Temporal horizons in forecasting: a performance-learnability trade-off

Anonymous authors
Paper under double-blind review

Abstract

When training autoregressive models to forecast dynamical systems, a critical question arises: how far into the future should the model be trained to predict? Too short a horizon may miss long-term trends, while too long a horizon can impede convergence due to accumulating prediction errors. In this work, we formalize this trade-off by analyzing how the geometry of the loss landscape depends on the training horizon. We prove that for chaotic systems, the loss landscape's roughness grows exponentially with the training horizon, while for limit cycles, it grows linearly, making long-horizon training inherently challenging. However, we also show that models trained on long horizons generalize well to short-term forecasts, whereas those trained on short horizons suffer exponentially (resp. linearly) worse long-term predictions in chaotic (resp. periodic) systems. We validate our theory through numerical experiments and discuss practical implications for selecting training horizons. Our results provide a principled foundation for hyperparameter optimization in autoregressive forecasting models.

1 Introduction

Forecasting the future state of dynamical systems is a fundamental challenge in scientific and engineering disciplines, from climate modeling to robotics. A central objective is to generate accurate predictions as far into the future as possible. Auto-regressive (AR) models, which iteratively feed their own predictions back as inputs, are a standard tool for this task in both traditional scientific computing and machine learning (Petropoulos et al., 2022; Lai & Lu, 2017; Azencot et al., 2020; Van den Oord et al., 2016; Lam et al., 2023).

Traditional AR models were linear and limited to low-dimensional time series (Petropoulos et al., 2022; Lai & Lu, 2017). Modern auto-regressive neural networks (ARNNs), trained via gradient descent, have emerged as powerful alternatives due to their ability to capture nonlinear dynamics in high-dimensional systems. Such nonlinearity is ubiquitous across scientific domains, including physics (Levien & Tan, 1993), robotics (Iqbal et al., 2014), ecology (Benincà et al., 2009; Huisman & Weissing, 1999), and climate science (Houghton et al., 2001). Consequently, ARNNs have demonstrated success in climate prediction (Lam et al., 2023; Kochkov et al., 2024), physics-informed dynamical system emulation (Azencot et al., 2020; Erichson et al., 2019; Li et al., 2020; Miller et al., 2022), and robotic control (Sasagawa et al., 2021; Williams et al., 2017; Heetmeyer et al., 2023). Notably, these systems are typically Markovian, meaning their future states depend only on the current state. This property renders architectures with explicit memory mechanisms, such as recurrent neural networks (RNNs) or transformers, unnecessarily complex and prone to overfitting.

A critical yet understudied challenge in training ARNNs is selecting an appropriate temporal horizon for prediction. Common approaches either use a single-step prediction scheme (Van den Oord et al., 2016; Azencot et al., 2020; Miller et al., 2022) or adopt longer horizons without rigorous justification (Lam et al., 2023; Kochkov et al., 2024; Price et al., 2025). Currently, no theoretical framework exists to guide this choice. This gap extends to mechanistic modeling with differential equations, where methods like piecewise regression (Pisarenko & Sornette, 2004; Doya et al., 1992; Boussange et al., 2024) and multiple shooting (Bock, 1981; England, 1983; Aydogmus & Tor, 2021) similarly lack principled criteria for horizon selection.

In this work, we rigorously analyze how the training temporal horizon affects ARNN performance. We derive practical guidelines for selecting this horizon based on the underlying system dynamics. Our contributions include:

- A theoretical analysis linking the training horizon to the geometry of the loss landscape
- Empirical validation of these relationships across different dynamical systems
- Practical recommendations for horizon selection in both data-driven and mechanistic modeling

The remainder of this paper is organized as follows. We first situate our work within the existing literature. Next, we introduce necessary background and notation. We then present our theoretical analysis of how temporal horizons shape the loss landscape, that we illustrate with numerical experiments. we examine the practical implications of our theoretical findings under the presence of noise and various model sizes, and how it connects with the task of fitting mechanistic models. Finally, we discuss broader implications and future research directions.

2 Related works

While ARNNs are a natural choice for forecasting Markovian systems, much of the literature on general time series forecasting has focused on recurrent neural networks (RNNs) (Grossberg, 2013). RNNs, such as LSTMs (Hochreiter, 1997), leverage recurrent connections to maintain memory of past inputs, enabling them to capture temporal dependencies. More recently, transformers (Vaswani et al., 2017) have emerged as a competitive alternative, often outperforming RNNs in practice (Feng et al., 2024).

Training RNNs or transformers requires backpropagation through time, where gradients are computed across sequences whose length corresponds to the temporal horizon. However, this process is often hampered by the vanishing or exploding gradient problem (EVGP) (Pascanu, 2013; Hochreiter et al., 2001; Bengio et al., 1994). Common strategies to mitigate EVGP include architectural modifications (e.g., gated mechanisms (Hochreiter, 1997)), specialized weight initializations (Narkhede et al., 2022), gradient clipping (Goodfellow, 2016; Zhang et al., 2020), and constrained dynamics (e.g., unitary RNNs (Arjovsky et al., 2016; Chang et al., 2019; Erichson et al., 2020)).

Despite these advances, RNNs often underperform for dynamical systems forecasting, as stability-focused designs can limit expressivity (Orhan & Pitkow, 2019; Schmidt et al., 2021; Mikhaeil et al., 2022). Recent work has proposed dynamically adjusting the BPTT horizon based on system properties like Lyapunov exponents (Mikhaeil et al., 2022), though such approaches still lag behind ARNNs in practice.

In contrast to RNNs, ARNNs lack a robust theoretical framework connecting their training horizon to the underlying system dynamics. Some works adapt the horizon during training to balance stability and accuracy (Boussange et al., 2024; Pisarenko & Sornette, 2004), while others implicitly control it via heuristic strategies. For example, in weather forecasting (Lam et al., 2023) and financial time series modeling (Bernaciak & Griffin, 2024), discount factors reduce the influence of distant prediction errors, effectively shortening the training horizon. Similarly, neural PDE solvers often prioritize high-frequency components, which are inherently tied to shorter temporal dependencies (Wiener, 1930; Lippe et al., 2023; Kurth et al., 2023).

Interestingly, the choice of training horizon extends beyond forecasting. In reinforcement learning, the effective horizon influences model success, leading to techniques like λ-returns (Sutton, 1995; Anand & Precup, 2021). Model predictive control also requires careful horizon selection (Hewing et al., 2020). Even in language modeling, recent empirical results suggest that multi-token prediction improves performance (Gloeckle et al., 2024; Cornille et al., 2024), though the theoretical basis remains unclear.

3 Background and Notation

3.1 Dynamical systems

Dynamical systems can generally be described by a set of update equations of the form

$$x(t) = \phi(x(t - \Delta t)) \tag{1}$$

where $x(t) \in \mathcal{X}$ is the state of the system and ϕ is a deterministic map and Δt is the timestep size, which in our derivations will be set to one for simplicity.

Dynamical systems theory categorizes systems in four main categories: stable, unstable, limit cycles, and chaotic systems (Jost, 2005; Brin & Stuck, 2002). Stable systems are those that converge to an attractor state x^*, and unstable those that diverge from a given point x^*, and in both cases the point x^* defines a constant orbit $\phi(x^*) = 0$. Those are well understood dynamics that are either inherently easy to predict – for stable systems, as the trajectories end in the same point – , or inherently impossible – for unstable systems, as the trajectories are constantly in a new state, and thus out of distribution.

The more interesting families for machine learning are thus limit cycles and chaotic systems, both of which appear exclusively in non-linear systems and which will be the focus of this work.

- **Limit cycles** are orbits that repeat themselves, and can be characterized by the a closed line that defines their orbit and by their period p or their **frequency** $\omega = \frac{2\pi}{p}$, which represents the time it takes for them to come back to a previous point, such that $x(t + p) = x(t)$(Brin & Stuck, 2002). .
- **Chaotic systems** on the other hand never repeat any point in the trajectory, and can be defined by their chaotic attractor – the set of points towards which the system will evolve, and their **Lyapunov exponent** λ which measures how two points that are initially close in the chaotic attractor diverge with time and is thus measures as $|x(t) - x'(t)| \approx e^{\lambda t}|x(0) - x'(0)|$, where $x(0)$ is infinitesimally close to $x'(0)$(Brin & Stuck, 2002).

Crucially, both chaotic or limit cycle systems are practically impossible to forecast very far into the future, as small differences between the model and the system being modeled will make long term predictions essentially random (A.5). Furthermore, notice that chaotic systems are locally unstable (Jost, 2005), so our derivations could be applied to systems that are unstable (even though we don't focus on them for the reasons noted before).

3.2 Examples of dynamical systems

In order to illustrate our theory, we will use the following examples of dynamical systems: A Lorenz attractor, a Double pendulum, a simple limit cylce, and a food web model. The lorenz attractor and the double pendulum are classical examples of chaotic systems, while the limit cycle is, as indicated, a limit cycle, and the food web is chaotic, but only in very long timescales, but for short timescales it is effectively a limit cycle. All the systems are described in detail in appendix B.

3.3 Autoregressive models

As dynamical systems have dynamics that are fixed and are autonomous, they are often modeled by autoregressive models, which are defined by a function f, such that for a given state $x(m)$, $f(x(m), \theta)$ gives the next state $x(m + 1)$ or an approximation thereof.

Since f is by construction made to output vectors with the same dimensionality as the input, we can generate a future state $x(m + \tau)$ by simple composition of the model

$$x(m + \tau) = f \circ \cdots \circ f\left(x(m), \theta\right) = f^{\tau}\left(x(m), \theta\right). \tag{2}$$

Note here that for simplifying notations we assume that one composition of f corresponds to a time step of unit length. Note that we use here the classical definition of autorregressive, since we are only considering

the last output state to compute the next (due to the Markovian nature of dynamical systems). This does not reflect the standard Transformer architecture, which has access to all previous states of the model.

We will focus on models where the function f is implemented by a neural network, and we will thus refer to autoregressive neural networks (ARNNs). While ARNNs can be implemented by multiple architectures (Lam et al., 2023; Azencot et al., 2020; Sangiorgio et al., 2022), we will focus on multi-layer perceptrons MLPs to validate our theory. For details about the models, please see appendix E.

3.4 Loss functions for ARNNs

Consider a time series of discrete states $\mathbf{x} = [x(0), x(1), \ldots, x(M)]$ sampled at times $[0, 1, 2, ..., M]$ which cover the full state of the system, we can define the ARNNs loss as

$$\mathcal{L}_{\mathbf{x}}(\theta, T) = \frac{1}{M-T} \sum_{m=1}^{M-T} \frac{1}{T} \sum_{\tau=1}^{T} \left\| x(m+\tau) - f^{\tau}\left(x(m), \theta\right) \right\|. \tag{3}$$

where τ indexes the number of prediction steps and T is the temporal horizon, which determines the extent to which future states influence learning. Note that it is possible to have different temporal horizons (Ts) for training and testing.

3.5 Theoretical assumptions and model set-up

To make our derivations we make three main assumptions:

- We assume that the loss function is smooth, and we can therefore use gradient descent. Note that most modern neural networks use ReLus, where the loss would not be smooth. To avoid this, we could assume that neurons use a soft-ReLu $\text{Softplus}(x) = \frac{1}{\beta} * \log(1 + \exp(\beta * x))$, which approximate ReLUs as β is high.
- We assume that our dynamical systems are ergodic (and therefore stationary), and deterministic. We also assume that our systems are fully observable, and as a consequence Markovian. This last assumption can be relaxed, as deterministic systems that are only partially observable (where a longer past needs to be considered) can be turned into Markovian systems by creating an extended state that incorporates previous observations (Takens, 1981; Noakes, 1991).
- We assume that we have enough data during training. In terms of dynamical systems, we assume that we have sample trajectories that are long enough to evaluate various temporal horizons, and that we have data covering all the state space of the model.

4 Temporal horizons and loss landscapes

In this section, we derive a connection between the training loss gradient and the training temporal horizon, and exploit it to understand the geometry of the loss landscape. Full proofs of our theorems and their corollaries are presented in the appendix A; in the main text we only provide an outline. We provide simulations for all our theorems.

4.1 The link between system dynamics and training loss

The following definition helps us characterize the region on the parameter space where the model $f(\cdot, \theta)$ is a reasonable approximation of the true dynamics ϕ.

Definition 4.1 (ϵ-bounded region)**.** For $\epsilon > 0$, an ϵ-bounded region of the parameter space Θ_ϵ for a model trained to forecast a dynamical system with a bounded stationary distribution is a convex set within the space of parameters of the model such that

$$\|f(x + \epsilon\vec{r}, \theta) - (f(x, \theta) + J_\phi(x)\vec{r}\epsilon)\| < \epsilon^2 \quad \forall x \in \text{Conv}(\mathbf{x})$$

where J_ϕ is the Jacobian matrix of $\phi(\cdot)$, $\vec{r}$ is a unitary random vector and $\text{Conv}(\mathbf{x})$ is the convex hull of the states $\mathbf{x}$.

We note that ϵ-bounded regions exist for arbitrarily small ϵ (lemma A.1), and have bounded loss (lemma A.8). Importantly, a small ϵ can only be achieved in models that have been partially trained, so that they are already a good approximation of the system dynamics. Thus, our theory applies to models that have been at least partially trained, not to models at initialization.

Now we can connect the dynamics of the system to the loss of a model trained to forecast the system. Consider the function

$$g(T) = \frac{\|\nabla_\theta \mathcal{L}_{\mathbf{x}}(\theta, T)\|}{\|\nabla_\theta \mathcal{L}_{\mathbf{x}}(\theta, 1)\|} \tag{4}$$

which represents the relative scaling of the gradient with respect to T. The following proposition and its validation in fig. 1 shows how g evolves with T.

