Generalized Teacher Forcing for Learning Chaotic Dynamics

Florian Hess [1 2 3] Zahra Monfared [1 3] Manuel Brenner [1 2] Daniel Durstewitz [1 2 4]

Abstract

Chaotic dynamical systems (DS) are ubiquitous in nature and society. Often we are interested in reconstructing such systems from observed time series for prediction or mechanistic insight, where by reconstruction we mean learning geometrical and invariant temporal properties of the system in question (like attractors). However, training reconstruction algorithms like recurrent neural networks (RNNs) on such systems by gradient-descent based techniques faces severe challenges. This is mainly due to exploding gradients caused by the exponential divergence of trajectories in chaotic systems. Moreover, for (scientific) interpretability we wish to have as low dimensional reconstructions as possible, preferably in a model which is mathematically tractable. Here we report that a surprisingly simple modification of teacher forcing leads to provably strictly all-time bounded gradients in training on chaotic systems, and, when paired with a simple architectural rearrangement of a tractable RNN design, piecewise-linear RNNs (PLRNNs), allows for faithful reconstruction in spaces of at most the dimensionality of the observed system. We show on several DS that with these amendments we can reconstruct DS better than current SOTA algorithms, in much lower dimensions. Performance differences were particularly compelling on real world data with which most other methods severely struggled. This work thus led to a simple yet powerful DS reconstruction algorithm which is highly interpretable at the same time.

[1]Department of Theoretical Neuroscience, Central Institute of Mental Health, Medical Faculty Mannheim, Heidelberg University, Mannheim, Germany [2]Faculty of Physics and Astronomy, Heidelberg University, Heidelberg, Germany [3]Cluster of Excellence STRUCTURES, Heidelberg University, Heidelberg, Germany [4]Interdisciplinary Center for Scientific Computing, Heidelberg University, Heidelberg, Germany. Correspondence to: Florian Hess <Florian.Hess@zi-mannheim.de>, Daniel Durstewitz <Daniel.Durstewitz@zi-mannheim.de>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

In many scientific and engineering settings we are interested in learning the dynamics of an unknown, or hard to tackle, underlying DS that we have observed through a set of time series measurements. This may, for instance, be temperature measurements to assess climate dynamics, tracking infected cases for epidemiological dynamics, or neural recordings from the brain. Having a faithful model of the underlying dynamics enables rigorous scientific and mathematical analysis, as well as optimal predictions of the system's future temporal evolution. For many complex DS such as the brain or social networks we have only rudimentary and insufficient knowledge about the governing equations. Given the success of deep learning in so many areas and disciplines, there has been a burgeoning interest in recent years in purely data-driven approaches to DS reconstruction to bypass the traditional tedious and protracted scientific model building process (Brunton et al., 2016; Chen et al., 2018; Koppe et al., 2019; Vlachas et al., 2020; Schmidt et al., 2021; Brenner et al., 2022; Lejarza & Baldea, 2022).

In DS reconstruction one aims to retrieve from data an (approximate) model of the underlying dynamics that encapsulates all its geometrical and invariant temporal properties, such as attractor states and other invariant sets (for some applications, even just capturing the topological structure of the underlying DS may be sufficient; Takens (1981); Sauer et al. (1991)). Training machine learning systems such as RNNs on time series observations from DS poses a number of severe challenges, however. In fact, DS reconstruction algorithms have so far mostly been probed only on *simulated* DS (Brunton et al., 2016; Vlachas et al., 2018; Pathak et al., 2018; Champion et al., 2019; Otto & Rowley, 2019; Bakarji et al., 2022), such as the Lorenz-63 (Lorenz, 1963) and Lorenz-96 (Lorenz, 1996) models of atmospheric convection, or Navier-Stokes equations for turbulent flow. While evaluation of algorithms on such ground truth benchmarks is obviously important, the relative sparsity of real-world applications demonstrates there is still a lot of ground to make (cf. Brenner et al. (2022); Mikhaeil et al. (2022)). One important challenge here is that most complex DS are inherently chaotic (Olsen et al., 1988; Van Vreeswijk & Sompolinsky, 1996; Durstewitz & Gabriel, 2007a; Duarte et al., 2010; Faggini, 2014; Kesmia et al., 2020; Mangiarotti et al., 2020; Inoue et al., 2021; Kamdjeu Kengne et al., 2021). Chaotic DS harbor trajectories diverging exponentially fast, which leads to exploding loss gradients when training with the most common, scalable gradient-based techniques (Engelken et al., 2020; Mikhaeil et al., 2022). Recent theoretical and empirical results emphasized that this a *principle* problem which cannot be addressed through specifically designed network architectures (Mikhaeil et al., 2022),[1] but needs to be addressed at the level of training algorithms. At the same time, especially for detailed scientific analysis, we are interested in reconstruction models that are tractable and reproduce the observed data (and the underlying dynamics) in as low dimensions as possible.[2]

Here we tackle both these challenges. In particular, we prove that a simple amendment of teacher forcing (TF), a classical technique to keep trajectories on track while training, leads to strictly contained loss gradients, for arbitrary time horizons, without diminishing a model's ability to capture chaotic dynamics. While simple in spirit, this solves an outstanding problem in the field (Mikhaeil et al., 2022) and leads to training results that surpass a wide range of previous SOTA techniques, often by large margins, especially on real-world data. We also rearrange the tractable PLRNN structure, previously suggested for DS reconstruction (Brenner et al., 2022), into a 'shallow' one-hidden-layer design. While this yields an almost standard multi-layer-perceptron (MLP), we observed that – somewhat surprisingly – it allows for reconstruction of DS with at most as many latent dynamical variables as those of the underlying system, and makes the resulting architecture particularly well-suited for our specific variant of TF. Meanwhile, we show the resulting model can still be rewritten in standard PLRNN form, preserving its tractable design.

2. Related Work

Dynamical Systems Reconstruction DS reconstruction aims at producing a generative model of an unknown DS underlying a set of observed time series variables. By 'generative' here we mean that, after successful training, the (not necessarily probabilistic) model should be capable of producing time series with the same temporal and geometrical properties as those produced by the true (but unknown) DS, where 'geometrical' refers to the geometry of dynamical objects (like attractors) in state space and 'temporal' includes invariant properties like a system's power spectrum. Commonly these models work by approximating the unknown governing equations (or their flow), e.g. through a sufficiently expressive library of basis functions (Brunton et al., 2016; Lejarza & Baldea, 2022), or by RNNs (Vlachas et al., 2018; Pathak et al., 2018; Koppe et al., 2019; Vlachas et al., 2020; Schmidt et al., 2021; Brenner et al., 2022), which are universal approximators of DS (Funahashi & Nakamura, 1993; Kimura & Nakano, 1998; Hanson & Raginsky, 2020). Algorithms for training RNNs on DS reconstruction problems were based on probabilistic techniques like Expectation-Maximization and nonlinear Kalman filters (Voss et al., 2004; Koppe et al., 2019; Zhao & Park, 2020) or variational inference (Hernandez et al., 2020; Kramer et al., 2022), but the to date most successful methods (cf. Brenner et al. (2022)) simply relied on gradient-based procedures like Back-Propagation-Through-Time (BPTT; Rumelhart et al. (1986)). Continuous-time RNNs like Neural ODEs (Chen et al., 2018) were also widely tested for DS reconstruction (Raissi, 2018), with extensions to systems of partial differential equations. Other recent ideas include Fourier Neural Operators for approximating spatially extended systems in function space (Li et al., 2021), RNNs or other DS approximators embedded within deep auto-encoders to extract suitable coordinate transformations or lower-dimensional state space representations (Champion et al., 2019; Bakarji et al., 2022), and approaches for extrapolating to 'unseen' dynamical regimes (Patel & Ott, 2022; Kirchmeyer et al., 2022; Ricci et al., 2022). For scientific applications, mathematical tractability and dynamical interpretability[3] of trained DS reconstruction models is imperative. To these ends, often locally linear techniques like piecewise-linear RNNs (PLRNNs; Durstewitz (2017); Koppe et al. (2019)) , Koopman operator theory (Lusch et al., 2018; Brunton et al., 2022) or co-training of switching linear DS (Smith et al., 2021), were designed, but most of these require moving to a higher-dimensional space. We show that this latter issue can be avoided by simply reshaping PLRNNs into a 1-hidden-layer structure.

Controlling exploding gradients The exploding and vanishing gradient problem has long been recognized as a severe challenge in training RNNs on time series prediction or classification tasks which involve large time spans between relevant pieces of information (Bengio et al., 1994; Hochreiter et al., 2001). For DS, a related problem is that these may contain dynamics evolving on widely differing, including very slow, time scales (Schmidt et al., 2021). Many approaches tried to address this issue by designing specific RNN architectures (Hochreiter & Schmidhuber, 1997; Cho et al., 2014; Rusch & Mishra, 2020; 2021; Rusch et al., 2022) or imposing specific constraints on RNN parameters

[1]For instance, both classical solutions like LSTM (Hochreiter & Schmidhuber, 1997) or GRU (Cho et al., 2014) as well as more recent architectures, like coRNN (Rusch & Mishra, 2020), do not offer any way out of this conundrum (Mikhaeil et al., 2022).

[2]Note that classical delay embedding reconstruction theorems (Takens, 1981; Sauer et al., 1991) demand that the reconstruction space is larger than two times the box-counting dimension of the underlying attractor, but latent variable models or deeper architectures may have other means to 'fill in' missing dimensions.

[3]Note we mean 'interpretability' in a DS sense, i.e. such that topological and geometrical properties of the resulting system can easily be analyzed, visualized, and understood.

(Arjovsky et al., 2016; Chang et al., 2019; Erichson et al., 2021). However, all of these either severely limit expressiveness such that chaotic dynamics cannot be learned to begin with (because constraints or design prevent maximum Lyapunov exponents from exceeding 0), or they still struggle with exploding gradients and hence time series from chaotic systems (Mikhaeil et al., 2022). Mikhaeil et al. (2022) therefore suggested to address the problem in training by controlling trajectory divergence through 'sparse TF', based on the older control-theoretic idea of TF (Pineda, 1988; Pearlmutter, 1989; Williams & Zipser, 1989; Jordan, 1990). Sparse TF (STF) resets RNN states to control values inferred from the data, thus pulling diverging trajectories back on track and cutting off exploding gradients, at times determined from the underlying system's Lyapunov spectrum. However, this requires knowledge or empirical estimates of the system's Lyapunov exponents, and – on the other hand – it does not strictly avoid exploding gradients either, just attempts to strike an optimal balance between gradient divergence and not loosing relevant longer time scales. Here we suggest a simple modification of TF that solves both these problems such that gradients are *strictly* contained yet chaotic dynamics can be learned, without any knowledge required of the underlying DS' maximum Lyapunov exponent. This is one of the main results of the present work.

3. Theoretical Analysis & Methods

This section will first discuss the basic problem in training RNNs on time series from chaotic systems. We will then outline a solution and its mathematical foundation, before addressing the specific RNN architecture for DS reconstruction championed in this work.

3.1. Problem setting: Loss gradients and chaotic dynamics

Our exposition here summarizes the key observations made in Mikhaeil et al. (2022). Most RNNs are parameterized discrete-time recursive maps of the form

$$\boldsymbol{z}_t = \boldsymbol{F}_{\boldsymbol{\theta}}\big(\boldsymbol{z}_{t-1}, \boldsymbol{s}_t\big), \tag{1}$$

with states $\boldsymbol{z}_t$, optional external inputs $\boldsymbol{s}_t$, and parameter set $\boldsymbol{\theta}$. If unconstrained, depending on its set of parameters, an RNN may exhibit any type of limit dynamics, like convergence to fixed points, cycles of any order, quasi-periodic behavior, or chaos (in fact, RNNs are dynamically universal, cf. Hanson & Raginsky (2020)). If an RNN is used to reconstruct a chaotic system, it needs to be chaotic itself (otherwise the reconstruction would have failed), which entails that its maximum Lyapunov exponent needs to be larger than 0. The Lyapunov exponent quantifies the exponential divergence of initially nearby trajectories. Defining the Jacobian of (1) by

$$\boldsymbol{J}_t := \frac{\partial \boldsymbol{F}_{\boldsymbol{\theta}}\big(\boldsymbol{z}_{t-1}, \boldsymbol{s}_t\big)}{\partial \boldsymbol{z}_{t-1}} = \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_{t-1}}, \tag{2}$$

the maximum Lyapunov exponent of an RNN orbit $\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \dots, \boldsymbol{z}_T, \dots\}$ is given through the product of Jacobians $\boldsymbol{J}_t$ along the orbit by

$$\lambda_{max} := \lim_{T\to\infty} \frac{1}{T} \log \left\| \prod_{r=0}^{T-2} \boldsymbol{J}_{T-r} \right\|_2, \tag{3}$$

where $\|\cdot\|_2$ denotes the spectral norm. Mikhaeil et al. (2022) showed that the problem of training RNNs on chaotic time series using gradient descent based algorithms is ill-posed, as the condition $\lambda_{max} > 0$ will inevitably lead to diverging loss gradients.

Let us illustrate this with the most common algorithm used for training RNNs, Backpropagation Through Time (BPTT; Rumelhart et al. (1986); Werbos (1990)). Given a loss function $\mathcal{L} = \sum_{t=1}^{T} \mathcal{L}_t(\boldsymbol{z}_t,\ \bar{\boldsymbol{z}}_t)$ where $\boldsymbol{z}_t$ are RNN-generated and $\bar{\boldsymbol{z}}_t$ target states, the BPTT algorithm employs the chain rule along the RNN unrolled in time to compute loss gradients w.r.t. model parameters $\theta_i \in \boldsymbol{\theta}$,

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}_t}{\partial \theta_i} \quad \text{with} \quad \frac{\partial \mathcal{L}_t}{\partial \theta_i} = \sum_{r=1}^{t} \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_t} \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \frac{\partial^+ \boldsymbol{z}_r}{\partial \theta_i}, \tag{4}$$

where ∂^+ denotes the immediate derivative. The crucial observation now is that Eq. (4) contains the same product of Jacobians,

$$\begin{aligned} \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} &= \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_{t-1}} \frac{\partial \boldsymbol{z}_{t-1}}{\partial \boldsymbol{z}_{t-2}} \cdots \frac{\partial \boldsymbol{z}_{r+1}}{\partial \boldsymbol{z}_r} \\ &= \boldsymbol{J}_t \boldsymbol{J}_{t-1} \cdots \boldsymbol{J}_{r+1} = \prod_{k=0}^{t-r-1} \boldsymbol{J}_{t-k}, \end{aligned} \tag{5}$$

that occurs in the definition of the maximum Lyapunov exponent in Eq. 3. As Mikhaeil et al. (2022) strictly prove, this entails exponentially exploding loss gradients for $T \to \infty$ when training RNNs on chaotic systems whose behavior they are supposed to reproduce. In practice, unreliable and ill-behaved training sets in even for moderate sequence lengths T, and architectures like LSTM designed to control gradient flows or simple gradient clipping techniques are not sufficient to contain the problem (Mikhaeil et al., 2022).

3.2. Generalized Teacher Forcing

What is needed in addition is a procedure for forcing diverging trajectories back onto their targets while training for DS reconstruction. This has been recognized long ago, and the classical control technique here is TF (Pineda, 1988;

Pearlmutter, 1989; Williams & Zipser, 1989; Jordan, 1990). TF replaces current RNN states *after* computing the loss by data-inferred values, classically at each time step, or at times strategically chosen according to the system's maximum Lyapunov exponent in sparse-TF (Mikhaeil et al., 2022). Motivated by the problem of unstable solutions after RNN training with classical TF, Doya et al. (1992) suggested to linearly interpolate between RNN-generated states $\boldsymbol{z}_t$ and target states $\bar{\boldsymbol{z}}_t$:

$$\tilde{\boldsymbol{z}}_t := (1-\alpha)\boldsymbol{z}_t + \alpha\bar{\boldsymbol{z}}_t, \tag{6}$$

with $0 \leq \alpha \leq 1$. This simple idea, which we will refer to as *Generalized Teacher Forcing* (GTF), has in fact not been systematically studied so far. It turns out that GTF, with the right choice of α, can *fully rectify* the exploding gradients problem in learning chaotic dynamics.

Now let us unwrap how GTF (6) will impact RNN (1) training. The state replacement, Eq. (6), is performed prior to applying RNN map $\boldsymbol{F_\theta}$ at each training time step, i.e. $\boldsymbol{z}_t = \boldsymbol{F_\theta}(\tilde{\boldsymbol{z}}_{t-1})$. According to the chain rule, this leads to the following factorization of Jacobians $\boldsymbol{J}_t$:

$$\begin{aligned}\boldsymbol{J}_t &= \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_{t-1}} = \frac{\partial \boldsymbol{z}_t}{\partial \tilde{\boldsymbol{z}}_{t-1}}\frac{\partial \tilde{\boldsymbol{z}}_{t-1}}{\partial \boldsymbol{z}_{t-1}} = \frac{\partial \boldsymbol{F_\theta}(\tilde{\boldsymbol{z}}_{t-1})}{\partial \tilde{\boldsymbol{z}}_{t-1}}\frac{\partial \tilde{\boldsymbol{z}}_{t-1}}{\partial \boldsymbol{z}_{t-1}} \\ &= (1-\alpha)\tilde{\boldsymbol{J}}_t,\end{aligned} \tag{7}$$

where $\tilde{\boldsymbol{J}}_t$ is the Jacobian of $\boldsymbol{F_\theta}$ evaluated at the forced state $\tilde{\boldsymbol{z}}_{t-1}$. Plugging into (5) gives the following expression for the product of Jacobians under GTF:

$$\begin{aligned}\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} = \prod_{k=0}^{t-r-1} \boldsymbol{J}_{t-k} &= \prod_{k=0}^{t-r-1}(1-\alpha)\tilde{\boldsymbol{J}}_{t-k} \\ &= (1-\alpha)^{t-r}\prod_{k=0}^{t-r-1}\tilde{\boldsymbol{J}}_{t-k}.\end{aligned} \tag{8}$$

For $\alpha = 0$ (no forcing) this simply yields vanilla BPTT (5), while for $\alpha = 1$ Eq. (8) evaluates to zero and there will be no gradient propagation. For values in between, GTF controls the Jacobian product norm $\left\|\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}\right\|$ as illustrated in Fig. 1. Fig. 2 summarizes the GTF principle and notation.

Now assume that for any fixed given set of parameters $\boldsymbol{\theta}$, $\mathcal{J} = \{\tilde{\boldsymbol{J}}_\kappa\}_{\kappa\in\mathcal{K}}$ is the set of all different Jacobians of an RNN (which may be countable or uncountable). We define

$$\begin{aligned}\tilde{\sigma}_{\max} &= \sup \mathcal{S}_1 := \sup\left\{\left\|\tilde{\boldsymbol{J}}_\kappa\right\| = \sigma_{\max}(\tilde{\boldsymbol{J}}_\kappa)\ :\ \tilde{\boldsymbol{J}}_\kappa \in \mathcal{J}\right\}, \\ \tilde{\lambda}_{\min} &= \inf \mathcal{S}_2 := \inf\left\{\lambda_{\min}(\tilde{\boldsymbol{J}}_\kappa) : \tilde{\boldsymbol{J}}_\kappa \in \mathcal{J}\right\} \geq 0,\end{aligned} \tag{9}$$

where $\lambda_{\min}(\tilde{\boldsymbol{J}}_\kappa) = \min\left\{|\lambda_j|\ :\ \lambda_j \in eig(\tilde{\boldsymbol{J}}_\kappa)\right\}$.

