Master's Thesis

# Topological Properties of Reservoir Computing Architectures as Probes for Hyperparameter Optimization

Spandan Pandya




Ludwig-Maximilians-Universität München
Faculty of Physics
Supervisor: PD. Dr. Christoph Räth

Masterarbeit

# Topologische Eigenschaften von Reservoir-Computing-Architekturen als Indikatoren für die Hyperparameteroptimierung

Spandan Pandya




Ludwig-Maximilians-Universität München
Fakultät für Physik
Betreuer: PD. Dr. Christoph Räth

*To Mumma, Daddy, Jaya, and Puppa*

# Acknowledgements

I am deeply indebted to Dr. Christoph Räth on many scales. While he uses his words sparingly, each word has sparked deep curiosity within me. His scientific temper, precision, and intuition are something I shall continually strive to attain in the future. I have also had the good fortune of being supervised by Davide Prosperino. I thank you deeply for your continuous ideation in the thesis, and bearing with my doubts and code bugs. I thank you also for teaching me a thing or two about scientific writing.

To my friends, many of whom made a completely alien place like Munich feel like home, I am perenially grateful. Thank you– Anna, Alice, Alex, Argyris, Keno, Kahan, Mahnoor, Nickos, Panagiotis, and Pranav. I shall deeply miss all of you.

Anjali, Shwetha, and Yerik, I am glad you had my back while I figured my life during my Master's.

To Ursula and Karl, who opened their homes and arms and welcomed me the warmest, I send lots of love.

To Tarjani, whose presence in my life has been the greatest source of comfort, I am grateful.

**Abstract**

The efficiency and performance of Reservoir Computing (RC) depend critically on various hyperparameters, which to date have been guided largely by heuristics. A deeper understanding is needed of how RCs learn and of the mathematical relationship linking hyperparameters to learning ability and prediction performance. In this thesis, we develop a novel approach that uses topological statistics such as continuity and differentiability on a specialized minimal RC architecture to optimize the minimal nonlinearity of the readout $\eta$ and the offset parameters in two system and three system multifunctional reservoirs. We provide our own numerical implementation of these topological statistics. We perform this study in the context of dynamical systems as synthetic data and use the minimal RC architecture, and the classical RC architecture as the basis for our conclusions. The methods we propose are useful in understanding the structure of the reservoir and the process of embedding the data within the reservoir. We provide a method that correctly predicts regions in parameter space conducive to multifunctionality.

# Contents

**1 Introduction** **1**

**2 Systems Overview** **4**
2.1 Selection of Attractors used in this work . . . . . 5

**3 Reservoir Computing Architectures** **11**
3.1 Regular RC . . . . . 12
3.1.1 Fundamental Concept . . . . . 12
3.1.2 Configuring $\mathbf{W_{in}}$ . . . . . 13
3.1.3 Reservoir . . . . . 13
3.1.4 Readout . . . . . 13
3.1.5 Training and Prediction . . . . . 14
3.1.6 Sample Prediction . . . . . 14
3.2 Minimal RC . . . . . 15
3.2.1 Combinatory Input Weights . . . . . 16
3.2.2 Block-Diagonal and Binary Reservoir . . . . . 16
3.2.3 Nonlinear Readout . . . . . 17
3.2.4 Sample Prediction . . . . . 17
3.2.5 Importance of $\eta$: . . . . . 18

**4 Topological Statistics** **20**
4.1 Motivation . . . . . 20
4.2 Continuity Statistics . . . . . 22
4.2.1 Mathematical Framework . . . . . 23
4.2.2 The Theiler window and false neighbours . . . . . 26
4.2.3 Algorithm . . . . . 27
4.2.4 Comparison and Numerical Implementation . . . . . 29
4.3 Differentiability Statistic . . . . . 31
4.3.1 Mathematical Framework . . . . . 31
4.3.2 Algorithm . . . . . 34
4.3.3 Note on Numerical Implementation . . . . . 35

**5 Finding the Minimum Nonlinearity** **36**
5.1 Classical RC . . . . . 37
5.1.1 Synthetic Systems . . . . . 37
5.1.2 Real Datasets . . . . . 41
5.1.3 AMOC climate data . . . . . 42
5.2 Minimal RC . . . . . 43
5.2.1 Lorenz System . . . . . 43
5.2.2 Halvorsen System . . . . . 44
5.2.3 Thomas System . . . . . 45

5.3 Usage Prescription . . . . . 46

**6 Multifunctionality** **47**
6.1 Architectures . . . . . 47
6.2 Inclusion of three attractors . . . . . 51
6.3 Continuity in Multifunctionality . . . . . 54
6.4 Results . . . . . 55

**7 Conclusion** **64**

# 1 Introduction

A majority of real-world data relies on the dictum of chaos, "The present determines the future, but the approximate present does not approximately determine the future"[1]. In this regard, Lorenz posits the idea that there exists a distinct class of systems that are increasingly hard to predict despite the knowledge of their approximate nature (or, in many cases their exact differential equations). We know today that these systems encompass a large body of fields ranging from weather and turbulence to population dynamics and finance. Therefore, finding efficient ways to predict the evolution of such systems becomes a central question. For the purposes of this thesis, we will restrict our attention to complex dynamical systems of dimension $d$ that are exponentially sensitive to their initial conditions and lie on a so called attractor; i.e., show 'chaotic behaviour'. Such a system can be described through coupled differential equations of the form:

$$\begin{aligned} \frac{dx_1}{dt} &= f_1(x_1, x_2, \ldots, x_d) \\ \frac{dx_2}{dt} &= f_2(x_1, x_2, \ldots, x_d) \\ &\vdots \\ \frac{dx_d}{dt} &= f_d(x_1, x_2, \ldots, x_d) \end{aligned} \tag{1}$$

with $\mathbf{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$. The functions $f_i$ may additionally have several time-independent parameters that allow for different dynamical behaviour for the same structure and mathematical terms of differential equations. Given an initial point $\mathbf{x}(t_0)$, a solution to the differential equations is called a trajectory. If two trajectories start infinitesimally close to each other but diverge exponentially fast while being bounded, the system exhibits chaotic behaviour and is called a strange attractor [2]. Even though the system dynamics are deterministic and bounded, their sensitivity to initial conditions imply that trajectories cannot be reconstructed without the exact knowledge of the initial state. From a numerical integration point of view, a finite-precision representation specifies the initial condition only to within machine epsilon, and every integration step injects additional round-off and truncation error. Because these perturbations are amplified at a rate set by the largest Lyapunov exponent, the time over which a computed trajectory tracks the true one grows only logarithmically with precision. The functions $f_i(\mathbf{x})$ that describe chaotic systems must have at least one nonlinear term, which makes them difficult or even impossible to solve analytically.

Among the various approaches used to predict such systems, artificial intelligence and machine learning models have shown significant success [3]. A fundamental breakthrough came when Takens in [4] showed that a chaotic attractor can be reconstructed through partial data. This implies that the data of the attractor carries information about its governing equations. The natural candidates in order to predict time series are Recurrent Neural Networks or RNNs that use a gradient based approach to learn sequential data.

A fundamental limitation of these RNN architectures in predicting chaotic series is their inability to remember long-term dependencies. This is referred to as the "vanishing gradient" problem [5]. In order to mitigate this issue, a neuromorphic computing approach was adopted to create a category of novel architectures referred to as "Reservoir Computing (RC)" [3]. A reservoir computing architecture operates by driving a fixed, high-dimensional recurrent network, the reservoir, with an input signal and training only a linear readout layer.

These architectures have made remarkable progress in predicting chaotic systems [6, 7], and have been quite successful in increasing performance in comparison to the standard RNN and fully connected network architectures [8, 9]. Additionally, these architectures are also robust in learning multiple datasets with the same reservoir. This ability is known as multifunctionality [10], and will be discussed in later parts of this thesis. In spite of this improvement in prediction, fundamental questions about its inner workings remain open: Why does it work? When does it fail? And how should its architecture be tailored to a given dataset?

Two key challenges persist in the field. First, *hyperparameter optimization* remains largely heuristic, with the spectral radius, reservoir size in classical RC [3], and the degree of nonlinearity in the reservoir states for minRC [11] all requiring careful tuning. Second, *explainability* is limited; because the reservoir's weights are typically initialized randomly, it is difficult to attribute the model's success or failure to specific structural properties.

Minimal reservoir computing (minRC) [12] addresses some of these concerns by replacing the random reservoir with a deterministic, block-diagonal architecture. This eliminates the confounding effects of random initialization and allows systematic study of the reservoir's dynamics. Recent work seen in [11] introduced a "tailored" minimal RC with a single tunable nonlinearity parameter $\eta$ in the generalized reservoir states. They evolve the reservoir states linearly and introduce a nonlinearity $\eta$ only during the readout. They showed that prediction performance peaks when $\eta$ matches the nonlinearity of the data. They further showed that for systems with multiple nonlinearities, the smallest one dominates prediction quality.

The work by Prosperino et al.[11] attempts to make the RC architecture less opaque by fixing the topology of the reservoir to a block diagonal matrix. This architecture enables the separation of reservoir states according to coordinate pairs. This form of the reservoir is just as capable of predicting systems as a classical RC system that uses a random network. This raises the natural question: How much information does the topology of the reservoir itself hold?

Pecora et al. in their work [13] showed that there exist certain statistics central to this thesis that aid in describing the topological relationship between the training data (which we will now refer to as the drive D) and the reservoir states (which we will now refer to as the reservoir R). Their work showed that one can optimise the embedding quality of the drive onto the reservoir by extremising these statistics.

In this thesis, we attempt to build on Pecora [13] and Prosperino's [11] work and show that the continuity statistic is a useful measure in performing a search of the nonlinearity in minRC and classical RC architectures with some limitations. We also show that the continuity statistics are useful in probing multifunctionality and provide parameter optimisation for learning many attractors. In this regard, this work provides a novel prediction free pipeline and diagnostic to perform hyperparameter search. The chapters and what they encompass are listed as follows:

- Chapter 2 defines the synthetic data and the various attractors used in this work.
- Chapter 3 outlines the RC architectures that were used in this work.
- Chapter 4 defines the continuity statistics and outlines Pecora's [13] algorithm.
- Chapter 5 applies the continuity statistics to probe the minimum required nonlinearity for a successful prediction.
- Chapter 6 focuses on multifunctionality and the use of the continuity statistic to optimise learning.
- Chapter 7 concludes the findings and proposes future directions and outlook.