Theorem 4.2 (Loss gradient growth)**.** *Consider a dynamical system which is either chaotic, contains locally unstable trajectories or limit cycles, and a corresponding model with an ϵ-bounded region of the parameter space $\Theta_\epsilon - \{\Theta^\rho_{\min}\}$ where $\Theta^\rho_{\min}$ are all the balls of radius $\rho \ll \epsilon$ around the minima of the loss. For a sufficiently small ϵ, when the forecasting horizon T is large, $g(T)$ scales with T as*

$$g(T) = \begin{cases} \mathcal{O}(e^{\lambda T}), & \textit{for chaotic/unstable} \\ \mathcal{O}(\omega T), & \textit{for limit cycles.} \end{cases} \tag{5}$$

Proof sketch. The Jacobian of the system over a time window T will scale exponentially or linearly with λ or ω. Furthermore, within definition 4.1, the Jacobian of the model remains close to the Jacobian of the true system (lemma A.6). The Jacobian of the model is always contained in the expression for $\nabla_\theta \mathcal{L}_{\mathbf{x}}(\theta, T)$, and thus it scales with T. See A.9 for the full proof. Note that we removed the minima in $\{\Theta^\rho_{\min}\}$ to avoid zeros, which would lead to a divergence of g. We will treat them in the next corollary. □

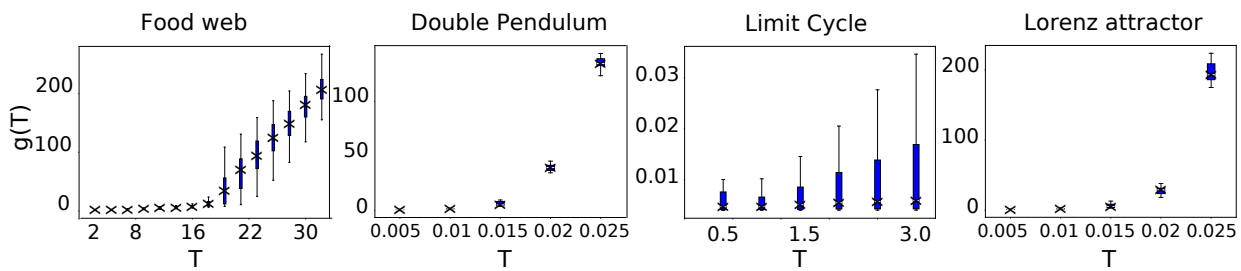

Figure 1: **Gradient scaling**: We measured the relative L_2-norm of the gradients ($g(T)$) as a function of T for MLPs $g(T)$ during training. As expected from theorem 4.2, we observe an exponential increase of $g(T)$ for the lorenz system and the double pendulum, and a linear one for the limit cycle and the food web (which initially shows periodic dynamics). Notice that we used the temporal units of the system dynamics as a unit to make our results independent of the choice of timestep size.

An important limitation of theorem 4.2 is that it does not inform us about the minima of the loss. Yet, we can use it to gain insights into the Hessian around those same minima,

Corollary 4.3. *Consider a dynamical system which is either chaotic or a limit cycle, and a corresponding model with with minima in an ϵ-bounded region of the parameter space Θ_ϵ. For a sufficiently small ϵ, when the forecasting horizon T is large, the Hessian around a minima scales with T as*

$$\frac{\|H(\theta, T)\|^*}{\|H(\theta, 1)\|^*} = \begin{cases} \mathcal{O}(e^{\lambda T}), & \textit{for chaotic systems} \\ \mathcal{O}(\omega T), & \textit{for limit cycles} \end{cases} \tag{6}$$

where $\|\cdot\|^*$ is the sum of the eigenvalues of the Hessian, also known as the nuclear norm.

Proof sketch. As the gradient grows with T, the second derivative must also grow to keep up. To build this connection, we consider small balls around the minima, and use the divergence theorem, where the gradient is the vector field. □

Having derived corollary 4.3, we find that there are two potential ways to interpret it:

- Classical statistics suggest that having a minimum with a high Fisher information – a quantity that scales with the Hessian – means that the model extracts a lot of information from the data Ly et al. (2017). This would suggest that it is better to pick a high T as it would make any minima found very precise.
- Flat minima seem to be better at generalization, because any difference in parameters (for example due to differences between training and testing data) will have little effect on the loss Hochreiter & Schmidhuber (1997); Keskar et al. (2016). This would suggest that it is better to pick a low T to improve generalization.

Clearly, both intuitions are at odds with each other. To gain further insights we will use theorem 4.2 to study the geometry of the loss landscape.

4.2 Generalization across temporal horizons

A key problem in comparing models trained with high and low T is that different time horizons correspond to different problems, and therefore we should not directly compare their minima in terms of their loss. Instead, we need a metric that directly relates the performance of a model trained to one temporal horizon to its performance in another temporal horizon.

We thus start by asking whether a minima found for a given time horizon will work for another time horizon. We formalize this notion as the ratio between two minima for different time horizons,

$$r(T_\mathrm{h}, T_\mathrm{l}) = \frac{\mathcal{L}_\mathbf{x}(\theta_\mathrm{h}^\mathrm{min}, T_\mathrm{h}) - \mathcal{L}_\mathbf{x}(\theta_\mathrm{l}^\mathrm{min}, T_\mathrm{h})}{\mathcal{L}_\mathbf{x}(\theta_\mathrm{l}^\mathrm{min}, T_\mathrm{l}) - \mathcal{L}_\mathbf{x}(\theta_\mathrm{h}^\mathrm{min}, T_\mathrm{l})} \tag{7}$$

where the parameter values $\theta_\mathrm{h}^\mathrm{min}, \theta_\mathrm{l}^\mathrm{min}$ are two minima found with losses evaluated at T_l and T_h respectively with $T_\mathrm{l} < T_\mathrm{h}$ (l stands for low and h for high).

Note that there is no guarantee that the distribution of minima is similar across temporal horizons, and therefore we need to restrict r to comparable minima. Thus, we will focus on specific pairs $(\theta_\mathrm{h}^\mathrm{min}, \theta_\mathrm{l}^\mathrm{min})$, which we define as being in each other basin of attraction: if the model is initialized with parameters $\theta_\mathrm{h}^\mathrm{min}$ and trained with the temporal horizon T_l, then it would converge to $\theta_\mathrm{l}^\mathrm{min}$, and conversely, if the model starts with parameters $\theta_\mathrm{l}^\mathrm{min}$, it would converge to $\theta_\mathrm{h}^\mathrm{min}$, if trained at T_h.

Theorem 4.4 (Minima with long forecasting horizons generalize to lower horizons)**.** *Consider a model $f(\cdot, \theta)$ with θ in an ϵ-bounded region. We assume the existence of two minima θ_l^min and θ_h^min for the losses $\mathcal{L}_\mathbf{x}(\theta, T_l)$ and $\mathcal{L}_\mathbf{x}(\theta, T_h)$ respectively with $T_h > T_l$, which are both within each other's basin of attraction. Then the difference in the change in losses follows*

$$r(T_h, T_l) = \begin{cases} \mathcal{O}(e^{\lambda(T_h - T_l)}), & \textit{for chaotic/unstable} \\ \mathcal{O}(\omega(T_h - T_l)), & \textit{for limit cycles} \end{cases} \tag{8}$$

Proof Sketch. Since the two minima are on each other's basin of attraction, we can follow the gradient of the loss for T_h or T_l, between the two minima. The change in loss is the integral of that gradient, which scales as theorem 4.2. Taking the ratio of the aforementioned change in loss yields our result. See A.11 for the full proof. □

Intuitively, the theorem reflects the fact that long temporal predictions rely on shorter predictions, implying that models that are able to make predictions over long temporal horizons must also make good predictions

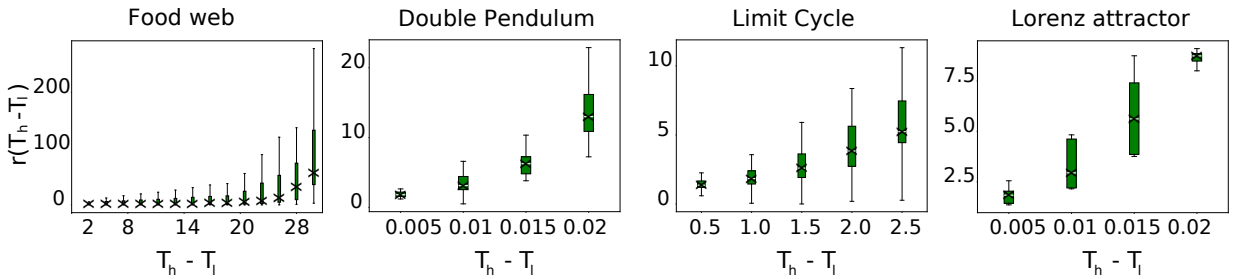

Figure 2: **Performance ratios for longer temporal horizons**: We measured the ratio of the difference between losses on connected minima found at different temporal horizons. As expected from theorem 4.4, we observe an upward trend for all the systems, with the limit cycle being linear, the double pendulum and food web being similar to an exponential and the Lorenz attractor being somewhat inconclusive due to the high variance. Notice that here the difference in losses is also implicitly linked to the accuracy of the minima we find, which depends on the optimizer and the randomness of the system, thus it is normal that we do not find a perfect match to our theory.

over short time horizons. However, the converse is not necessarily true, and valid short-term predictions might diverge in the long run, suggesting that $\theta_{\mathrm{h}}^{\min}$ might capture better the global dynamics of the system than $\theta_{\mathrm{l}}^{\min}$.

4.3 Loss landscape roughness and the temporal horizon

Although theorem 4.4 would suggest that we should train models with a long T, we need to beware that such training might be extremely hard. We quantify this through a notion of roughness, which we take as the number of maxima and minima of the loss $z(T, \theta_1, \theta_2)$ found over a line in parameter space $\overrightarrow{\theta_1\theta_2}$, and prove that it scales with the temporal horizon in theorem 4.5, which is validated empirically in fig. 3.

Theorem 4.5 (Loss landscape roughness)**.** *For any two points θ_1, θ_2 in an ϵ-bounded region of the parameter space that are not in a connected region of zero loss*[1]*, the number of minima and maxima along the line segment that connects them will grow as*

$$z(T, \theta_1, \theta_2) = \begin{cases} \mathcal{O}(e^{\lambda T}\|\theta_1 - \theta_2\|), & \textit{for chaotic/unstable} \\ \mathcal{O}(\omega T\|\theta_1 - \theta_2\|), & \textit{for limit cycles} \end{cases} \tag{9}$$

Proof Sketch. Consider two arbitrary elements θ_1, θ_2 in an ϵ-bounded region. We consider the variation in the loss function through a line segment l connecting θ_1 and θ_2, it takes the form of an integral dependent on $|\partial_l \mathcal{L}_{\mathbf{x}}(l, T)|$. This derivative is the projection of $\nabla_\theta \mathcal{L}_{\mathbf{x}}(l, T)$ onto $\vec{u}_{\theta_1 \to \theta_2}$, the unit vector of the line segment, and thus grows with T as shown in theorem 4.2. As T grows, the loss variation grows beyond the maximum possible loss $\mathcal{L}_\epsilon^{\max}$, thus it must go up and down, and hence have a minimum or maximum. By letting T grow further, it will have more and more minima and maxima. See A.13 for the full proof. □

A curious corollary from this theorem shows that as the temporal horizon grows towards $T \to \infty$, the loss landscape becomes a fractal (A.14). Since a fractal is non-differentiable, this result implies that gradient descent methods would inevitably fail. In the more realistic case of $T < \infty$, theorem 4.5 simply states that the loss landscape becomes harder to navigate as T grows.

Our theory shows that there are better minima when we use longer temporal horizons during training (theorem 4.4), but also that those minima are harder to find (theorem 4.5). This would suggest, our loss surface analysis suggests that increasing T will lead to better performance by theorem 4.4 . However, as a

[1]such scenario is not covered in the assumptions on theorem 4.2, and in this case there is no learning

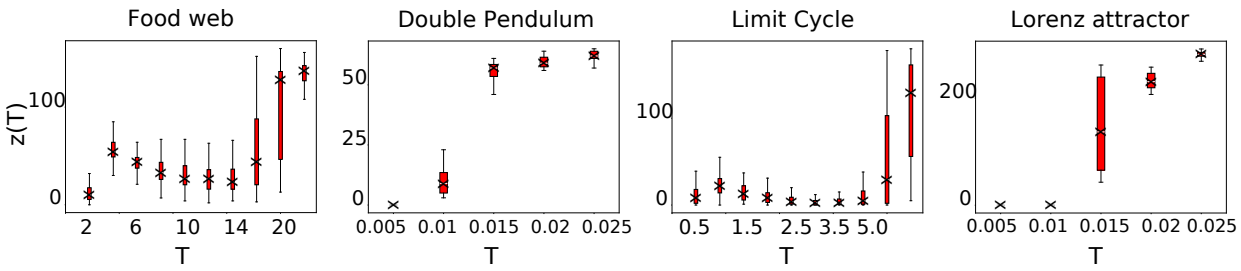

Figure 3: **Loss landscape roughness**: We measured the number of minima and maxima on a cross-section between two parameter sets θ_1, θ_2, both of which were found at late training stages and had low loss. As expected from theorem 4.5, we see a clear increase in the number of zeros found. Note the number of minima and maxima that we can detect is limited by how much can we discretize the line between θ_1 and θ_2, and since we have to evaluate the loss at each point in that line, our results underestimate the number of local minima. This explains the saturation seen in the Double Pendulum and the Lorenz attractor which is not expected from our theory.

result of theorem 4.5 , we would expect that a large value of T will render learning impractical. In the next section we explore in more depth this trade-off and other practical implications of our theory.