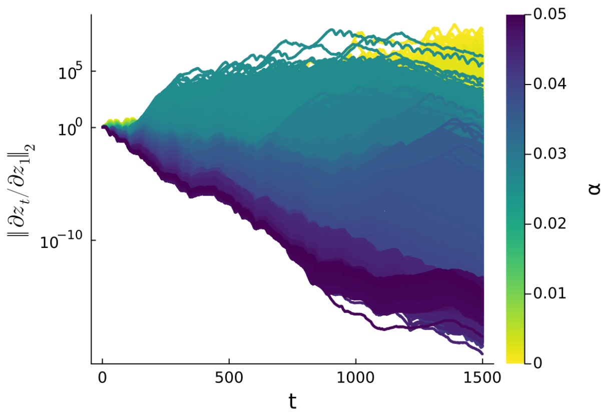


Figure 1. Norms of Jacobian product series given in Eq. (8) for an RNN trained by BPTT+GTF on the Lorenz-63 system as function of sequence length t. The RNN state is initialized based on the ground truth and then propagated for t time steps. The Jacobians diverge if α is chosen too small, but converge if too large.

Obviously the nonempty set $\mathcal{S}_2 \subset \mathbb{R}$ is bounded from below by definition, but the set $\mathcal{S}_1 \subset \mathbb{R}$ is not necessarily bounded from above. However, here we will consider the set of extended real numbers $\bar{\mathbb{R}} := \mathbb{R} \cup \{-\infty, +\infty\}$ and define the supremum of a set not bounded from above as $+\infty$. The following proposition states a necessary condition for the existence of chaos in RNNs:

Proposition 1. *If the RNN (1) is chaotic, then $\tilde{\sigma}_{\max} > 1$.*

Proof. See Appx. 6.1.1. □

The next proposition now sets the stage for a proper choice of the GTF parameter α:

Proposition 2. *Consider an RNN given by (1) and let $\tilde{\sigma}_{\max} \geq 1$.*

(i) If $\alpha = \alpha^* := 1 - \frac{1}{\tilde{\sigma}_{\max}}$, then the Jacobian product series $\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}$ will be bounded from above, i.e. its norm will not diverge as $t - r \to \infty$.

(ii) Assume that $\alpha = \alpha^*$ and define $\tilde{\gamma} := \frac{\tilde{\lambda}_{\min}}{\tilde{\sigma}_{\max}}$ $(0 \leq \tilde{\gamma} \leq 1)$. If $\tilde{\gamma} = 1$, then $\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}$ neither explodes nor vanishes for $t - r \to \infty$. When $\tilde{\gamma} \neq 1$, the Jacobian $\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}$ will not explode, but may potentially vanish as $t - r$ goes to infinity. Furthermore if $\lim_{t-r\to\infty}\left\|\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}\right\| = 0$, then the closer $\tilde{\gamma}$ is to 1, the slower $\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}$ tends to zero for $t - r \to \infty$.

Proof. See Appx. 6.1.2. □

3.3. Reconstruction models

We will base our experiments on the specific class of PLRNNs (Durstewitz, 2017; Koppe et al., 2019), since they

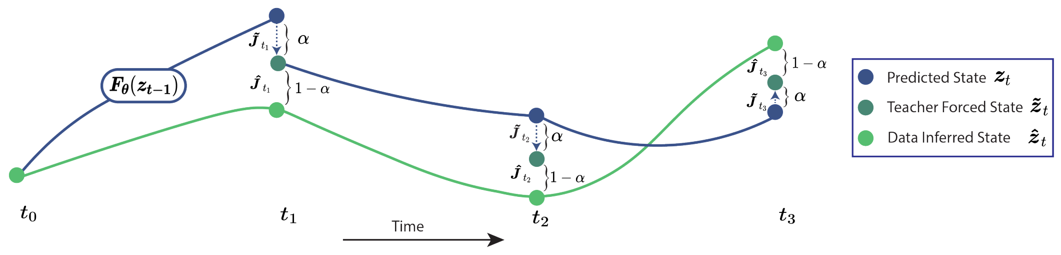


Figure 2. Principle of Generalized Teacher Forcing.

are mathematically tractable in the sense that fixed points, cycles, and other invariant sets can, in principle, be computed exactly and semi-analytically for them (Monfared & Durstewitz, 2020; Schmidt et al., 2021; Brenner et al., 2022). This facilitates their post-training DS analysis in scientific contexts. In its simplest form, the PLRNN is given by

$$z_t = A z_{t-1} + W \phi(z_{t-1}) + h_0, \tag{10}$$

with diagonal matrix $A \in \mathbb{R}^{M\times M}$, off-diagonal matrix $W \in \mathbb{R}^{M\times M}$, bias $h_0 \in \mathbb{R}^M$, and rectified-linear-unit nonlinearity $\phi(\cdot) = \mathrm{ReLU}(\cdot) = \max(0, \cdot)$. Brenner et al. (2022) extended this basic structure by adding a linear spline basis expansion, dubbed the dendritic PLRNN (dendPLRNN), to increase the expressivity of each unit's nonlinearity and thus enable DS reconstructions in lower dimensions,

$$z_t = A z_{t-1} + W \sum_{b=1}^{B} \alpha_b \phi(z_{t-1} - h_b) + h_0, \tag{11}$$

with slope-threshold pairs $\{\alpha_b, h_b\}_{b=1}^B$, where B is the number of bases. It can be shown that the dendPLRNN can be reformulated as a higher-dimensional conventional PLRNN (Brenner et al., 2022).

However, in this work we will specifically consider the following "1-hidden-layer" ReLU-based RNN, which we will refer to as *shallow* PLRNN (shPLRNN):

$$z_t = A z_{t-1} + W_1 \phi(W_2 z_{t-1} + h_2) + h_1, \tag{12}$$

with latent states $z_t \in \mathbb{R}^M$, diagonal matrix $A \in \mathbb{R}^{M\times M}$, rectangular connectivity matrices $W_1 \in \mathbb{R}^{M\times L}$ and $W_2 \in \mathbb{R}^{L\times M}$, and thresholds $h_2 \in \mathbb{R}^L$ and $h_1 \in \mathbb{R}^M$. With $L > M$, by expanding each unit's activation into a weighted sum of ReLU nonlinearities, this formulation appears similar to the dendPLRNN, and indeed this intuition is confirmed by the following Proposition:

Proposition 3. *Any shPLRNN given by* (12) *can be rewritten in the form of the dendPLRNN* (11)*. It follows, in particular, that fixed points and cycles of* (12) *can be computed in an analogous way as for the dendPLRNN. Vice versa, any M-dimensional dendPLRNN can be reformulated as an M-dimensional shPLRNN with hidden layer size $L = M \cdot B$.*

Proof. See Appx. 6.1.3 □

Similar to the dendPLRNN, the shPLRNN can be equipped with a clipping mechanism that prevents states from diverging to infinity due to the unbounded ReLU nonlinearity:

$$\begin{aligned} z_t = A z_{t-1} + W_1 \big[&\phi(W_2 z_{t-1} + h_2) \\ &-\phi(W_2 z_{t-1})\big] + h_1. \end{aligned} \tag{13}$$

This guarantees bounded orbits provided the largest absolute eigenvalue of A is smaller than 1 (as shown in Appx. 6.1.4).

Finally, to map from the PLRNN's M-dimensional latent to the N-dimensional observation space, a linear observation model (i.e., linear output layer) is used:

$$\hat{x}_t = G_\varphi(z_t) = B z_t. \tag{14}$$

3.4. Model training

We use BPTT combined with GTF to train RNNs on time series $X \in \mathbb{R}^{N\times T}$ from chaotic DS, where T is the length of the time series and N is the number of observed variables. Control signals $\hat{z}_t$ used for TF are inferred from the observations by inversion of the observation model G_φ (Mikhaeil et al., 2022),

$$\hat{z}_t := G_\varphi^{-1}(x) = B^+ x_t, \tag{15}$$

where B^+ denotes the Moore–Penrose (pseudo-) inverse of B.[4] In the simplest case, if the RNN's latent dimension and the dimension of the observations match, i.e. $N = M$, one could also simply fix $B = \mathbb{1}_{N\times N}$ and train on the observations directly, taking $\hat{z}_t = x_t$ as the control signal (see Brenner et al. (2022)).

How to choose the GTF parameter α in practice? Building on Proposition 2, to avoid exploding gradients altogether, one would need to set α according to the maximum singular value of the RNN Jacobians, Eq. (9). However, exactly computing $\tilde{\sigma}_{\max}$ is intractable for most RNN architectures.

[4]More generally, one may think of invertible neural networks (INNs; Dinh et al. (2017); Ardizzone et al. (2019)) for linking latent states to observations, but here we contend ourselves with a simple linear model.

This can easily be seen for the shPLRNN (12), for which the Jacobian (2) is given by

$$\boldsymbol{J}_t^{(sh)} = \boldsymbol{A} + \boldsymbol{W}_1 \tilde{\boldsymbol{D}}_{\Omega(t-1)} \boldsymbol{W}_2, \tag{16}$$

where $\tilde{\boldsymbol{D}}_{\Omega(t-1)} = \mathrm{diag}\big(\tilde{d}_{1,t-1}, \tilde{d}_{2,t-1}, \cdots, \tilde{d}_{L,t-1}\big)$ is an $L \times L$ diagonal binary indicator matrix with $d_{l,t-1} = 1$ if $\sum_{j=1}^{M} w_{lj}^{(2)} z_{j,t-1} > -h_l^{(2)}$, for $\boldsymbol{W}_2 = \big[w_{ij}^{(2)}\big]$, $\boldsymbol{h}_2 = \big[h_i^{(2)}\big]$, and 0 otherwise. Each possible configuration of 1's and 0's on the diagonal of $\tilde{\boldsymbol{D}}_\Omega$ corresponds to a different linear subregion of the state space, in which the Jacobian (16) is constant and the dynamics of (12) is linear. Hence, to compute $\tilde{\sigma}_{\max}$ one would need to evaluate the Jacobians of all 2^L linear subregions, which is generally infeasible, especially since $\tilde{\sigma}_{\max}$ would need to be re-evaluated after each parameter update during training. The cheapest way around this issue is to choose α according to an upper bound of $\tilde{\sigma}_{\max}$:

$$\alpha^{(n)} = 1 - \frac{1}{\lceil \tilde{\sigma}_{\max} \rceil^{(n)}}, \tag{17}$$

where n denotes the n-th optimization step and $\lceil \tilde{\sigma}_{\max} \rceil$ is the upper bound given by

$$\begin{aligned} \lceil \tilde{\sigma}_{\max} \rceil &:= \|\boldsymbol{A}\| + \|\boldsymbol{W}_1\| \, \|\boldsymbol{W}_2\| \\ &\geq \max_{\tilde{\boldsymbol{D}}_\Omega} \left\| \boldsymbol{A} + \boldsymbol{W}_1 \tilde{\boldsymbol{D}}_\Omega \boldsymbol{W}_2 \right\| = \tilde{\sigma}_{\max}. \end{aligned} \tag{18}$$

A slightly more expensive alternative heuristic is to approximate $\tilde{\sigma}_{\max}$ by computing the Jacobians at states given by the teacher signals of a given training batch $\boldsymbol{X}$ (since, ultimately, the RNN is required to generate time series with these same signatures). Letting $\boldsymbol{x}_t^{(p)}$ denote the observation vector of the p-th sequence in the batch at time t, the teacher signals are given by $\hat{\boldsymbol{z}}_t^{(p)} = \boldsymbol{G}_{\boldsymbol{\varphi}}^{-1}(\boldsymbol{x}_t^{(p)})$, for which the Jacobians $\hat{\boldsymbol{J}}_t^{(p)}$ (cf. Fig. 2) can be computed using (16). We then have an estimate $\hat{\sigma}_{\max}$ as

$$\hat{\sigma}_{\max} = \max_{t,p} \left\| \hat{\boldsymbol{J}}_t^{(p)} \right\|, \tag{19}$$

and can choose α accordingly.

However, we find that in practice the estimates above are too conservative, leading to suboptimal performance. Instead, we derive an estimate directly based on Eq. (8). First, note that ideal error propagation is achieved when the chain of Jacobians connecting temporally most distal states in a training sequence of length T is close to the identity,

$$\frac{\partial \boldsymbol{z}_T}{\partial \boldsymbol{z}_1} = (1-\alpha)^{T-1} \prod_{k=0}^{T-2} \tilde{\boldsymbol{J}}_{T-k} \overset{!}{=} \mathbb{1}. \tag{20}$$

Provided that the Jacobian product series is non-singular, it follows that

$$(1-\alpha)\mathcal{G}(\tilde{\boldsymbol{J}}_{T:2}) \overset{!}{=} \mathbb{1}, \tag{21}$$

$$\text{where} \quad \mathcal{G}(\tilde{\boldsymbol{J}}_{T:2}) := \left(\prod_{k=0}^{T-2} \tilde{\boldsymbol{J}}_{T-k} \right)^{\frac{1}{T-1}}. \tag{22}$$

Taking the norm on both sides of Eq. (22), rearranging, and assuming, as above, that we can replace forced Jacobians $\tilde{\boldsymbol{J}}_t^{(p)}$ of the p-th sequence in the current batch with Jacobians $\hat{\boldsymbol{J}}_t^{(p)}$ evaluated at data-inferred states, we obtain a collection $\{\alpha^{(p)}\}$ which we condense into a single estimate by taking

$$\alpha = \max_p \alpha^{(p)} = \max_p \left[1 - \frac{1}{\left\| \mathcal{G}(\hat{\boldsymbol{J}}_{T:2}^{(p)}) \right\|} \right]. \tag{23}$$

Since computing (22) requires evaluating Jacobian products which cause exploding gradients in the first place, we use an approximation which foregoes computing those products altogether, see Appx. 6.2 for details. Furthermore, to ensure that replacing Jacobians $\tilde{\boldsymbol{J}}_t$ at forced states with data inferred Jacobians $\hat{\boldsymbol{J}}_t$ remains valid throughout training, we employ an annealing strategy, which starts with strong forcing ($\alpha = 1$) that decays throughout training while remaining bounded from below by (23).[5] The full training protocol is detailed in Appx. 6.2, and will be referred to as *adaptive* GTF (aGTF). As a reference, in our experiments below we will also employ fine-tuning α via grid search. Note that GTF is only used for *training* the model, not during actual testing.

4. Results

We evaluate GTF using the shPLRNN on simulated DS and real-world data sets, and compare its performance to a variety of other DS reconstruction algorithms. We will first introduce the data sets and evaluation measures used.

4.1. DS data sets

Lorenz-63, Lorenz-96, and multiscale Lorenz-96 Both the 3d Lorenz-63 (Lorenz, 1963) and the higher-dimensional, spatially extended Lorenz-96 (Lorenz, 1996) ODE systems were conceived as simple models of atmospheric convection with chaotic behavior in some regime (see Appx. 6.3 for more details). Here we include them mainly because they often served as DS benchmarks in the past. But by now almost all DS reconstruction algorithms achieve satisfactory performance on them, so that the focus here will be on the much more challenging empirical data sets. As a somewhat more challenging benchmark, we also consider a (partially observed) multiscale variant of the Lorenz-96 system (Thornes et al., 2017).

[5] More generally, annealing protocols have previously been observed to improve TF-based training in RNNs (Bengio et al., 2015; Vlachas & Koumoutsakos, 2023).

Electrocardiogram (ECG) data ECG recordings of a human subject were taken from the open-access PPG-DaLiA dataset (Reiss et al., 2019). The ECG is a scalar physiological (heart muscle potential) time series, which we delay-embed (Sauer et al., 1991) into $5d$ using the PECUZAL algorithm (Krämer et al., 2021). The time series shows signs of chaotic behavior, as indicated by its empirically determined maximum Lyapunov exponent (cf. Appx 6.3).

Electroencephalogram (EEG) data As another challenging real-world data set we use open-access EEG recordings from a human subject under task-free, quiescent conditions with eyes open (Schalk et al., 2000). The $64d$ dataset consists of 60 s of brain activity measured through 64 electrodes placed across the scalp. This dataset has previously been shown to bear signatures of chaotic activity, like a fractal dimensionality and positive maximum Lyapunov exponent ($\lambda_{max} \approx 0.017$; Mikhaeil et al. (2022)).

We emphasize that – unlike common simulated benchmarks – empirical data often come with a number of additional challenges, like only a fraction of all dynamical variables observed, from potentially very high-dimensional generating systems, through only partly known and often intricate observation functions, with possibly high levels of dynamical and observation noise and unknown external perturbations, multiple temporal and spatial scales (brain recordings in particular), and potentially non-stationary behavior.

4.2. Evaluation measures

To assess DS reconstruction quality, besides short-term prediction, we need to ensure that *invariant* properties of the underlying system, like an attractor's geometrical structure in state space and its temporal signatures, have been adequately captured. In our choice of measures we follow Koppe et al. (2019); Brenner et al. (2022); Mikhaeil et al. (2022). For implementation details we refer to Appx. 6.4.

Geometrical Measure To assess the agreement in true and reconstructed attractor geometries, we compute a Kullback-Leibler (KL) divergence based on the natural measure induced by system trajectories in state space on the invariant set (D_{stsp}). Specifically, D_{stsp} is defined in the (potentially extended; Sauer et al. (1991)) observation space between estimated probability distributions $p(\boldsymbol{x})$, generated by trajectories of the true system, and $q(\boldsymbol{x})$, generated by orbits of the reconstructed system, see Appx. 6.4 for details.

Temporal Measure As in Mikhaeil et al. (2022), we check for invariant temporal agreement of ground truth and generated orbits by computing the Hellinger distance (H) between the power spectra of all dynamical variables (see Appx. 6.4 for details). The Hellinger distance is a proper metric with $0 \leq H \leq 1$, where 0 represents perfect agreement between its arguments. To produce a single number, the Hellinger distance is averaged across all dynamical variables of the observed system, referred to here as D_H.

Prediction error (PE) We further employ a short-term prediction error to assess each model's capability to perform forecasts on the chaotic time series. Note that, in general, accurate prediction of chaotic systems is an ill-posed problem as even tiny model approximation errors, differences in initial conditions, or noise sources are exponentially amplified, leading to rapid divergence from ground truth orbits. Hence, measures based on prediction accuracy are only of limited validity or unsuitable for assessing reconstruction quality, and only work for limited time horizons determined by the system's Lyapunov spectrum (Mikhaeil et al., 2022). To assess short-term predictability, here we compute a mean-squared n-step prediction error between n-step forward propagated initial states from the test set and their corresponding ground truth values, see Appx. 6.4.

4.3. Experimental evaluation

We compared DS reconstructions on the benchmarks defined above for the shPLRNN trained by GTF to a variety of other SOTA reconstruction methods, chosen to represent four major classes of model architectures and training strategies in use. These include DS reconstruction techniques based on 1) gated RNN models, specifically LSTMs trained using truncated BPTT (TBPTT) (Vlachas et al., 2018), 2) Reservoir Computing (RC) / Echo State Machines (Pathak et al., 2018), 3) library methods employing symbolic regression, namely Sparse Identification of Nonlinear Dynamical Systems (SINDy) (Brunton et al., 2016), and 4) ODE-based formulations like Neural ODEs (N-ODE) (Chen et al., 2018) and Long-Expressive-Memory (LEM) (Rusch et al., 2022)[6] (we also tested other N-ODE variants like Latent-ODE and ODE-RNN (Rubanova et al., 2019), with similar results, see Appx. 6.6). Moreover, we compare our method to the similar approach of Brenner et al. (2022) using dendPLRNNs trained with BPTT and a specific sparse TF protocol, dubbed identity TF (id-TF). For all comparison methods we determined optimal hyper-parameters through grid search (see Appx. 6.6), while trying to keep the total number of trainable parameters about the same (see Table 1).[7] For our shPLRNN we report in Table 1 results with both fixed GTF parameter α determined by grid search (see Fig. S3) as well as for aGTF, obtaining similar performance. Of course, GTF – like TF more generally – is only employed for model training, not during testing where systems evolve autonomously.

