \Phi^{-1}(1 - 1/K)$, where σ is the per-cell sampling std and Φ^{-1} is the inverse normal CDF. For our typical $K \in [10, 150]$ candidate cells per (method, system) and $\sigma \approx \mathrm{SE}_{\mathrm{HM}}$, the per-cell inflation is 0.02–0.06 in the iid-null limit; correlation across candidates pulls this back to a typical 0.01–0.03.

Appendix C.4.2 repeats the column-wise selection on held-out data; the column means change by at most .02. Differences between methods at the per-system level should therefore be interpreted as upper bounds on the true gap rather than unbiased point estimates.

C.5.1 Per-system bootstrap confidence intervals

The standard errors above treat query windows as independent and ignore the correlation between lead times. Here we instead resample the independent units from which the query windows are drawn and report confidence intervals for the TipPFN columns of Table 1 and for the change in AUROC with context. We use the scores and settings of Table 1; the point estimate is its balanced AUROC averaged over $\Delta > 0$ and reproduces the table. The cluster bootstrap [72] ($B = 10{,}000$) resamples the generating draw ψ for the simulated systems, the recording for TAC and the population episode for DaphniaExt; each replicate uses the same units for all lead times and context counts, so differences between context counts are paired. We report two-sided 90% percentile intervals and bootstrap p-values, Holm-corrected [73] across the 14 rows within each column (cells against 0.5, differences against 0); the smallest attainable corrected p-value is about 0.003. SWEC-iEEG is not included (Appendix C.4.5).

With two context episodes, TipPFN is significantly above chance on all 14 rows (Table S24). At zero context, B-SEIRx, TAC and DaphniaExt are not distinguishable from chance, and R-Bautin and R-AMOC are significantly below chance. The change from zero to two context episodes is significant on every row except TAC and positive on all of them except R-Compost, where zero-context TipPFN performs best ($-.102$, 90% CI $[-.159, -.038]$); the first context episode significantly lowers AUROC on R-Compost. For the real-world datasets, the intervals do not account for the post hoc choice of window and are therefore optimistic; for TAC, 1,100 of the 10,000 replicates contain a lead time with only one class, which makes its intervals approximate.

C.5.2 Score Analysis

The CNN baselines (Bury, Huang, Zhuge) require fixed-length input windows; signals shorter than the network's expected length are extended via one of three `short_mode` schemes (`pad`, `backfill`, `resample`). Rather than pinning a single short-mode globally, each baseline column in the headline AUROC table independently picks its column-best `short_mode`, and the row-wise variant (Table S12) re-picks per system; the picked token is reported in every baseline cell. Across all baseline cells of the row-wise table, `resample` dominates ($\sim$71%), with `backfill` ($\sim$19%) and `pad` ($\sim$10%) chosen on a minority of systems where the signal-to-padding scaling differs. Because the picked token is already disclosed at cell granularity and no global pin is asserted, we do not include a dedicated short-mode sensitivity SI table; the row-wise picks are self-documenting.

A handful of baseline scores are anti-correlated with criticality on specific datasets, producing AUROCs significantly below chance (< 0.5). The clearest case is Zhuge's `signed_tip_margin` on SWEC-iEEG, which is constant within each class (all non-critical queries score 0, all critical queries at a given Δ share one negative value), so its AUROC is exactly 0 for every window and band; we therefore omit Zhuge on SWEC-iEEG. While excluding degenerate scores, we report all

Table S24: **Cluster-bootstrap confidence intervals for TipPFN.** AUROC averaged over $\Delta > 0$ with 90% confidence interval for 0, 1 and 2 context episodes (c), and paired changes from 0c. *Units* is the number of independent units resampled. Daggers mark values whose Holm-corrected p-value exceeds 0.05 (cells tested against 0.5, changes against 0); underlined cells lie significantly below 0.5.