5 Practical insights

To demonstrate the connection of our theory to practice, we evaluate the forecasting performance of MLPs for different temporal horizons. We focus on three main points: (1) the choice of the optimal training temporal horizon T and the factors that affect it, and the application of our theory to fitting the parameters of mechanistic models in physical or biological systems.

5.1 Temporal horizon as an hyper-parameter

We trained residual MLPs with different training temporal horizons for the four dynamical systems presented in appendix B until they appeared to reach convergence or a large total wall time cutoff. Based on our theory, we would expect to observe the following set of behaviors: (1) an initial increase in the performance of models with larger prediction horizons (theorem 4.4) but (2) worse performance for models trained on time horizons sufficiently long such that the loss landscape becomes harder to navigate (theorem 4.5). We find the expected U-shaped curve in all the dynamical systems considered, as seen in fig. 5. Importantly, the optimal training horizon is neither one nor the time that we wish to use for forecasting, suggesting that setting the temporal horizon for training naively would have significant effects on performance.

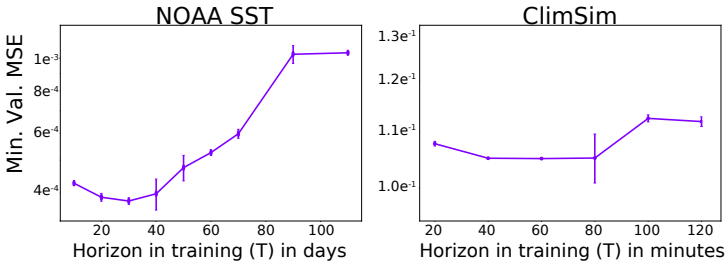

Figure 4: **Performance of various architectures trained with different training horizons on the NOAA SST and ClimSim datasets**. We evaluated the average performance 5 time steps into the future for both tasks, which correspond to 50 days in the case of SST and 100 minutes in the case of ClimSim. We plotted the median of the MSE with upper and lower quartiles. The loss is convex with respect to the temporal horizon in training, and the optimal training horizon is not the evaluation horizon.

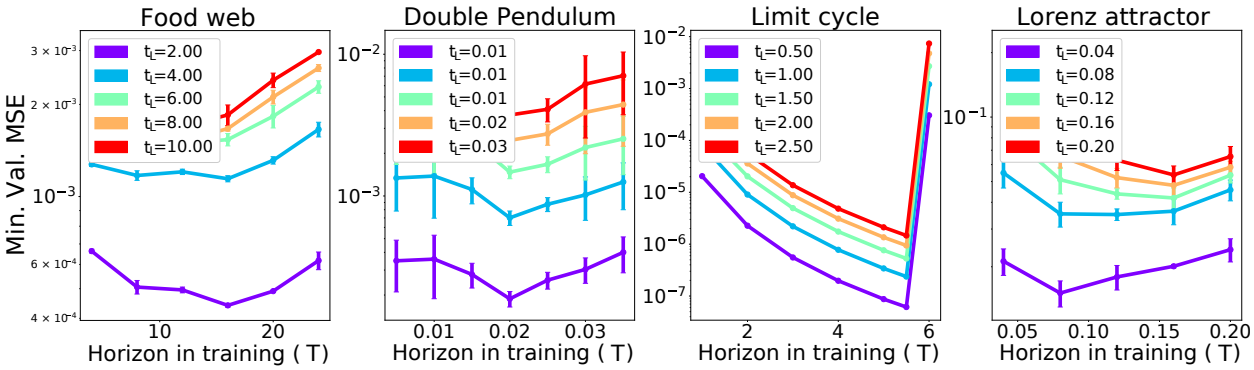

Figure 5: **Performance of residual MLPs trained to predict four dynamical systems with minimum validation MSE on the y-axis and the temporal horizon T used in training on the x-axis**. Each line in the figure is associated with a given evaluation horizon marked by T_l. Each point as one goes along the line represents the median performance of a model (with interquartile range error bars) trained with an increasing training horizon T when evaluated T_l time into the future. That is, if we have a model $f(x, \theta_T)$ trained on T, the points above T on the graph are $\|x(T_l) - f^{n_t}(x(0), \theta_T)\|$ where n_t is the number of auto-regressive steps to get to T_l. As expected, the loss grows with T_l, the loss is convex, and the optimal predictive horizon for training rarely coincides with T_l.

To validate the last point, we trained MLPs on two real-world datasets taken from climate science that are often used as benchmarks, namely the National Oceanographic Atmospheric Administration Sea Surface Temperature dataset (Huang et al., 2021), and ClimSIM dataset (see appendix C for details), both of which are routinely evaluated at one timestep in the future. In both cases we find that the optimum training horizon is longer than a single step-ahead.

However, a high T gives worse losses because navigating the loss landscape becomes harder (theorem 4.5). Thus, according to our theory an increase in the computational resources used during training should allow an optimizer to find minima with higher T, which should generalize better (theorem 4.4). We performed this analysis in appendix G for the ecology system, finding that the computing time used for training does indeed correlate with the temporal horizon for training T.

Interestingly, the optimal training time horizons are not equivalent to the testing horizon, further suggesting that the optimal training horizons are inextricably linked to the system's dynamics rather than the desired prediction horizon. This was found not only on our simple dynamical systems, but also on more complex benchmark datasets on climate modeling (sea surface temperature data National Oceanic and Atmospheric Administration Huang et al. (2021) and an adapted version of the ClimSim Yu et al. (2023) dataset).

Notice that our results on the National Oceanic and Atmospheric Administration dataset Huang et al. (2021) are consistent with our theory, even though the data contains noise. To evaluate whether our theory is compatible with noise, we performed an in-depth experiment for the ecology model, which we present in appendix F. Our simulations suggest that our theory remains valid for systems with small noise, where the neural networks can approximate the dynamics of the model well (see lemma A.3). However, over long horizons it is impossible to predict the state of a stochastic system (see lemma A.5), thus models will converge to a fixed bad forecast, as observed in fig. 4. It might be tempting to add more parameters to the model to improve forecasting capacity under increased noise. However, doing so only increases the performance of the model in small noise scenarios (appendix F). Furthermore, adding parameters increases the computational resources needed to train the model, and thus we might achieve lower performance if computational resources are finite (appendix F).

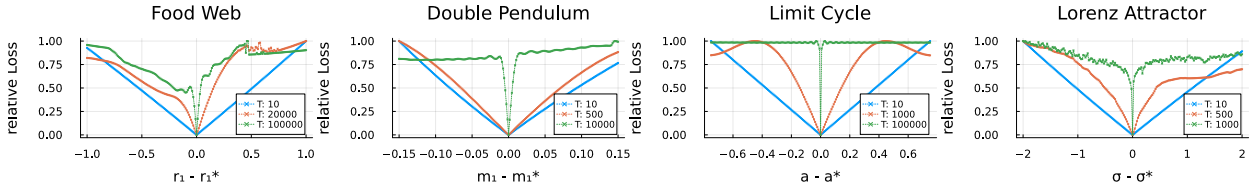

Figure 6: Normalized values of $\mathcal{L}_{\mathbf{x}}(\theta, T)$ from eq. (3) when altering one dimension of θ for different values of T in several dynamical systems models. The loss is normalized for each T so that its maximum value is 1. For all the systems, the loss is generally flat with a single minima for low T and becomes steeper and has more minima for high T. Notice that exact models can approximate the system much better than neural networks, thus it is possible to reach zero loss, and the differences between model and data become evident at long timescales.

5.2 Loss landscape geometry for mechanistic models

Our theory is built for general AR models, including mechanistic models such as differential equation-based models, where the processes driving the dynamics of the system under study are known and already encoded in the model's structure, but where the parameters of the model are unknown. Here we extend our theory to dynamical systems governed by known models.

We use the same systems as in the previous section, but this time we use the models described in appendix B, both to generate time series data and as generative models (with unknown parameters). We study the geometry of the forecasting loss from eq. (3) in fig. 6, where we changed a single parameter per model and evaluated the loss function at multiple temporal horizons, and a more complete view over all parameters is presented for the Lorenz system in appendix D.

All models achieve a zero-loss when the parameters of the model are the same as those that generated the data, because the model is a perfect copy of the original data-generating equations. This implies that the generalization ratio $r(T_{\mathrm{l}}, T_{\mathrm{h}})$ in theorem 4.4 is undetermined because both long and short T provide a loss of zero, implying that a short T generalizes as well as a long T. We also observe that the loss landscape becomes rougher as T grows, in line with theorem 4.5, and for extremely large T the loss is similar across almost all parameters, as per lemma A.5. Notice that the T that we use are much larger than in the previous sections, because the models here capture the time series perfectly, thus it takes a much longer for our theory to be reflected in the loss function.

Thus, our theory would suggest that it is always better to train with the smallest possible T, when the system is known and perfectly deterministic. However, this statement should be contrasted with the practical limitations of mechanistic models, which are often not perfect, and real observations, which are contaminated with noise. Indeed, empirical studies considering noise and imperfect models suggest that there is a trade-off in the choice of T similar to the one we found with MLPs (Pisarenko & Sornette, 2004; Boussange et al., 2024). Thus, an adapted version of our theory for dynamical systems should include imperfections or noise where the loss is never exactly zero thus we do not get the indeterminacy. This could be done by using lemma A.7 as a starting building block and derive a variant of theorem 4.4.

6 Discussion

Our work establishes a fundamental connection between the training temporal horizon T and the underlying dynamics of the system. Theoretically and empirically, we show that common training paradigms—such as single-step prediction or matching the testing horizon—are rarely optimal. Instead, the ideal T depends on the system's intrinsic dynamics (e.g., Lyapunov exponent λ), but practical considerations like model capacity, computational budget, and stochasticity also play a role. Our findings motivate a principled approach to hyperparameter search, where T is selected based on the system's dynamical properties.

A key practical limitation is that we currently lack a robust guidance to choose T besides a standard hyperparameter search methods. We investigated an iterative algorithm that automatically adapts T for a given an initial learning rate (see appendix H, algorithm 1). The scheme showed promise for low λ or limit cycles, outperforming fixed choices of T for fixed computation time, learning rates and algorithm hyperparameters, but did not perform well on more chaotic systems. Future works should investigate this further, for example by jointly optimizing T and the learning rate by leveraging loss landscape curvature, building on ideas from Roulet et al. (2024). Such an approach could extend beyond fixed horizons: many systems exhibit time-varying predictability (e.g., intermittent chaos or quasi-periodicity Zhou et al. (2006)), where a dynamic T could sharpen the trade-off between learnability and generalization. For instance, shorter horizons might focus training on unpredictable regions, while longer horizons capture coherent dynamics. This could unlock more efficient training for complex, non-stationary systems.

References

Anna Åkesson, Alva Curtsdotter, Anna Eklöf, Bo Ebenman, Jon Norberg, and György Barabás. The importance of species interactions in eco-evolutionary community dynamics under climate change. *Nature Communications*, 12(1):4759, 2021.

Nishanth Anand and Doina Precup. Preferential temporal difference learning. *arXiv preprint arXiv:2106.06508*, 2021.

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

Ozgur Aydogmus and Ali Hakan Tor. A modified multiple shooting algorithm for parameter estimation in odes using adjoint sensitivity analysis. *Applied Mathematics and Computation*, 390:125644, 2021.

Omri Azencot, N. Benjamin Erichson, Vanessa Lin, and Michael Mahoney. Forecasting sequential data using consistent koopman autoencoders. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 475–485. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/azencot20a.html`.

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

Elisa Benincà, Klaus D Jöhnk, Reinhard Heerkloss, and Jef Huisman. Coupled predator–prey oscillations in a chaotic food web. *Ecology letters*, 12(12):1367–1378, 2009.

Dawid Bernaciak and Jim E Griffin. A loss discounting framework for model averaging and selection in time series models. *International Journal of Forecasting*, 2024.

Hans Georg Bock. Numerical treatment of inverse problems in chemical reaction kinetics. In *Modelling of Chemical Reaction Systems: Proceedings of an International Workshop, Heidelberg, Fed. Rep. of Germany, September 1–5, 1980*, pp. 102–125. Springer, 1981.

Victor Boussange, Pau Vilimelis Aceituno, Frank Schäfer, and Loïc Pellissier. Partitioning time series to improve process-based models with machine learning. *bioRxiv*, 2024. doi: 10.1101/2022.07.25.501365. URL `https://www.biorxiv.org/content/early/2024/04/11/2022.07.25.501365`.

Michael Brin and Garrett Stuck. *Introduction to dynamical systems*. Cambridge university press, 2002.

Bo Chang, Minmin Chen, Eldad Haber, and Ed H Chi. Antisymmetricrnn: A dynamical system view on recurrent neural networks. *arXiv preprint arXiv:1902.09689*, 2019.

Nathan Cornille, Marie-Francine Moens, and Florian Mai. Learning to plan for language modeling from unlabeled data. *arXiv preprint arXiv:2404.00614*, 2024.

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

Gerald A Edgar and Gerald A Edgar. *Measure, topology, and fractal geometry*, volume 2. Springer, 2008.