[6]LEMs, although not specifically designed for DS reconstruction, are universal approximators that have been exemplified on DS problems and are a current SOTA for addressing the EVGP.

[7]This was, however, not fully possible. For instance, RC required many more trainable parameters than other methods to yield any sensible reconstruction results.

Table 1. SOTA comparisons. Reported values are median $\pm$ median absolute deviation over 20 independent training runs. 'dim' refers to the model's state space dimensionality (number of dynamical variables). $|\boldsymbol{\theta}|$ denotes the total number of *trainable* parameters.

Dataset	Method	$D_{\text{stsp}} \downarrow$	$D_H \downarrow$	PE(20) $\downarrow$	dim	$\lvert\boldsymbol{\theta}\rvert$
ECG (5d)	shPLRNN + GTF	$\mathbf{4.3 \pm 0.6}$	$\mathbf{0.34 \pm 0.02}$	$(\mathbf{2.4 \pm 0.1}) \cdot \mathbf{10^{-3}}$	5	2785
	shPLRNN + aGTF	$\mathbf{4.5 \pm 0.4}$	$\mathbf{0.34 \pm 0.02}$	$(\mathbf{2.4 \pm 0.2}) \cdot \mathbf{10^{-3}}$	5	2785
	shPLRNN + STF	7.1 ± 1.8	0.38 ± 0.03	$(5 \pm 2) \cdot 10^{-3}$	5	2785
	dendPLRNN + id-TF	5.8 ± 0.6	0.37 ± 0.06	$(4.0 \pm 0.4) \cdot 10^{-3}$	35	3245
	RC	5.3 ± 1.7	0.39 ± 0.05	$(4 \pm 1) \cdot 10^{-3}$	1000	5000
	LSTM-TBPTT	15.2 ± 0.5	0.73 ± 0.02	$(2.5 \pm 0.5) \cdot 10^{-2}$	70	5920
	SINDy	diverging	diverging	diverging	5	3960
	N-ODE	12.2 ± 0.7	0.7 ± 0.03	$(4.1 \pm 0.1) \cdot 10^{-1}$	5	4955
	LEM	16.3 ± 0.2	0.56 ± 0.04	$(7.4 \pm 0.1) \cdot 10^{-1}$	62	4872
EEG (64d)	shPLRNN + GTF	$\mathbf{2.1 \pm 0.2}$	$\mathbf{0.11 \pm 0.01}$	$(5.5 \pm 0.1) \cdot 10^{-1}$	16	17952
	shPLRNN + aGTF	$\mathbf{2.4 \pm 0.2}$	$\mathbf{0.13 \pm 0.01}$	$(5.4 \pm 0.6) \cdot 10^{-1}$	16	17952
	shPLRNN + STF	14 ± 7	0.50 ± 0.16	$(\mathbf{2.5 \pm 0.3}) \cdot \mathbf{10^{-1}}$	16	17952
	dendPLRNN + id-TF	3 ± 1	0.13 ± 0.04	$(3.4 \pm 0.1) \cdot 10^{-1}$	105	18099
	RC	14 ± 7	0.54 ± 0.15	$(5.9 \pm 0.3) \cdot 10^{-1}$	448	28672
	LSTM-TBPTT	30 ± 21	0.2 ± 0.1	$(9.2 \pm 2.3) \cdot 10^{-1}$	160	51584
	SINDy	diverging	diverging	diverging	64	133120
	N-ODE	20 ± 0.5	0.47 ± 0.01	$(5.5 \pm 0.2) \cdot 10^{-1}$	64	17995
	LEM	10.2 ± 1.5	0.38 ± 0.06	$(8.2 \pm 0.6) \cdot 10^{-1}$	76	18304

As Table S3 in Appx. 6.6 reveals, shPLRNN+GTF outperforms, or performs about on par with, the other methods on the simulated DS benchmarks Lorenz-63, Lorenz-96, and multiscale Lorenz-96 (see also Figs. S6, S7, S8). However, as noted above, by now almost any recent DS reconstruction method performs reasonably well on these, essentially leading to a 'ceiling effect'. More importantly, shPLRNN+GTF has a clear edge over all other methods on the empirical EEG and ECG data, in some comparisons by almost an order of magnitude (Table 1).[8] This is significant, as DS reconstruction methods have so far mostly been tested on simulated DS models only, like the Lorenz equations, but rarely on empirical data. As noted above, real-world data usually contain multiple additional complexities not present in most simulated benchmarks. The reconstruction algorithm advanced here is apparently better able to deal with these various complicating factors met in empirical data.

This is particularly evident in Fig. 3 which provides examples of reconstructed time series on the EEG data for all of the compared methods, illustrating the trained models' *long-term* behavior (see Fig. S11 for all EEG channels, and Fig. S12 for ECG data). For creating these graphs, each method was initialized with a value inferred from the data, and then freely evolved forward for a long time (unconstrained by the data) from that one initial condition. As the underlying system is likely chaotic (according to an estimate of its maximal Lyapunov exponent and fractal dimensionality; Mikhaeil et al. (2022)), in each case reliable forecasts can only be obtained for a couple of time steps ahead. Importantly, however, the temporal structure of the real data is conserved in the simulated data for the shPLRNN+GTF, and a bit less so for dendPLRNN, but for none of the other methods which often just converge to an equilibrium point after some time, or even diverge. Thus, other methods were not able to reconstruct the systems' attractors (although they may still produce reasonable short-term predictions; cf. Tab. 1 & Fig. S13). At the same time, shPLRNN+GTF requires only 16 latent states to reconstruct the EEG, compared to 105 for dendPLRNN (and 64 for some of the others methods).

Finally, replacing GTF by the previously proposed STF (Mikhaeil et al., 2022) in shPLRNN training led to worse performance on the reconstruction measures (Table 1, shPLRNN+STF), suggesting an important role for GTF.[9]

5. Conclusions

In this paper we develop and test the idea of GTF for learning chaotic DS. GTF addresses the central challenge posed in Mikhaeil et al. (2022), namely that exploding gradients cannot be avoided by architectural or parameter constraints

[8] As observed previously (Brenner et al., 2022), SINDy severely struggles if the "right" functional terms are not in its library, as is likely for any empirical situation. Unlike for the Lorenz-63/96, we only observed diverging solutions for SINDy on the EEG and ECG data.

[9] As in Mikhaeil et al. (2022), the forcing interval was simply set according to the predictability time (ECG: $\tau_{pred} \approx 221$, EEG: $\tau_{pred} \approx 41$, see Appx. 6.3) and not by systematic grid search.

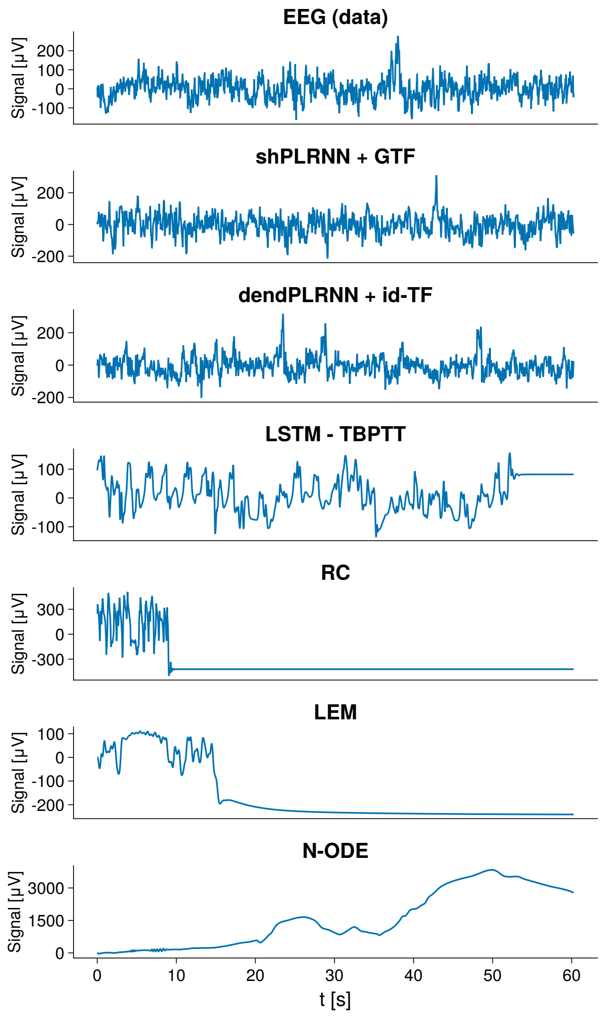


Figure 3. Example time traces of EEG reconstructions provided by the methods employed in Table 1. For each method we took the best model out of 20. For most runs ($\approx 90\%$), the shPLRNN trained with GTF and the dendPLRNN (Brenner et al., 2022) provide similarly good results, while all other methods have highly variable outcomes and overall poor reconstructions of the long-term behavior.

in RNNs trained on chaotic systems, and hence should be targeted at the level of the training procedure itself. Although the basic idea of GTF has been introduced a long time ago (Doya et al., 1992), to our knowledge it never has been systematically empirically tested or theoretically examined. Here, for the first time, we show this training strategy can be amended such that exploding gradients can be completely avoided along theoretically infinite-length trajectories, resolving the training issue for chaotic systems. Another mechanism by which GTF may enhance training is by smoothing the loss landscape (Fig. S10), as observed for other control and annealing methods (Abarbanel, 2013).

Especially when paired with a simple modification of the PLRNN, the shPLRNN, GTF results in a powerful DS reconstruction algorithm which outperforms a diversity of other models and techniques on the empirical datasets by sometimes large margins. At the same time, it returns a dynamically interpretable DS model for which fixed points and cycles can be determined semi-analytically (cf. 3.3), and which achieves the lowest-dimensional reconstructions of all methods tested, requiring at most as many dimensions as the observed system, potentially even less than empirically observed (Tab. 1). Although (or perhaps because) conceptually our advancements are simple, this sets a new standard in the field, especially for more relevant real-world data with which many algorithms profoundly struggle with. In fact, up to now most of the DS reconstruction literature has focused on simulated DS benchmarks only. While studying ground truth systems with known properties is clearly important, ultimately, of course, these algorithms are to be taken to the real world, where additional and partly unforeseeable issues may wait (some of them pointed out in sect. 4.1).

Chaotic dynamics are ubiquitous in natural and many human-made systems (Durstewitz & Gabriel, 2007a; Duarte et al., 2010; Faggini, 2014; Mangiarotti et al., 2020; Kamdjeu Kengne et al., 2021). In fact they are essentially the rule in any type of complex multi-component system with some heterogeneity in its elements or connections (Van Vreeswijk & Sompolinsky, 1996), including the brain (Durstewitz & Gabriel, 2007b), climate systems (Bury et al., 2021), or social and economical networks (Xu et al., 2011). By strictly controlling the gradients in training, without imposing any other constraints, GTF may prove valuable more generally for any time series prediction, regression or classification tasks. While in principle applicable to any other RNN architecture, however, we point out that it is also the specific design of the shPLRNN which may make it particularly well suited for straightforward implementation of GTF. For instance, we were not able to obtain similar performance boosts through GTF for the dendPLRNN (at least in a naive implementation) which lacks the shPLRNN's 1:1 relation between observations and latent states (nor did we achieve similar performance with $\tanh$ rather than ReLU activation; see Appx. 6.5). Thus, despite its general design, how to best utilize GTF in other model architectures, as well as how it compares to STF, will still require further research.

All code is available at `https://github.com/DurstewitzLab/GTF-shPLRNN`.

Acknowledgements

This work was funded by the German Research Foundation (DFG) within Germany's Excellence Strategy EXC 2181/1 – 390900948 (STRUCTURES), by DFG grants Du354/10-1 & Du354/15-1 to DD, and by the European Union Horizon-2020 consortium SC1-DTH-13-2020 (IMMERSE).

References

Abarbanel, H. *Predicting the future: completing models of observed complex systems*. Springer, 2013.

Ardizzone, L., Kruse, J., Rother, C., and Köthe, U. Analyzing inverse problems with invertible neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJed6j0cKX.

Arjovsky, M., Shah, A., and Bengio, Y. Unitary evolution recurrent neural networks. In *International conference on machine learning*, pp. 1120–1128. PMLR, 2016.

Bakarji, J., Champion, K., Kutz, J. N., and Brunton, S. L. Discovering governing equations from partial measurements with deep delay autoencoders. *arXiv preprint arXiv:2201.05136*, 2022.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

Bengio, Y., Simard, P., and Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Böttcher, A. and Wenzel, D. The frobenius norm and the commutator. *Linear algebra and its applications*, 429 (8-9):1864–1885, 2008.

Brenner, M., Hess, F., Mikhaeil, J. M., Bereska, L. F., Monfared, Z., Kuo, P.-C., and Durstewitz, D. Tractable dendritic RNNs for reconstructing nonlinear dynamical systems. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 2292–2320. PMLR, 2022.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

Brunton, S. L., Budisic, M., Kaiser, E., and Kutz, J. N. Modern koopman theory for dynamical systems. *SIAM Review*, 64(2):229–340, 2022.

Bury, T. M., Sujith, R. I., Pavithran, I., Scheffer, M., Lenton, T. M., Anand, M., and Bauch, C. T. Deep learning for early warning signals of tipping points. *Proceedings of the National Academy of Sciences*, 118(39):e2106140118, 2021. doi: 10.1073/pnas.2106140118. URL https://www.pnas.org/doi/10.1073/pnas.2106140118.

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

Chang, B., Chen, M., Haber, E., and Chi, E. H. AntisymmetricRNN: A dynamical system view on recurrent neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryxepo0cFX.

Chattopadhyay, A., Hassanzadeh, P., and Subramanian, D. Data-driven predictions of a multiscale lorenz 96 chaotic system using machine-learning methods: reservoir computing, artificial neural network, and long short-term memory network. *Nonlinear Processes in Geophysics*, 27(3):373–389, 2020.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Cooley, J. W. and Tukey, J. W. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301, 1965. ISSN 0025-5718. doi: 10.2307/2003354. URL https://www.jstor.org/stable/2003354. Publisher: American Mathematical Society.

Datseris, G. Dynamicalsystems.jl: A julia software library for chaos and nonlinear dynamics. *Journal of Open Source Software*, 3(23):598, mar 2018. doi: 10.21105/joss.00598. URL https://doi.org/10.21105/joss.00598.

de Silva, B., Champion, K., Quade, M., Loiseau, J.-C., Kutz, J., and Brunton, S. Pysindy: A python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, 5(49):2104, 2020. doi: 10.21105/joss.02104. URL https://doi.org/10.21105/joss.02104.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

Doya, K. et al. Bifurcations in the learning of recurrent neural networks 3. *learning (RTRL)*, 3:17, 1992.

Duarte, J., Januário, C., Martins, N., and Sardanyés, J. Quantifying chaos for ecological stoichiometry. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(3): 033105, September 2010.

Durstewitz, D. A state space approach for piecewise-linear recurrent neural networks for identifying computational dynamics from neural measurements. *PLoS computational biology*, 13(6):e1005542, 2017.

Durstewitz, D. and Gabriel, T. Dynamical basis of irregular spiking in nmda-driven prefrontal cortex neurons. *Cerebral cortex*, 17(4):894–908, 2007a.

Durstewitz, D. and Gabriel, T. Dynamical basis of irregular spiking in NMDA-driven prefrontal cortex neurons. *Cerebral Cortex (New York, N.Y.: 1991)*, 17(4):894–908, 2007b. ISSN 1047-3211. doi: 10.1093/cercor/bhk044.

Engelken, R., Wolf, F., and Abbott, L. F. Lyapunov spectra of chaotic recurrent neural networks. *arXiv preprint arXiv:2006.02427*, 2020.

Erichson, N. B., Azencot, O., Queiruga, A., Hodgkinson, L., and Mahoney, M. W. Lipschitz recurrent neural networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=-N7PBXqOUJZ.

Faggini, M. Chaotic time series analysis in economics: Balance and perspectives. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 24(4):042101, 2014. Publisher: American Institute of Physics.

Funahashi, K.-i. and Nakamura, Y. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6):801–806, 1993.

Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23): e215–e220, 2000.

Govindan, R., Narayanan, K., and Gopinathan, M. On the evidence of deterministic chaos in ecg: Surrogate and predictability analysis. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(2):495–502, 1998.

Hanson, J. and Raginsky, M. Universal simulation of stable dynamical systems by recurrent neural nets. In *Learning for Dynamics and Control*, pp. 384–392. PMLR, 2020.

Hernandez, D., Moretti, A. K., Saxena, S., Wei, Z., Cunningham, J., and Paninski, L. Nonlinear evolution via spatially-dependent linear dynamics for electrophysiology and calcium data. *Neurons, Behavior, Data analysis, and Theory*, 3(3):13476, 2020.

Hershey, J. R. and Olsen, P. A. Approximating the kullback leibler divergence between gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pp. IV–317. IEEE, 2007.

Higham, N. J. and Lin, L. A schur-padé algorithm for fractional powers of a matrix. *SIAM Journal on Matrix Analysis and Applications*, 32(3):1056–1078, 2011.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

Inoue, K., Ohara, S., Kuniyoshi, Y., and Nakajima, K. Transient chaos in bert. *arXiv:2106.03181*, 2021.

Jordan, M. I. Attractor dynamics and parallelism in a connectionist sequential machine. In *Artificial neural networks: concept learning*, pp. 112–127. IEEE Press, 1990.

Kamdjeu Kengne, L., Mboupda Pone, J. R., and Fotsin, H. B. On the dynamics of chaotic circuits based on memristive diode-bridge with variable symmetry: A case study. *Chaos, Solitons & Fractals*, 145:110795, 2021.

Kantz, H. A robust method to estimate the maximal lyapunov exponent of a time series. *Physics letters A*, 185 (1):77–87, 1994.

Kaptanoglu, A. A., de Silva, B. M., Fasel, U., Kaheman, K., Goldschmidt, A. J., Callaham, J., Delahunt, C. B., Nicolaou, Z. G., Champion, K., Loiseau, J.-C., Kutz, J. N., and Brunton, S. L. Pysindy: A comprehensive python package for robust sparse system identification. *Journal of Open Source Software*, 7(69):3994, 2022. doi: 10.21105/joss.03994. URL https://doi.org/10.21105/joss.03994.

Kesmia, M., Boughaba, S., and Jacquir, S. Control of continuous dynamical systems modeling physiological states. *Chaos, Solitons & Fractals*, 136:109805, 2020.

Kimura, M. and Nakano, R. Learning dynamical systems by recurrent neural networks from orbits. *Neural Networks*, 11(9):1589–1599, 1998.

Kirchmeyer, M., Yin, Y., Dona, J., Baskiotis, N., Rakotomamonjy, A., and Gallinari, P. Generalizing to new physical systems via context-informed dynamics model. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11283–11301. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/kirchmeyer22a.html.

Koppe, G., Toutounji, H., Kirsch, P., Lis, S., and Durstewitz, D. Identifying nonlinear dynamical systems via generative recurrent neural networks with applications to fmri. *PLoS computational biology*, 15(8):e1007263, 2019.

Kramer, D., Bommer, P. L., Durstewitz, D., Tombolini, C., and Koppe, G. Reconstructing nonlinear dynamical systems from multi-modal time series. In *International Conference on Machine Learning*, pp. 11613–11633. PMLR, 2022.

Krämer, K.-H., Datseris, G., Kurths, J., Kiss, I. Z., Ocampo-Espindola, J. L., and Marwan, N. A unified and automated approach to attractor reconstruction. *New Journal of Physics*, 23(3):033017, 2021.

Lejarza, F. and Baldea, M. Data-driven discovery of the governing equations of dynamical systems via moving horizon optimization. *Scientific Reports*, 12(1):1–15, 2022.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. On the variance of the adaptive learning rate and beyond. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*, April 2020.