# 2 Systems Overview

While no universal definition of a complex system exists, chaotic systems can be characterized by a variety of quantities, chief among them being the spectrum of Lyapunov exponents, "which has proven to be the most useful dynamical diagnostic for chaotic systems" [14]. Consider two trajectories denoted by $\mathbf{x}_1(t)$ and $\mathbf{x}_2(t)$ which are both infinitesimally seperated at the beginning. $||\mathbf{x_1}(0) - \mathbf{x_2}(0)|| = \delta(\delta \to 0)$. If at a given time $t$, these trajectories diverge exponentially, and yet remain globally bounded. If the trajectories converge to or remain close to this globally bounded structure that we define as an attractor, then we classify these systems as chaotic. This rate of divergence is referred to as the largest Lyapunov exponent. From now on, we refer to the largest Lyapunov exponent simply as the "Lyapunov exponent". If we assume $\delta(t)$ as the euclidian distance between two infinitesimally close trajectories at a given time $t$, and they show the behaviour as described in equation 2, then $\lambda$ is its Lyapunov exponent.

$$\delta(t) = ||\mathbf{x_1}(t) - \mathbf{x_2}(t)|| \sim \delta(0)e^{\lambda t} \tag{2}$$

The systems chosen in this thesis are all categorised by a positive Lyapunov exponent. There are several merits for using chaotic systems as a source of synthetic data. Primarily, these systems show exponential sensitivity to initial conditions [15]. In other words, given a long trajectory, we cannot necessarily infer the initial conditions of the data. This mimics the real world quite well, where we often only have access to a small fraction of noisy data without any knowledge of the initial conditions that cause the behaviour. A further important property of complex systems is that their trajectories neither settle into fixed points nor converge to periodic orbits. Rather, they exhibit sustained aperiodic behaviour over time. Consequently, learning from a limited segment of the data, such as the training set in a machine learning context, the model cannot simply learn a fixed repeating pattern and expect it to reappear later in the same form. Additionally, we reiterate that complex systems remain deterministic and globally bound on an attractor. The aperiodic behaviour is a consequence of the nonlinear properties of the dynamical system itself and not of stochasticity or random forces. This allows for a simple simulation pipeline where we solve a differential equation to obtain how each coordinate of the attractor evolves. To ensure that the trajectory has settled onto the attractor, we discard the 5000 initial datapoints as transients.
To quantify the efficacy of how well this synthetic data is learnt by the machine learning model, we are interested in two quantities. First, how much do the predicted values drift from the true trajectory. This measure is known as the forecast horizon of the model. This forecast horizon is formally defined by setting a threshold of deviation [16]. Therefore we define the horizon time $T_H$ as:

$$\begin{aligned} N_H &\equiv ||x_{\text{True}} - x_{\text{Predicted}}|| < \varepsilon \\ T_H &= \Delta t \times N_H \end{aligned} \tag{3}$$

In the equation above, the equality gives the number of data points that fall within the threshold, and then this threshold is converted to time units through the $\Delta t$ of our integrator. In this work, we set $\varepsilon = \text{std}(\mathbf{x})$ as the precision threshold to define a formal departure from true values. 'std' here refers to the largest coordinate-wise standard

deviation. This quantity is a standard accepted measure across various studies. Now, it is obvious that $T_H$ is not an inherently efficient measure when we compare two dynamical systems. This is essentially where the second quantity of the Lyapunov exponent comes in. A positive Lyapunov exponent indicates the presence of chaos, and every dynamical system has a different Lyapunov exponent. A lower Lyapunov exponent alludes to a lower rate of separation between nearby initial conditions, and hence, may be a system that is easier to predict than a system with a higher Lyapunov exponent. Therefore, the time scale at which the attractor's nearby points diverge is set by Lyapunov time $\tau_\lambda = 1/\lambda$. In this context, when we compare two dynamical systems, we must normalise the horizon time by the system's corresponding Lyapunov exponent.

$$T_{FH} \equiv \frac{1}{n} \sum_i \lambda_i T_{H(i)} \tag{4}$$

We calculate the Lyapunov exponent of the sytem through the method introduced by Rosenstein et al. [17]. For this, we track the evolution of initially close points in phase space by identifying nearest neighbors in the time series and measuring the average logarithmic divergence over time. The slope of this divergence computed over a selected time interval yields an estimate of the maximal Lyapunov exponent.

## 2.1 Selection of Attractors used in this work

The thesis consists of results from seven attractors. Each attractor has a different positive Lyapunov exponent and a different geometrical structure. The underlying equations and the justification for choosing these attractors can be listed as follows.

### Lorenz Attractor

The Lorenz attractor [18] is governed by

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

The parameters used are

$$\sigma = 10, \qquad \rho = 28, \qquad \beta = \frac{8}{3}.$$

The Lorenz system is included as a canonical three-dimensional chaotic flow, originally intended for weather modelling [18]. The attractor is a standard benchmark for testing sensitivity to initial conditions, prediction accuracy, and attractor reconstruction. The Lorenz system has the Lyapunov exponent $\lambda \approx 0.91$ [19].

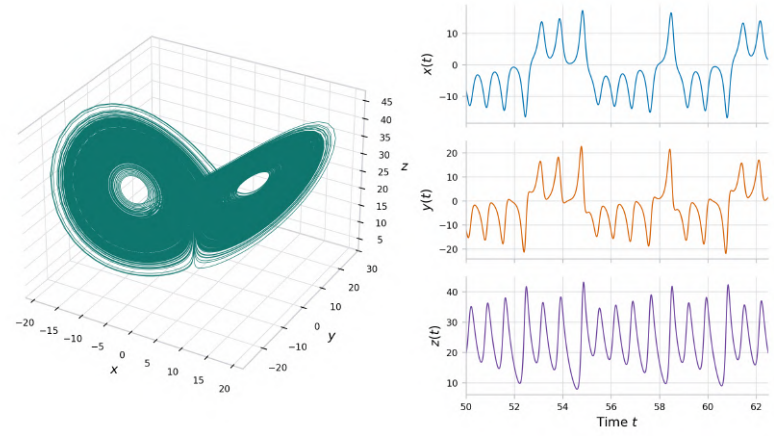



Figure 1: The Lorenz attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

### Halvorsen Attractor

The Halvorsen attractor is governed by

$$\begin{aligned}\dot{x} &= -ax - by - bz - y^2,\\ \dot{y} &= -ay - bz - bx - z^2,\\ \dot{z} &= -az - bx - by - x^2.\end{aligned}$$

The parameters used are $a = 1.4, \quad b = 4$. The Halvorsen attractor is included because it provides a cyclically symmetric chaotic system with quadratic nonlinearities in each coordinate. Its structure differs from the Lorenz and Rössler systems, making it useful for evaluating whether the analysis generalizes across distinct chaotic geometries.The Lyapunov exponent is $\lambda = 0.81$ [19]. While not formally published, a form of the attractor can be found in [20].

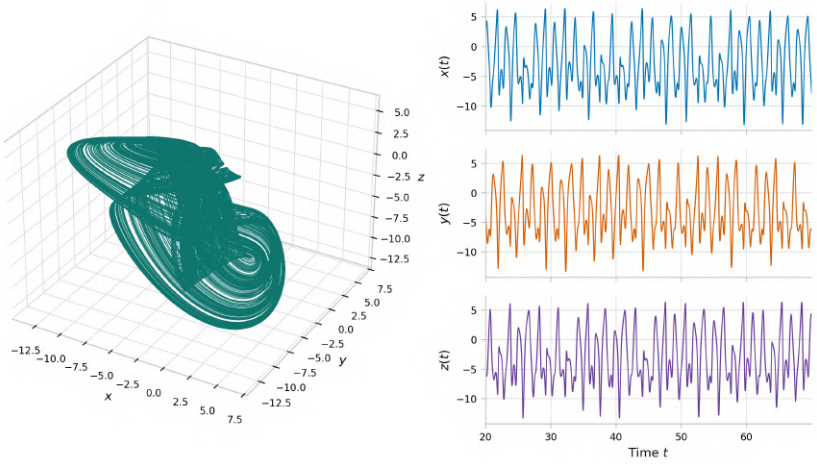



Figure 2: The Halvorsen attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

### Thomas Attractor

The Thomas attractor [21] cyclically symmetric attractor is governed by

$$\begin{aligned}\dot{x} &= -bx + \sin(y),\\ \dot{y} &= -by + \sin(z),\\ \dot{z} &= -bz + \sin(x).\end{aligned}$$

The parameter used is

$$b = 0.18.$$

The Thomas attractor was included because its smooth sinusoidal coupling and cyclic symmetry provide dynamics that are qualitatively different from polynomial chaotic systems. The Lyapunov exponent is $\lambda = 0.0349$, being barely chaotic.

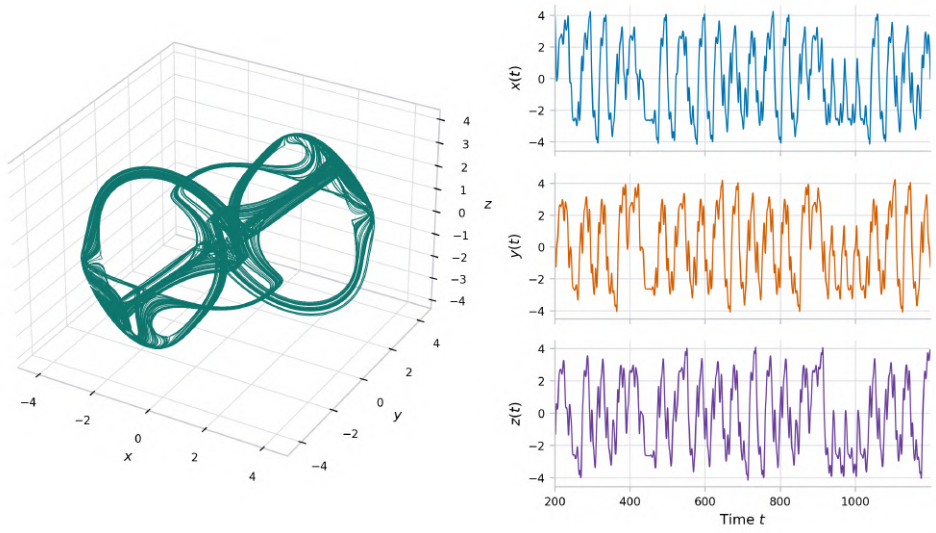



Figure 3: The Thomas attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

### Rössler Attractor

The Rössler attractor [22] is governed by

$$\begin{aligned}\dot{x} &= -y - z,\\ \dot{y} &= x + ay,\\ \dot{z} &= b + z(x - c).\end{aligned}$$

The parameters used are $a = 0.2$, $b = 0.2$, $c = 5.7$. The Rössler system is included as a classical spiral-type chaotic attractor. Its relatively simple structure and weakly positive largest Lyapunov exponent make it a useful comparison case. The Lyapunov exponent for this system is $\lambda = 0.0714$ [19].

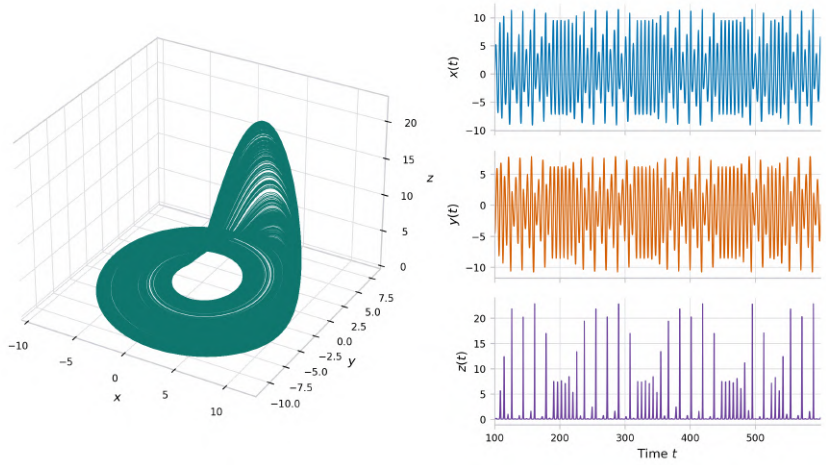



Figure 4: The Rössler attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

### Rucklidge Attractor

The Rucklidge attractor [23] is governed by

$$\begin{aligned}\dot{x} &= -\kappa x + \zeta y - yz,\\ \dot{y} &= x,\\ \dot{z} &= -z + y^2.\end{aligned}$$

The parameters used are $\kappa = 2,\ \zeta = 6.7$.

The Rucklidge attractor is included because it combines bilinear and quadratic nonlinearities. This makes it useful for testing performance on dynamics that differ structurally from the Lorenz and Rössler systems. The Lyapunov exponent is $\lambda = 0.0643$ [19] which is again a barely chaotic system.