Roland England. *Some Examples of Parameter Estimation by Multiple Shooting*, pp. 122–136. Birkhäuser Boston, Boston, MA, 1983. ISBN 978-1-4684-7324-7. doi: 10.1007/978-1-4684-7324-7_8. URL https://doi.org/10.1007/978-1-4684-7324-7_8.

N. Benjamin Erichson, Michael Muehlebach, and Michael W. Mahoney. Physics-informed autoencoders for lyapunov-stable fluid flow prediction, 2019.

N Benjamin Erichson, Omri Azencot, Alejandro Queiruga, Liam Hodgkinson, and Michael W Mahoney. Lipschitz recurrent neural networks. *arXiv preprint arXiv:2006.12070*, 2020.

Leo Feng, Frederick Tung, Mohamed Osama Ahmed, Yoshua Bengio, and Hossein Hajimirsadegh. Were rnns all we needed? *arXiv preprint arXiv:2410.01201*, 2024.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*, 2024.

Ian Goodfellow. Deep learning, 2016.

Stephen Grossberg. Recurrent neural networks. *Scholarpedia*, 8(2):1888, 2013.

Mukul K. Gupta, Kamal Bansal, and Arun K. Singh. Mass and length dependent chaotic behavior of a double pendulum. *IFAC Proceedings Volumes*, 47(1):297–301, 2014. ISSN 1474-6670. doi: https://doi.org/10.3182/20140313-3-IN-3024.00071. URL https://www.sciencedirect.com/science/article/pii/S1474667016326714. 3rd International Conference on Advances in Control and Optimization of Dynamical Systems (2014).

Alan Hastings. Structured models of metapopulation dynamics. *Biological Journal of the Linnean Society*, 42 (1-2):57–71, 1991.

Frederik Heetmeyer, Marcin Paluch, Diego Bolliger, Florian Bolli, Xiang Deng, Ennio Filicicchia, and Tobi Delbruck. Rpgd: A small-batch parallel gradient descent optimizer with explorative resampling for nonlinear model predictive control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3218–3224. IEEE, 2023.

Lukas Hewing, Kim P Wabersich, Marcel Menner, and Melanie N Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3 (1):269–296, 2020.

S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

John Theodore Houghton, YDJG Ding, David J Griggs, Maria Noguer, Paul J van der Linden, Xiaosu Dai, Kathy Maskell, Colin A Johnson, et al. *Climate change 2001: the scientific basis*, volume 881. Cambridge university press, 2001.

Boyin Huang, Chunying Liu, Viva Banzon, Eric Freeman, Garrett Graham, Bill Hankins, Tom Smith, and Huai-Min Zhang. Improvements of the daily optimum interpolation sea surface temperature (doisst) version 2.1. *Journal of Climate*, 34(8):2923 – 2939, 2021. doi: 10.1175/JCLI-D-20-0166.1. URL https://journals.ametsoc.org/view/journals/clim/34/8/JCLI-D-20-0166.1.xml.

Jef Huisman and Franz J. Weissing. Biodiversity of plankton by species oscillations and chaos. *Nature*, 402 (6760):407–410, 1999.

Sajid Iqbal, Xizhe Zang, Yanhe Zhu, and Jie Zhao. Bifurcations and chaos in passive dynamic walking: A review. *Robotics and Autonomous Systems*, 62(6):889–909, 2014.

Jürgen Jost. *Dynamical systems: examples of complex behaviour*. Springer Science & Business Media, 2005.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.

Aaron Klebanoff and Alan Hastings. Chaos in three species food chains. *Journal of Mathematical Biology*, 32: 427–451, 1994.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, Sam Hatfield, Peter Battaglia, Alvaro Sanchez-Gonzalez, Matthew Willson, Michael P. Brenner, and Stephan Hoyer. Neural general circulation models for weather and climate, 2024.

Thorsten Kurth, Shashank Subramanian, Peter Harrington, Jaideep Pathak, Morteza Mardani, David Hall, Andrea Miele, Karthik Kashinath, and Anima Anandkumar. Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *Proceedings of the platform for advanced scientific computing conference*, pp. 1–11, 2023.

Dihui Lai and Bingfeng Lu. Autoregressive model for time series as a deterministic dynamic system. *Predictive Analytics and Futurism*, 15:7–9, 2017.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Graphcast: Learning skillful medium-range global weather forecasting, 2023.

RB Levien and SM Tan. Double pendulum: An experiment in chaos. *American Journal of Physics*, 61(11): 1038–1044, 1993.

Yunzhu Li, Hao He, Jiajun Wu, Dina Katabi, and Antonio Torralba. Learning compositional koopman operators for model-based control. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=H1ldzA4tPr`.

Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. Pde-refiner: Achieving accurate long rollouts with neural pde solvers. *Advances in Neural Information Processing Systems*, 36:67398–67433, 2023.

Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of Atmospheric Sciences*, 20(2):130 – 141, 1963. doi: https://doi.org/10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

B.B. Mandelbrot. *The Fractal Geometry of Nature*. Einaudi paperbacks. Henry Holt and Company, 1983. ISBN 9780716711865. URL `https://books.google.ch/books?id=0R2LkE3N7-oC`.

Noah Marti. ChaoticInference - Julia library for parameter inference specialized for chaotic systems., 2024. URL `https://gitlab.com/chaoticinference/chaotic-inference`.

Kevin McCann and Peter Yodzis. Biological conditions for chaos in a three-species food chain. *Ecology*, 75 (2):561–564, 1994.

Jonas Mikhaeil, Zahra Monfared, and Daniel Durstewitz. On the difficulty of learning chaotic dynamics with rnns, 2022.

Jack W. Miller, Charles O'Neill, Navid C. Constantinou, and Omri Azencot. Eigenvalue initialisation and regularisation for koopman autoencoders, 2022.

Meenal V Narkhede, Prashant P Bartakke, and Mukul S Sutaone. A review on weight initialization strategies for neural networks. *Artificial intelligence review*, 55(1):291–322, 2022.

Lyle Noakes. The takens embedding theorem. *International Journal of Bifurcation and Chaos*, 1(04):867–872, 1991.

A Emin Orhan and Xaq Pitkow. Improved memory in recurrent neural networks with sequential non-normal dynamics. *arXiv preprint arXiv:1905.13715*, 2019.

R Pascanu. On the difficulty of training recurrent neural networks. *arXiv preprint arXiv:1211.5063*, 2013.

Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J Bessa, Jakub Bijak, John E Boylan, et al. Forecasting: theory and practice. *International Journal of Forecasting*, 38(3):705–871, 2022.

V. F. Pisarenko and D. Sornette. Statistical methods of parameter estimation for deterministically chaotic time series. *Physical Review E*, 69(3):036122, March 2004. ISSN 1539-3755. doi: 10.1103/PhysRevE.69.036122.

David M Post, Michael L Pace, and Nelson G Hairston Jr. Ecosystem size determines food-chain length in lakes. *Nature*, 405(6790):1047–1049, 2000.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044):84–90, 2025.

Christopher Rackauckas and Qing Nie. DifferentialEquations.jl–a performant and feature-rich ecosystem for solving differential equations in Julia. *Journal of Open Research Software*, 5(1), 2017.

Vincent Roulet, Atish Agarwala, Jean-Bastien Grill, Grzegorz Swirszcz, Mathieu Blondel, and Fabian Pedregosa. Stepping on the edge: Curvature aware learning rate tuners. *arXiv preprint arXiv:2407.06183*, 2024.

Matteo Sangiorgio, Fabio Dercole, and Giorgio Guariso. *Deep learning in multi-step prediction of chaotic dynamics: from deterministic models to real-world systems*. Springer, 2022.

Ayumu Sasagawa, Sho Sakaino, and Toshiaki Tsuji. Motion generation using bilateral control-based imitation learning with autoregressive learning. *IEEE Access*, 9:20508–20520, 2021.

Dominik Schmidt, Georgia Koppe, Zahra Monfared, Max Beutelspacher, and Daniel Durstewitz. Identifying nonlinear dynamical systems with multiple time scales and long-range dependencies, 2021. URL `https://arxiv.org/abs/1910.03471`.

Richard S Sutton. Td models: Modeling the world at a mixture of time scales. In *Machine learning proceedings 1995*, pp. 531–539. Elsevier, 1995.

Floris Takens. Dynamical systems and turbulence. *Warwick, 1980*, pp. 366–381, 1981.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Norbert Wiener. Generalized harmonic analysis. *Acta mathematica*, 55(1):117–258, 1930.

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M Rehg, Byron Boots, and Evangelos A Theodorou. Information theoretic mpc for model-based reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 1714–1721. IEEE, 2017.

Alan Wolf, Jack B Swift, Harry L Swinney, and John A Vastano. Determining lyapunov exponents from a time series. *Physica D: nonlinear phenomena*, 16(3):285–317, 1985.

Sungduk Yu, Walter Hannah, Liran Peng, Jerry Lin, Mohamed Aziz Bhouri, Ritwik Gupta, Björn Lütjens, Justus Christopher Will, Gunnar Behrens, Julius Busecke, Nora Loose, Charles I Stern, Tom Beucler, Bryce Harrop, Benjamin R Hillman, Andrea Jenney, Savannah Ferretti, Nana Liu, Anima Anandkumar, Noah D Brenowitz, Veronika Eyring, Nicholas Geneva, Pierre Gentine, Stephan Mandt, Jaideep Pathak, Akshay Subramaniam, Carl Vondrick, Rose Yu, Laure Zanna, Tian Zheng, Ryan Abernathey, Fiaz Ahmed, David C Bader, Pierre Baldi, Elizabeth Barnes, Christopher Bretherton, Peter Caldwell, Wayne Chuang, Yilun Han, YU HUANG, Fernando Iglesias-Suarez, Sanket Jantre, Karthik Kashinath, Marat Khairoutdinov, Thorsten Kurth, Nicholas Lutsko, Po-Lun Ma, Griffin Mooers, J. David Neelin, David Randall, Sara Shamekh, Mark A Taylor, Nathan Urban, Janni Yuval, Guang Zhang, and Michael Pritchard. Climsim: A large multi-scale dataset for hybrid physics-ML climate emulation. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL `https://openreview.net/forum?id=W5If9P1xqO`.

Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. *ICLR*, 2020.

Bin Zhou, Li Gao, and Yu-Hong Dai. Gradient methods with adaptive step-sizes. *Computational optimization and applications*, 35:69–86, 2006.

A Proofs for propositions and theorems

In the following subsections, we provide full proofs or sketches for all of the non-trivial mathematical claims made in the main text. Note that these subsections have been arranged so that the relevant theorems or propositions follow the same order as in the main text.

A.1 Existence of neural network for epsilon-regions

Lemma A.1. *For any $\epsilon > 0$ and any real $p \in [0, 1]$, there exits a number of observations M_ϵ and a sufficiently large neural network that has an ϵ-bounded region with a probability p*

Proof. By ergodicity, for a sufficiently large M, the distribution of samples will converge to the stationary distribution. At some point there will be enough points in the (bounded) stationary distribution to guarantee that all points have a neighbour at a distance of ϵ with a probability larger than p. With a sufficiently large neural network, we can set the parameters such that every point x in the neighborhood of $x(t)$ gets sent to $x(t+1)$. By construction, this will always map $f(x + \epsilon\vec{r}, \theta)$ onto either $f(x)$ or a close neighbour which will, by the compressive nature of the projection, fulfill the conditions of our definition. □

Lemma A.2. *In a multi-layer perceptron with at least one hidden variable, the training loss that can be achieved decreases with the size of the hidden layers.*

Proof. Consider a multi-layer perceptron with at least one hidden layer and N_l neurons per hidden layer and trajectory of length M in a system with D variables where the MLP achieves a training error per sample of $e(m) = \|x(m) - f(x(m-1, \theta)\|$. Then add $D + 1$ neurons to the first hidden layer and one neuron to any subsequent layers. We can create a bounding box with the extra $D + 1$ neurons for the sample $m_{\max}$ corresponding to the maximum $e(m_{\max})$, and have one neuron in any subsequent layer whose only task is to transfer that signal to the output layer, correcting the error. □

Lemma A.3. *For a stochastic dynamical system of the form $\dot{x} = \phi(x) + \xi$, where*

$$\|f(x + \epsilon\vec{r}, \theta) - (f(x, \theta) + J_\phi(x)\vec{r}\epsilon)\| < \epsilon^2 \tag{10}$$

and ξ is a random random vector such that $Pr\left[\|\xi\| > b\right] < \alpha$,

$$Pr\left[\|f(x + \epsilon\vec{r}, \theta) - (f(x, \theta) + J_\phi(x)\vec{r}\epsilon)\| < \epsilon^2 + b\right] < \alpha \quad \forall x \in Conv(\mathbf{x}),$$

Proof. We are simply adding the noise term to the definition of ϵ-bounded region, and note that since the noise is unrelated to the dynamics, the errors add up. □

Remark A.4. If b is small with α approaching one, then the system is effectively deterministic in the short term horizon.

Lemma A.5. *Consider a model that approximates a dynamical system that is not stable and has a bounded state space in its stationary distribution. For an initial position $x(0)$, the forecasting error as $T \to \infty$ behaves as a random variable with expected value*

$$E_{x(0)}\left[\|f^\tau(T) - x\left(T|x(0)\right)\|^2\right] = \begin{cases} 0 & \textit{If the model perfectly predicts the system} \\ \mathcal{O}\left(Var_{\mathcal{X}}\left[x(t)\right]\right) & \textit{If the model is not strictly perfect} \end{cases} \tag{11}$$

where $x\left(T|x(0)\right)$ is the state of the system at time T for a starting position $x(0)$ and $Var_{\mathcal{X}}\left[x(t)\right]$ is the variance of the dynamics.