Lorenz, E. N. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

Lorenz, E. N. Predictability: A problem partly solved. In *Proc. Seminar on predictability*, volume 1, 1996.

Lusch, B., Kutz, J. N., and Brunton, S. L. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):1–10, 2018.

Mangiarotti, S., Peyre, M., Zhang, Y., Huc, M., Roger, F., and Kerr, Y. Chaos theory applied to the outbreak of COVID-19: an ancillary approach to decision making in pandemic context. *Epidemiology and Infection*, 148:e95, 2020.

Mikhaeil, J. M., Monfared, Z., and Durstewitz, D. On the difficulty of learning chaotic dynamics with RNNs. In *Proceedings of the 36th Conference on Neural Information Processing Systems*. NeurIPS, 2022.

Monfared, Z. and Durstewitz, D. Transformation of relu-based recurrent neural networks from discrete-time to continuous-time. In *International Conference on Machine Learning*, pp. 6999–7009. PMLR, 2020.

Olsen, L. F., Truty, G. L., and Schaffer, W. M. Oscillations and chaos in epidemics: a nonlinear dynamic study of six childhood diseases in Copenhagen, Denmark. *Theoretical Population Biology*, 33(3):344–370, 1988.

Otto, S. E. and Rowley, C. W. Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, 2019. doi: 10.1137/18M1177846. URL https://epubs.siam.org/doi/abs/10.1137/18M1177846. Publisher: Society for Industrial and Applied Mathematics.

Patel, D. and Ott, E. Using machine learning to anticipate tipping points and extrapolate to post-tipping dynamics of non-stationary dynamical systems. *arXiv preprint arXiv:2207.00521*, 2022.

Pathak, J., Hunt, B., Girvan, M., Lu, Z., and Ott, E. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical review letters*, 120(2):024102, 2018.

Pearlmutter, B. A. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1(2):263–269, 1989.

Perko, L. *Differential Equations and Dynamical Systems*, volume 7. Springer, New York, NY, 2001.

Pineda, F. J. Dynamics and architecture for neural computation. *Journal of Complexity*, 4(3):216–245, 1988.

Raissi, M. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

Reiss, A., Indlekofer, I., Schmidt, P., and Van Laerhoven, K. Deep ppg: Large-scale heart rate estimation with convolutional neural networks. *Sensors*, 19(14), 2019. ISSN 1424-8220. doi: 10.3390/s19143079. URL https://www.mdpi.com/1424-8220/19/14/3079.

Ricci, M., Moriel, N., Piran, Z., and Nitzan, M. Phase2vec: Dynamical systems embedding with a physics-informed convolutional network. *arXiv preprint arXiv:2212.03857*, 2022.

Rubanova, Y., Chen, R. T., and Duvenaud, D. K. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Rusch, T. K. and Mishra, S. Coupled oscillatory recurrent neural network (cornn): An accurate and (gradient) stable architecture for learning long time dependencies. *arXiv preprint arXiv:2010.00951*, 2020.

Rusch, T. K. and Mishra, S. Coupled oscillatory recurrent neural network (co{rnn}): An accurate and (gradient) stable architecture for learning long time dependencies. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=F3s69XzWOia.

Rusch, T. K., Mishra, S., Erichson, N. B., and Mahoney, M. W. Long expressive memory for sequence modeling. In *International Conference on Learning Representations*, 2022.

Sauer, T., Yorke, J. A., and Casdagli, M. Embedology. *Journal of statistical Physics*, 65(3):579–616, 1991.

Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., and Wolpaw, J. R. BCI2000: a general-purpose brain-computer interface (BCI) system. *IEEE transactions on bio-medical engineering*, 51(6):1034–1043, 2000. ISSN 0018-9294. doi: 10.1109/TBME.2004.827072.

Schmidt, D., Koppe, G., Monfared, Z., Beutelspacher, M., and Durstewitz, D. Identifying nonlinear dynamical systems with multiple time scales and long-range dependencies. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=_XYzwxPIQu6.

Skokos, C. H., Gottwald, G. A., and Laskar, J. *Chaos Detection and Predictability*, volume 1. Springer, 2016.

Smith, J., Linderman, S., and Sussillo, D. Reverse engineering recurrent neural networks with jacobian switching linear dynamical systems. *Advances in Neural Information Processing Systems*, 34:16700–16713, 2021.

Takens, F. Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980*, pp. 366–381. Springer, 1981.

Thornes, T., Düben, P., and Palmer, T. On the use of scale-dependent precision in earth system modelling. *Quarterly Journal of the Royal Meteorological Society*, 143(703): 897–908, 2017.

Van Vreeswijk, C. and Sompolinsky, H. Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science (New York, N.Y.)*, 274(5293):1724–1726, 1996. ISSN 0036-8075. doi: 10.1126/science.274.5293.1724.

Vlachas, P. R. and Koumoutsakos, P. Learning from predictions: Fusing training and autoregressive inference for long-term spatiotemporal forecasts. *arXiv preprint arXiv:2302.11101*, 2023.

Vlachas, P. R., Byeon, W., Wan, Z. Y., Sapsis, T. P., and Koumoutsakos, P. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2213): 20170844, 2018.

Vlachas, P. R., Pathak, J., Hunt, B. R., Sapsis, T. P., Girvan, M., Ott, E., and Koumoutsakos, P. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126:191–217, 2020.

Voss, H. U., Timmer, J., and Kurths, J. Nonlinear dynamical system identification from uncertain and indirect measurements. *International Journal of Bifurcation and Chaos*, 14(06):1905–1933, 2004.

Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

Williams, R. J. and Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Xu, X., Chen, Z., Si, G., Hu, X., Jiang, Y., and Xu, X. The chaotic dynamics of the social behavior selection networks in crowd simulation. *Nonlinear Dynamics*, 64 (1):117–126, 2011. ISSN 1573-269X. doi: 10.1007/s11071-010-9850-z. URL https://doi.org/10.1007/s11071-010-9850-z.

Zhao, Y. and Park, I. M. Variational online learning of neural dynamics. *Frontiers in computational neuroscience*, 14: 71, 2020.

6. Appendix

6.1. Theorems: Proofs

6.1.1. Proof of proposition 1

Proof. Let $\mathcal{J} = \{\tilde{\boldsymbol{J}}_\kappa\}_{\kappa \in \mathcal{K}}$ be the set of all Jacobians of the RNN (1), and $\mathcal{O}_{\boldsymbol{z}_1} = \{\boldsymbol{z}_T\}_{T=1}^{\infty}$ be a chaotic orbit of the system. Then, the largest Lyapunov exponent of $\mathcal{O}_{\boldsymbol{z}_1}$ is positive, i.e.

$$\lambda = \lim_{T\to\infty} \frac{1}{T} \log \left\| \tilde{\boldsymbol{J}}_T \, \tilde{\boldsymbol{J}}_{T-1} \cdots \tilde{\boldsymbol{J}}_2 \right\| > 0, \tag{24}$$

which implies

$$\lim_{T\to\infty} \left\| \tilde{\boldsymbol{J}}_T \, \tilde{\boldsymbol{J}}_{T-1} \cdots \tilde{\boldsymbol{J}}_2 \right\| = \lim_{T\to\infty} \left\| D\boldsymbol{F}_{\boldsymbol{\theta}}^{T}(\boldsymbol{z}_1) \right\| = \infty. \tag{25}$$

Accordingly,

$$\exists\, \hat{n} \in \mathbb{N} \quad s.t. \quad \forall m \geq \hat{n} \qquad \left\| D\boldsymbol{F}_{\boldsymbol{\theta}}^{m}(\boldsymbol{z}_1) \right\| > 1. \tag{26}$$

Therefore $\left\| \tilde{\boldsymbol{J}}_s \right\| > 1$ for some $s \in \{2, 3, \cdots, m\}$ $(m \geq \hat{n})$. Otherwise

$$\left\| D\boldsymbol{F}_{\boldsymbol{\theta}}^{m}(\boldsymbol{z}_1) \right\| = \left\| \tilde{\boldsymbol{J}}_m \, \tilde{\boldsymbol{J}}_{m-1} \cdots \tilde{\boldsymbol{J}}_2 \right\| \leq \left\| \tilde{\boldsymbol{J}}_m \right\| \left\| \tilde{\boldsymbol{J}}_{m-1} \right\| \cdots \left\| \tilde{\boldsymbol{J}}_2 \right\| \leq 1, \tag{27}$$

which is in contradiction to eq. (26). Since $\tilde{\boldsymbol{J}}_s \in \mathcal{J}$, so $\tilde{\sigma}_{\max} = \sup \left\{ \left\| \tilde{\boldsymbol{J}}_k \right\| = \sigma_{\max}(\tilde{\boldsymbol{J}}_k) \, : \, \tilde{\boldsymbol{J}}_k \in \mathcal{J} \right\} > 1.$ □

6.1.2. Proof of proposition 2

Proof. (i) According to (8) we have

$$\left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 = \left\| (1-\alpha)^{t-r} \prod_{k=0}^{t-r-1} \tilde{\boldsymbol{J}}_{t-k} \right\|_2 \leq (1-\alpha)^{t-r} \prod_{k=0}^{t-r-1} \left\| \tilde{\boldsymbol{J}}_{t-k} \right\|_2 \leq \left[(1-\alpha)\tilde{\sigma}_{\max} \right]^{t-r}, \tag{28}$$

and

$$\begin{aligned} \left[(1-\alpha)\tilde{\lambda}_{\min} \right]^{t-r} &\leq (1-\alpha)^{t-r} \prod_{k=0}^{t-r-1} \lambda_{\min}(\tilde{\boldsymbol{J}}_{t-k}) \leq (1-\alpha)^{t-r} \lambda_{\min}\Big(\prod_{k=0}^{t-r-1} \tilde{\boldsymbol{J}}_{t-k} \Big) \\ &\leq (1-\alpha)^{t-r} \rho\Big(\prod_{k=0}^{t-r-1} \tilde{\boldsymbol{J}}_{t-k} \Big) \leq \left\| (1-\alpha)^{t-r} \prod_{k=0}^{t-r-1} \tilde{\boldsymbol{J}}_{t-k} \right\|_2 = \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 . \end{aligned} \tag{29}$$

Therefore,

$$\left[(1-\alpha)\tilde{\lambda}_{\min} \right]^{t-r} \leq \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 \leq \left[(1-\alpha)\tilde{\sigma}_{\max} \right]^{t-r}. \tag{30}$$

Inserting $\alpha = \alpha^* = 1 - \frac{1}{\tilde{\sigma}_{\max}}$ (for $\tilde{\sigma}_{\max} \geq 1$) into the r.h.s. of (30) gives

$$\lim_{t\to\infty} \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 \leq \lim_{t\to\infty} \left[\Big(\frac{1}{\tilde{\sigma}_{\max}} \Big) \tilde{\sigma}_{\max} \right]^{t-r} = 1, \tag{31}$$

and so $\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}$ will not diverge for $t \to \infty$.

(ii) If $\tilde{\gamma} = 1$, then $\tilde{\sigma}_{\max} = \tilde{\lambda}_{\min} \geq 1$. Hence, substituting $\alpha^* = 1 - \frac{1}{\tilde{\sigma}_{\max}} = 1 - \frac{1}{\tilde{\lambda}_{\min}}$ in (30) yields

$$\lim_{t\to\infty} \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 = 1. \tag{32}$$

For $\tilde{\gamma} \neq 1$, inserting $\alpha^* = 1 - \frac{1}{\tilde{\sigma}_{\max}}$ in (30) results in

$$0 = \lim_{t\to\infty} \big(\tilde{\gamma}\big)^{t-r} \leq \lim_{t\to\infty} \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 \leq 1, \tag{33}$$

and so $\lim_{t\to\infty} \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2$ may go to zero for $t \to \infty$. Moreover, obviously, the closer $\tilde{\gamma}$ is to 1, the slower $\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r}$ may vanish as $t \to \infty$. □

Remark 1. According to Rademacher's theorem, for Lipschitz-continuous RNNs $F_{\boldsymbol{\theta}}$ is differentiable almost everywhere and has a bounded derivative. Therefore, for such RNNs, the nonempty set $\mathcal{S}_1$ (defined in (9)) is bounded from above and so always has a supremum in $\mathbb{R}$.

Remark 2. Assume that

$$\tilde{\sigma}_{\min} = \inf \left\{ \sigma_{\min}(\tilde{\boldsymbol{J}}_k) \, : \, \tilde{\boldsymbol{J}}_k \in \mathcal{J} \right\} \geq 0. \tag{34}$$

Since $\tilde{\lambda}_{\min} \geq \tilde{\sigma}_{\min} \geq 0$, from (30) we have

$$\big[(1-\alpha)\tilde{\sigma}_{\min}\big]^{t-r} \leq \big[(1-\alpha)\tilde{\lambda}_{\min}\big]^{t-r} \leq \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{z}_r} \right\|_2 \leq \big[(1-\alpha)\tilde{\sigma}_{\max}\big]^{t-r}. \tag{35}$$

6.1.3. Proof of proposition 3

Proof. First, the dendPLRNN (11) can be rewritten in the following form:

$$\boldsymbol{z}_t = \boldsymbol{W}^B_{\Omega(t-1)} \, \boldsymbol{z}_{t-1} + \boldsymbol{W} \, \boldsymbol{h}^B_{\Omega(t-1)} + \boldsymbol{h}_0, \tag{36}$$

in which

$$\begin{aligned} \boldsymbol{D}^B_{\Omega(t-1)} &:= \sum_{b=1}^{B} \alpha_b \, \boldsymbol{D}^{(b)}_{\Omega(t-1)}, \\ \boldsymbol{h}^B_{\Omega(t-1)} &:= \sum_{b=1}^{B} \alpha_b \, \boldsymbol{D}^{(b)}_{\Omega(t-1)}(-\boldsymbol{h}_b), \\ \boldsymbol{W}^B_{\Omega(t-1)} &:= \boldsymbol{A} + \boldsymbol{W} \, \boldsymbol{D}^B_{\Omega(t-1)}, \end{aligned} \tag{37}$$

and $\boldsymbol{D}^{(b)}_{\Omega(t-1)} = \mathrm{diag}\big(d^{(b)}_{1,t-1}, d^{(b)}_{2,t-1}, \cdots, d^{(b)}_{M,t-1}\big)$ are diagonal binary indicator matrices with $d^{(b)}_{m,t-1} = 1$ if $z_{m,t-1} > h_{m,b}$ and 0 otherwise. Similarly, the shPLRNN (12) can be brought into the form

$$\begin{aligned} \boldsymbol{z}_t &= \Big(\boldsymbol{A} + \boldsymbol{W}_1 \, \tilde{\boldsymbol{D}}_{\Omega(t-1)} \boldsymbol{W}_2\Big) \boldsymbol{z}_{t-1} + \boldsymbol{W}_1 \tilde{\boldsymbol{D}}_{\Omega(t-1)} \boldsymbol{h}_2 + \boldsymbol{h}_1 \\ &=: \tilde{\boldsymbol{W}}_{\Omega(t-1)} \boldsymbol{z}_{t-1} + \boldsymbol{W}_1 \tilde{\boldsymbol{h}}_{\Omega(t-1)} + \boldsymbol{h}_1, \end{aligned} \tag{38}$$

where $\tilde{\boldsymbol{D}}_{\Omega(t-1)} = \mathrm{diag}\big(\tilde{d}_{1,t-1}, \tilde{d}_{2,t-1}, \cdots, \tilde{d}_{L,t-1}\big)$ denotes an $L \times L$ diagonal binary indicator matrix with $d_{l,t-1} = 1$ if $\sum_{j=1}^{M} w^{(2)}_{lj} \, z_{j,t-1} > -h^{(2)}_l$ and 0 otherwise, where we used the notation $\boldsymbol{W}_2 = \big[w^{(2)}_{ij}\big]$ and $\boldsymbol{h}_2 = \big[h^{(2)}_i\big]$. In other words, it can be rewritten as the dendPLRNN (36). Consequently, fixed points of (12) can be computed analogously to dendPLRNNs as

$$\boldsymbol{z}^{*1} = \Big(\boldsymbol{I} - \tilde{\boldsymbol{W}}_{\Omega(t^{*1})}\Big)^{-1} \Big[\boldsymbol{W}_1 \, \tilde{\boldsymbol{h}}_{\Omega(t^{*1})} + \boldsymbol{h}_1\Big], \tag{39}$$

where $\boldsymbol{z}^{*1} = \boldsymbol{z}_{t^{*1}} = \boldsymbol{z}_{t^{*1}-1}$, and $\det(\boldsymbol{I} - \tilde{\boldsymbol{W}}_{\Omega(t^{*1})}) = P_{\tilde{\boldsymbol{W}}_{\Omega(t^{*1})}}(1) \neq 0$, i.e. $\tilde{\boldsymbol{W}}_{\Omega(t^{*1})}$ has no eigenvalue equal to 1 (otherwise we are dealing with a bifurcation or with a continuous set of marginally stable points). The same holds for cycles of (12): Letting $t + n - 1 =: t^{*n}$, the periodic point $\boldsymbol{z}^{*n}$ of an n-cycle is

$$\begin{aligned}\boldsymbol{z}^{*n} \;=\; & \Big(\boldsymbol{I} - \prod_{i=1}^{n} \tilde{\boldsymbol{W}}_{\Omega(t^{*n}-i)}\Big)^{-1} \Big(\sum_{j=2}^{n} \Big[\prod_{i=1}^{n-j+1} \tilde{\boldsymbol{W}}_{\Omega(t^{*n}-i)} \boldsymbol{W}_1\, \tilde{\boldsymbol{h}}_{\Omega(t^{*n}-n+j-2)}\Big] \\ & + \boldsymbol{W}_1\, \tilde{\boldsymbol{h}}_{\Omega(t^{*n}-1)} \;+\; \Big(\sum_{j=2}^{n} \prod_{i=1}^{n-j+1} \tilde{\boldsymbol{W}}_{\Omega(t^{*n}-i)} + \boldsymbol{I}\Big)\boldsymbol{h}_1\Big), \end{aligned} \tag{40}$$

where $\boldsymbol{z}^{*n} = \boldsymbol{z}_{t^{*n}} = \boldsymbol{z}_{t^{*n}-n}$, if $(\boldsymbol{I} - \prod_{i=1}^{n} \tilde{\boldsymbol{W}}_{\Omega(t^{*n}-i)})$ is invertible, i.e.

$$\det\Big(\boldsymbol{I} - \prod_{i=1}^{n} \tilde{\boldsymbol{W}}_{\Omega(t^{*n}-i)}\Big) = P_{\prod_{i=1}^{n} \tilde{\boldsymbol{W}}_{\Omega(t^{*n}-i)}}(1) \neq 0.$$

Now consider an M-dimensional dendPLRNN given by (11) with B bases and $L = M \cdot B$. Furthermore, set $\widetilde{\boldsymbol{A}} := \boldsymbol{A}$, $\widetilde{\boldsymbol{h}}_1 := \boldsymbol{h}_0$, and

$$\widetilde{\boldsymbol{W}}_1 := \begin{bmatrix}\boldsymbol{W}\alpha_1 & \boldsymbol{W}\alpha_2 & \dots & \boldsymbol{W}\alpha_B\end{bmatrix} \in \mathbb{R}^{M\times L}, \tag{41}$$

$$\widetilde{\boldsymbol{W}}_2 := \begin{bmatrix}\mathbb{1}_{M\times M}\\ \mathbb{1}_{M\times M}\\ \vdots \\ \mathbb{1}_{M\times M}\end{bmatrix} \in \mathbb{R}^{L\times M}, \qquad \widetilde{\boldsymbol{h}}_2 := -\begin{bmatrix}\boldsymbol{h}_1\\ \boldsymbol{h}_2\\ \vdots\\ \boldsymbol{h}_B\end{bmatrix} \in \mathbb{R}^{L}. \tag{42}$$

Then it follows that eq. (11) can be written as an M-dimensional shPLRNN with hidden layer size L,

$$\boldsymbol{z}_t = \widetilde{\boldsymbol{A}}\boldsymbol{z}_{t-1} + \widetilde{\boldsymbol{W}}_1 \phi\left(\widetilde{\boldsymbol{W}}_2 \boldsymbol{z}_{t-1} + \widetilde{\boldsymbol{h}}_2\right) + \widetilde{\boldsymbol{h}}_1, \tag{43}$$

which completes the proof. □

6.1.4. PROOF OF BOUNDED ORBITS OF (13)

Proposition 4. *If $\rho(\boldsymbol{A}) = \|\boldsymbol{A}\| < 1$, then every orbit of the clipped shPLRNN (13) will be bounded.*

Proof. Consider the clipped shPLRNN of the form

$$\begin{aligned}\boldsymbol{z}_t \;&=\; \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}_1\big[\phi(\boldsymbol{W}_2\boldsymbol{z}_{t-1} + \boldsymbol{h}_2) - \phi\left(\boldsymbol{W}_2\boldsymbol{z}_{t-1}\right)\big] + \boldsymbol{h}_1 \\ &:=\; \boldsymbol{A}\,\boldsymbol{z}_{t-1} \;+\; \boldsymbol{W}_1\,\psi(\boldsymbol{z}_{t-1}) \;+\; \boldsymbol{h}_1. \end{aligned} \tag{44}$$

Obviously, for $\boldsymbol{h}_2 = \big[h_l^{(2)}\big]$ and every $l \in \{1, 2, \cdots, L\}$

$$\psi_l(\boldsymbol{z}_{t-1}) \;=\; \max\big(0, \sum_{j=1}^{M} w_{lj}^{(2)}\, z_{j,t-1} + h_l^{(2)}\big) - \max\big(0, \sum_{j=1}^{M} w_{lj}^{(2)}\, z_{j,t-1}\big) \;\leq\; \big|h_l^{(2)}\big|. \tag{45}$$

This implies

$$\|\psi(\boldsymbol{z}_{t-1})\| \;=\; \sqrt{\sum_{l=1}^{L} \big(\psi_l(z_{t-1})\big)^2} \;\leq\; \sqrt{L}\, h_{\max} \;:=\; \bar{h}_{\max}, \tag{46}$$

where $h_{\max} \;=\; \max\limits_{1\leq l\leq L} \big\{\big|h_l^{(2)}\big|\big\}$.