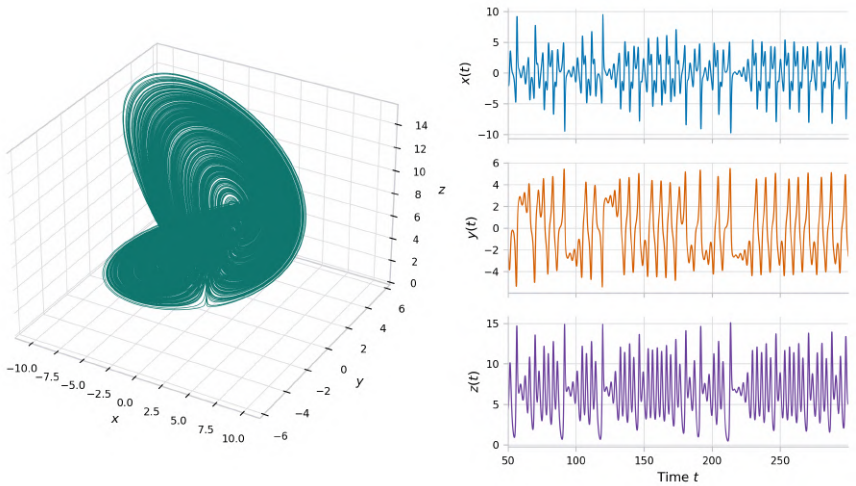



Figure 5: The Rucklidge attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

## Chua Attractor

The Chua circuit [24] is governed by

$$\begin{aligned}\dot{x} &= \alpha\big(y - x + \phi(x)\big),\\ \dot{y} &= x - y + z,\\ \dot{z} &= -\beta y,\end{aligned}$$

where the piecewise-linear nonlinearity is

$$\phi(x) = bx + \frac{1}{2}(a-b)\left(|x+1| - |x-1|\right).$$

The parameters used are

$$\alpha = 9, \qquad \beta = \frac{100}{7}, \qquad a = \frac{8}{7}, \qquad b = \frac{5}{7}.$$

The Chua attractor is included because it is a physically motivated electronic-circuit system with a piecewise-linear nonlinearity. This provides a useful nonsmooth benchmark in contrast to the smooth polynomial and sinusoidal systems. The Lyapunov exponent is $\lambda = 0.3271$ [19].

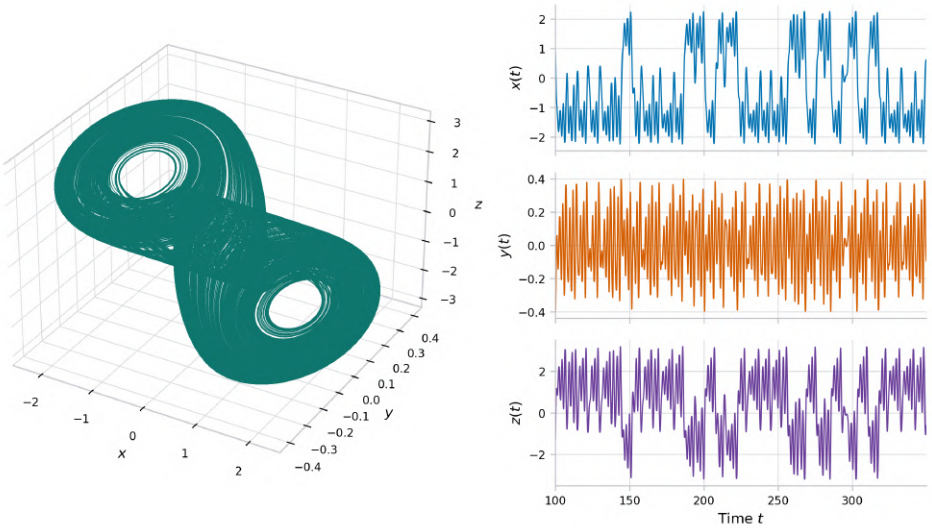



Figure 6: The Chua attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

## Complex Butterfly Attractor

The Complex Butterfly attractor [25] is governed by

$$\begin{aligned}\dot{x} &= a(y - x),\\ \dot{y} &= -z\ \mathrm{sgn}(x),\\ \dot{z} &= |x| - 1.\end{aligned}$$

The parameter used is $a = 0.55$. The Complex Butterfly attractor was included because it has a nonsmooth butterfly-shaped geometry involving both the sign and absolute-value functions. This makes it a useful test case for attractors with discontinuous or piecewise-defined nonlinear structure.The Lyapunov exponent is $\lambda = 0.169$ [19].

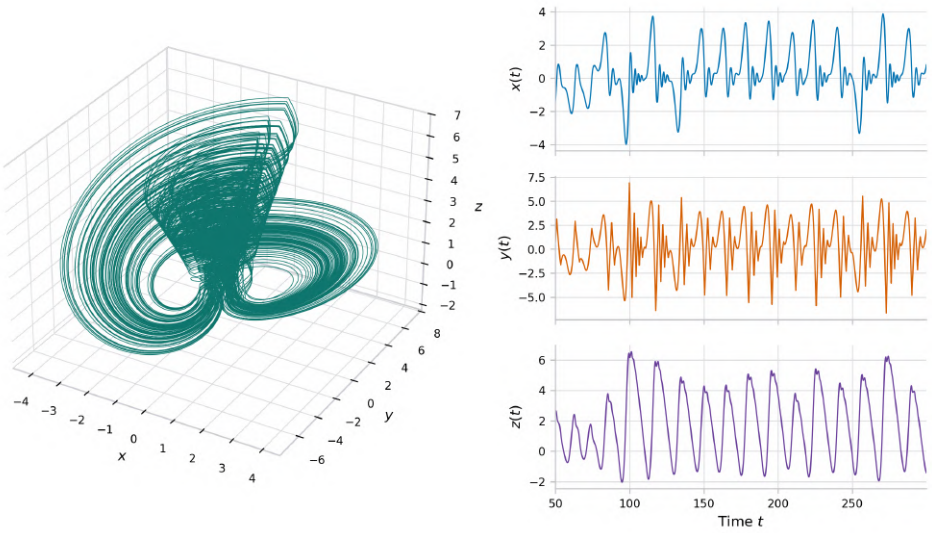



Figure 7: The Complex Butterfly attractor. Left: The trajectory in $(x, y, z)$ phase space. Right: The corresponding time series $x(t)$, $y(t)$ and $z(t)$.

# 3 Reservoir Computing Architectures

Reservoir Computing (RC) is a novel class of Artificial Intelligence (AI) and Machine Learning (ML) models that are used in processing, learning, and predicting sequential, time-series data. Contrary to conventional deep neural architectures that depend heavily on weight optimisation, RC is a tripartite architecture consisting of the input layer, the "Reservoir", and the readout layer. The reservoir itself is a dynamic, nonlinear computational substrate, whose input weights transform input data into a higher-dimensional representation [26]. The reservoir itself is driven by the data, and helps evolves the dynamic. The resulting representation generated by this reservoir is combined with a suitable readout layer to produce the desired output or prediction.

One of the key strengths of reservoir computing is its simplified training procedure. Unlike conventional neural networks, which require extensive weight tuning throughout the training phase, reservoir computing concentrates on training only the readout layer reducing most problems to solving a linear regression problem. Because the reservoir naturally generates rich, complex representations of the input data, the readout layer can exploit these representations directly, ensuring that computational demands for learning remain manageable. Consequently, RCs become an attractive, environmentally friendly approach to ML.

In this thesis we focus our attention to the behaviour of two ML architectures which we refer to as classical or regular RC, and minimal RC. The regular RC refers to the architecture where the Reservoir itself is a network whose connections can be described by an adjacency matrix. The adjacency matrix is a sparse random matrix dimension $D_r \times D_r$ where $D_r$ are the number of nodes. The network was chosen to be a sparse and random in tandem with the conclusions of Haluszczysnki et al. [16]. The form of the network could be small world, or scale free, but we choose an Erdős–Rényi network as suggested in Carroll's work [27]. Contrary to the classical RC approach, the work of Ma et al. [28] replaces the random network reservoir with a block-diagonal matrix with fixed weights. This architecture's reservoir state evolves linearly and the non-linearity is added after the evolution. This single nonlinearity becomes a hyperparameter we can vary.

## 3.1 Regular RC

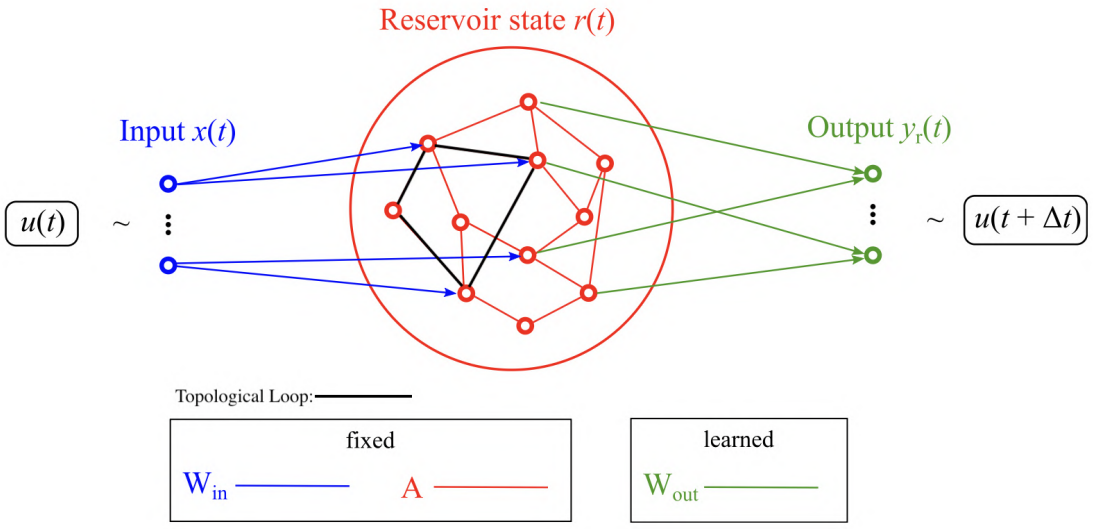



Figure 8: The tripartite RC architecture with a random network as its reservoir. The black loop hints at the role of topology in learning. This figure is modified from [29].

Unlike a feed forward neural network, RC networks exhibit topological loops. These topological loops make RC especially well-suited for temporal/sequential data, as they enable the network to "remember" previous inputs. This structure is particularly amenable to physical systems [30]. The architecture can be visualised in figure 8. The input matrix $\mathbf{w}_{in}$ is represented in blue whereas the network is represented by the adjacency matrix $\mathbf{A}$ in red. The readout matrix is illustrated in green.

### 3.1.1 Fundamental Concept

The reservoir computing architecture connects the input data $\mathbf{u}(t)$ to the desired output data $\mathbf{y}(t)$. Essentially, the time series data is injected into a nonlinear dynamical system (the network) of a higher dimension and it essentially drives the system. The two spaces, the drive space $\mathbf{u}(t)$ and the reservoir space $\mathbf{r}(t)$ are related through the dynamical equation

$$\mathbf{r}(t+1) = f\left(\mathbf{W}_{in}\mathbf{u}(t) + \mathbf{A}\mathbf{r}(t)\right) \tag{5}$$

In this equation $\mathbf{r}(t)$ is the reservoir's $r_{\text{dim}}$ -dimensional state vector at time $t$, $\mathbf{u}(t)$ is the input vector of dimension $u_{\text{dim}}$ at time $t$. Another parameter that appropriately weighs the input is the matrix $\mathbf{W}_{\text{in}}$ with the dimension $r_{\text{dim}} \times u_{\text{dim}}$ . $\mathbf{A}$ is an $r_{\text{dim}} \times r_{\text{dim}}$ adjacency matrix, creating the reservoir network structure, and $f$ is a nonlinear activation function. In our work we take two activation functions, the standard hyperbolic tangent commonly used in classical RC, and the identity function used for the minRC architecture.