Proof. The gist of our argument is that as the model and the system are ergodic, their difference is also ergodic (if they are different) or non-existent (if the model is perfect). Thus, for infinite time horizons the error in the forecasting follows a 0-1 law. If the model perfectly captures the dynamics, the error is clearly 0. If the model is not perfect at any region in the stationary distribution of the dynamical system, there will

be a perturbation of the trajectory. Such perturbations will not fade away since our system is not stable. Because the system is bounded but the trajectory is infinitely long, any region will be visited infinite times. Thus the error in the trajectory will accumulate. Eventually, any correlation between the model and the dynamical system will disappear. By ergodicity, the model and the system will generate points that are effectively independent with a variance of order $\mathrm{Var}_{\mathcal{X}}[x(t)]$. The expected value of the difference between two random variables is the sum of their variances, and since the model is similar to the system, the variances are within the same order of magnitude. □

A.2 The model captures the dynamics

Lemma A.6 (Approximate Jacobian)**.** *For $\epsilon \to 0$, in an ϵ-bounded region of the parameter space the Jacobian of the model converges to the Jacobian of the true dynamics*

$$\lim_{\epsilon \to 0} \|J_\phi(x) - J_f(x)\|_{OP} = 0 \tag{12}$$

where $\|\|_{OP}$ is the operator norm.

Proof. By using the equation in the definition of ϵ-bounded region,

$$\left\| f(x + \sqrt{\epsilon}\vec{r}, \theta) - \left(f(x,\theta) + J_\phi(x)\vec{r}\sqrt{\epsilon} \right) \right\| = \left\| \left(f(x + \sqrt{\epsilon}\vec{r}, \theta) - f(x,\theta) \right) - J_\phi(x)\vec{r}\sqrt{\epsilon} \right\| < \epsilon$$

and taking the a Taylor expansion,

$$\left\| \left(f(x + \sqrt{\epsilon}\vec{r}, \theta) - f(x,\theta) \right) - J_\phi(x)\vec{r}\sqrt{\epsilon} \right\| \approx \|J_f(x)\vec{r}\sqrt{\epsilon} - J_\phi(x)\vec{r}\sqrt{\epsilon}\| < \epsilon$$

which converges to an equality in the limit $\epsilon \to 0$, yielding

$$\begin{aligned} &\|J_f(x)\vec{r}\sqrt{\epsilon} - J_\phi(x)\vec{r}\sqrt{\epsilon}\| = \sqrt{\epsilon}\|J_f(x)\vec{r} - J_\phi(x)\vec{r}\| < \epsilon \\ &\|J_f(x)\vec{r} - J_\phi(x)\vec{r}\| < \sqrt{\epsilon} \end{aligned}$$

□

Lemma A.7 (Temporal horizon limits on systems with noise)**.** *In a dynamical system whose dynamics are given by limit cycle or chaos, and where there is an extra term inducing random noise with a variance uniform in all dimensions and with value σ^2 (per dimension), there is a maximum temporal horizon $T_{\max}$ after which forecasting is effectively a random guess given by*

$$T_{\max} = \begin{cases} \mathcal{O}\left(\dfrac{\ln(S) - \ln(\sigma)}{\lambda} \right), & \textit{for chaotic or unstable systems} \\ \mathcal{O}\left(\dfrac{L}{\sigma} \right), & \textit{for limit cycles} \end{cases} \tag{13}$$

where L is the length of the limit cycle and S the maximum radius of the state space in the chaotic system.

Proof. The gist is that in any dynamical system where the noise accumulates, at some point the contribution of the noise to the current state of the system is going to be as large as the contribution of the deterministic dynamics. For a chaotic system a perturbation of size σ will scale exponentially as $\sigma e^{\lambda T}$, and when $\sigma e^{\lambda T} > S$, the effect of that perturbation is as large as the state space. For a limit cycle we can apply the same logic, except that the noise grows as $T\sigma$ and we use the length of the limit cycle. □

A.3 Bounded loss

Lemma A.8 (Bounded loss)**.** *In an ϵ-bounded region of the parameter space, the loss is bounded by*

$$\mathcal{L}(\theta, T) \leq \left(\max_{x_1, x_2 \in \mathcal{X}_0} \|x_1 - x_2\|_2^2 + 2\epsilon \right) = \mathcal{L}_\epsilon^{\max} \quad \forall T \tag{14}$$

Proof. For any dynamical system with a bounded state space, there is a maximum distance between its two farthest points. Since the model itself is also at a maximum distance from any point in the original system,

$$\mathcal{L}(\theta, T) \leq \max_{x_1, x_2 \in \mathcal{X}_\epsilon} \|x_1 - x_2\|_2^2 \leq \left(\max_{x_1, x_2 \in \mathcal{X}_0} \|x_1 - x_2\|_2^2 + 2\epsilon \right) = \mathcal{L}_{\max,\epsilon} \quad \forall T \tag{15}$$

Note that for $\epsilon \to 0$, the upper bound is the variance of the ergodic distribution of the system. □

A.4 Relating gradients and temporal horizons

Theorem A.9 (Unbounded loss gradient)**.** *Consider a dynamical system which is either chaotic, contains locally unstable trajectories or contains limit cycles, and a corresponding model with an ϵ-bounded region of the parameter space Θ_ϵ with non-zero loss. When the forecasting horizon T is large, the expected gradient magnitude grows with T as*

$$\frac{\|\nabla_\theta \mathcal{L}(\theta, T)\|}{\|\nabla_\theta \mathcal{L}(\theta, 1)\|} = \begin{cases} \mathcal{O}(e^{\lambda T}), & \textit{for chaotic systems} \\ \mathcal{O}(\omega T), & \textit{for limit cycles} \end{cases} \tag{16}$$

where $\|\cdot\|$ is the Euclidean norm.

Proof. We start by analyzing the loss given in eq. (3), which is a function of the parameters θ given the true trajectory at any time t by $x(t)$. To understand how learning operates under this loss function, we study how the landscape induced by $\mathcal{L}(\theta, T)$ varies under different choices of T. We can get the gradient of eq. (3), our loss function, by evaluating

$$\nabla_\theta \mathcal{L}(\theta, T) = \frac{\alpha_T}{M - T} \sum_{k=1}^{M-T} \sum_{\tau=1}^{T} \left. \frac{\partial f^\tau(x, \theta)}{\partial \theta} \right|_{x(k), \theta} \left(x(k + \tau) - f^\tau\left(x(k), \theta\right) \right), \tag{17}$$

where we must still compute the partial derivatives of the function f^τ. For $\tau = 1$, we denote the derivatives of $f^1 = f(x, \theta)$ as

$$J_\theta(x, \theta) = \nabla_\theta f(x, \theta), \quad J_x(x, \theta) = \nabla_x f(x, \theta) \tag{18}$$

Using the chain rule, we obtain the expression

$$J_\theta^\tau(x, \theta) = \nabla_\theta f^\tau(x, \theta) = \sum_{l=1}^{\tau} J_x^l(f^{\tau - l}(x), \theta) J_\theta\left(f^l(x), \theta\right), \tag{19}$$

where $J_x^k(x, \theta) = \prod_{k=1}^{\tau} J_x(f^k(x), \theta)$. To analyze eq. (17) and thus eq. (3), we can use some knowledge we have about the properties of the state space and parameter space Jacobians involved in eq. (19). Specifically, the state space Jacobian, $J_x(x, \theta)$, is directly dependent on the trajectories of the system and indirectly dependent on the parameters. The notable implication of this is that any set of parameters where the model captures the dynamics reasonably well, the Jacobian in the state space of the model will be very closely aligned with the Jacobian of the system dynamics, which is given by the data (see Lemma A.6). In contrast, the Jacobian of the parameters $J_\theta(x, \theta)$ depends on the model and its parametrization.

We can therefore compute the ratio of gradients, $\dfrac{\|\nabla_\theta \mathcal{L}(\theta, T)\|}{\|\nabla_\theta \mathcal{L}(\theta, 1)\|}$ for a given θ as

$$\frac{\frac{1}{M-T} \sum_{k=1}^{M-T} \frac{1}{T} \sum_{\tau=1}^{T} \sum_{l=1}^{\tau} J_x^l(f^{\tau-l}(x), \theta) J_\theta\left(f^l(x), \theta\right) \left(x(k + \tau) - f^\tau\left(x(k), \theta\right)\right)}{\frac{1}{M-1} \sum_{k=1}^{M-T} J_\theta(x(k), \theta) \left(x(k + 1) - f\left(x(k), \theta\right)\right)} \tag{20}$$

and here we can assume that $\|x(k + \tau) - f^\tau\left(x(k), \theta\right)\| \geq \|x(k + 1) - f\left(x(k), \theta\right)\|$ for the simple reason that a prediction that passes through τ iterations of the model is likelly to be worse than a prediction that passes only once. Furthermore, the value of J_θ is applied to the same values and with $M \gg T$, we get the estimate of the ratio as

$$\frac{\|\nabla_\theta \mathcal{L}(\theta, T)\|}{\|\nabla_\theta \mathcal{L}(\theta, 1)\|} = \mathcal{O}\left(\frac{1}{T}\sum_{\tau=1}^{T} \|\mathrm{E}_{\mathbf{y}}\left[J_x^T\right]\|\right) \tag{21}$$

Where $\mathrm{E}_{\mathbf{y}}\left[J_x^T\right]$ is the expectation of the norm of the Jacobian.

Now the main question is to estimate the values of J_x. If a system is chaotic, at least one Lyapunov exponent must be positive (Wolf et al., 1985), meaning that

$$\lim_{\tau\to\infty} \frac{1}{\tau} \ln\left(\prod_{l=1}^{\tau} J_x(f^l(x), \theta)\right) = \lambda > 0. \tag{22}$$

Hence the term $J_x^\tau(x, \theta)$ in Eq. 19, and thus $\mathrm{E}_{\mathbf{y}}\left[J_x^T\right]$, grows exponentially with $\tau \to \infty$. For locally unstable trajectories, we can simply note that the largest eigenvalue of the Jacobian follows

$$\mathrm{Re}\left[\ln\left(\lambda\left(J_x(x, \theta)\right)\right)\right] > 0 \tag{23}$$

and therefore term $J_x^\tau(x, \theta)$ will also grow exponentially.

We also analyze limit cycles by focusing on the phase of the rotation. A trajectory produced by $f^\tau(x, \theta)$ where the period is not exactly the same as in the original dynamical system will slowly drift as the phases will change move at slightly different angular speeds. In more formal terms, if we parameterize the limit cycle by the phase φ and ignore its support we obtain

$$J_\theta^\tau(\varphi, \theta) = \sum_{l=1}^{\tau}\left[\prod_{m=l+1}^{\tau} J_\varphi(f^m(x), \theta)\right] J_\theta\left((f^l(\varphi), \theta\right) = \sum_{l=1}^{\tau} J_\theta\left(f^l(\varphi), \theta\right) \tag{24}$$

where the position is repeated for large τ (this is a limit cycle), with a difference given by the quality of the approximation. To make this simple, we can consider the growth per cycle, and count the cycles,

$$J_\theta^\tau(\varphi, \theta) \approx c\int_0^{2\pi} J_\theta\left(\varphi, \theta\right) d\varphi = \tau\omega \int_0^1 J_\theta\left(a, \theta\right) da \tag{25}$$

where the integral gives the average Jacobian through the cylce, and $c = \frac{\tau}{p} = \frac{1}{2\pi}\tau\omega$ is the number of cycles (notice the change of variable in the integral.

As such with chaotic behavior and locally unstable trajectories, the norm of the Jacobian grows exponentially with τ. For limit cycles, we know $J_\theta^\tau(\varphi, \theta)$ grows at least linearly with $\omega\tau$.

□

Corollary A.10. *For the models considered in A.9, the hessian of the model around its minima grows as*

$$\frac{\|H(\theta, T)\|^*}{\|H(\theta, 1)\|^*} = \begin{cases} \mathcal{O}(e^{\lambda T}), & \textit{for chaotic systems} \\ \mathcal{O}(\omega T), & \textit{for limit cycles} \end{cases} \tag{26}$$

where $\|\cdot\|^*$ is the Nuclear norm.

Proof. The gist of the proof is to show that as the gradient grows with T, the second derivative must also grow to keep up. We build this connection between the gradient and the Hessian through the divergence theorem.