Let $\{z_1, z_2, \cdots, z_t, \cdots\}$ be an orbit of (44). For $T \in \mathbb{N}$, recursively computing $z_2, z_3, \cdots, z_T$ yields

$$\begin{aligned} z_2 &= A\, z_1 + W_1\, \psi(z_1) + h_1 \\ z_3 &= A^2\, z_1 + A\, W_1\, \psi(z_1) + W_1\, \psi(z_2) + \big[A + I\big] h_1 \\ &\vdots \\ z_T &= A^{T-1}\, z_1 + \sum_{j=0}^{T-2} A^j\, W_1\, \psi(z_{T-1-j}) + \sum_{j=0}^{T-2} A^j\, h_1. \end{aligned} \tag{47}$$

Hence, due to (46), one concludes that

$$\|z_T\| \leq \|A\|^{T-1} \|z_1\| + \bar{h}_{\max} \|W_1\| \sum_{j=0}^{T-2} \|A\|^j + \sum_{j=0}^{T-2} \|A\|^j \|h_1\|. \tag{48}$$

If $\|A\| < 1$, then $\lim_{T\to\infty} \|A\|^{T-1} = 0$, and so

$$\lim_{T\to\infty} \|z_T\| \leq \bar{h}_{\max} \|W_1\| \sum_{j=0}^{\infty} \|A\|^j + \sum_{j=0}^{\infty} \|A\|^j \|h_1\| = \frac{\bar{h}_{\max} \|W_1\| + \|h_1\|}{1 - \|A\|} < \infty, \tag{49}$$

which completes the proof. □

6.2. Training Protocol

General procedure Given a times series $x_{1:T}$, we train the model using BPTT with GTF in the following manner: Per epoch, we sample sub-sequences of length $\tilde{T}$ from the observed orbit, $\tilde{x}^{(p)}_{1:\tilde{T}} := x_{t_p:t_p+\tilde{T}}$, where $t_p \in [1, T - \tilde{T}]$ is drawn at random. We arrange multiple sequences in a batch $\left\{\tilde{x}^{(p)}_{1:\tilde{T}}\right\}_{p=1}^{S}$, where S denotes the batch size. For a single parameter update, we first estimate forcing signals for the entire batch using Eq. (15) applied to each time step in each sequence, i.e. $\hat{z}^{(p)}_t = G^{-1}_{\varphi}(\tilde{x}^{(p)}_t)$. We then take the estimated teacher signal for the first time step of each batch, $\hat{z}^{(p)}_1$, as the initial condition for the RNN, which we propagate forward in time according to $z_t = F_{\theta}(\tilde{z}_{t-1})$ (see sect. 3.3) to produce predictions $z^{(p)}_{2:\tilde{T}}$. These are mapped back into observation space via $\hat{\tilde{x}}^{(p)}_t = G_{\varphi}(z^{(p)}_t)$. The MSE between ground truth and predicted orbits is then minimized according to

$$\mathcal{L}_{\text{MSE}}\left(\left\{\tilde{x}^{(p)}_{2:\tilde{T}}\right\}, \left\{\hat{\tilde{x}}^{(p)}_{2:\tilde{T}}\right\}\right) = \frac{1}{S(\tilde{T}-1)} \sum_{p=1}^{S} \sum_{t=2}^{\tilde{T}} \left\| \tilde{x}^{(p)}_t - \hat{\tilde{x}}^{(p)}_t \right\|_2^2. \tag{50}$$

In all experiments, we use the RAdam (Liu et al., 2020) optimizer with a learning rate starting at 10^{-3} which is exponentially reduced to reach 10^{-6} at the end of training. For all datasets, we trained for 5000 epochs, where one epoch is defined as processing of 50 batches of size $S = 16$. This comes down to a total of $250,000$ parameter updates in each training run. For the Lorenz-63 and Lorenz-96 and the empirical ECG data, we used a sequence length of $\tilde{T} = 200$. For the EEG data, we used only $\tilde{T} = 50$.

aGTF annealing protocol Note that the Jacobians $\tilde{J}_t$ in Eq. (22) implicitly depend on α. Hence, we replace the Jacobians of the forced model by data-inferred Jacobians $\hat{J}_t$ evaluated at estimated teacher signals $\hat{z}_t$ in latent space. This approximation generally holds when the model is either strongly forced or when the model dynamics is already close to that of the observed teacher system. To approach this scenario, we employ an α-annealing protocol which starts with strong forcing at the beginning of training when the model is still far off from the observed system, and then smoothly decreases forcing throughout training depending on the model's Jacobians as the observed system is captured increasingly better. This procedure is formally described in Algorithm 1. Given a batch of teacher signals, we first compute Jacobians at each time step for each sequence in the batch and the norm of Eq. (22). We then set $\alpha = 0$ if the maximum norm for the current batch is smaller than 1, and else compute α according to (23). Starting with strong forcing at the beginning of training, i.e.

$\alpha_0 = 1$, α_n is set up to decay exponentially over the course of training until it hits a lower bound given by the norm of (22) of the converged model. This is achieved by using an exponential moving average for α_n throughout training, which is controlled by the hyperparameter γ:

$$\alpha_n = \begin{cases} (1-\gamma)\alpha + \gamma\alpha_{n-1}, & \text{if } \alpha < \alpha_{n-1} \\ \alpha, & \text{otherwise,} \end{cases} \tag{51}$$

where the index-free α is given by Eq. (23). Note that we replace α_n with the estimated α of the current training batch if it exceeds the current value α_{n-1}. This is to avoid exploding gradients caused by sudden bifurcations during RNN training.

Algorithm 1 Adaptive GTF

```
Require: $\alpha_{n-1}$ estimate at previous optimization step, update interval $k$, decay rate $\gamma$, batch of forcing signals $\{\hat{\boldsymbol{z}}_{1:T}^{(p)}\}_{p=1}^{S}$,
    RNN $\boldsymbol{F}_{\theta_n}$
 1: if $n \mod k == 0$ then                                  ▷ Update only every $k$-th opt. step
 2:     $\hat{\boldsymbol{J}}_t^{(p)} = Jacobian(\boldsymbol{F}_{\theta_n}, \hat{\boldsymbol{z}}_t^{(p)})$      ▷ Compute Jacobians manually or via automatic differentiation
 3:     $\kappa = \max_p \left\| \mathcal{G}(\hat{\boldsymbol{J}}_{T:2}^{(p)}) \right\|$              ▷ Approximate $\|\mathcal{G}\|$ using one of (52), (53), (54)
 4:     $\alpha = \max\left(0, 1 - \frac{1}{\kappa}\right)$                 ▷ Set $\alpha = 0$ when Jacobians converge
 5:     if $\alpha > \alpha_{n-1}$ then
 6:         $\alpha_n = \alpha$
 7:     else
 8:         $\alpha_n = (1-\gamma)\alpha + \gamma\alpha_{n-1}$
 9:     end if
10: else
11:     $\alpha_n = \alpha_{n-1}$
12: end if
```

To reduce the resulting training slow-down caused mainly by computation of Jacobians for each and every batch, one could update α_n only every k-th optimization step. On the other hand, too sparse α-updates may lead to rather lazy reactions to sudden bifurcations in RNN dynamics. In practice, we found $k = 5$ to work well, not significantly harming convergence in training. For all experiments on aGTF we fixed $k = 5$, $\alpha_0 = 1$, and $\gamma = 0.999$ (i.e., these parameters do not need a grid search but could simply be set to the reasonable ad-hoc values used here).

Approximation of $\mathcal{G}$ To determine the optimal forcing α (23), we need an efficient and numerically stable estimate of Eq. (22). To this end, we approximate the principal logarithm of the Jacobian product by a sum of principal logarithms:

$$\mathcal{G}(\hat{\boldsymbol{J}}_{T:2}) = \left(\prod_{k=0}^{T-2} \hat{\boldsymbol{J}}_{T-k}\right)^{\frac{1}{T-1}} \overset{①}{=} \exp\left(\frac{1}{T-1}\log\left(\prod_{k=0}^{T-2}\hat{\boldsymbol{J}}_{T-k}\right)\right) \approx \exp\left(\frac{1}{T-1}\sum_{t=2}^{T}\log\hat{\boldsymbol{J}}_t\right). \tag{52}$$

Note that equality ① strictly holds if the Jacobian product is *non-singular* (Higham & Lin, 2011). The approximation made here is exact if Jacobians $\hat{\boldsymbol{J}}_{T:2}$ commute under multiplication. While this assumption does not hold in general, we find that at least for the shPLRNN trained on the systems and data introduced in this work, Jacobians approximately commute (Fig. S1). Furthermore, in practice we find that in most cases the arithmetic mean of Jacobians

$$\mathcal{G}\left(\hat{\boldsymbol{J}}_{T:2}\right) \approx \frac{1}{T-1}\sum_{t=2}^{T}\hat{\boldsymbol{J}}_t \tag{53}$$

can be used as a plug-in replacement for the approximation above, as it produces similar α estimates, as shown in Fig. S2. This replacement is mainly motivated by the runtime improvement for the arithmetic mean over the matrix exponentiation and logarithms involved in computing (52).

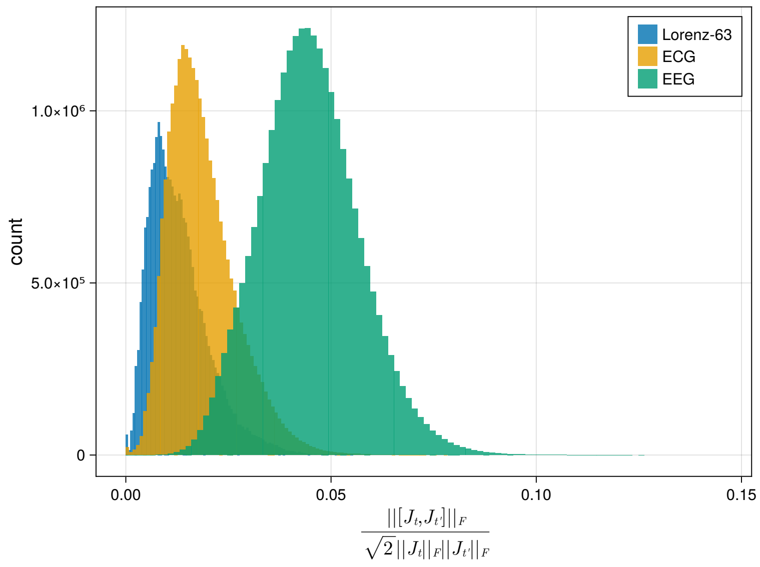


Figure S1. Histograms of relative Jacobian commutator norms for converged shPLRNNs (13) trained on different datasets. We measure the commutativity of Jacobians by first drawing an orbit of length $T = 5000$ and computing the ratio between the Frobenius norm of the commutator of all possible permutations of pair-wise Jacobians in the orbit, and the respective tight upper bound on the commutator norm (Böttcher & Wenzel, 2008). For all of the systems shown (Lorenz-63, ECG and EEG), most mass is concentrated below 5% of the upper bound, which is a reference point for maximally non-commuting Jacobians.

We can also approximate $\left\|\mathcal{G}(\hat{\boldsymbol{J}}_{T:2})\right\|$ by using the upper bound

$$\begin{aligned}\left\|\mathcal{G}(\hat{\boldsymbol{J}}_{T:2})\right\| = \left\|\left(\prod_{k=0}^{T-2} \hat{\boldsymbol{J}}_{T-k}\right)^{\frac{1}{T-1}}\right\| &\leq \left(\prod_{k=0}^{T-2} \left\|\hat{\boldsymbol{J}}_{T-k}\right\|\right)^{\frac{1}{T-1}} \\ &= \exp\left(\frac{1}{T-1}\sum_{t=2}^{T} \log \left\|\hat{\boldsymbol{J}}_{T-k}\right\|\right) \\ &= \exp\left(\frac{1}{T-1}\sum_{t=2}^{T} \log \sigma_{max}\left(\hat{\boldsymbol{J}}_{T-k}\right)\right).\end{aligned} \tag{54}$$

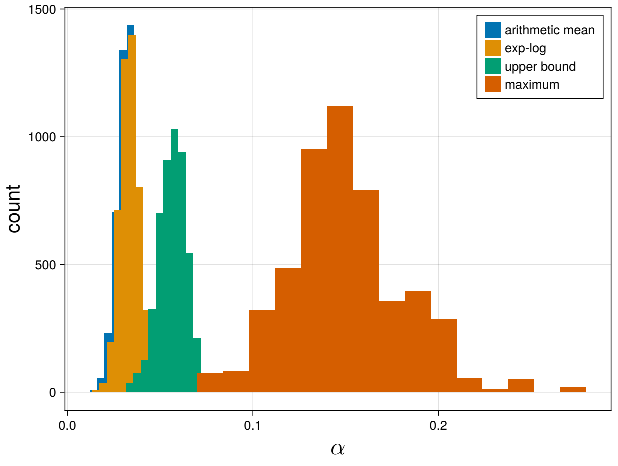
While technically accurate, the upper bound is biased towards higher values for α than generally required in practice, similar to the estimates in Eq. (18) and (19) (cf. Fig. S2).

Line search for optimal α For comparison to the performance of our aGTF procedure outlined above, we also determined optimal settings for α through a line search over different values $\alpha \in [0, 1]$ in steps of 0.05. Figure S3 shows D_{stsp} and D_H values for shPLRNNs (13) on different datasets against α. As some of the graphs indicate, performance is often not overly sensitive to the precise choice of α, and generally on par with the automatic adjustment of α in aGTF, remediating the need for grid search.

Latent model regularization Performance of the annealing protocol can be improved even further with the manifold attractor regularization suggested in Schmidt et al. (2021), which adds the following term with regularization factor λ_{reg} to the loss:

$$\mathcal{L}_{\text{reg}} = \lambda_{\text{reg}}\left(\|\mathbb{1} - \boldsymbol{A}\|_F^2 + \|\boldsymbol{W}_1\|_F^2 + \|\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{h}_1\|_2^2 + \|\boldsymbol{h}_2\|_2^2\right) \tag{55}$$

This has two benefits: 1) It pushes the shPLRNN's Jacobian towards the identity, thereby improving error propagation (Schmidt et al., 2021) and reducing the amount of necessary forcing by already taming exploding/ vanishing gradients, and 2) it improves accuracy of the approximation of $\mathcal{G}$ (see Eq. (52)).



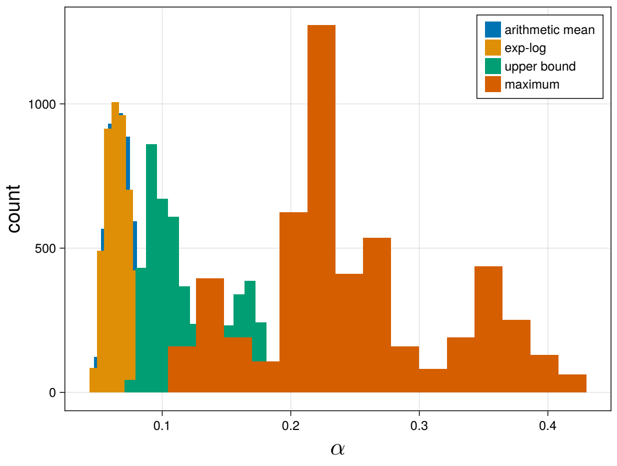


Figure S2. Histograms of different α estimates for a shPLRNN trained on the Lorenz-63 system (left) and on the ECG data (right) after convergence to the desired dynamics. Legend labels refer to the different approximations: 'arithmetic mean' (53), 'exp-log' (52), 'upper bound' (54) and 'maximum' (19). To produce this plot, we sampled 5000 sequences of length $T \in [100, 200]$ and computed α according to Eq. (23). Estimates for the 'arithmetic mean' and 'exp-log' approximations highly correlate for both datasets (Pearson's $r \approx 0.996$ across all data points in both datasets). For comparison, estimates of α computed through the Jacobian norm upper bound of the shPLRNN given by Eq. (18) are $\alpha \approx 0.87$ for the ECG and $\alpha \approx 0.79$ for the Lorenz-63.

Observation model regularization When using a trainable linear observation model (14), we need to invert the model to obtain teacher signals in latent space (15). However, we found that very occasionally $\boldsymbol{B}$ becomes ill-conditioned, i.e. close to singular. To avoid this, one can regularize the condition number of $\boldsymbol{B}$:

$$\mathcal{L}_{cn} = \lambda_{cn} \left(1 - \frac{\sigma_{max}(\boldsymbol{B})}{\sigma_{min}(\boldsymbol{B}) + \epsilon}\right)^2, \tag{56}$$

where $\sigma_{max}(\boldsymbol{B})$ and $\sigma_{min}(\boldsymbol{B})$ are the largest and smallest singular values of $\boldsymbol{B}$, respectively, λ_{cn} is a hyperparameter and ϵ $(= 10^{-8})$ a small number added for numerical stability. This regularization pushes the condition number towards 1, thus ensuring invertibility.