### 3.1.2 Configuring $\mathbf{W_{in}}$

The input weight matrix allows for the embedding of the input data into the reservoir such that each $W_{in,ij}$, quantify the connection strength between the input variable $u_j$ and the reservoir node $r_i$. This parameter is responsible for creating the higher dimensional embedding of the drive in the reservoir space. The $\mathbf{W}_{in}$ is populated with random numbers originating from a uniform distribution spanning $[-\sigma, \sigma]$. The $\sigma$ is the signifier of input strength. We set $\sigma = 1$. Higher values allude to a higher linkage from input to reservoir [7].

### 3.1.3 Reservoir

The specific definition of the classical RC architecture we used is inspired from the work on Echo State Networks (ESNs) [3]. A large sparse and random network is created as the reservoir. We begin by initializing an all-zero adjacency matrix. Further, with probability $p$, we assign a value of 1 to every off-diagonal element. The average connectivity of each node is $d = p \times (r_{\text{dim}} - 1)$ distinct nodes. This is a crucial parameter referred to as the average node degree. For an optimal implementation of the Echo State Property (ESP), which argues that the reservoir states are independent of initial conditions, we scale the reservoir matrix to a target spectral radius $\rho$. This spectral radius is the largest absolute value of its eigenvalue distribuition. Therefore this is represented as,

$$\rho(\mathbf{A}) = \max\left\{|\lambda_i(\mathbf{A})|\right\}, \tag{6}$$

where $\lambda_i(\mathbf{A})$ are the eigenvalues of $\mathbf{A}$. The spectral radius in this regard determines the impact of previous reservoir states on the future ones, and significantly influences the memory of the reservoir.

$$\mathbf{A} \mapsto \frac{\rho^*}{\rho(\mathbf{A})}\mathbf{A}. \tag{7}$$

### 3.1.4 Readout

The iterative equation for $\mathbf{r}(t)$ shown earlier is not enough for prediction accuracy. To enhance prediction accuracy by breaking symmetries, Herteux & Räth [31] introduced a nonlinear readout layer where the reservoir states are further augmented with the squared reservoir states resulting in the new reservoir state matrix taking the $\tilde{\mathbf{r}}$. We refer to $\tilde{\mathbf{r}}$ as the generalised reservoir state:

$$\tilde{\mathbf{r}} = \left[r_1, \ldots, r_{r_{dim}}, r_1^2, \ldots, r_{r_{dim}}^2\right] \tag{8}$$

Each element in the reservoir states is squared, and hence the generalized reservoir states are referred to as $\tilde{\mathbf{r}}(t)$. If this symmetry breaking term is not included, we find the presence of mirror attractors– attractors that are inverted forms of the original attractors. This mirror attractor is a direct consequence of the fact that the activation function is of the form $f(-x) = -f(x)$.

### 3.1.5 Training and Prediction

The only trained parameter in this architecture is the $\mathbf{W}_{\text{out}}$. $\mathbf{W}_{\text{out}}$ is just a matrix that picks the linear combination of those random temporal features in the reservoir space that best matches your target. The reservoir maps the input history into a high-dimensional state space in which the target is assumed to be well approximated by a linear functional of the reservoir state [32]. Under these conditions a linear readout is sufficient, since echo state networks with linear readouts are universal approximators of fading-memory filters [33]. A simple ridge regression on the high-dimensional representation is assumed to be enough. The fundamental distinction between ridge regression and ordinary least squares (OLS) regression is the presence of a regularization term in the loss function [34]. The ridge regression is given by:

$$\mathbf{W}_{\text{out}} = \left(\tilde{\mathbf{R}}^T\tilde{\mathbf{R}} + \beta I\right)^{-1}\tilde{\mathbf{R}}^T\mathbf{Y}. \tag{9}$$

Now $\mathbf{W}_{\text{out}}$ is the output weights matrix with dimensions $u_{\text{dim}} \times 2r_{\text{dim}}$. It is the only component of the RC that undergoes training. $\mathbf{Y}$ is the training data matrix, with dimensions $u_{\text{dim}} \times T$, that carries the desired output values. $\tilde{\mathbf{R}}$ is the matrix of size $2r_{\text{dim}} \times T$ containing the stacked generalized reservoir states. The regularization parameter $\beta$ controls the strength of the penalty for large parameter values, which ensures stability. For $\beta = 0$ the ridge regression reduces to an OLS regression. $I$ is is the identity matrix. Therefore the final recursion relation to predict the data is simply:

$$\mathbf{y}(t) = \mathbf{W}_{\text{out}}\,\tilde{\mathbf{r}}(t). \tag{10}$$

We also collect the vector reservoir states and their outputs into matrices of dimensionality matching the training length. For any arbitrary length,

$$\mathbf{r}(t+1) = f\left(\mathbf{W}_{\text{in}}\,\mathbf{W}_{\text{out}}\,\tilde{\mathbf{r}}(t) + \mathbf{A}\mathbf{r}(t)\right). \tag{11}$$

### 3.1.6 Sample Prediction

Using DLR's SCAN library [19], a chaotic Lorenz63 trajectory of 50,000 steps is generated (integrated at $\Delta t = 10^{-2}$) from the initial condition $\mathbf{x}_0 = (-14.0,\ -20.9,\ 25.5)$. An Echo State Network with a reservoir of 150 nodes, spectral radius $\rho = 0.9$, and hyperbolic tangent activation is then trained on this data: the first 1,000 steps synchronize the reservoir, 45,000 steps are used for fitting a $[\mathbf{r},\ \mathbf{r}^2]$ readout, and the network autonomously forecasts the subsequent 1,000 steps ($\mathbf{y}_{\text{pred}}$) for comparison against the ground truth ($\mathbf{y}_{\text{test}}$). This is shown in figure 9. The prediction matches very closely with initial time steps and then deviates.

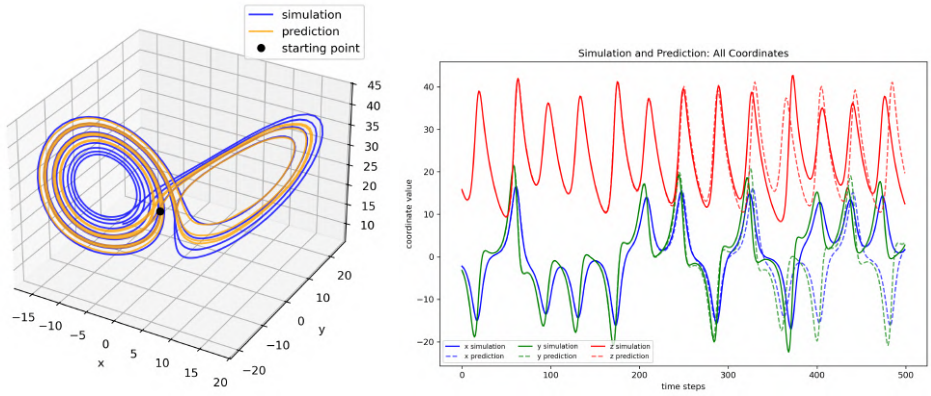



Figure 9: Attractor reconstruction and the prediction quality of a given coordinate for a regular RC architecture is shown in the figure.

## 3.2 Minimal RC

The choice of a random network as a reservoir may not be the only choice for a reservoir to successfuly predict chaotic systems. While randomness does affect predictive capability, its not neccesary [16]. Additionally, the random components of the RC also make it less transparent and interpretable. In this regard, Ma et al in [12] introduces a new approach that eliminates randomness. The architecture is refered to as minimal reservoir computing which we will call minRC. The main motivation behind the architecture is to eliminate the randomness present in the classical reservoir computing architectures. Minimal RC is still a subset of RCs. The brief overview of the minRC architecture is as follows:

**Minimal Reservoir Computing**

1. The reservoir matrix $\mathbf{A}$ has a block-diagonal structure in which each block is an all-ones matrix of dimension $b$. Minimal RC thinks in "features," and we use one block diagonal matrix for each feature, eliminating randomness.

2. **Linear Reservoir Dynamics**. No nonlinear activation is applied when forming the reservoir states $\mathbf{r}(t)$.

$$\mathbf{r}(t+1) = \mathbf{A} \times \mathbf{r}(t) + \mathbf{W}_{in} \times \mathbf{u}(t)\,, \tag{12}$$

3. **Nonlinear Readout**. Rather than appending only the squared reservoir states, we form generalized states $\tilde{\mathbf{r}}$ that collect every power up to a chosen nonlinearity order $\eta$:

$$\tilde{\mathbf{r}} = \mathbf{r}, \mathbf{r}^2, \ldots, \mathbf{r}^{\eta-1}, \mathbf{r}^{\eta}$$

.

In this way, the readout captures the full nonlinear structure of the signal.

**Training**. Following standard RC practice, the training inputs $\mathbf{u}$ and their associated reservoir states $\tilde{\mathbf{r}}$ are assembled column-wise into matrices $\mathbf{U}$ and $\tilde{\mathbf{R}}$. The solution of $\mathbf{W}_{out} \cdot \tilde{\mathbf{R}} = \mathbf{U}$ is then obtained via ridge regression with the equation outlined in equation 9.

**Prediction**. Forecasting proceeds in the same manner, driven by the update rule:

$$\mathbf{r}(t) = \mathbf{A} \cdot \mathbf{r}(t) + \mathbf{W}_{in} \cdot \mathbf{W}_{out} \cdot \tilde{\mathbf{r}}(t)\,. \tag{14}$$

Observe that $\mathbf{A}$ acts only on the *simple* reservoir state $\mathbf{r}$, whereas the companion term multiplying $\mathbf{W}_{out}$ uses $\tilde{\mathbf{r}}$, which carries all the nonlinear powers. The forecast time series $\mathbf{y}(t)$ then follows directly from the product:

$$\mathbf{y}(t) = \mathbf{W}_{out} \cdot \tilde{\mathbf{r}}(t)\,. \tag{15}$$

### 3.2.1 Combinatory Input Weights

The novel framework chooses elements of $\mathbf{W}_{in}$ in a structured manner. Primarily, to eliminate randomness, we select a set of equally spaced values between 0 and 1 for each block size of $b$:

$$\mathbf{w} = (w_1, w_2, \ldots, w_b)^T = \left(1, \frac{b-2}{b-1}, \ldots, \frac{1}{b-1}, 0\right)^T. \tag{16}$$

The non-invertibility of matrices and numerical instabilities are avoided through the square root of the weights: $\mathbf{w} = \left(\sqrt{w_1}, \ldots, \sqrt{w_b}\right)^T$. Additionally, the input matrix is structured such that each feature (coordinate) is fed seperately. For a 3D system with coordinates $\mathbf{u}(t) = (x, y, z)^T(t)$ and a nonlinearity order $\eta = 2$, the input matrix (multiplication) is performed as follows:

$$\mathbf{W}_{in} \cdot \mathbf{u}(t) = \begin{pmatrix} \mathbf{w} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{w} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{w} \\ \mathbf{w} & \mathbf{w} & \mathbf{0} \\ \mathbf{w} & \mathbf{0} & \mathbf{w} \\ \mathbf{0} & \mathbf{w} & \mathbf{w} \\ \mathbf{w} & \mathbf{w} & \mathbf{w} \end{pmatrix} \cdot \mathbf{u}(t) = \begin{pmatrix} x \\ y \\ z \\ x+y \\ x+z \\ y+z \\ x+y+z \end{pmatrix} (t) \otimes \mathbf{w}, \tag{17}$$

In the equation above $\otimes$ denotes the outer product and hence each block represents one feature $f$. The feature space has $n_f = 2^n - 1$ elements. and the dimension of the reservoir is $d = n_f \cdot b$.