System/data	Units	0c	1c	2c	0c→1c	0c→2c
Canonical						
B-Fold	41	.876 [.839, .914]	.929 [.906, .953]	.955 [.936, .971]	+.053 [+.014, +.096]†	+.078 [+.039, +.118]
B-Hopf	46	.861 [.822, .897]	.935 [.898, .964]	.952 [.913, .982]	+.074 [+.029, +.119]	+.091 [+.046, +.136]
B-Trans.	41	.864 [.819, .902]	.925 [.886, .956]	.950 [.918, .974]	+.061 [+.018, +.108]†	+.086 [+.045, +.130]
Semi-real						
B-Harv.	20	.846 [.790, .891]	.958 [.919, .982]	.974 [.951, .991]	+.112 [+.072, +.157]	+.127 [+.087, +.182]
B-RM TC	31	.815 [.762, .874]	.951 [.928, .972]	.963 [.943, .978]	+.136 [+.080, +.183]	+.148 [+.092, +.195]
B-RM Hopf	33	.769 [.717, .817]	.829 [.768, .885]	.904 [.866, .940]	+.061 [−.008, +.128]†	+.136 [+.071, +.201]
B-SEIRx	17	.582 [.455, .687]†	.822 [.713, .912]	.905 [.857, .956]	+.240 [+.082, +.420]†	+.324 [+.205, +.477]
B-AMOC	27	.853 [.801, .900]	.898 [.847, .942]	.940 [.901, .971]	+.045 [−.004, +.098]†	+.087 [+.033, +.142]
R-Bautin	34	.308 [.272, .341]	.543 [.485, .607]†	.700 [.632, .777]	+.235 [+.171, +.311]	+.392 [+.322, +.477]
R-SN	35	.644 [.569, .717]	.802 [.749, .850]	.892 [.846, .932]	+.158 [+.104, +.213]	+.248 [+.186, +.313]
R-Compost	31	.767 [.726, .804]	.623 [.561, .689]	.665 [.615, .721]	−.144 [−.214, −.069]	−.102 [−.159, −.038]
R-AMOC	28	.257 [.206, .296]	.782 [.704, .848]	.913 [.863, .951]	+.525 [+.436, +.614]	+.656 [+.597, +.719]
Real-world & Sim-to-Real						
TAC	12	.521 [.445, .542]†	.512 [.498, .523]†	.519 [.505, .556]	−.009 [−.029, +.061]†	−.002 [−.025, +.104]†
DaphniaExt	55	.527 [.459, .598]†	.663 [.623, .700]	.833 [.790, .868]	+.136 [+.058, +.210]	+.307 [+.219, +.388]

baseline AUROCs as-is without auto-flipping anti-correlated scores – flipping at evaluation time would amount to a per-cell sign-tuning.

Table S25: Selected configuration for each column of the headline AUROC table (Table 1). Score names are defined in Table S10. For TipPFN and TabPFN the configuration spans the score head, context size, query-window length, and feature budget; for the baseline methods only the parameters consumed by that method are listed (e.g. EWS reads window and indicator; Bury, Huang and Zhuge additionally read *short_mode*). Each baseline column independently picks its best query-window length (column-best over $\{64, 96, 128\}$) and, for SWEC where the score is computed across multiple bandpower bands, its best signal band. Daphnia uses an incomparable window range $\{8, 12, 16\}$, so every column reports the per-cell best-of-$\{8, 12, 16\}$ for daphnia (matching the row-wise convention). The row-wise best variant (Table S12) re-selects window, *short_mode*, and (on SWEC) signal band separately per system.

Column	Score name	Configuration
TipPFN / 0c	`tippfn:fc:1-median`	head=fc:1-median, ctx=0, w=128, features=16
TipPFN / 1c	`tippfn:fc:1-median`	head=fc:1-median, ctx=1, w=128, features=16
TipPFN / 2c	`tippfn:fc:1-median`	head=fc:1-median, ctx=2, w=128, features=16
TabPFN / 1c	`tabpfn:fc:1-median`	head=fc:1-median, ctx=1, w=64, features=16
TabPFN / 2c	`tabpfn:fc:1-median`	head=fc:1-median, ctx=2, w=64, features=16
EWS / best	`dews:w64:var_tau`	w64, indicator=var_tau
Bury / best	`bury:resample:w128:prob_tip`	short_mode=resample, w128, head=prob_tip
Huang / best	`huang:resample:w64:prob_tip`	short_mode=resample, w64, head=prob_tip
Zhuge / best	`zhuge:rdtc:resample:w128:signed_tip_margin`	control=rdtc, short_mode=resample, w128, head=signed_tip_margin

Table S26: **Head sensitivity of the All-datasets summary row.** Mean AUROC over $\Delta > 0$ across all 15 system/data rows, broken down by TipPFN/TabPFN context column (column groups), score scope+head (rows), and query-window length (sub-columns) at features=16. Heads: `fc`/`nc` use the forecast/nowcast query position; `1-median` and `1-mean` are one minus the median or mean of the predicted RDTC distribution, and `P(rdtc<`τ`)` is the predicted probability that RDTC falls below τ (Table S10). The headline col-wise table (Table 1) pins TipPFN/TabPFN to `fc:1-median` for every column. The best cell per column group is shown in bold with an asterisk ($\cdot^*$); other cells within $\sqrt{\sigma^2_{\text{best}} + \sigma^2_{\text{cell}}}$ of the best (cross-system SE) are also bold (no asterisk), indicating statistically indistinguishable choices.