Consider the gradient of the loss $\nabla\mathcal{L}(\theta, T)$ as a vector field in the space of parameters. Pick a minima of the loss $\theta_T^{\min}$, and a ball around it denoted by $B_{\theta_T^{\min}}$ and its spherical boundary $S_{\theta_T^{\min}}$. Then, by the divergence theorem,

$$\int_{B_{\theta_T^{\min}}} \mathbf{div}\nabla\mathcal{L}(\theta, T) dV = \int_{S_{\theta_T^{\min}}} \nabla\mathcal{L}(\theta, T) d\vec{S} \tag{27}$$

where $d\vec{S}$ is a vector normal to the sphere scaled by a differential surface element, and dV a differential element of volume. Since $\theta_T^{\min}$ is a minima, for a small enough ball the gradient of the loss is always pointing towards the inside, hence the product with $d\vec{S}$ will have a constant sign. Thus,

$$\int_{B_{\theta_T^{\min}}} \mathbf{div} \nabla \mathcal{L}(\theta, T) dV = \int_{S_{\theta_T^{\min}}} \|\nabla \mathcal{L}(\theta, T) d\vec{S}\|. \tag{28}$$

Now, note that the divergence of the gradient is the trace of the Hessian,

$$\int_{B_{\theta_T^{\min}}} \mathbf{Tr}\left[H(\theta, T)\right] dV = \int_{S_{\theta_T^{\min}}} \|\nabla \mathcal{L}(\theta, T) d\vec{S}\|. \tag{29}$$

Now we simply need to remember that the ball was relatively small and the loss smooth, so that we can use the approximation

$$\mathbf{Tr}\left[H(\theta, T)\right] \approx \mathbf{Tr}\left[H(\theta_T^{\min}, T)\right] \quad \forall \theta \in B_{\theta_T^{\min}}, \tag{30}$$

and noticing that the trace is sum of eigenvalues of a matrix, which is the nuclear norm,

$$\mathbf{Tr}\left[H(\theta_T^{\min}, T)\right] \int_{B_{\theta_T^{\min}}} dV = \|H(\theta_T^{\min}, T)\|^* \int_{B_{\theta_T^{\min}}} dV = \int_{S_{\theta_T^{\min}}} \|\nabla \mathcal{L}(\theta, T) d\vec{S}\|. \tag{31}$$

and if we apply this for $T = 1$ and divide,

$$\frac{\|H(\theta_T^{\min}, T)\|^*}{\|H(\theta_1^{\min}, 1)\|^*} = \frac{\int_{S_{\theta_T^{\min}}} \|\nabla \mathcal{L}(\theta, T) d\vec{S}\|}{\int_{S_{\theta_1^{\min}}} \|\nabla \mathcal{L}(\theta, 1) d\vec{S}\|} = \begin{cases} \mathcal{O}(e^{\lambda T}), & \text{for chaotic systems} \\ \mathcal{O}(\omega T), & \text{for limit cycles} \end{cases}, \tag{32}$$

where the last step comes from theorem 4.2,

□

A.5 Minima with longer forecasting horizons generalize better

Theorem A.11 (Minima with longer forecasting horizons generalize better)**.** *Consider a model $f(\cdot, \theta)$ with θ in an ϵ-bounded region. Now consider two parameter sets $\theta_l^{\min}$ and $\theta_h^{\min}$ that minimize the losses $\mathcal{L}(\theta, T_l)$ and $\mathcal{L}(\theta, T_h)$ respectively with $T_h > T_l$, and which are both within each other's basin of attraction, meaning that $\theta_l^{\min}$ would converge $\theta_h^{\min}$ by following the gradient of $\mathcal{L}(\theta, T_h)$ and vice versa. Then the difference in the change in losses*

$$\frac{\mathcal{L}(\theta_h^{\min}, T_h) - \mathcal{L}(\theta_l^{\min}, T_h)}{\mathcal{L}(\theta_l^{\min}, T_l) - \mathcal{L}(\theta_h^{\min}, T_l)} = \begin{cases} \mathcal{O}(e^{\lambda(T_h - T_l)}), & \textit{for chaotic or unstable systems} \\ \mathcal{O}(T_h - T_l), & \textit{for limit cycles} \end{cases} \tag{33}$$

Proof. Consider the path p of the gradient between the minima $\theta_T^{\min}$ for the temporal horizon T and another point θ^{basin} on the basin of attraction of $\theta_T^{\min}$. On this path we can compute the change in loss,

$$\mathcal{L}(\theta_T^{\min}, T) - \mathcal{L}(\theta^{\text{basin}}, T) = \int_{\theta \in p} \langle \nabla \mathcal{L}(\theta, T), \vec{u}(\theta, p) \rangle d\theta = \int_{\theta \in p} \|\nabla \mathcal{L}(\theta, T)\| d\theta \tag{34}$$

$$= \int_{\theta \in p} \|\nabla \mathcal{L}(\theta, 1)\| \frac{\|\nabla \mathcal{L}(\theta, T)\|}{\|\nabla \mathcal{L}(\theta, 1)\|} d\theta \tag{35}$$

where $\vec{u}(\theta, p)$ is the unit vector on the direction of the path of the gradient, and since the path is defined by the gradient itself, $\langle \nabla \mathcal{L}(\theta, T), \vec{u}(\theta, p) \rangle = \|\nabla \mathcal{L}(\theta, T)\|$. By theorem 4.2, the ratio of loss norms in the integral either grows exponentially or linearly with T. By substituting $T = T_h$ and T_l then taking the fraction in eq. (33) we complete the proof. □

A.6 Loss landscape roughness

Lemma A.12. *Consider a continuous differentiable function $g(s)$ on an interval $s \in [s_a, s_b]$ bounded from above and below,*

$$g_{\min} < g(s) < g_{\max} \tag{36}$$

with a variation in value

$$v_g = \int_{z_a}^{z_b} \left| \frac{dg(s)}{dz} \right| ds > 2n(g_{\max} - g_{\min}), \quad n \in \mathbb{N}^+. \tag{37}$$

then $g(s)$ has at least n minima and n maxima in the interval considered.

Proof. Assume without loss of generality that g is initially increasing. That is, $\frac{dg(s_a)}{ds} > 0$. Further define,

$$v_g(s) = \int_{s_a}^{s} \left| \frac{dg(s)}{ds} \right| ds \tag{38}$$

which we know to be a semi-positive monotonically increasing function with range $[0, U]$ where $U > 2n(g_{\max} - g_{\min})$. Consider the first point z_1 such that $v_g(s_1) > g_{\max} - g_{\min}$. Assume for contradiction that g does not have a maxima on the interval $[s_a, s_1]$. We know that,

$$g(s_1) = g(s_a) + \int_{s_a}^{s_1} \frac{dg(s')}{ds'} ds'. \tag{39}$$

By assumption, $\dfrac{dg(s')}{ds'}$ is semi-positive and must not change sign on the interval. As such,

$$g(s_1) = g(s_a) + \int_{s_a}^{s_1} \left| \frac{dg(s')}{ds'} \right| ds' > g(s_a) + g_{\max} - g_{\min} \Rightarrow g(s_1) - g(s_a) > g_{\max} - g_{\min}. \tag{40}$$

This implies that $g(s_1) - g(s_a) > g_{\max} - g_{\min}$, which is a contradiction. Therefore on the interval $[s_a, s_1]$ there must exist a maximum s_1^*. Applying the same logic to the interval $[s_1^*, s_2]$ such that $v_g(s_2) > 2(g_{\max} - g_{\min})$ proves there must be a minimum on that interval. Further applying these two arguments $n-1$ more times shows g must have at least n minima and n maxima on $[s_a, s_b]$. □

Theorem A.13 (Loss landscape roughness)**.** *For any two points θ_1, θ_2 in an ϵ-bounded region of the parameter space that are not in a connected region of zero loss*[2]*, the number of minima and maxima along the line segment that connects them will grow as*

$$z(T) = \begin{cases} \mathcal{O}(e^{\lambda T} \|\theta_1 - \theta_2\|), & \textit{for chaotic or unstable systems} \\ \mathcal{O}(T \|\theta_1 - \theta_2\|), & \textit{for limit cycles} \end{cases} \tag{41}$$

Proof. Let us look at arbitrary elements of the parameter space $\theta_1, \theta_2 \in \Theta$. By lemma A.8, the difference in their loss is bounded,

$$|\mathcal{L}(\theta_1, T) - \mathcal{L}(\theta_2, T)| \leq \mathcal{L}_\epsilon^{\max}. \tag{42}$$

We now consider the variation in the loss function that goes from θ_1 to θ_2 through a straight line,

$$v_T(\theta_1, \theta_2) = \int_{l=\theta_1}^{\theta_2} \left| \frac{\partial \mathcal{L}(l, T)}{\partial l} \right| dl, \tag{43}$$

where l is the variable that represents points in a line from θ_1 to θ_2. We note that the derivative of the loss is a projection of the gradient onto the line,

$$\frac{\partial \mathcal{L}(l, T)}{\partial l} = \langle \nabla_\theta \mathcal{L}(l, T), \vec{u}_{\theta_1 \to \theta_2} \rangle \tag{44}$$

[2]This case is not covered in the assumptions on theorem 4.2, but if we are in such a region there is no learning

where $\vec{u}_{\theta_1 \to \theta_2}$ is the unitary vector of the line going from θ_1 to θ_2. By theorem 4.2, the value of this projection grows exponentially in a chaotic or locally unstable trajectory. In a limit cycle, the growth is or linearly with T. Thus, we only need linear growth in eq. (44) for an unbounded value of $v_T(\theta_1, \theta_2)$. Taking the integral over the line,

$$\mathcal{O}\left[\int_{l=\theta_1}^{\theta_2} \left|\frac{\partial \mathcal{L}(l,T)}{\partial l}\right| dl\right] = \begin{cases} e^{\lambda T}\|\theta_1 - \theta_2\|, & \text{for chaotic or unstable systems} \\ T\|\theta_1 - \theta_2\|, & \text{for limit cycles} \end{cases} \tag{45}$$

and by lemma A.12, we then have that $z(T)$ must grow with lower bound given by eq. (45), thus completing the proof.

□

Corollary A.14 (The loss function is a fractal)**.** *For an ϵ-bounded region of the parameter space, in the limit $T \to \infty$, the surface of the loss is a fractal occupying a volume that converges to*

$$|\Theta_\epsilon|\sqrt{\frac{Var[x \in \mathcal{X}]}{M}} < V < |\Theta_\epsilon|\sqrt{\frac{Var[x \in \mathcal{X}] + \epsilon^2}{M}} \tag{46}$$

where $|\Theta_\epsilon|$ is the area of the ϵ-bounded region and $Var[x \in \mathcal{X}]$ is the variance of the invariant probability distribution in the dynamical system.

Proof. We will use the notion that if a curve has a box counting dimension that is higher than its topological dimension, it is a fractal Mandelbrot (1983); Edgar & Edgar (2008). The topological dimension is N, the number of parameters. For the box counting dimension, we consider a cover of the parameter space by infinitesimally small open balls. For each one of those balls, however, the loss function has unbounded gradients and thus it is a space-filling curve. Hence, we would need $N + 1$ dimensions to cover the loss landscape surface. From the dynamics perspective, any parameter setting where $\mathcal{L}(\theta, 1) > 0$ implies that there is at least some cases where the model deviates from the system. For some high value of t^*, the deviations accumulate until the position of the model is uncorrelated with the position of the dynamical system observed as $x(t^*)$. If $T \gg t^*$, the observations are uncorrelated with the model, and thus we are sampling a variable with variance $\text{Var}[x \in \mathcal{X}]$. We have M samples and a margin of error of ϵ, which gives us the term in the square root. The multiplication by Θ_ϵ is simply because we need to consider all the parameter sets. Note that even if some parameters result in a zero-loss, they form surfaces of lower dimensionality and thus have zero volume. □

A.7 Effects of noise

Our theory builds on the assumption that the ϵ in theorem 4.2 is small enough so that models with $\theta \in \Theta_\epsilon$ will approximate the system dynamics well (lemma A.6). In stochastic systems the ϵ is bounded away from zero, meaning that in large temporal horizons any model is effectively a random guess (A.5).

B Dynamical systems equations

Here we describe the differential equations governing the four dynamical systems used in the main text.

B.1 Lorenz attractor

The Lorenz attractor is a three dimensional dynamical system described by the following equations

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(y - x), \quad \frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho - z) - y, \quad \frac{\mathrm{d}z}{\mathrm{d}t} = xy - \beta z \tag{47}$$

where we take $\sigma = 10.0$, $\rho = 28.0$ and $\beta = \frac{8}{3}$. The system was originally described by Edward Lorenz and is a simplification of hydrodynamic flows. We note that its trajectories are bounded and non-periodic (Lorenz, 1963).

B.2 Double pendulum

The double pendulum is a simple mechanical system that exhibits chaotic behavior Gupta et al. (2014). It consists of two masses rigidly linked to each other, both can rotate freely around their anchor point. The first mass is connected to a center point and the second mass is linked to the first mass.

The dynamics are determined by applying the standard equations of motion from Newtonian mechanics, resulting in eq. (48)ff.

$$x_1 = \frac{l}{2}\sin\theta_1 \tag{48}$$

$$y_1 = -\frac{l}{2}\cos\theta_1 \tag{49}$$

$$x_2 = x_1 + l_2 \sin\theta_2 \tag{50}$$

$$y_2 = y_1 - l_2 \cos\theta_2 \tag{51}$$

B.3 Pood web model

We consider a generalized 7-species food-web model inspired by Hastings (1991); McCann & Yodzis (1994); Klebanoff & Hastings (1994); Post et al. (2000); Åkesson et al. (2021), where species interact through trophic and competitive relationships.

Denoting the abundance of species i by N_i, and defining $N = (N_1, \ldots, N_7)$, the generalized community model can be expressed as

$$\frac{1}{N_i}\frac{d}{dt}N_i = r_i(u(t)) - \sum_{j=1}^{7} N_j \left[\alpha_{i,j} + [F(N)]_{i,j} - [F(N)]_{j,i}\right]. \tag{52}$$

In eq. (52), the first two terms capture intrinsic growth rate and intra- and interspecific competition. The last two terms capture growth and loss due to trophic interactions.