### 3.2.2 Block-Diagonal and Binary Reservoir

To maintain the independence of individual features, we choose the reservoir as a block diagonal matrix composed of blocks of ones of size $b$. $\mathbf{J}_i$ is specifically associated with a particular feature:

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{J}_{n_f} \end{pmatrix} \longmapsto \begin{pmatrix} \mathbf{J}_x & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_y & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{J}_{y+z} \end{pmatrix}. \tag{18}$$

These are again scaled with the spectral radius $\rho^*$ between $(0, 1)$.

### 3.2.3 Nonlinear Readout

The entire nonlinearity of the prediction now depends on the readout. Therefore we compile higher orders of nonlinearity in generalised states: $\tilde{\mathbf{r}}$. For a given degree of nonlinearity $\eta$, they take the following form:

$$\tilde{\mathbf{r}} = \left\{\mathbf{r}, \mathbf{r}^2, \ldots, \mathbf{r}^{\eta-1}, \mathbf{r}^{\eta}\right\} \tag{19}$$

As proposed by Prosperino et al in [11], we reduce this collection to simply the reservoir and its nonlinear $\eta$ raised readout counterpart defined as:

$$\tilde{\mathbf{r}} = \left\{\mathbf{r}, \mathbf{r}^{\eta}\right\}. \tag{20}$$

The exponentiation is applied element wise.

### 3.2.4 Sample Prediction

The efficacy of a non-random and block diagonal reservoir in predicting chaotic systems can be seen below:

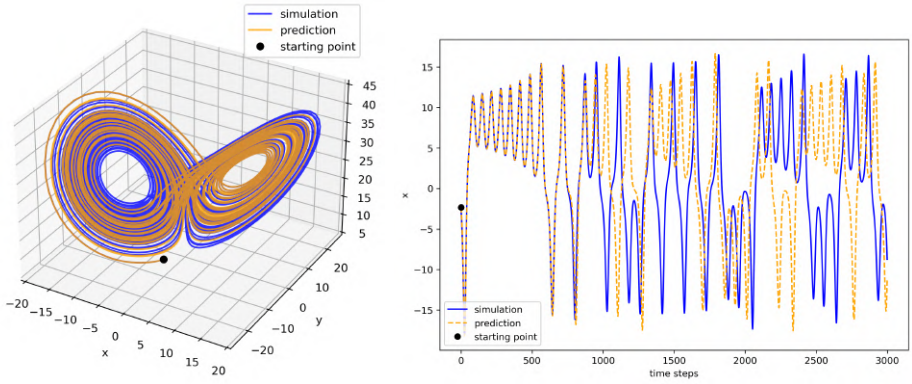



Figure 10: There attractor reconstruction and the prediction quality of a given coordinate for a minimal RC architecture is shown in the figure.

A Lorenz trajectory of 50,000 steps is generated at $\Delta t = 10^{-2}$ (with $\sigma = 10$, $\rho = 28$, $\beta = 8/3$) from the initial condition $\mathbf{x}_0 = (-14.0,\ -20.9,\ 25.5)$. The first 1,000 steps are dropped as transient. Of the remaining 49,000 steps, 39,200 are used for training and 3000 for forecasting. The model is the minimal reservoir computer of Prosperino et al. [11], which uses a block-diagonal reservoir (block size $b = 3$, spectral radius $\rho^* = 10^{-3}$) instead of a random one, and readout states of order $\eta = [1, 2]$. The trained model then predicts the next 3000 steps.

### 3.2.5 Importance of $\eta$:

In Prosperino et al. [11], they argue that prediction performance peaks when $\eta$ matches the nonlinearity of the data. They further showed that for systems with multiple nonlinearities, the smallest one dominates prediction quality. For instance, the Lorenz attractor consists of the coupling terms $xz$ and $xy$, which means that they have a nonlinearity order of $\eta = 2$. In comparison, a Thomas system has the term $\sin(x_1)$ which, if one uses Taylor expansion, has the lowest term as $x^3/3!$ and hence has $\eta = 3$. Using a deterministic, minimal reservoir framework and a novel chaotic system with tunable fractional exponents, we isolate this relationship and demonstrate that the smallest nonlinearity in the data plays a key role. This insight enables a method to estimate nonlinearity from time series alone. This phenomena is shown in Figure 11. The details of obtaining the specific plot and the importance of the nonlinearity are explained below.

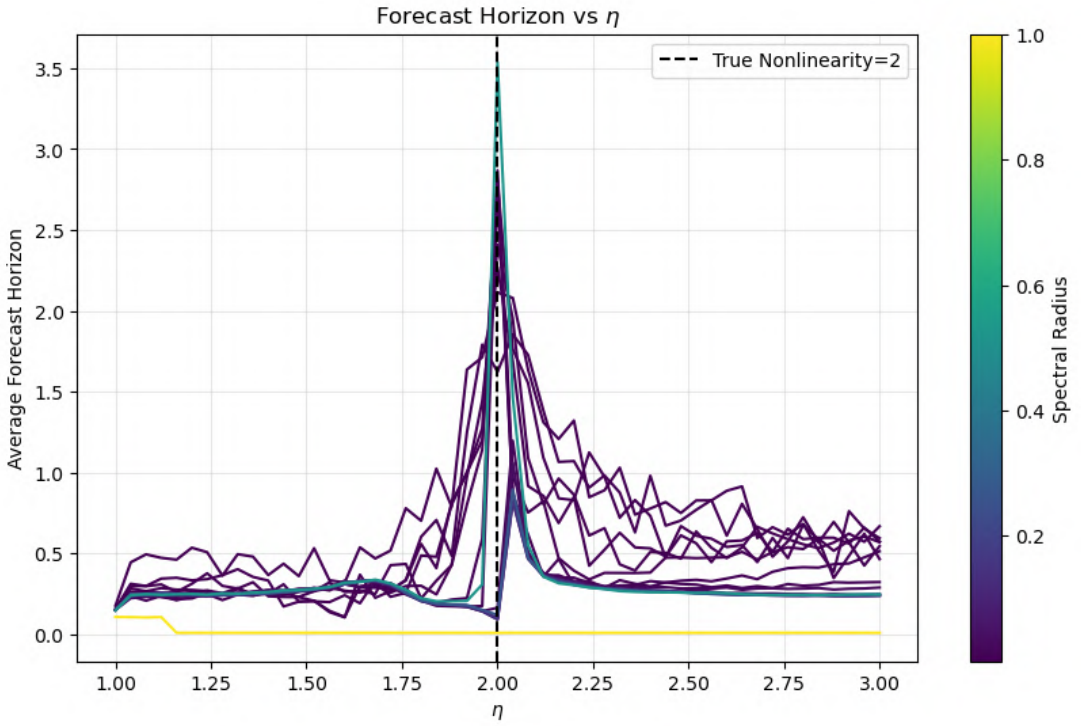



Figure 11: Prediction performance of the minimal reservoir computer on Lorenz as a function of the tunable reservoir-state nonlinearity $\eta$, colour-coded by spectral radius and averaged over 20 initial conditions. The forecast horizon peaks at $\eta = 2$ (dashed line), the nonlinearity order of the Lorenz coupling terms $xy$ and $xz$.

Trajectories are generated by integrating the standard Lorenz equations ($\sigma = 10$, $\rho = 28$, $\beta = 8/3$) with a fourth-order Runge–Kutta scheme at a step size of $\Delta t = 0.01$ for $3 \times 10^4$ steps; the first $10^4$ steps are discarded as transient, leaving $2 \times 10^4$ points on the attractor. Each trajectory is split 80/20 into a training set (16,000 steps) and a forecasting window ($\sim$4,000 steps), and the experiment is repeated for 20 random initial conditions drawn uniformly from $[-20, 20]$. Prediction is carried out with the deterministic, minimal reservoir computer (minRC), configured with block size $b = 3$, and a reservoir-state readout nonlinearity of the form $[1, \eta]$, where the tunable fractional exponent $\eta$ is the quantity of interest. We define this $\eta$ with its numerator component running from 50 to 150 and we fix the denominator $d$ to 50 for numerical stability.

$$r_i^{\eta} = \left(r_i^{\text{numerator}}\right)^{\frac{1}{50}} \tag{21}$$

We sweep $\eta$ over $[1, 3]$ in steps of 0.04 (51 values) and the target spectral radius over 15 logarithmically spaced values in $[10^{-4}, 1]$, evaluating all $51 \times 15 \times 20 = 15{,}300$ configurations (parallelised across initial conditions). Forecast quality is quantified by the forecast horizon (the longest run for which the per-coordinate error stays below that coordinate's standard deviation). This is expressed in Lyapunov times using the largest Lyapunov exponent estimated directly from the true trajectory. Our results confirm that the smallest nonlinearity present in the data governs prediction quality. This work raises a fundamental question about the definition of the reservoir and what it must constitute in order to predict complex systems. One fundamental question that remains to be clarified is what mathematical quantity enables the formation of the peak? Is it related to embedding? These questions are answered in the later chapter.

# 4 Topological Statistics

There exists many time series that are tediously long and multidimensional. As the number of features or dimensions in a dataset increase, the data space becomes increasingly empty and sparse. Due to this curse of dimensionality, it is often that researchers have no applicable models or mathematical relationships that can link two corresponding datasets or subsets of the time series data. For higher dimensional data sets such as the ones we often encounter in the case of reservoir computing, even plotting is inefficient. In this regard there exists a need to develop statistical analyses that can at least come up with functional mathematical relationships like continuity, and differentiability between the datasets we are comparing. In this regard, Pecora et al's work [13] provides a functional toolkit in the form of "continuity" and "differentiability" statistics. These statistics are able to take any two multidimensional datasets that are time twin comparable (meaning that dataset $a(t)$ is related to dataset $b(t)$ at the same time $t$). The toolkit attempts to answer what can we infer about functions that map $a(t)$ to $b(t)$? What properties can we say such functions have? Continuity? Invertibility? Differentiability? In this work, we assume that these functional relations exist on the sample of points from each dataset, and these points themselves are representative of the entire dataset. Mathematically, we assume that these samples exist on some manifold in their space, but are not strictly dense in the mathematical sense. There exists a need for good statistic based on definitions of distances between sets of points in a metric space. The metric space definitions are based on the work of Eggar et al. [35]. In this thesis, we try to come up with our numerical implementation of these statistics and use them in the context of hyperparameter optimisation for reservoir computers. We apply these statistics for two hyperparameters, the minimum required nonlinearity $\eta$, and the offsets set up for multifunctional reservoirs in minimal RCs(described in chapter 6).