	TipPFN / 0c			TipPFN / 1c			TipPFN / 2c			TabPFN / 1c			TabPFN / 2c		
Score head	w64	w96	w128	w64	w96	w128	w64	w96	w128	w64	w96	w128	w64	w96	w128
fc:1-median	**.683**	**.688**	**.670**	.708	**.759**	**.779**	.775	**.819**	**.838**	.522	.514	.512	.666	.655	.652
fc:1-mean	**.679**	**.675**	**.681**	.726	**.765**	**.780**	.790	**.825**	**.842**	.518	.511	.507	.647	.637	.635
fc:P(rdtc<0.05)	**.676**	**.672**	**.673**	.726	**.764**	**.781**	.786	**.822**	**.842**	.498	.494	.487	.605	.597	.590
fc:P(rdtc<0.1)	**.676**	**.674**	**.674**	.728	**.765**	**.780**	.787	**.823**	**.843**	.513	.507	.501	.617	.609	.604
fc:P(rdtc<0.2)	**.676**	**.676**	**.670**	.729	**.763**	**.779**	.788	**.819**	**.840**	**.531**	.524	.525	.634	.625	.628
fc:P(rdtc<0.3)	**.671**	**.679**	**.669**	.733	**.764**	**.777**	.788	**.817**	**.838**	**.546**	**.539**	**.541**	.658	.649	.650
nc:1-median	**.634**	**.685**	**.663**	.662	**.755**	**.789**	.770	**.833**	**.856**	**.559***	**.540**	**.526**	**.729***	**.707**	**.692**
nc:1-mean	**.645**	**.689***	**.674**	.681	**.771**	**.794**	.792	**.846**	**.864***	**.544**	**.525**	.511	**.712**	**.689**	.673
nc:P(rdtc<0.05)	.501	**.662**	**.659**	.533	.582	.634	.578	.654	.703	.516	.485	.442	.660	.617	.563
nc:P(rdtc<0.1)	.502	**.671**	**.664**	.558	.612	.707	.620	.702	.765	.519	.487	.459	.660	.626	.597
nc:P(rdtc<0.2)	.507	**.681**	**.659**	.606	.694	**.777**	.695	.777	**.836**	**.534**	.518	.523	**.679**	.660	.655
nc:P(rdtc<0.3)	.610	**.676**	**.655**	.668	**.769**	**.800***	.759	**.834**	**.862**	**.542**	**.539**	**.551**	**.705**	**.689**	**.684**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly outline our contribution to the paper and scope of the work in the abstract and introduction, including on headline result in Figure 1 and contribution overview in Figure 2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not include any theoretical results. Where necessary published theoretical background has been included in the paper and properly referenced.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide extensive description on data generation in Appendix C.1 and modeling choices in Section 3, and release the code, the trained checkpoint and the data (Appendix B.9).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Yes, instructions on the data generating pipeline, pretraining, and validation are detailed in Section 3, and the related Appendix B, C. We release the code, the trained checkpoint, the training data and the evaluation sets with instructions to reproduce the main results; SWEC-iEEG cannot be redistributed, and we provide a guide and preprocessing script instead (Appendix B.9).

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, experimental details, including sweep parameters in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Yes. We report sample sizes and standard errors for all results in Tables S14 and S23, per-system 95% cluster-bootstrap confidence intervals in Appendix C.5.1, and paired differences with cluster-bootstrap intervals for the ablations in Appendix C.4.4. Where reasonable, we show asymmetric confidence intervals as error bands. Details on the balanced AUROC statistical procedure are in Appendix C.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide descriptions on the used compute resources in Appendix B.8.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research conforms with the NeurIPS Code of Ethics, as we do not conduct research with human participants (we use the public SWEC-iEEG recordings under their research licence and do not redistribute them), follow norms of fair use and copyright for any algorithms or datasets we use, and present a method with no risk of potential harmful consequences or societal impact, such as privacy, surveillance, discrimination, human rights, etc.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: As we provide a methodological advance for foundational research, we do not see any opportunity for potential negative societal impacts such as disinformation, deepfakes, surveillance, unfair impacts of specific groups, privacy considerations, or security considerations.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We do not provide any data or models that are at high risk of misuse such as LLMs, image generators or from the internet scraped datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original papers of all baselines, simulators and datasets, and list the source and licence of each asset we use or redistribute in Appendix B.9 (e.g., TAC and DaphniaExt: CC0; Huang et al. code: CC BY 4.0; SWEC-iEEG: research and education use only; TabPFN 2.6 weights: Prior Labs licence).

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We detailed the data generating pipeline, summary of the systems, and modeling training and inference setups in e.g., Appendix C. The released checkpoint, datasets and baseline checkpoints come with model and dataset cards that state their training details, licences and intended use (Appendix B.9).

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: Our paper does not include any crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our paper does not include any crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We do not use any LLM for the core methodology, scientific rigorousness, or originality of the research, except for copy edit to fix grammar, misspelling, and double word counting.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.