The per capita growth rate r_i may depend on a time-dependent environmental forcing $u(t)$. The competition coefficient between species i and j is denoted by $\alpha_{i,j}$. The feeding rate $[F(N)]_{i,j}$ of species i on j follows a functional response of type II

$$[F(N)]_{i,j} = \frac{q_{i,j}W_{i,j}}{1 + q_{i,j}H_{i,j}\sum_{k=1}^{7} W_{i,k}N_k}, \tag{53}$$

where $q_{i,j}$ and $H_{i,j}$ represent the attack rate and handling time of species i when feeding on species j, respectively. We used the biologically realistic parameter values with $r(u(t))$ being constant. If one lets $\omega = 0.2$ our system is defined by

$$r = \begin{pmatrix} 1 \\ -0.15 \\ -0.08 \\ 1.0 \\ -0.15 \\ -0.01 \\ -0.005 \end{pmatrix}, \alpha = \begin{pmatrix} 1.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, W = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \omega & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-\omega & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix},$$

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2.89855 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7.35294 & 0 & 0 & 7.35294 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.89855 & 0 & 0 & 0 \\ 0 & 0 & 8.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 12.0 & 0 \end{pmatrix},$$

$$q = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.38 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.272 & 0 & 0 & 0.272 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.38 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.05 & 0 \end{pmatrix}.$$

The dynamics of the system are chaotic for this set of parameter values, but in short timescales they resemble a limit cycle. All non-zero coefficients were set as free parameters, excluding the coefficients of the adjacency matrix W, where only ω was considered as a free parameter.

B.4 Limit cycle

The basic limit cycle system is defined below,

$$\frac{1}{a}\frac{\mathrm{d}x}{\mathrm{d}t} = \mu x - y - x(x^2 + y^2), \quad \frac{1}{a}\frac{\mathrm{d}y}{\mathrm{d}t} = x + \mu y - x(x^2 + y^2) \tag{54}$$

with parameters $a = 1.0, \mu = 0.4$.

B.5 Implementation Details

The exact way in which the dynamical systems were solved depended on the figure. For fig. 6 they were solved in Julia using `ChaoticInference`Marti (2024) whereas for the other plots they were solved in Python using `diffrax` (Kidger, 2021).

B.5.1 Fitting dynamical systems

The four systems, Lorenz, double pendulum, ecological model and the limit cycle are implemented as dynamical systems, where an ODE gives rise to a trajectory.

As the dynamical systems are solved by numerical integration over some time and not autoregressive, an adjusted loss function (Equation 55) is used.

$$\mathcal{L}_{\mathbf{x}}(\theta, T) = \sum_{m=0}^{\frac{M-T}{T}} \sum_{\tau=1}^{T} \left\| x(mT + \tau) - F^{\tau}\left(x(mT), \theta\right) \right\|. \tag{55}$$

Where $F^\tau(x_0, \dots)$ denotes the numerical solution for $x(\tau)$.

Heavy lifting is done by the `ChaoticInference` library (Marti, 2024), namely data generation, loss evaluation and plotting. Data generation was done with the fully implicit, fifth order `RadauIIA5` solver(Rackauckas & Nie, 2017). For more details regarding initial conditions, timespan, etc. see the respective implementations.

B.5.2 Fitting neural networks

The four systems were solved using the `diffrax` package developed by Kidger (2021) with 10 initial conditions sampled from a Gaussian distribution. For the dataset associated with the food web, we sample the system every 2 seconds, for those associated with the double pendulum they were sampled every 0.005 seconds, for the Lorenz they were either sampled every 0.005 or 0.04 depending on the experiment and for the limit cycle they were sampled every 0.5.

B.6 Timestep choice

We selected the sampling timestep in such a way that we could observe the growths of the gradients within a reasonable range of T. Having a higher sampling rate would require computing a lot of values of T and overburden our plots, while a too low T would prevent the model from learning.

C Geo-spatial datasets

In the following section, we expand on the geo-spatial datasets used in the main text and how we processed them for training with our limited computational budget.

C.1 NOAA SST dataset

Huang et al. (2021) provide daily sea surface temperature data from September 1981 to the present day on a $1/4^\circ$ global grid. We sub-sample this grid, converting it to a 4° resolution and also sub-sample the temporal horizon to get states every 10 days. We split the training and validation datasets using given years. 2000 to 2009 was used for training and 2011 to 2017 was used for validation. Notably, we deal with missing data over land by setting values to 0. The data was also normalized for training and validation.

C.2 ClimSim dataset

The ClimSim dataset was originally developed by Yu et al. (2023) to provide training data for models predicting complex climate variables from more simplistic variables. However, we noticed it could also serve as an excellent candidate for exploring the learnability trade-off as data is reported every 20 minutes of the simulation. As such, we converted the low-resolution version the dataset, labeled `LEAP/ClimSim_low-res` for use in training our autoregressive models. We took a subset of the variables, namely the lowest u and v fields for prediction. The training and validation sets were taken from non-overlaping year-long periods. As with the SST data, we also normalised this dataset.

D Loss plots for Lorenz system

The Lorenz system is parameterized by three parameters $\theta_1, \theta_2, \theta_3$ corresponding to ρ, σ, β in appendix B.1. In this section we expand the visualization from Fig. 6 to the other parameters for a given temporal horizon T. Note that in systems with more parameters such as the food web model, the plots become too complex.

We evaluate this function for $T \in [10, 500, 1000]$. See plots fig. 7, and fig. 8. In these figures we see, that the prediction already verified in a one dimensional loss in figure fig. 6 holds also for higher dimensional losses in dynamical systems.

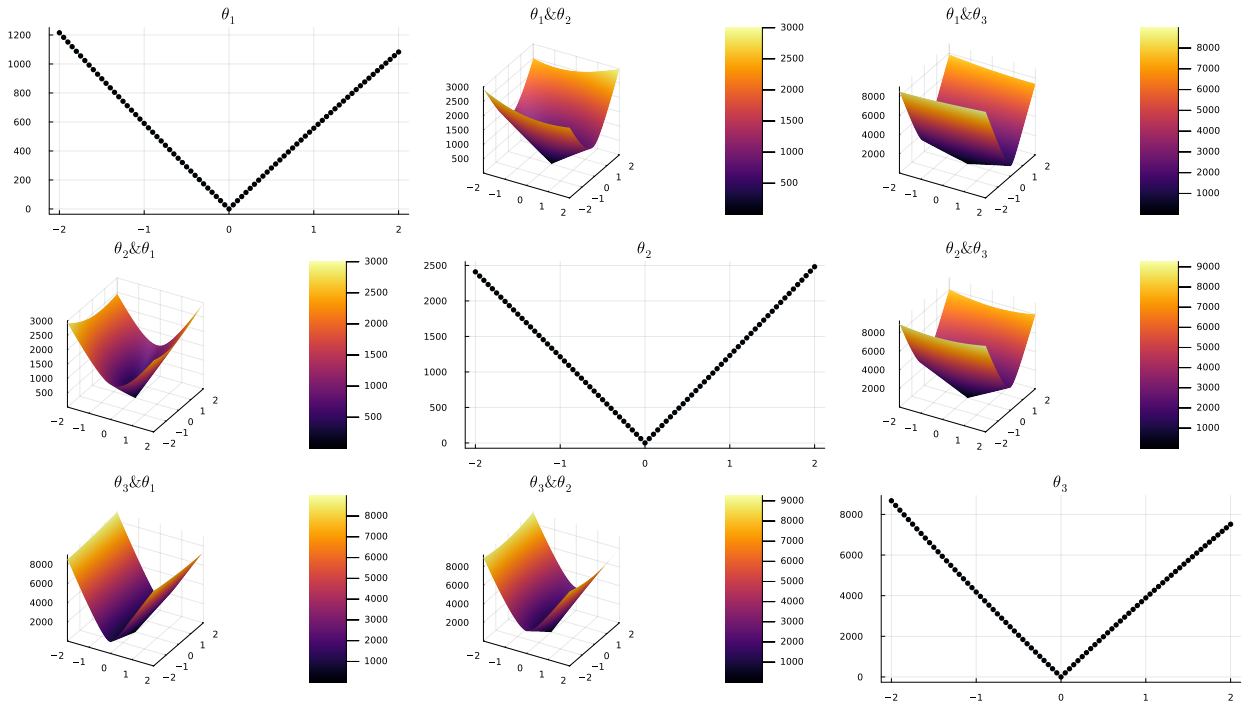

Figure 7: Surface plot of the piecewise loss function of the Lorenz system with $T = 10$

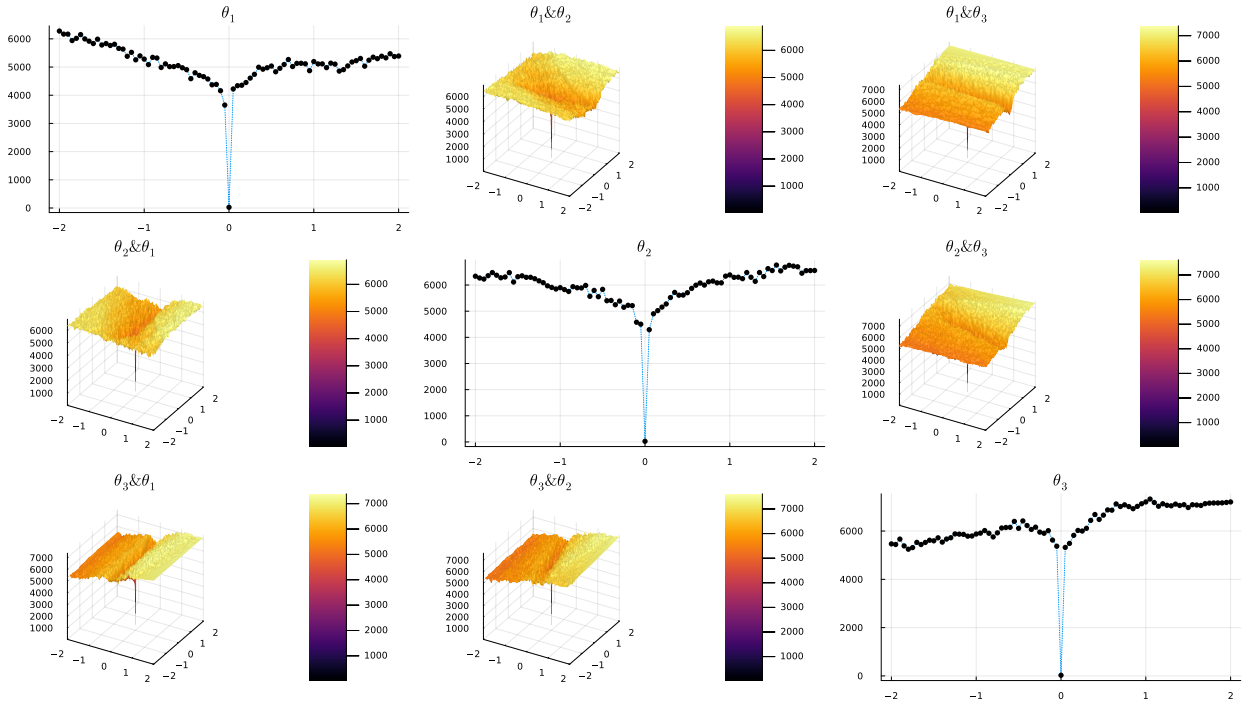

Figure 8: Surface plot of the piecewise loss function of the Lorenz system with $T = 1000$

E Training neural networks

In the following section we provide additional details on the training setup used for the four dynamical systems, as well as the networks used for the two climate-related datasets. We used a both MLPs andresidual MLPs with the same architectures.

E.1 MLPs

We used two types of MLPs, a plain MLP without residual connections and one with residual connections.

Definition E.1 (MLP)**.** We define a multi-layer to consist of an embedding layer (W_{embed}), an unembedding layer (W_{unembed}) and a series of blocks of the form:

$$\text{block}(x) = \text{ReLU}\left(W_{\text{block}}\text{LN}(x) + b_{\text{block}}\right) \tag{56}$$

where LN is a layer normalization. The network makes a given prediction by first applying the embedding layer, a series of blocks and then the unembed layer.

If we let v be the input size of the example, we generally take $W_{\text{embed}} \in \mathbb{R}^{av \times v}$, $W_{\text{block}} \in \mathbb{R}^{av \times av}$ and $W_{\text{unembed}} \in \mathbb{R}^{av \times v}$ with varying values of a and the number of blocks depending on the complexity of the dataset.

E.2 Computer resources

We use less than 34000 core hours, or equivalent time of 1888 hours on a single V100 GPU.

E.3 Hyperparameter choices

As a general rule, we tried to maintain some consistency between the hyperparameter choices for various datasets and architectures (for example we typically used a batch size of 512). However, due to the variability of both the data and the inductive biases, this was not always possible. For a detailed set of these, one would need to refer to the source code.

F Relationships between model complexity, learning rate, temporal horizon and noise on performance

In the autoregressive setting studied in this paper, there are many hyperparameters to set. One needs to decide, among other things, upon the complexity of the model, what learning rate to use and which temporal horizon to chose. To provide guidance on this front, we explored how performance was related to temporal horizon, learning rate and model complexity under different levels of observational noise. Specifically, we repeated the setup of section 5 for the ecology system with increasing label noise for a small, medium and large MLP using a total wall-time cutoff. We believe the following conclusions are evident from fig. 9:

- For (relatively) high forecasting horizons and learning rates the models fail to learn (red areas in all plots). This aligns with our theory showing that longer time horizons make the loss landscape harder to navigate, which can be alleviated with small learning rates.
- For any training horizon, the learning rates have an optimal value which is not at the extrema: high learning rates lead to a full failure to learn (red), but decreasing the learning rate can alleviate this. However, having a too low learning rate also decreases the performance of the model, likely because training has not been completed.

Those observations are both compatible with our theory and also fairly intuitive for machine learning practitioners. Taken together, we can conclude that with increasing time horizon and model complexity, the upper bound on effective learning rates decreases.