## 4.1 Motivation

We can view the training data and the reservoir as two separate dynamical systems such that the training data drives the reservoir. Consequently, the reservoir itself possesses some properties of the drive. This idea was solidified further by Takens' embedding theorem [4]. We can show this embedding phenomenon in the reservoir for dynamical systems that follow iterative dynamics (such that $\mathbf{x}(n+1) = \mathbf{F}(\mathbf{x}(n))$). These have simpler mathematical expressions than flows (continuous-time systems generated through ODEs). However, to see a more rigorous mathematical proof involving flows, one can refer to [36]. Let us now consider the dynamical system such that

$$\mathbf{x}(n+1) = \mathbf{F}\big(\mathbf{x}(n)\big), \tag{22}$$

where $\mathbf{x}(n) \in \mathscr{M} \subseteq \mathbb{R}^k$ for all $n$, and $\mathbf{F} : \mathscr{M} \to \mathscr{M}$ is a diffeomorphism. We do not observe $\mathbf{x}(n)$ directly. Instead we record a single scalar measurement through a smooth observation function $h : \mathscr{M} \to \mathbb{R}$,

$$u(n) = h\big(\mathbf{x}(n)\big). \tag{23}$$

The sequence $\{u(n)\}$ is the training signal. The reservoir has state $\mathbf{r}(n) \in \mathbb{R}^{NM}$ ($M$ coupled reservoirs of $N$ nodes each) and is *driven* by $u(n)$ through a smooth map $\mathbf{G} : \mathbb{R} \times \mathbb{R}^{NM} \to \mathbb{R}^{NM}$:

$$\mathbf{r}(n+1) = \mathbf{G}\big(u(n),\, \mathbf{r}(n)\big). \tag{24}$$

Substitute (24) into itself repeatedly. Each equality replaces the current state $\mathbf{r}(m)$ by $\mathbf{G}\big(u(m-1), \mathbf{r}(m-1)\big)$, exposing one more past input

$$\begin{aligned}
\mathbf{r}(n+1) &= \mathbf{G}\big(u(n),\, \mathbf{r}(n)\big) \\
&= \mathbf{G}\big(u(n),\, \mathbf{G}(u(n-1),\, \mathbf{r}(n-1))\big) \\
&= \mathbf{G}\big(u(n),\, \mathbf{G}(u(n-1),\, \mathbf{G}(u(n-2),\, \mathbf{r}(n-2)))\big) \\
&\;\;\vdots \\
&= \mathbf{G}\big(u(n),\, \mathbf{G}(u(n-1),\, \mathbf{G}(u(n-2),\, \dots\, \mathbf{G}(u(0),\, \mathbf{r}(0))))\big).
\end{aligned} \tag{25}$$

So the reservoir state is an explicit function of the *whole input history* $u(n), u(n-1), \dots, u(0)$ together with the initial state $\mathbf{r}(0)$. Since each $u(j)$ is a scalar read from the drive, $\mathbf{r}(n+1)$ is a function of the current and past values of that one measured signal, together with $\mathbf{r}(0)$. Under the echo state property the dependence on $\mathbf{r}(0)$ decays as $n$ grows, so for large $n$ the reservoir state is determined by the input history alone. This has the same structure as a delay-coordinate map. For a generic observation function $h$ and a delay $\tau \in \mathbb{N}$, define the *delay-coordinate map* $\mathbf{E} : \mathscr{M} \to \mathbb{R}^d$ by

$$\mathbf{E}(\mathbf{x}) = \Big(h(\mathbf{x}),\, h\big(\mathbf{F}^{-\tau}(\mathbf{x})\big),\, h\big(\mathbf{F}^{-2\tau}(\mathbf{x})\big),\, \dots,\, h\big(\mathbf{F}^{-(d-1)\tau}(\mathbf{x})\big)\Big). \tag{26}$$

Takens' theorem states that $E$ is an embedding of $\mathscr{M}$ whenever $d > 2\dim \mathscr{M}$. That is, the delay vector built from the measured signal,

$$\big(u(n),\, u(n-\tau),\, \dots,\, u(n-(d-1)\tau)\big) = E\big(\mathbf{x}(n)\big), \tag{27}$$

traces out a set diffeomorphic to the attractor $\mathscr{A}$; more precisely, $E|_{\mathscr{A}}$ is a diffeomorphism copy of its own attractor [4].

In the context of embeddings, continuity and differentiability are the two fundamental mathematical properties that need to hold in order to create a map from drive space to reservoir space $\mathbf{x} \mapsto \mathbf{r}$, into a proper embedding of the drive attractor. An embedding is a map $E$ that is continuous, one-to-one, and smoothly invertible onto its image (a diffeomorphism). Continuity means that nearby drive states are sent to nearby reservoir states, so the attractor is carried into reservoir space without tearing and its image $\mathscr{M} = E(\mathscr{A})$ is a faithful copy rather than a scattered cloud of points. Being injective or one-to-one implies that two different drive states must never land on the same reservoir state, because if they were the same state, then a folding of the attractor could be inferred, information would be lost, and no readout could tell those states apart. Differentiability then makes the copy smooth rather than merely continuous, since it preserves the local structure of the attractor, meaning its tangent directions, its dimension, and the smooth way coordinates change from point to point.

A well-behaved derivative also helps keep the map one-to-one because it rules out local folding that would allow nearby parts of the attractor to cross. Once the map is an embedding in this full sense, the reservoir state fixes the entire drive state, so every drive variable becomes a smooth function of $\mathbf{r}$. And because $\mathscr{M}$ is a low-dimensional surface sitting inside the high-dimensional reservoir space $\mathbb{R}^{NM}$, those smooth functions can be closely matched by a simple linear map, which is exactly why a trained linear readout can recover the unseen drive variables. Continuity and differentiability are therefore not optional extras; they are the conditions under which the reservoir holds a true embedded image of the drive attractor instead of a distorted or collapsed shadow of it. In this thesis we therefore monitor the continuity and differentiability between the drive and the reservoir as a function of changing hyperparameters and make conclusions about the embedding quality.

## 4.2 Continuity Statistics

In this chapter, we develop the numerical definitions for continuity and differentiability that can be calculated on finite datasets rather than abstract mathematical ideas that work only in the infinitesimal limit. Let us assume that $\{x_t\}_{t=0}^{T-1} \subset \mathbb{R}^d$ denotes the trajectory of the drive system and $\{y_t\}_{t=0}^{T-1} \subset \mathbb{R}^r$ is the simultaneously sampled trajectory of the reservoir. Pairs $(x_t, y_t)$ carrying the same time index are called *time twins* [13]: The twins supply a point-by-point pairing between the two data clouds without our ever having to write down the map that produces it. The work in [37, 38] provides a way for us to define notions of continuity statistically, without requiring the infinitesimal limit. This limit is not plausible in real-life data since we often have access to only a finite set. We ask the most basic question: Is there statistical evidence, in the data alone, that a map $\phi$ from the reservoir to the drive is *continuous*? This question must be settled before any attempt at fitting $\phi$, because if no continuous relation exists, every subsequent regression or reconstruction is built on sand. Throughout this implementation, we assume that distances are Euclidean. We use a pre-programmed $k$-dimensional tree, a tool that automatically segregates points based on the ranges of the different distances found. This tool allows us to query different points and calculate Euclidean distances between them, and categorise them. Since a relation may be single-valued in one direction but one-to-many in the other (for instance $f(x) = |x|$, whose points over $y > 0$ each contain two separated inverse points), the statistic is therefore directional: It is evaluated once for drive $\to$ reservoir and once for reservoir $\to$ drive. We write $x$ for the source and $y$ for the target of whichever direction is under test.

### 4.2.1 Mathematical Framework

The starting point is the calculus definition of continuity at a point $x_0$,

$$\forall\,\varepsilon > 0 \;\; \exists\,\delta > 0 : \qquad \|x - x_0\| < \delta \implies \|\phi(x) - \phi(x_0)\| < \varepsilon, \tag{28}$$

which reads: Points close to $x_0$ in the source must have images close to $\phi(x_0)$ in the target, and this must hold at every resolution $\varepsilon$, however fine. A finite, noisy sample can never satisfy this condition since the dataset cannot be extrapolated to the infinitesimal limit. We do not have enough points. In this statistic, we therefore invert the logic of the definition. Rather than reducing the radius of $\varepsilon$ to the infinitesimal limit, we fix a pair $(x_i, y_i)$ and identify a candidate $\delta$-neighbourhood in the source $x_i$. We use the points in the neighborhood and look at their time-matched counterparts in the target space $y_i$ to ask: What is the *smallest* $\varepsilon$ at which the transported points are so tightly clustered around $y_i$ that chance can be ruled out? The "smaller" the radius in the context of the scale of the data, the more continuous it may be deemed. The $\delta$-neighborhood is taken to be the $n_\delta$ nearest neighbors of $x_i$. For each fiducial point choose a bunch of neighbourhood points (except self) and arrange these distance wise.

$$\mathscr{N}_i(n_\delta) = \{j_1, \ldots, j_{n_\delta}\}, \qquad \|x_{j_1} - x_i\| \le \cdots \le \|x_{j_{n_\delta}} - x_i\|, \tag{29}$$

so that the radius $\delta_i(n_\delta) = \|x_{j_{n_\delta}} - x_i\|$ is implicit rather than prescribed. The twins of the gathered neighbours sit at target distances:

$$d_k = \|y_{j_k} - y_i\|, \qquad k = 1, \ldots, n_\delta, \qquad \text{and} \qquad n_\varepsilon(\varepsilon) = \#\{k : \; d_k \le \varepsilon\} \tag{30}$$

counts how many of them the closed $\varepsilon$-ball around $y_i$ captures. We check this for all fiducial points.

**A geometric null hypothesis:** To decide whether an observed count $n_\varepsilon$ is not random, one needs a model of what "random" looks like. The null hypothesis $H_0$ is that *no functional relationship* ties $y$ to $x$, so that knowing a point is a source-neighbour of $x_i$ tells us nothing about where its twin lands. Therefore Pecora et al. [13] assume the twins fall at random into a ball centred on $y_i$ of *twice the volume* of the $\varepsilon$-ball; each twin then lands inside the $\varepsilon$-ball itself with probability[13]:

$$p_\varepsilon = \tfrac{1}{2}. \tag{31}$$

These suggestions are heuristic. The implementation fixes $p_\varepsilon = \frac{1}{2}$, i.e. a maximally conservative "coin-toss" null in which the $\varepsilon$-ball is already assumed to capture half of the points by chance, so rejecting it means the observed clustering is stronger than even a very generous accident could produce. Under $H_0$ the twins land independently, so the number of them inside the ball is a binomial random variable,

$$\Pr(X = j) = \binom{n_\delta}{j} p_\varepsilon^{\,j} \, (1 - p_\varepsilon)^{\,n_\delta - j}, \tag{32}$$

the probability that *exactly* $j$ of the $n_\delta$ twins fall inside. Summing (32) gives the probability of at most $n$ successes, the cumulative binomial distribution

$$C(n; n_\delta, p_\varepsilon) = \sum_{j=0}^{n} \binom{n_\delta}{j} p_\varepsilon^j \, (1-p_\varepsilon)^{\,n_\delta - j}, \tag{33}$$

(Eq. (8) of [13], and of the implementation), and the complement of "at most $n_\varepsilon - 1$" is the quantity the test actually needs: the probability of seeing *at least* the observed count by chance,

$$P_>(n_\varepsilon; n_\delta, p_\varepsilon) = 1 - C(n_\varepsilon - 1; n_\delta, p_\varepsilon) = \sum_{j=n_\varepsilon}^{n_\delta} \binom{n_\delta}{j} p_\varepsilon^j \, (1-p_\varepsilon)^{\,n_\delta - j} \tag{34}$$

Equation (34) is nothing but the $p$-value of the observed clustering under the geometric null, and $H_0$ is rejected whenever

$$P_>\big(n_\varepsilon; n_\delta, p_\varepsilon\big) < \alpha, \qquad \alpha = 0.02, \tag{35}$$

a threshold that sits roughly two standard deviations into the tail of the distribution [13].