6.3. Benchmark Systems and Real-World Data

Benchmark: Lorenz-63 Introduced in Lorenz (1963) as one of the first ODE systems for which chaotic behavior was demonstrated, the 3d Lorenz-63 system has become the most common benchmark in the DS reconstruction literature. Designed as a simple model of atmospheric convection, the system exhibits different routes to chaos like period doubling and homoclinic bifurcations and, due to a fundamental symmetry, exhibits the famous butterfly-wing shape (see Fig. S4, Perko (2001)). The system is described by the following set of ODEs:

$$\begin{aligned}\dot{x} &= \sigma(y - x)\\ \dot{y} &= x(\rho - z) - y\\ \dot{z} &= xy - \beta z,\end{aligned}$$

where we place the system into the chaotic dynamical regime with $\sigma = 10$, $\beta = \frac{8}{3}$ and $\rho = 28$. For each of the training and test sets we simulate a trajectory of length $T = 10^5$, each starting at a different random initial condition $\boldsymbol{u}_0 = (x_0, y_0, z_0)^T \sim \mathcal{N}(\boldsymbol{0}, \mathbb{1}_{3\times 3})$, using the DynamicalSystems.jl (Datseris, 2018) Julia library. The ODE system is integrated using a Runge-Kutta scheme with adaptive step size and a read-out interval of $\Delta t = 0.01$. Furthermore, we contaminate the training set with Gaussian observation noise, using a noise level of 5% of the data standard deviation. Finally, we standardize each dimension of both training and test set to zero mean and unit variance.

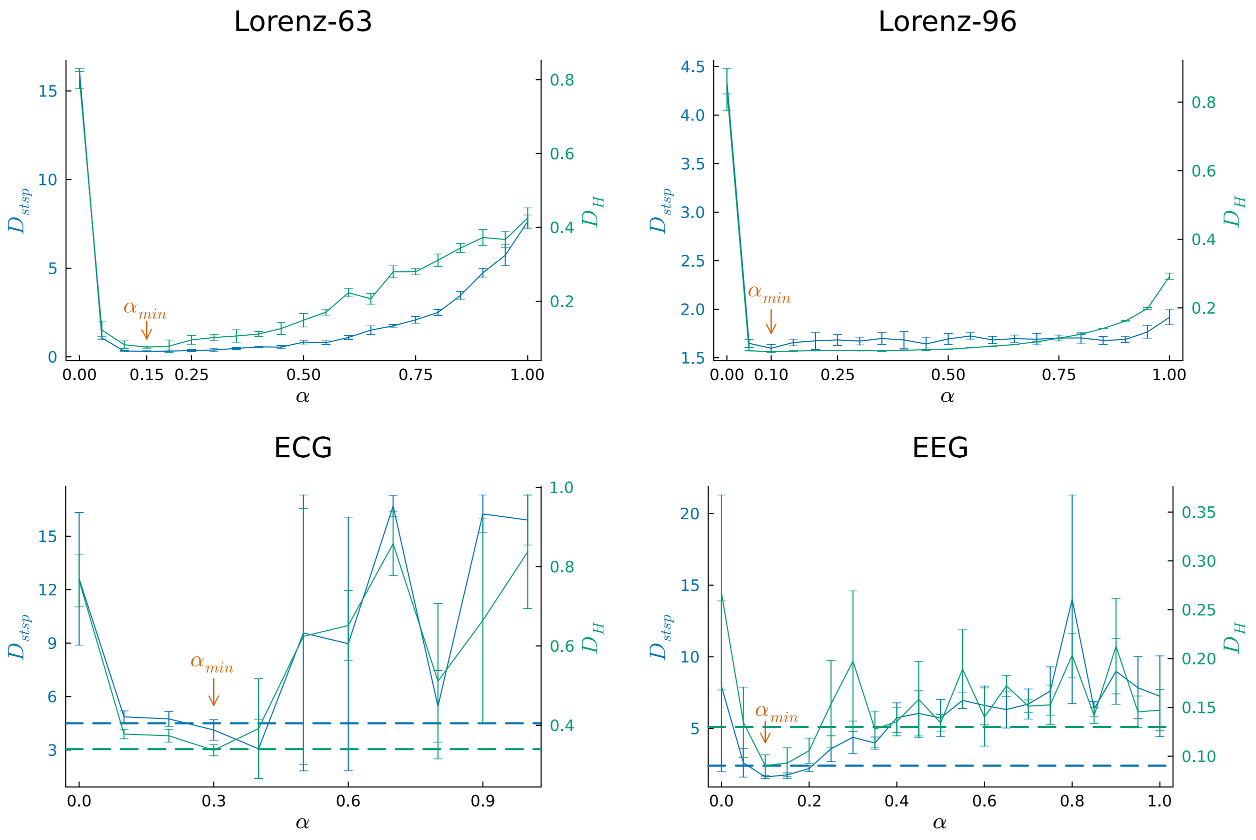


Figure S3. Line search over different α values. The minima lie at $\alpha_{min} \approx 0.15$ for the Lorenz-63, and $\alpha_{min} \approx 0.1$ for the Lorenz-96 and EEG data, respectively, while for the ECG data we find $\alpha_{min} \approx 0.3$. There appears to be a somewhat larger α-range over which performance is similarly good, and thus some lenience regarding the precise adjustment of this value. Dashed blue (D_{stsp}) and green (D_H) lines indicate performance levels of aGTF on ECG and EEG data, respectively.

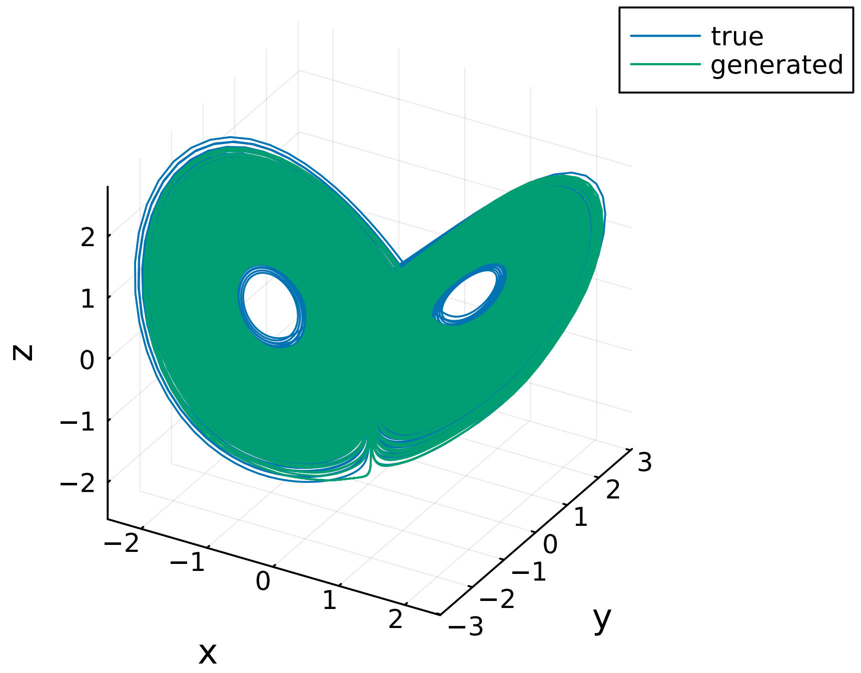


Figure S4. Example reconstruction of the Lorenz-63 system by shPLRNN+GTF. Note that the model freely generates the dynamics given the first time point of the test set.

Benchmark: Lorenz-96 A higher-dimensional, spatially extended system for atmospheric convection with local neighborhood interactions was suggested in Lorenz (1996). The system can be formulated in arbitrarily high dimensions:

$$\dot{x}_k = (x_{k+1} - x_{k-2})x_{k-1} - x_k + F, \qquad k = 1 \ldots N$$

with dynamical variables $\{x_k\}$, constant forcing term F and dimension N. For our experiments we choose $N = 20$ and $F = 16$, such that the system is situated within the chaotic regime. We create a training and test set using the same protocol as for the Lorenz-63 model, i.e. we draw two trajectories, one for training and one for testing. The training trajectory is contaminated with 5% Gaussian noise, and both resulting datasets are standardized to have zero mean and unit variance on each dimension. See Fig. S5 for an excerpt of the system dynamics in form of a heatmap, together with a reconstruction using the method employed in this work.

Benchmark: Multiscale Lorenz-96 Building on the original Lorenz-96 model, Thornes et al. (2017) introduced an extended version which models atmospheric weather phenomena evolving on multiple temporal and spatial scales through nested sets of dynamical variables. The set of ODEs is given by

$$\begin{aligned}
\frac{dX_k}{dt} &= X_{k-1}(X_{k+1} - X_{k-2}) - X_k + F - \frac{hc}{b}\sum_{j=1}^{J} Y_{j,k}, \\
\frac{dY_{j,k}}{dt} &= -cbY_{j+1,k}(Y_{j+2,k} - Y_{j-1,k}) - cY_{j,k} + \frac{hc}{b}X_k - \frac{he}{d}\sum_{i=1}^{I} Z_{i,j,k}, \\
\frac{dZ_{i,j,k}}{dt} &= edZ_{i-1,j,k}(Z_{i+1,j,k} - Z_{i-2,j,k}) - g_Z e Z_{i,j,k} + \frac{he}{d}Y_{j,k}.
\end{aligned} \tag{57}$$

These equations describe a system with K slow large-scale variables X, each of which coupled to J faster and smaller-scale variables Y, which in turn are coupled to I very fast small-scale variables Z. Parameters h, b, c, e and d determine the coupling strength between different scales, F is the external forcing strength and g_Z is a damping parameter. It has

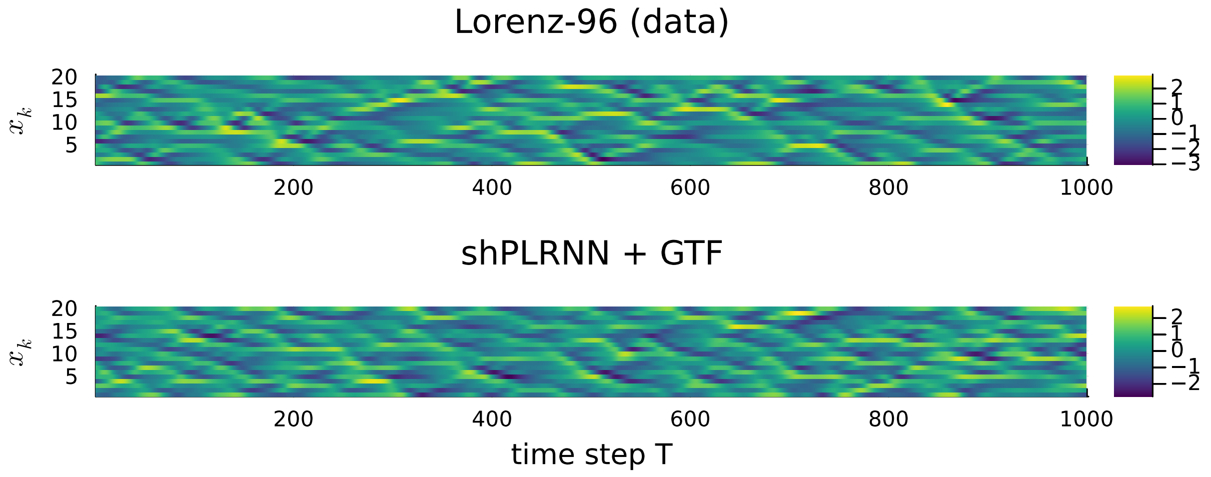


Figure S5. Heatmaps of an excerpt of the Lorenz-96 test set (see 6.3) and an example reconstruction using the shPLRNN+GTF.

previously been used to assess the capability of different machine learning models to predict future DS states (Chattopadhyay et al., 2020), some of which – like RCs and LSTMs – also included in this work. Chattopadhyay et al. (2020) assess forecasting abilities on a subset of the dynamical variables, using $I = J = K = 8$, which results in a 584-dimensional system, with parameters $b = c = e = d = g_Z = 10$, $h = 1$, and forcing $F = 20$. This places the system into a highly chaotic regime. However, the authors assumed that only the $K = 8$ slow, large-scale variables X are observed, i.e. all methods were trained on partial observations from the 584-dimensional system. To enable direct comparison, we trained N-ODE, dendPLRNN and the shPLRNN on the dataset provided within the authors' codebase. We find that all tested methods (N-ODE, shPLRNN+GTF, dendPLRNN+id-TF) produce both accurate short-term forecasts of the partially observed system,[10] as well as statistically indistinguishable reconstructions of the long-term behavior (all within a 10% median absolute deviation and 34% SEM margin; Kruskal-Wallis test $p = 0.84$; see Tab. S1 and Figs. S6, S7, S8).

Table S1. Results for the partially observed multiscale Lorenz-96 system. Reported values are median ± median absolute deviation (MAD) over 20 independent training runs. 'dim' refers to the model's state space dimensionality (number of dynamical variables). $|\boldsymbol{\theta}|$ denotes the total number of *trainable* parameters.

Dataset	Method	$D_{\text{stsp}} \downarrow$	$D_H \downarrow$	PE(20) $\downarrow$	dim	$\|\boldsymbol{\theta}\|$
multiscale	shPLRNN + GTF	$(6.1 \pm 1.1) \cdot 10^{-2}$	$(7.3 \pm 0.4) \cdot 10^{-2}$	$(5.7 \pm 0.6) \cdot 10^{-3}$	8	1780
Lorenz-	dendPLRNN + id-TF	$(6.3 \pm 1.4) \cdot 10^{-2}$	$(6.9 \pm 0.4) \cdot 10^{-2}$	$(3.4 \pm 1.6) \cdot 10^{-3}$	25	1845
96	Neural-ODE	$(6.2 \pm 0.3) \cdot 10^{-2}$	$(7.8 \pm 0.3) \cdot 10^{-2}$	$(4.6 \pm 0.3) \cdot 10^{-3}$	8	1708

Empirical Dataset: ECG The ECG data used here consists of a single time series with $T = 419,973$ time points. Given a sampling frequency of 700Hz, this corresponds to 600s of recording time. We first preprocessed the ECG data by smoothing the time series using a Gaussian filter ($\sigma = 6$, $l = 8\sigma + 1 = 49$), followed by standardization of the time series. We then delay-embed the signal using the PECUZAL algorithm (Krämer et al., 2021) implemented in the DynamicalSystems.jl (Datseris, 2018) Julia library. The algorithm uses the L-statistic and non-uniform delays to find an optimal delay embedding (for details, see Krämer et al. (2021) or the documentation of the algorithm). We set $\Delta L = 0.05$, and use a Theiler window based on the first minimum of the mutual information, leading to an embedding dimension of $m = 5$. For our experiments, we use the first $T = 100,000$ samples ($\approx$ 143s) as the training set and the next $T = 100,000$ samples as the test set. The maximum Lyapunov exponent is estimated as $\lambda_{max} = (2.19 \pm 0.05)\ \frac{1}{\text{s}}$ by fitting a line to the average log-distance $log(d(t))$ between trajectories evolving from neighboring states for different embedding dimensions m (Fig. S9; cf. Kantz (1994); Skokos et al. (2016)), and agrees well with the literature (Govindan et al., 1998).

Empirical Dataset: EEG The Electroencephalogram (EEG) dataset used here is the same as in Brenner et al. (2022) and Mikhaeil et al. (2022) (the latter authors using a different preprocessing). The dataset consists of recordings from 64

[10]See Chattopadhyay et al. (2020) for performance of RCs and LSTMs.

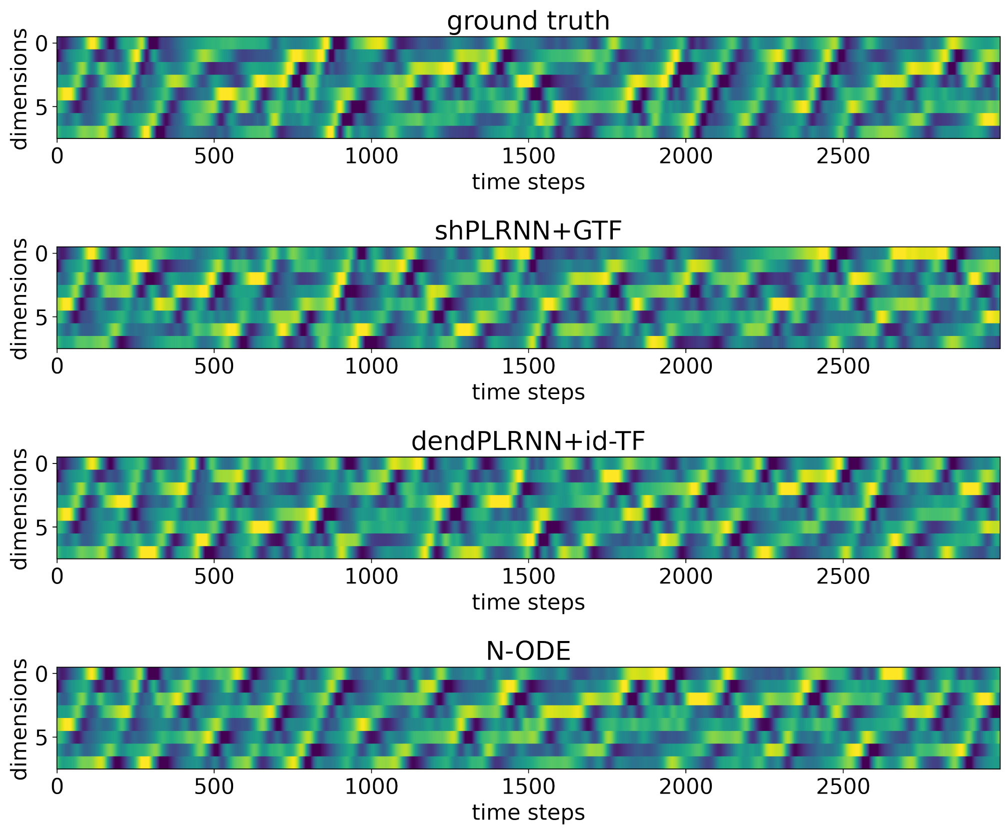


Figure S6. Long term spatio-temporal behavior for the three best performing methods trained on the partially observed multiscale Lorenz-96 model from Chattopadhyay et al. (2020).

electrodes placed across the scalp of a human subject sitting still in a chair with eyes open (first subject performing task 1 labeled "S001R01"). The EEG sampling frequency was 160Hz. These data are part of a larger study conducted by Schalk et al. (2000), openly available at PhysioNet (Goldberger et al., 2000). Here the data was preprocessed as in Brenner et al. (2022), standardizing and smoothing each signal using a Hann filter with window length 15 time bins. For computing the invariant (long-term) statistics D_{stsp} and D_H on the EEG data, tested models were (as for all other comparisons) simulated freely starting from just a data-inferred initial condition, but EEG reference values were taken from the training data since EEG time series were much shorter than ECG series (less than $10,000$ time steps for EEG compared to more than $400,000$ for ECG). Robust and reliable estimation of long-term properties like power spectra and attractor geometries would thus have been difficult for only a short (left-out) fraction of that time series. Apart from the fact that this was of course handled equally for all methods tested, however, note that this should not affect our measures D_{stsp} and D_H much, since (by definition) these long-term properties would not be expected to change on shorter time scales. To also make predictions more challenging under these conditions (without separate left-out set), we chose a relatively large prediction step n. For STF, the predictabiliy time for the EEG data was directly taken from Mikhaeil et al. (2022). However, reliably determining the predictability time from data is exactly one of the issues with STF as proposed, and while we did not perform systematic grid search on the forcing interval for STF, we observed that other intervals could improve performance for shPLRNN+STF. A more systematic comparison between GTF and STF therefore still warrants further research.