It is worth noticing that our theory was focused on deterministic systems with very small ϵ values, and both assumptions are not necessarily preserved in some subplots of Fig. 9. Thus, by comparing the different subplots we can reach the following conclusions:

- For small noise levels, the magnitude of the forecasting error for the optimal time horizon decreases with the model size (note that the error magnitudes decrease across rows). This can be understood as saying that the model can approximate the dynamics better (ϵ reduces).
- For high noise levels, all models achieve similar optimal performances, regardless of size. Our interpretation is that as ϵ is bounded from below by the randomness of the dynamics, the model size does not contribute to make better predictions

Taken together, those observations suggest that our theory is valid for small noise levels, but is not informative for large noise levels. However, at such levels no prediction is possible due to the inherently stochastic nature of the dynamics.

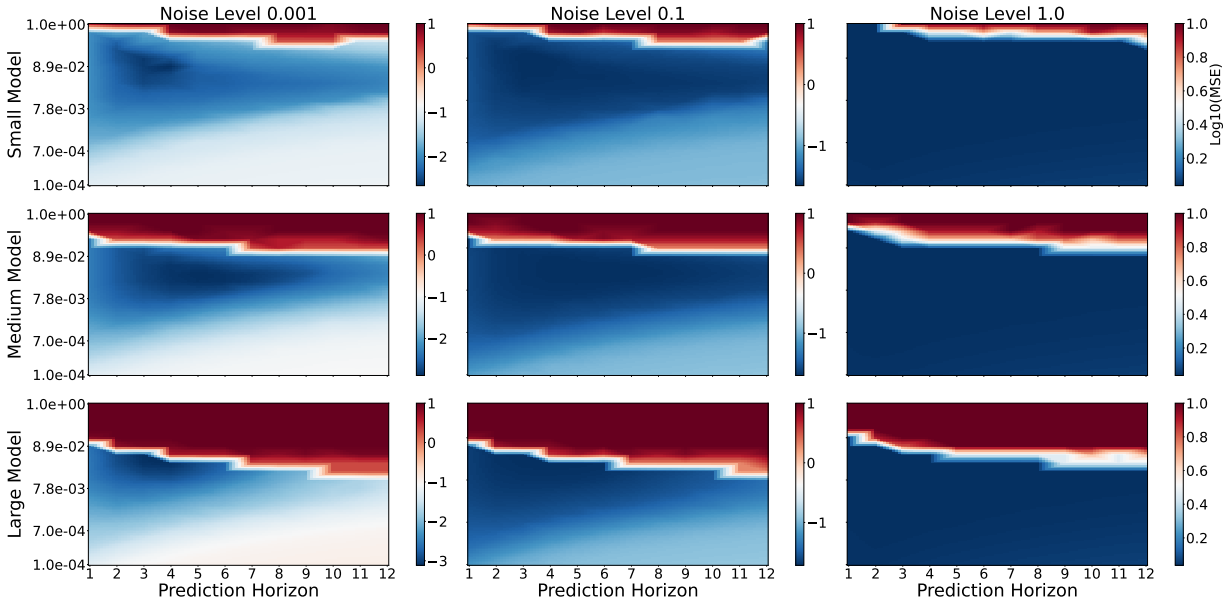

Figure 9: We trained our models at different levels of noise (columns) and with different parameter sizes (rows). For every combination, we tested different forecasting horizons (x-axis) and learning rates (y-axis).

G Increasing the temporal horizon increases performance of a sufficiently converged model

In this section, we provide empirical evidence for a down-stream effect that we should expect from the presented theory. Specifically from theorem 4.4. That is, for a sufficiently converged model, we expect that increasing the temporal horizon will increase performance so long as the increase in temporal horizon not sufficiently large to disrupt the existence of the minima. To do so, we consider MLPs trained on the ecology time series presented in the main text. We assume that we have a fixed training time B that we split equally among different choices of temporal horizon. Given this split, we train the model by starting it on $T = 1$, then switching to $T = 2$ and so on until we reach $T_{\max}$. This process is visualized in fig. 10.

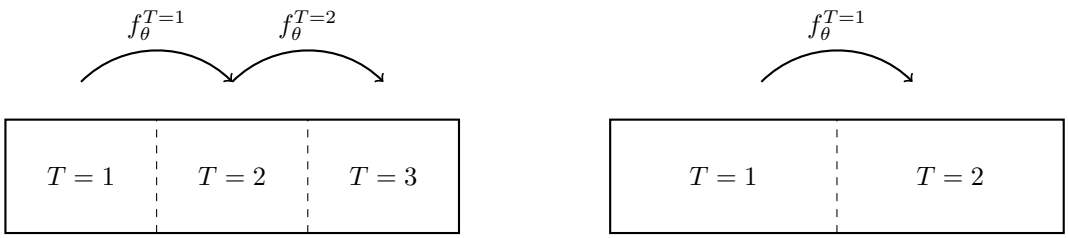

Figure 10: Experimental setup demonstrating the utility of increasing the temporal horizon of a converged model.

If our hypothesis is true, for sufficiently large values of B up to a certain $T_{\max}$ we should get better performance from a greater splitting. fig. 11 reports the results of the experiment. As can be seen there, we do indeed get the expected trend, with greater total time favoring greater splitting with a larger total time horizon.

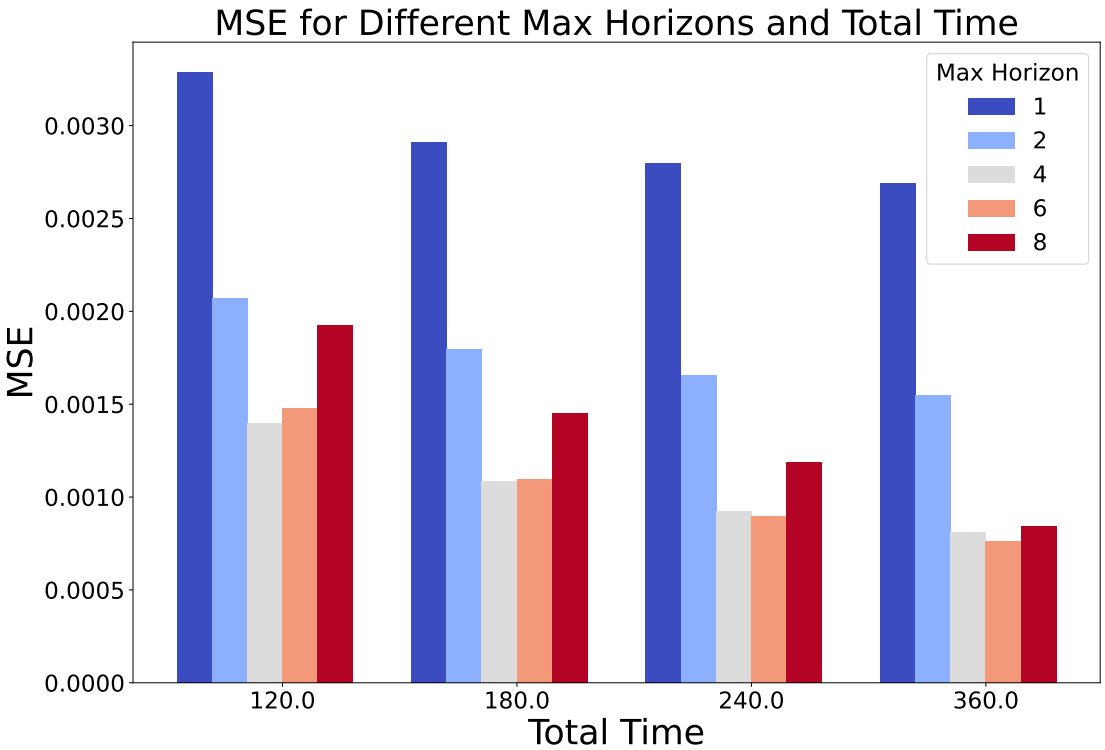

Figure 11: MSE compared to total time for various maximum horizons where training is split equally among time horizons up to the maximum horizon. We used the ecology system and computed the MSE

H Choosing time horizons and learning rates

Although we did not deem the results robust enough for inclusion in the main text, we did develop a scheme consistent with our theory for automatically deciding upon the training temporal horizon and learning rate. We would tentatively recommend this scheme to practitioners first applying eq. (3) to a well-behaved time series problem as given an initial η_0, it can sometimes out-performs the best T but may also be able to uncover a reasonable choice for T in cases where it does not perform as well.

H.1 Motivation and specification of the algorithm

The scheme, detailed in algorithm 1, seeks to automatically choose T and η to overcome a common difficulty in gradient descent: the presence of plateaus. As we saw in theorem 4.5, the loss landscape becomes less flat with increasing T, suggesting that it should be possible to increase T in order to deal with the plateaus. There is, however, some subtlety, because plateaus come in two forms: a minimum plateau and saddle-plateau. A minimum plateau is a region of low loss surrounded by higher losses, and a saddle-plateau is connected to some region of lower loss. In the later case, we'd like to speed up optimization at the expense of skipping small improvements that could be made in the region. In contrast, for a minimum plateau we would like to identify the best points within the plateau, and thus we would like to keep the size of the parameter update equal or smaller to explore a more informative minima. If we want to increase the step size to traverse a saddle-plateau, it suffices to increase T, and the gradient will increase either linearly or exponentially, depending on the underlying dynamics of the system as per Prop. 4.2. Translating this logic into practice lead to development of the algorithm presented in algorithm 1. As can be read there, the algorithm works by

Algorithm 1 Iterative scheduling of T and η

Initialize: $T \leftarrow 1$, $\eta \leftarrow \eta_0$, $\theta_{\text{prev}} \leftarrow \theta_0$, $\gamma \leftarrow 1.5 \cdot 10^{-4}$, look $\leftarrow$ True, succeeded $\leftarrow$ False
Set: $t_{start} \leftarrow$ time(), $t_{curr} \leftarrow$ time(),
while $t_{curr} - t_{start} < T_{max}$ **do**
  if look **then**
    $E \leftarrow 20$
    $\theta \leftarrow \theta_{\text{prev}}$
  else
    $E \leftarrow$ None
    time_limit $\leftarrow t_{max} - (t_{curr} - t_{start})$
  end if
  Train MLP with T for E epochs while recording $\mathcal{L}$, $\|\nabla_\theta \mathcal{L}\|$
  if validation loss improves **then**
    succeeded $\leftarrow$ True
    look $\leftarrow$ False
  else
    Adjust η using exponential fit of trend in past $\|\nabla_\theta \mathcal{L}\|$
    $\theta \leftarrow \theta_{prev}$
  end if
  if early stop due to gradient using $\|\nabla_\theta \mathcal{L}\| < \gamma$ **then**
    $T \leftarrow T + 1$
    look $\leftarrow$ False
    succeeded $\leftarrow$ False
  end if
  $t_{curr} \leftarrow$ time()
end while

waiting until we reach a flat region, looking ahead to see if we can obtain lower loss by increasing T without decreasing η and otherwise decreasing η by fitting an exponential to the existing data (mimicking the scaling implied by theorem 4.2). Note that the algorithm is modified for the limit cycle system by replacing the exponential fit by a linear one.

H.2 Results in comparison to normal training

In fig. 12, we compare algorithm 1 with networks trained using the same wall time with varying training temporal horizons and a constant learning rate on our four dynamical systems, changing the algorithm to fit a linear trend on the limit cycle. Note that some tuning to γ, η_0 and the wall time was needed to find examples where the iterative scheme showed promise. Nonetheless, as presented fig. 12, there are certain wall times, dynamical systems, learning rates and values of γ one can choose which show that the iterative scheme can, in some cases, outperform or match the best choice of training temporal horizon without knowledge of that horizon beforehand. Some general observations from initial testing was that the scheme appeared to perform better in cases where η_0 was relatively low and γ was also reasonably low. However, there was some non-robust sensitivity to γ especially in the case of the limit cycle.

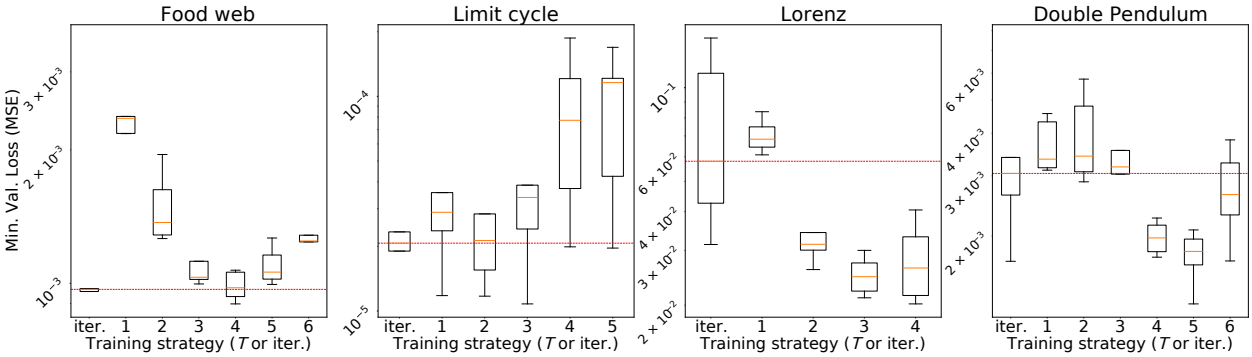

Figure 12: Comparison of iterative scheme to regular training on the four dynamical systems from the main text, as a box plot of the performances taken to be $\mathcal{L}(\theta, T)$ where T is the maximum horizon trained on and the data is from the validation set. For each system and each time limit, we plot the temporal horizon for training (integers) or the iterative increase. The red line is the median of these scores for the iterative scheme. Note that η_0, wall time and γ have been tuned in favor of the iterative scheme (although a full search was not applied).