**From the test to a length scale:** Since the tail probability $P_>$ shrinks as $k$ grows, every neighbourhood size $n_\delta$ has a minimal significant count $k^*(n_\delta)$: the smallest $k$ with $P_>(k; n_\delta, \frac{1}{2}) < \alpha$. With ten neighbours, for example, at least nine twins must land inside the ball: nine gives $P_> \approx 0.011 < \alpha$, while eight gives $P_> \approx 0.055$ which is still plausible by chance. The generous null (31) demands a supermajority, not a mere majority. Now sort the twin distances, $d_{(1)} \le \cdots \le d_{(n_\delta)}$. The $k^*$-th smallest is the smallest radius at which the null is rejected, so we set

$$\varepsilon_i(n_\delta) = d_{(k^*(n_\delta))}. \tag{36}$$

We try several $n_\delta$ rather than fix one because the map can reshuffle distances: Even a smooth $\phi$ stretches space unevenly. Small $n_\delta$ probes fine scales but has weak statistics; large $n_\delta$ has power to spare but coarse resolution. At each fiducial point we therefore keep the smallest radius found over $n_\delta = 5, \ldots, 30$ (never exceeding the available neighbours),

$$\varepsilon_i^\star = \min_{n_\delta} \varepsilon_i(n_\delta). \tag{37}$$

Continuity is a local property, so $\varepsilon_i^\star$ describes a single point. The attractor-wide statistic is its average over the random fiducial points $\mathscr{E}$ where it is defined,

$$\langle \varepsilon^\star \rangle = \frac{1}{|\mathscr{E}|} \sum_{i \in \mathscr{E}} \varepsilon_i^\star, \tag{38}$$

**Two normalising parameters: $\varepsilon_{\min}$ and $\sigma_{\text{std}}$.**
On its own, $\langle \varepsilon^\star \rangle$ is a length in the units of the target space: There is always some $\varepsilon$ large enough to swallow the required number of twins. To become a statistic it must be *normalised*, and the implementation provides two normalising parameters that answer two different questions. The first is the global scale of the target attractor,

$$\sigma_{\mathrm{std}} = \Big(\frac{1}{T}\sum_{t=1}^{T}\|y_t - \bar{y}\|^2\Big)^{1/2}, \qquad \bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t, \tag{39}$$

the root-mean-square radius of the data cloud about its centroid. Since $\sigma_{\mathrm{std}}^2 = \operatorname{tr}\Sigma_y$ (covariance matrix) is the total variance summed over the coordinates of $y$, this is the isotropic generalisation of the standard deviation (for scalar data). Dividing by it expresses $\varepsilon^\star$ as a *fraction of the attractor size*. A value $\langle\varepsilon^\star\rangle/\sigma_{\mathrm{std}} \approx 0.05$ says the images of source neighbourhoods cluster within 5% of the attractor's radius, whereas a value approaching one says the ball must engulf the attractor before the supermajority of twins is reached. Therefore for this value the map has no resolvable continuity at all.

The second is the resolution scale of the target,

$$\varepsilon_{\min} = \big\langle\, d_{\mathrm{NN}}^{(w)}(t)\,\big\rangle_{t\in S}, \qquad d_{\mathrm{NN}}^{(w)}(t) = \min_{\substack{|j-t|>w \\ \|y_j - y_t\|>0}} \|y_j - y_t\|, \tag{40}$$

which is the mean distance from a point to its nearest temporally uncorrelated neighbour (the nearest valid point after Theiler window exclusion discussed in figure 13). Dividing by it expresses $\varepsilon^\star$ in *multiples of the finest scale the sampling can resolve*. A ratio $\langle\varepsilon^\star\rangle/\varepsilon_{\min} \approx 1$ means the mapped neighbours land as tightly as the data are capable of demonstrating: The statistic has saturated at its resolution floor, and no stronger claim can be made without a longer or finer time series. We hence calculate all values as,

$$\Big(\ \langle\varepsilon^\star\rangle,\ \ \frac{\langle\varepsilon^\star\rangle}{\varepsilon_{\min}},\ \ \frac{\langle\varepsilon^\star\rangle}{\sigma_{\mathrm{std}}}\ \Big), \tag{41}$$

Continuity can be seen through a visual analogy in figure 12.

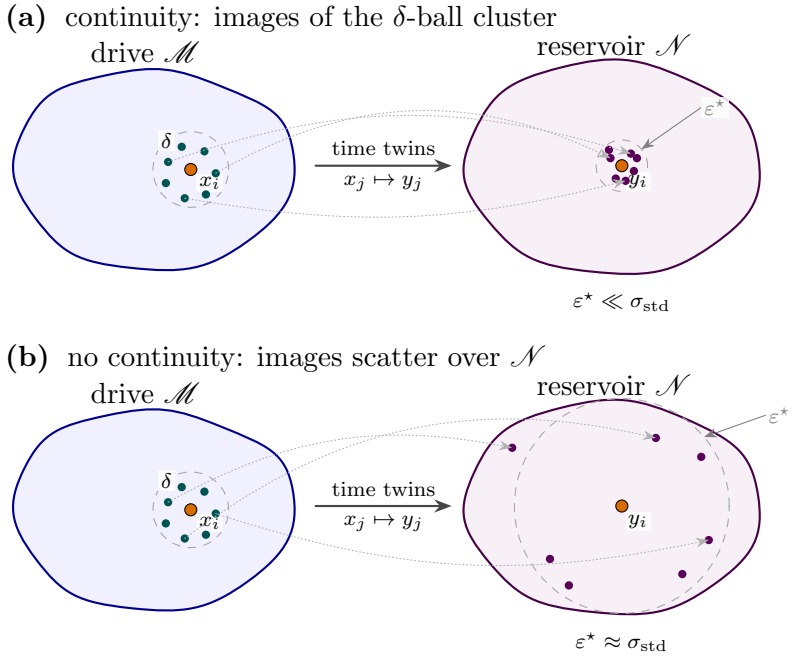



Figure 12: The two regimes of the continuity statistic. In both panels the same $\delta$-neighbourhood of a fiducial point $x_i$ (teal points inside the dashed $\delta$-ball) is transported to the reservoir attractor $\mathscr{N}$ through the time twins $x_j \mapsto y_j$ (dotted arrows show three representatives). **(a)** If a continuous map $\phi : \mathscr{M} \to \mathscr{N}$ underlies the data, the images (violet) cluster around the fiducial twin $y_i$; the binomial null (34) is rejected already at a small radius, and $\varepsilon^\star \ll \sigma_{\text{std}}$. **(b)** Without a continuous relation the images scatter over $\mathscr{N}$; the supermajority demanded is reached only once the $\varepsilon$-ball covers a large fraction of the attractor, so the smallest significant radius saturates at $\varepsilon^\star \approx \sigma_{\text{std}}$.

### 4.2.2 The Theiler window and false neighbours

Everything above rests on one tacit assumption: The $n_\delta$ neighbours are *independent* visits of the trajectory to the neighbourhood of $x_i$, so that the binomial model (32) which assumes independent coin tosses is valid. Samples of a flow that are adjacent in time violate this assumption in the worst possible way. Because trajectories are continuous in time, the samples $x_{i\pm1}, x_{i\pm2}, \ldots$ are automatically close to $x_i$ in state space: They are *false neighbours*, close by temporal correlation rather than by geometric recurrence. Worse, their twins $y_{i\pm1}, y_{i\pm2}, \ldots$ are just as automatically close to $y_i$, for the same reason. Admitting them would let the test "discover" continuity even between completely unrelated systems. It would merely re-detect the continuity of trajectories in time, not probe the map $\phi$. And such points can be manufactured at will. Sampling the same trajectory twice as fast doubles the number of temporal neighbours without adding any genuine geometric information [13]. The implementation therefore rejects a candidate neighbour with index $j$ if

$$\text{(i)}\ \ |j-i| \le w \qquad \text{or} \qquad \text{(ii)}\ \ \exists\, j' \text{ already accepted} : \ |j-j'| \le w, \tag{42}$$

The concept we discuss now is the Theiler window. We choose a Theiler window $w = 40$ time steps here, chosen on the recommendation of Pecora et al. [13]. Rule (i) is the classical Theiler exclusion around the fiducial point [39]: It carves a temporal moat around $i$ so that no sample from the fiducial point's own strand of trajectory can pose as a neighbour. Rule (ii) proposed by Kennel and Abarbanel [40] is also integrated for further safety. Once a genuine neighbour is accepted, all samples within $w$ steps of it belong to the same pass through the neighbourhood and carry no new information, so at most one representative per strand is kept. The accepted neighbours are thus pairwise separated by more than $w$ steps,one point per independent recurrence, which is precisely what the independence assumption behind (32) requires. Figure 13 illustrates both rules.

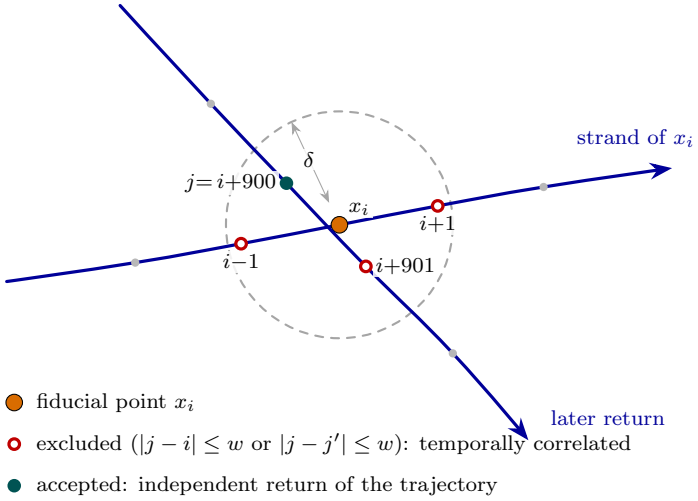



Figure 13: Theiler window and false neighbours, illustrated with two passes (strands) of the same trajectory through the $\delta$-ball of a fiducial point $x_i$. The samples $x_{i\pm1}$ lie inside the ball only because they are adjacent to $x_i$ *in time* ,the trajectory has simply not moved far yet. They are false neighbours: their time twins $y_{i\pm1}$ are automatically close to $y_i$ for the same reason, so counting them would fake evidence for continuity. Rule (i) of Eq. (42) therefore discards every sample within $w$ steps of $i$ (open red markers). The point $j = i+900$ is a genuine neighbour, an independent return of the trajectory into the ball, and is accepted (teal). Its successor $i+901$ lies inside the ball too, but it belongs to the same pass as $j$ and adds no new information; rule (ii) discards it, so that every strand contributes exactly one neighbour.

### 4.2.3 Algorithm

The exact mathematical steps and procedures to get an estimate of the continuity statistic have been outlined so far. Now we provide a pseudocode that performs these calculations. We also provide a list of steps to take care of.