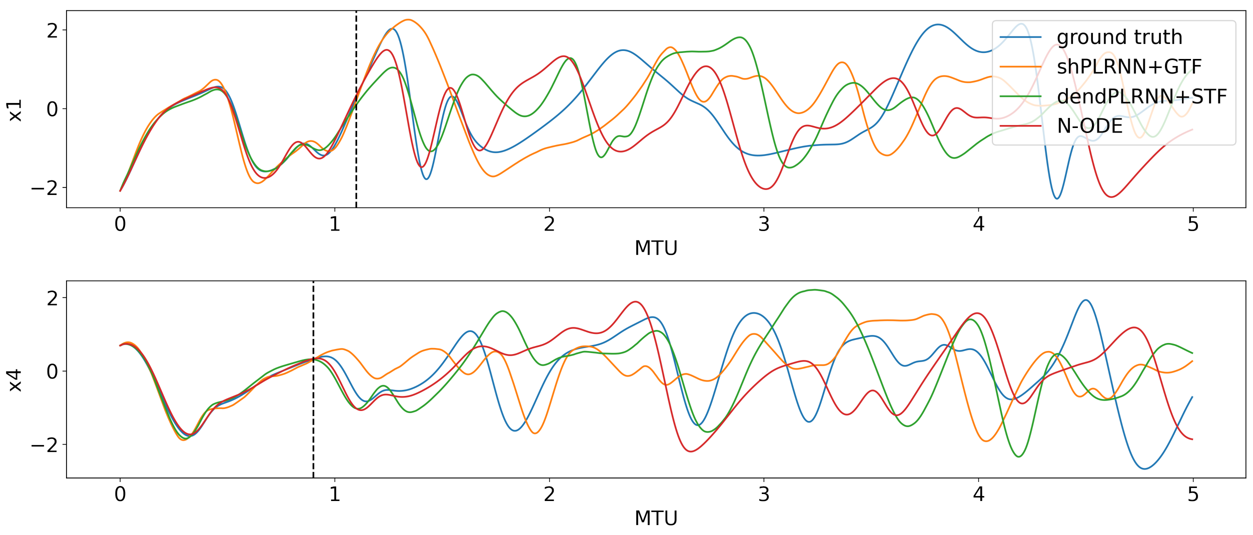


Figure S7. For direct comparison with Fig. 5 in Chattopadhyay et al. (2020): Short-term forecasts on the partially observed multiscale Lorenz-96 system for the three best performing methods, starting from the first time step of the test set. Time is in model time units (MTU), where 1 MTU$= 200\Delta t \approx 4.5/\lambda_{max}$. The vertical line indicates the approximate prediction horizon, which for chaotic systems is limited by the system's maximum Lyapunov exponent.

6.4. Details on Evaluation Measures

Geometrical measure (D_{stsp}) Given $p(\boldsymbol{x})$, generated by trajectories of the true system, and $q(\boldsymbol{x})$, generated by the model, the state space divergence is defined as

$$D_{\text{stsp}} := D_{\text{KL}}\left(p(\boldsymbol{x}) \,||\, q(\boldsymbol{x})\right) = \int_{\boldsymbol{x}\in\mathbb{R}^N} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x} \tag{58}$$

For low dimensional observation spaces, $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ can be estimated directly from a binning of space (Koppe et al., 2019; Brenner et al., 2022), with the minimum-to-maximum range for binning determined by the extent of the observed (ground truth) attractor. Eq. (58) is then approximately given by

$$D_{\text{stsp}} = D_{\text{KL}}\left(\hat{p}(\boldsymbol{x}) \,||\, \hat{q}(\boldsymbol{x})\right) \approx \sum_{k=1}^{K} \hat{p}_k(\boldsymbol{x}) \log \frac{\hat{p}_k(\boldsymbol{x})}{\hat{q}_k(\boldsymbol{x})} \tag{59}$$

where $K = m^N$ is the total number of bins, with m the number of bins per dimension, $\hat{p}_k(\boldsymbol{x})$ is the relative number of ground truth orbit data points in bin k and, likewise, $\hat{q}_k(\boldsymbol{x})$ that for generated orbits. For high-dimensional systems, a binning approach is no longer sensible. Instead, Gaussian Mixture Models (GMMs) were placed along orbits (see (Brenner et al., 2022)), i.e. $\hat{p}(\boldsymbol{x}) = 1/T \sum_{t=1}^{T} \mathcal{N}(\boldsymbol{x};\ \boldsymbol{x}_t, \boldsymbol{\Sigma})$ and $\hat{q}(\boldsymbol{x}) = 1/T \sum_{t=1}^{T} \mathcal{N}(\boldsymbol{x};\ \hat{\boldsymbol{x}}_t, \boldsymbol{\Sigma})$, where $\boldsymbol{x}_t$ and $\hat{\boldsymbol{x}}_t$ are observed and generated states, respectively, $\mathcal{N}(\boldsymbol{x};\ \boldsymbol{x}_t, \boldsymbol{\Sigma})$ is a multivariate Gaussian with mean vector $\boldsymbol{x}_t$ and covariance matrix $\boldsymbol{\Sigma} = \sigma^2 \mathbb{1}_{N\times N}$, and T is the orbit length. Hershey & Olsen (2007) provide a Monte Carlo approximation to the KL divergence between two GMMs, which is given by

$$D_{\text{stsp}} = D_{\text{KL}}\left(\hat{p}(\boldsymbol{x}) \,||\, \hat{q}(\boldsymbol{x})\right) \approx \frac{1}{n} \sum_{i=1}^{n} \log \frac{\hat{p}(\boldsymbol{x}^{(i)})}{\hat{q}(\boldsymbol{x}^{(i)})}, \tag{60}$$

with n Monte Carlo samples $\boldsymbol{x}^{(i)}$ randomly drawn from the GMM based on observed orbits. We follow Brenner et al. (2022) in using $m = 30$ for the binning approach and $\sigma^2 = 1.0$ for the GMM approach.

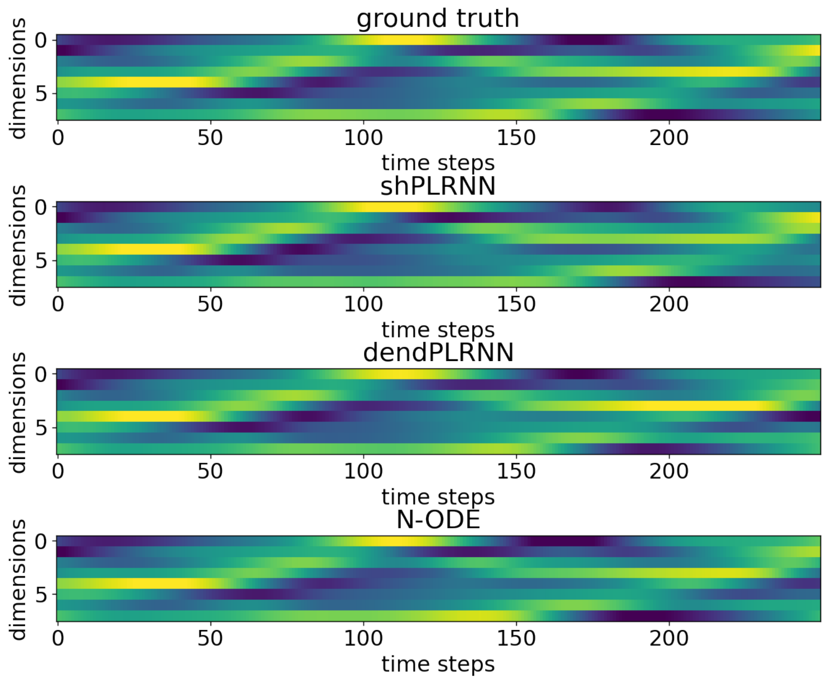


Figure S8. Short term spatio-temporal forecasts for the three best performing methods trained on the partially observed multiscale Lorenz-96 model from Chattopadhyay et al. (2020).

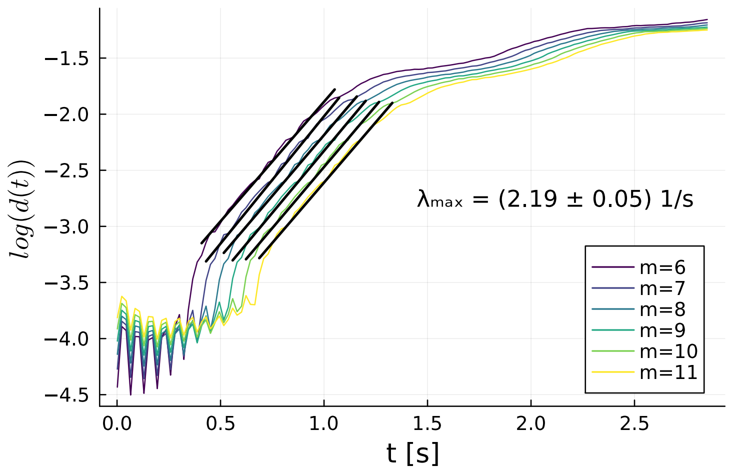


Figure S9. Estimation of the maximum Lyapunov exponent of the ECG time series for different embedding dimensions m and fixed delay $\tau = 59$. The delay was determined as the first minimum of the mutual information of the original time series.

Temporal Measure (D_H) Given power spectra $f_i(\omega)$ and $g_i(\omega)$ of the i-th dynamical variable of the observed and reconstructed orbits, respectively, with $\int_{-\infty}^{\infty} f_i(\omega)d\omega = 1$ and $\int_{-\infty}^{\infty} g_i(\omega)d\omega = 1$, the Hellinger distance is given by

$$H(f_i(\omega), g_i(\omega)) = \sqrt{1 - \int_{-\infty}^{\infty} \sqrt{f_i(\omega)g_i(\omega)}\, d\omega} \tag{61}$$

In practice, we approximate power spectra by performing a Fast Fourier Transform (FFT; Cooley & Tukey (1965)) yielding $\hat{\boldsymbol{f}}_i = |\mathcal{F}x_{i,1:T}|^2$ and $\hat{\boldsymbol{g}}_i = |\mathcal{F}\hat{x}_{i,1:T}|^2$, with vectors $\hat{\boldsymbol{f}}_i$ and $\hat{\boldsymbol{g}}_i$ discrete power spectra of ground truth traces $x_{i,1:T}$ and model generated traces $\hat{x}_{i,1:T}$. Power spectra are then smoothed using a Gaussian filter with standard deviation $\tilde{\sigma} = 20$ and window length $l = 8\sigma + 1$. Before computing the Hellinger distance, the power spectra are normalized to fulfill $\sum_\omega \hat{f}_{i,\omega} = 1$ and $\sum_\omega \hat{g}_{i,\omega} = 1$. Finally, H (61) is computed as

$$H(\hat{\boldsymbol{f}}_i, \hat{\boldsymbol{g}}_i) = \frac{1}{\sqrt{2}} \left\| \sqrt{\hat{\boldsymbol{f}}_i} - \sqrt{\hat{\boldsymbol{g}}_i} \right\|_2, \tag{62}$$

where the square root is applied elementwise. The final measure D_H is obtained by averaging H across all dimensions:

$$D_H = \frac{1}{N} \sum_{i=1}^{N} H(\hat{\boldsymbol{f}}_i, \hat{\boldsymbol{g}}_i). \tag{63}$$

Prediction Error We compute an n-step prediction error as the MSE between ground truth data and n-step ahead predictions of the model:

$$\text{PE}(n) = \frac{1}{N(T-n)} \sum_{t=1}^{T-n} \left\| \boldsymbol{x}_{t+n} - \hat{\boldsymbol{x}}_{t+n} \right\|_2^2. \tag{64}$$

Note that for chaotic systems, due to the exponential divergence of trajectories, the invariant statistics (D_{stsp}, D_H) and the n-step prediction error may dissociate, as illustrated in Koppe et al. (2019) and Schmidt et al. (2021).

Evaluation Setup For computing D_{stsp} and D_H, we first draw a single long orbit of the generative model at hand (i.e. either the shPLRNN or any of the comparison methods employed, cf. Table 1) of length $1.25 \cdot T$, where T is the total length of the available (test) data. We then discard the first $0.25 \cdot T$ time steps of the model-generated orbit to make sure the measures are evaluated on the limit sets and not on transients of the dynamics.[11] We then use the remaining T time steps to compute both measures. The initial condition for the model is determined from the first time step of the observed ground truth orbit. For comparison methods requiring a dynamical warm-up phase, such as RC, multiple time steps of the ground truth orbit are provided. The prediction error (PE(n)) is computed across the entire test set.

6.5. GTF and RNN architecture

In theory, GTF is independent of the specific RNN architecture or, more generally, map $\boldsymbol{F}_\theta$ employed. In practice, we observed training the dendPLRNN (Brenner et al., 2022) with GTF does not lead to similarly strong performance boosts over STF-based training as observed for the shPLRNN (Tab. S2), implying that certain RNN designs like the shPLRNN might be more amenable to GTF. This could be because for the dendPLRNN we do not have a 1:1 relation between observations and latent states as for the shPLRNN (rather, we need to go to higher dimensions, cf. Tab. 1). This makes implementation of GTF for the dendPLRNN less straightforward, since - unlike for the shPLRNN - the mapping onto latent states is underdetermined. For the experiments in Tab. S2, we used an implementation of GTF similar to id-TF (Brenner et al., 2022), only forcing the dendPLRNN's first N latent states and leaving the remaining $M - N$ states unforced. Alternatively, one may use inversion of a trainable linear observation model for projecting the teacher signal into the model's latent space (Mikhaeil et al., 2022). However, this does not resolve the underdeterminacy (on the contrary, it seemed to even exacerbate the problem, possibly because the additional degrees of freedom introduced by the trainable projection operator might make the GTF forcing even less tight). Hence, it needs to be concluded that the shPLRNN also appears to bear a specific architectural advantage, while for other models the best way for implementing GTF needs more consideration. We also observed that replacing the ReLU activation of the shPLRNN by $\tanh$ diminished performance (Tab. S2).

[11]This is important, since otherwise erroneously a good reconstruction may be indicated, while truly the reconstructed system may converge to a limit set topologically different from that of the true system, e.g. an equilibrium rather than a chaotic attractor.

Table S2. Ablation study. Reported values are median ± median absolute deviation (MAD) over 20 independent training runs. 'dim' refers to the model's state space dimensionality (number of dynamical variables). $|\boldsymbol{\theta}|$ denotes the total number of *trainable* parameters. Values for shPLRNN+GTF are copied from Tab. 1.

Dataset	Method	$D_{\text{stsp}} \downarrow$	$D_H \downarrow$	PE(20) $\downarrow$	dim	$\|\boldsymbol{\theta}\|$
ECG (5d)	shPLRNN + GTF	$\mathbf{4.3 \pm 0.6}$	$\mathbf{0.34 \pm 0.02}$	$\mathbf{(2.4 \pm 0.1) \cdot 10^{-3}}$	5	2785
	shPLRNN + GTF ($tanh$)	7.8 ± 2.1	0.37 ± 0.03	$(8.6 \pm 0.3) \cdot 10^{-3}$	5	2785
	dendPLRNN + GTF	5.2 ± 1.2	0.34 ± 0.02	$(5.5 \pm 1.3) \cdot 10^{-3}$	35	3245
	dendPLRNN + STF	5.8 ± 0.6	0.37 ± 0.06	$(4.0 \pm 0.4) \cdot 10^{-3}$	35	3245
EEG (64d)	shPLRNN + GTF	$\mathbf{2.1 \pm 0.2}$	$\mathbf{0.11 \pm 0.01}$	$\mathbf{(5.5 \pm 0.1) \cdot 10^{-1}}$	16	17952
	shPLRNN + GTF ($tanh$)	17 ± 3	0.37 ± 0.07	$(6.6 \pm 0.1) \cdot 10^{-1}$	16	17952

6.6. More Details on Comparison Methods

Table S3. SOTA comparisons on the Lorenz-63 and Lorenz-96 benchmark systems. Reported values are median ± median absolute deviation over 20 independent training runs. 'dim' refers to the model's state space dimensionality (number of dynamical variables). $|\boldsymbol{\theta}|$ denotes the total number of *trainable* parameters. Note that SINDy has an inbuilt advantage for these two benchmarks over all other methods: Both the Lorenz-63 and Lorenz-96 ODEs are second-order polynomials, and SINDy's polynomial function library included terms up to second order as well. Hence, all that SINDy needs to do in these cases is to determine the appropriate parameters (rather than approximating the underlying system). While the library method may be an advantage for systems with known functional form (as in these cases), it may become a severe disadvantage if no detailed structural knowledge is available (as commonly the case in complex empirical scenarios like those evaluated in Table 1).

Dataset	Method	$D_{\text{stsp}} \downarrow$	$D_H \downarrow$	PE(20) $\downarrow$	dim	$\|\boldsymbol{\theta}\|$
Lorenz-63 (3d)	shPLRNN + GTF	$\mathbf{0.26 \pm 0.03}$	$\mathbf{0.090 \pm 0.007}$	$\mathbf{(6.0 \pm 0.5) \cdot 10^{-4}}$	3	365
	dendPLRNN + id-TF	0.9 ± 0.2	0.15 ± 0.03	$(2.2 \pm 0.7) \cdot 10^{-3}$	10	361
	RC	0.52 ± 0.12	0.19 ± 0.04	$(5 \pm 2) \cdot 10^{-3}$	201	603
	LSTM-TBPTT	0.46 ± 0.22	0.11 ± 0.03	$(1.1 \pm 0.3) \cdot 10^{-3}$	30	1188
	SINDy	$\mathbf{0.24 \pm 0.00}$	$\mathbf{0.091 \pm 0.000}$	$\mathbf{(6.1 \pm 0.0) \cdot 10^{-4}}$	3	30
	N-ODE	0.63 ± 0.2	0.15 ± 0.05	$(2.3 \pm 0.3) \cdot 10^{-3}$	3	353
	LEM	0.39 ± 0.24	0.12 ± 0.05	$(6.0 \pm 0.9) \cdot 10^{-3}$	14	360
Lorenz-96 (20d)	shPLRNN + GTF	1.68 ± 0.06	$\mathbf{0.072 \pm 0.001}$	$(1.21 \pm 0.02) \cdot 10^{-1}$	20	4540
	dendPLRNN + id-TF	$\mathbf{1.65 \pm 0.05}$	0.083 ± 0.005	$\mathbf{(1.1 \pm 0.1) \cdot 10^{-1}}$	60	5740
	RC	2.40 ± 0.15	0.14 ± 0.02	$(4.9 \pm 0.4) \cdot 10^{-1}$	600	12000
	LSTM-TBPTT	5 ± 1	0.31 ± 0.04	$(1.14 \pm 0.04) \cdot 10^{0}$	80	10580
	SINDy	$\mathbf{1.59 \pm 0.00}$	$\mathbf{0.06 \pm 0.00}$	$\mathbf{(4.6 \pm 0.0) \cdot 10^{-3}}$	20	4620
	N-ODE	1.77 ± 0.07	0.076 ± 0.01	$(2.5 \pm 0.02) \cdot 10^{-1}$	20	4530
	LEM	7.2 ± 2.3	0.54 ± 0.13	$(1.3 \pm 0.06) \cdot 10^{0}$	46	4620

SINDy For SINDy we used the PySINDy package (de Silva et al., 2020; Kaptanoglu et al., 2022) with the sequentially thresholded least squares (STLSQ) optimizer. We scanned the threshold hyperparameter, which determines the sparsity of the final solution (the higher the threshold, the sparser the solution), in the range $[0,\ 1)$. Furthermore, since all employed datasets are noisy, we used PySINDy's `SmoothedFiniteDifference` for empirical vector field estimates, leading to more robust estimates. For the Lorenz-63 and Lorenz-96 systems, we used a polynomial basis library (`PolynomialLibrary`) up to order 2. Note that this essentially means SINDy only needs to figure out the right values of the parameters for these problems, as its set of equations is almost the same as in the ground truth systems to begin with. For the empirical ECG and EEG data, we experimented with terms up to 5-th order and also tried a Fourier basis (`FourierLibrary`), but did not manage to obtain any solution which would not quickly diverge during numerical integration. Orbits for the Lorenz-63 and Lorenz-96 systems were drawn using the `LSODA` integrator with absolute tolerance 10^{-4} and relative tolerance 10^{-6}.

RC and LSTM For RC and LSTM we used the official repository provided by the authors (Pathak et al., 2018; Vlachas et al., 2018; 2020). For RC (Pathak et al., 2018), we searched for optimal hyperparameters for {`dynamics_length`, `regularization`, `noise_level`}. For LSTMs trained with truncated BPTT (Vlachas et al., 2018), we determined optimal hyperparameters for {`hidden_state_propagation_length`, `regularization`, `learning_rate`, `noise_level`, `sequence_length`}.

dendPLRNN + id-TF For comparison to the method in Brenner et al. (2022), we used the corresponding repository. We scanned across different values for the sparse TF interval τ (`teacher_forcing_interval`) and sequence length (`seq_len`), biased by the values reported in Brenner et al. (2022).

LEM For comparison to the method proposed in Rusch et al. (2022), we used code provided by the authors on their public respository. The size of the network was determined such that a comparable number of trainable parameters was obtained as for the other methods, while optimal hyperparameters for the learning rate and time constant Δt were selected via grid search.

Neural ODE For comparison to N-ODEs (Chen et al., 2018), we used the implementation provided in the `torchdiffeq` package. As hyperparameters we considered the employed activation function $\{relu, tanh\}$ and the sequence length $\in \{1, 5, 10, 25, 50\}$ used per minibatch. We further tried several fixed-step numerical solvers (`rk4`, `euler`, `midpoint`), which had little influence on the results, while an adaptive-step solver (`adaptive_heun`) led to unacceptably long training times. We further probed the N-ODE variants Latent-ODE and ODE-RNN proposed in (Rubanova et al., 2019), using the implementation provided on the authors' github page. The results in Table S4 imply that neither of these accomplishes more faithful reconstructions of EEG data than 'vanilla' N-ODE. In fact, the values for D_{stsp} and D_H obtained for ODE-based methods essentially all reflect "chance level" in the sense that the resulting long-term dynamics bears no similarity with the one observed.

We stress that most of these methods achieved fairly good reconstruction results on the simulated benchmarks (i.e., the various Lorenz systems introduced in Appx. 6.3), see Tables S1 and S3. They may also be competitive in their short-term forecasts on the empirical data (EEG & ECG, see Table 1 and Fig. S13). However, as Figs. 3, S11 and S12 demonstrate, unlike shPLRNN+GTF and dendPLRNN, they failed to reproduce the long-term behavior of the empirically observed systems, i.e. they failed to reconstruct underlying geometrical and invariant temporal properties as essential in this context.

Table S4. Results for Latent-ODE and ODE-RNN (Rubanova et al., 2019) on the EEG data. Reported values are median ± median absolute deviation (MAD) over 20 independent training runs. 'dim' refers to the model's state space dimensionality (number of dynamical variables). $|\boldsymbol{\theta}|$ denotes the total number of *trainable* parameters. Values for shPLRNN+GTF and N-ODE are copied from Table 1.

Dataset	Method	$D_{\text{stsp}} \downarrow$	$D_H \downarrow$	PE(20) $\downarrow$	dim	$\lvert\boldsymbol{\theta}\rvert$
	shPLRNN + GTF	$\mathbf{2.1 \pm 0.2}$	$\mathbf{0.11 \pm 0.01}$	$\mathbf{(5.5 \pm 0.1) \cdot 10^{-1}}$	16	17952
EEG	N-ODE	20 ± 0.5	0.47 ± 0.01	$(5.5 \pm 0.2) \cdot 10^{-1}$	64	17995
(64d)	Latent ODE	16.1 ± 3	0.47 ± 0.02	$(5.6 \pm 0.2) \cdot 10^{-1}$	64	17915
	ODE-RNN	13.9 ± 2.1	0.59 ± 0.03	$(9.1 \pm 0.6) \cdot 10^{-1}$	64	17859

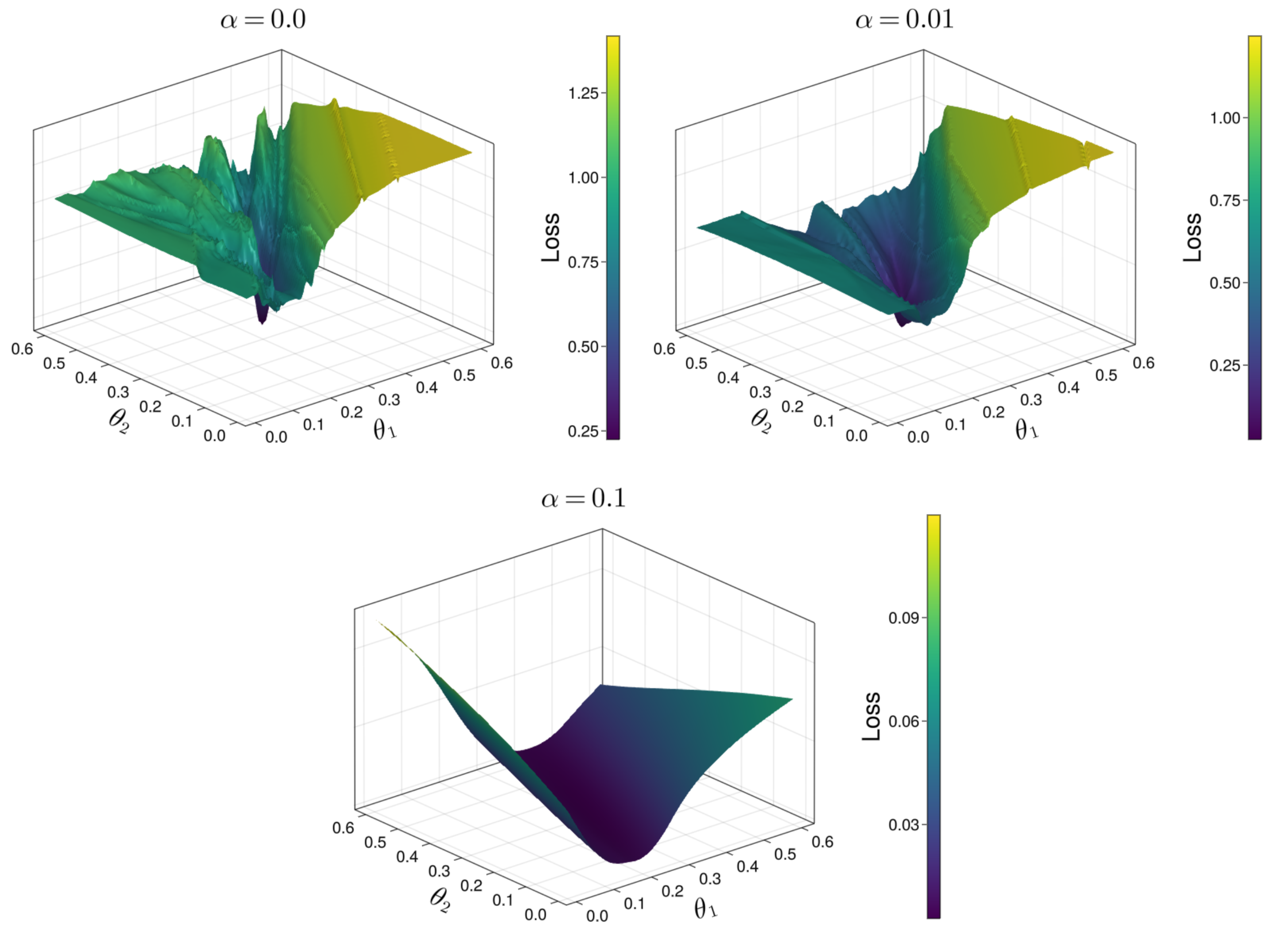


Figure S10. GTF smoothens loss landscapes. Loss as a function of two arbitrarily chosen ($\theta_1 = w^{(1)}_{1,2}$, $\theta_2 = w^{(1)}_{2,1}$) shPLRNN parameters. The shPLRNN (13) was first trained on the Lorenz-63 system for a couple of epochs ($\alpha = 0.15$, $\tilde{T} = 200$, $S = 16$), after which the loss was determined for various α values based on a random batch of the training data. For larger α the loss landscape smoothens out.

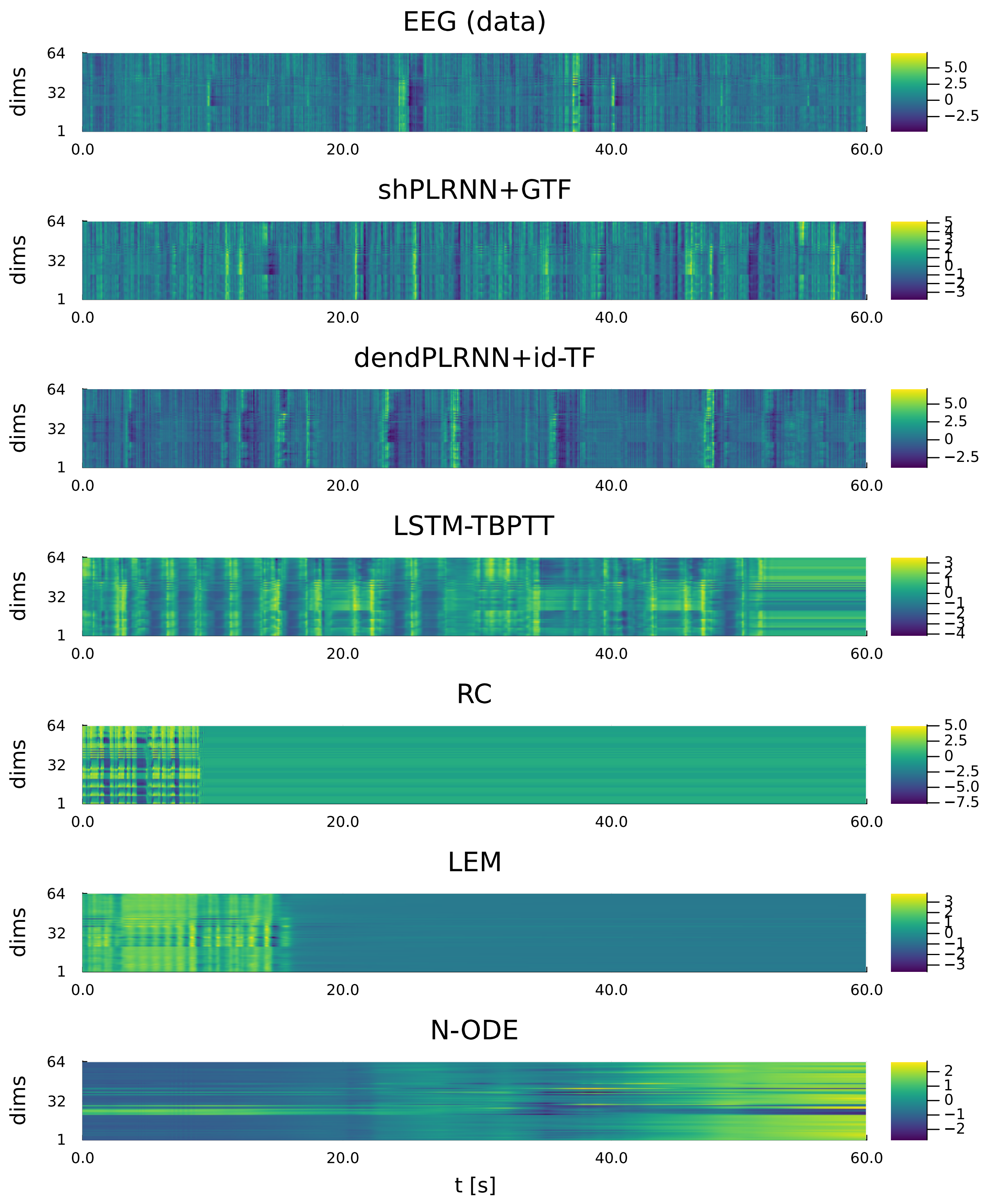


Figure S11. Example heatmaps of EEG reconstructions provided by the methods employed in Table 1. We used the same models as for Fig. 3. To make the heatmaps comparable, each channel was standardized for each method separately.

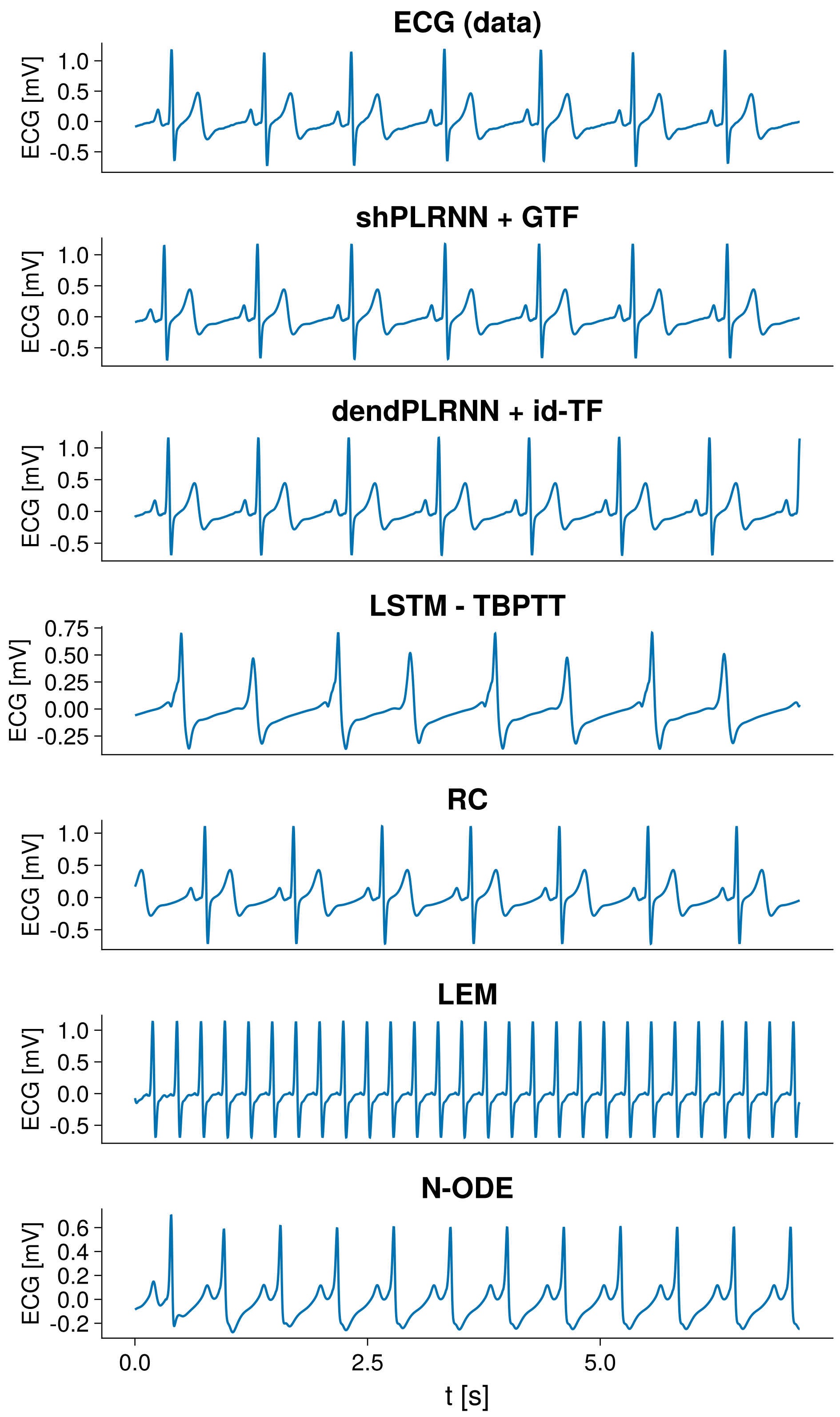


Figure S12. Example time traces of ECG reconstructions provided by the methods employed in Table 1. For each method we picked the best reconstruction out of 20.

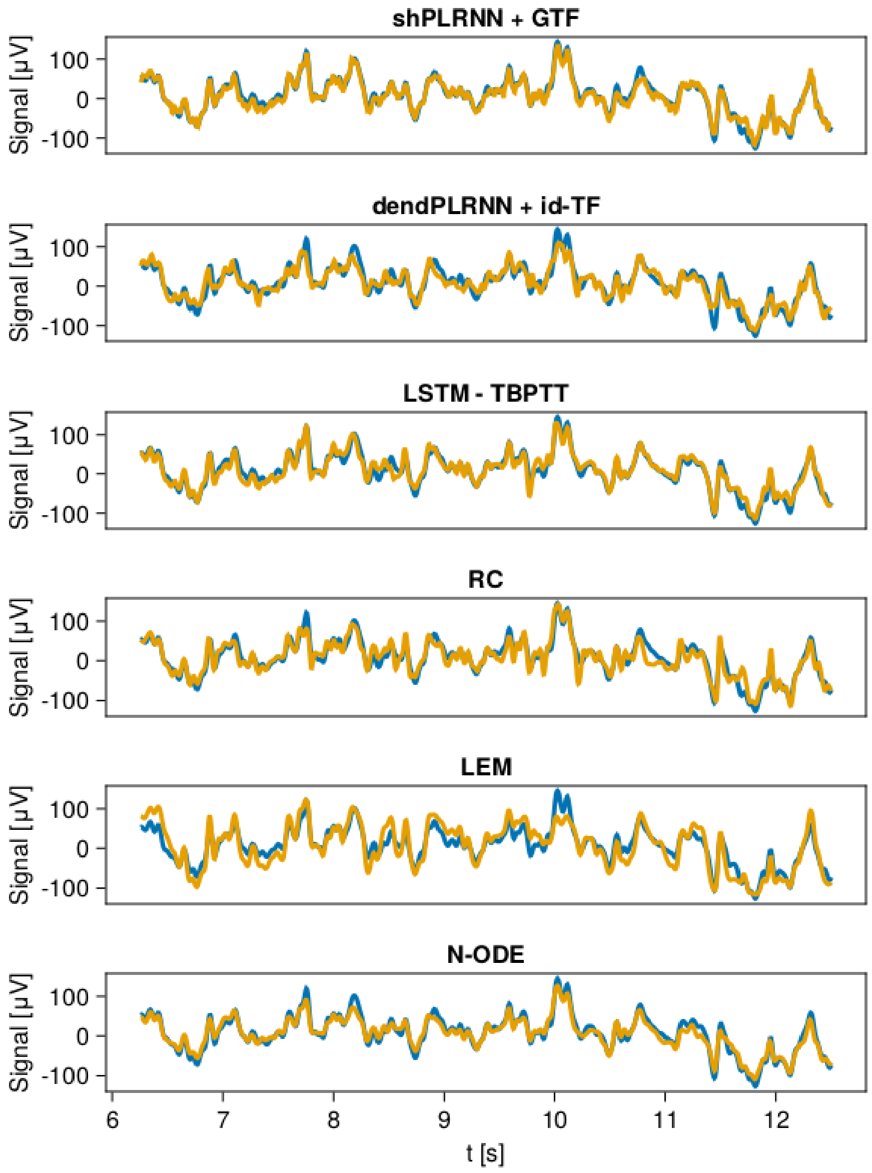


Figure S13. Model *short-term* predictions: Excerpt of EEG time series (blue) vs. 5-step-ahead predictions (yellow) for the different DS reconstruction models & methods compared in this work. Note that essentially all methods provide reasonable short-term forecasts, yet most fail to produce non-trivial limiting dynamics (cf. Fig. 3).