**Algorithm 1** Continuity statistic between paired datasets $\mathscr{D}$ and $\mathscr{R}$

**Require:** Paired time series $\{(\mathbf{d}_i, \mathbf{r}_i)\}_{i=1}^{N}$; $\alpha = 0.02$; $p_\varepsilon = 0.5$; Theiler window $w$; neighbourhood sizes $n_{\min} \le n_\delta \le n_{\max}$
**Ensure:** $\varepsilon^*$, $\varepsilon^*/\varepsilon_{\min}$, $\varepsilon^*/\sigma_{\mathscr{R}}$
$\sigma_{\mathscr{R}} \leftarrow$ RMS distance of $\{r_i\}$ from its centroid
$\varepsilon_{\min} \leftarrow$ mean Theiler-excluded nearest-neighbour distance in $\mathscr{R}$
**for** each fiducial pair $(\mathbf{d}_0, \mathbf{r}_0)$ **do**
    $\mathscr{S} \leftarrow$ the $n_{\max}$ nearest neighbours of $\mathbf{d}_0$ in $\mathscr{D}$, keeping only points more than $w$ time steps from the fiducial and from each other
    $\varepsilon^*_{d_0} \leftarrow \infty$
    **for** $n_\delta = n_{\min}, \ldots, |\mathscr{S}|$ **do**
        $s_{(1)} \le \cdots \le s_{(n_\delta)} \leftarrow$ sorted twin distances $\|r_i - r_0\|$ over first $n_\delta$ points of $\mathscr{S}$
        $n^* \leftarrow \min\{\, n : P_>(n, n_\delta, p_\varepsilon) < \alpha \,\}$ ▷ Eq. (32)
        **if** $n^*$ exists **then**
            $\varepsilon^*_{d_0} \leftarrow \min\big(\varepsilon^*_{d_0},\, s_{(n^*)}\big)$
        **end if**
    **end for**
**end for**
$\varepsilon^* \leftarrow$ mean of all finite $\varepsilon^*_{d_0}$
**return** $\big(\varepsilon^*,\, \varepsilon^*/\varepsilon_{\min},\, \varepsilon^*/\sigma_{\mathscr{R}}\big)$

1. Compute $\sigma_{\mathscr{R}}$, the root-mean-square distance of the points $\{\mathbf{r}_i\}$ from their centroid.
2. Compute $\varepsilon_{\min}$, the mean nearest-neighbour distance in $\mathscr{R}$, excluding neighbours within the Theiler window $w$.
3. Build a $k$-d tree on $\mathscr{D}$ and take each pair $(\mathbf{d}_0, \mathbf{r}_0)$ in turn as the fiducial point.
4. Query the tree for the $n_{\max}$ nearest neighbours of $\mathbf{d}_0$ in $\mathscr{D}$, retaining only those separated by more than $w$ time steps from the fiducial point and from one another. Call the retained set $\mathscr{S}$.
5. For each neighbourhood size $n_\delta = n_{\min}, \ldots, |\mathscr{S}|$, take the first $n_\delta$ points of $\mathscr{S}$ and sort their image distances $\|\mathbf{r}_i - \mathbf{r}_0\|$ in ascending order to give $s_{(1)} \le \cdots \le s_{(n_\delta)}$.
6. Find the smallest rank $n^*$ for which the upper-tail binomial probability satisfies $P_>(n, n_\delta, p_\varepsilon) < \alpha$, with $p_\varepsilon = 0.5$ and $\alpha = 0.02$.
7. If such an $n^*$ exists, record $s_{(n^*)}$ as a candidate value and keep the smallest candidate obtained over all $n_\delta$ as $\varepsilon^*_{d_0}$ for that fiducial point; otherwise discard the fiducial point.
8. Average the finite $\varepsilon^*_{d_0}$ over all fiducial points to obtain $\varepsilon^*$.
9. Report $\varepsilon^*$ together with the normalised ratios $\varepsilon^*/\varepsilon_{\min}$ and $\varepsilon^*/\sigma_{\mathscr{R}}$.

### 4.2.4 Comparison and Numerical Implementation

Our implementation of the continuity statistic is coherent with the one prescribed in [13]. We verify this by taking the system outlined in their paper and implementing it with our code. We also show that the statistics show the same numerical behaviour irrespective of the architecture used (applied to our architecture instead of the architecture outlined in the paper). The specifications of the numerics are first summarised in the table below.

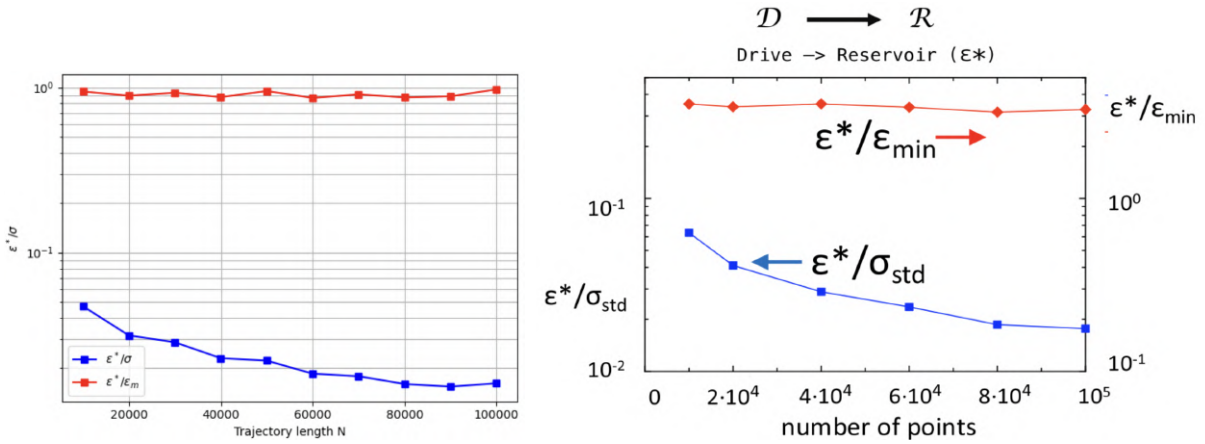
Table 1: Hyperparameters of the continuity statistic. The Theiler window $w_T$ sets the temporal exclusion radius around each fiducial point; $\delta_{\max}$ denotes the maximum neighbourhood size; $T$ is the time-series length.

| **Parameter** | **Value** |
|---|---|
| Number of fiducial points | At least $0.15 \times$ Drive |
| Minimum neighbourhood size | 5 |
| Maximum neighbourhood size | 30 |
| Theiler window | 40 |
| Null-hypothesis probability | 0.5 |
| Significance level | 0.02 |
| *Internal constraints* | |
| Neighbour search breadth | $\delta_{\max} + 4\,w_T + 50$ |
| Sample size for local scale | $\min(2000, T)$ |
| Nearest-neighbour query depth | $2\,w_T + 5$ |

In Pecora et al's [13] work they work with the reservoir state $\mathbf{r}(t)$ such that it is the solution of the differential equation:

$$\frac{dr_i}{dt} = \alpha \left[ \kappa \left( p_1 r_i + p_2 r_i^2 \right) + p_3 r_i^3 + \sum_{j=1}^{N} A_{ij} r_j + w_i x(t) \right]. \tag{43}$$

Our implementation of this RC network with $\kappa = 1$ and the input as the Lorenz system yields the same numerical results as expected in their paper. This numerically verifies the fact that our implementation of their algorithm is valid.




(a) Results obtained using our implementation.

(b) Results reported in the reference paper.

Figure 14: Comparison of the dependence of the continuity length scale $\varepsilon^*$ on the number of trajectory points $N$. Both implementations exhibit the same qualitative scaling. The normalized quantity $\varepsilon^*/\sigma$ decreases as $N$ increases, showing that the absolute neighbourhood size required to establish continuity becomes smaller as the trajectory is sampled more densely. In contrast, $\varepsilon^*/\varepsilon_{\min}$ remains approximately independent of $N$ in both panels.

Additionally, we also applied the same statistic to the architecture of a regular RC model that works on the recursive definition as outlined in 9. This model shows the same scaling as the one obtained for a polynomial RC outlined above. This is indicative of the fact that we can apply our statistics to a variety of RC architectures while being numerically faithful.

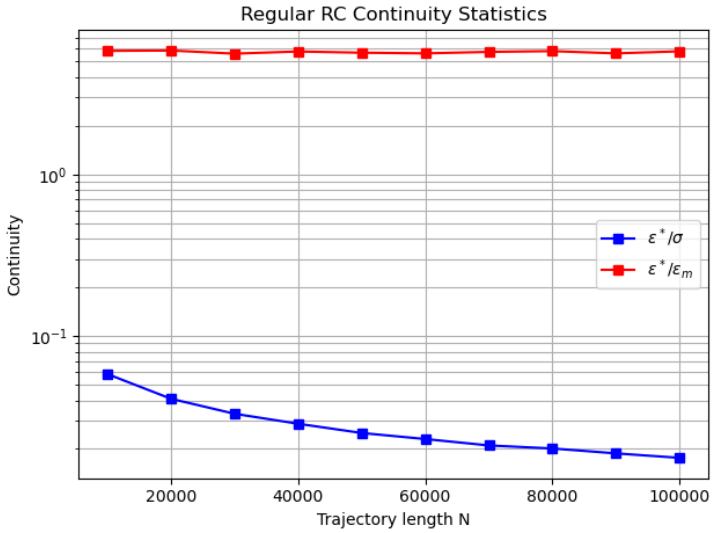



Figure 15: Continuity statistics for the regular reservoir computing (RC) architecture as a function of trajectory length $N$. The results are consistent with those reported in the reference paper.

## 4.3 Differentiability Statistic

The other core aspect of a good embedding is that the map that takes the data (drive) to the reservoir itself is differentiable. A continuous map does not necessarily imply differentiability. A fractal graph, typically seen during the onset of generalised synchronization [38] is a typical example for continuous but non differentiable mapping. If we use a simple analogy, a function is differentiable at a point if, when you zoom in far enough, it looks like a straight line. In higher dimensions, "straight line" becomes "linear map", which in practice means a matrix. So the whole differentiability statistic boils down to: Is there a matrix that describes what this map does in a tiny neighbourhood, and does that matrix actually map the points from one space to another? The statistic tests whether local source secants are mapped to local target secants by a single linear transformation that generalises across the neighbourhood [41]. We perform calculations for this statistic on fiducial points that are seperated by the Theiler window according to the procedures outlined above. This allows us to get rid of any temporal correlations that may be present in the data.

### 4.3.1 Mathematical Framework

A map $\phi$ (between the drive and the reservoir) is differentiable at $\mathbf{m}_i$ (a randomly chosen fiducial point at a given time $t$) if, for a sufficiently small source displacement (translating our chosen point in a vector space) $\mathbf{u}$,

$$\phi(\mathbf{m}_i + \mathbf{u}) - \phi(\mathbf{m}_i) = D\phi(\mathbf{m}_i)\mathbf{u} + \mathscr{O}(||\mathbf{u}||), \tag{44}$$

Where $\mathscr{O}(||\mathbf{u}||) \to 0$ as $||\mathbf{u}|| \to 0$. Once again, this definition relies on the infinitesimal limit. Therefore using the same approach for finite system continuity we outlined in the chapter before, we now try to choose a neighbourhood in the source space and see whether a corresponding time-paired neighbourhood in the target space can be approximated through a linear transformation. The first step is to choose a fiducial point $(\mathbf{m}_i, \mathbf{n}_i)$ where $\mathbf{m}_i \in \mathbb{R}^{d_s}$ source point at time index $i$, and $\mathbf{n}_i \in \mathbb{R}^{d_t}$ target point at time index $i$, the time twin of $\mathbf{m}_i$. $d_s$ and $d_t$ refer to the dimensions of the source and the target respectively. Once this point has been selected, we apply the Theiler, and strand exclusion method outlined in equation 42. Now we utilise a $k$-DTree computational package to calculate and sort points according to the Euclidean distance from the chosen fiducial. Let $j_1, \ldots, j_K$ be the time indices of the neighbours of $\mathbf{m}_i$ that survive the Theiler-window and strand-exclusion filters. The paired source and target secants are hence defined as,

$$\mathbf{u}_\ell = \mathbf{m}_{j_\ell} - \mathbf{m}_i, \qquad \mathbf{v}_\ell = \mathbf{n}_{j_\ell} - \mathbf{n}_i, \qquad \ell = 1, \ldots, K. \tag{45}$$

Note that the neighbour search runs in the source space only. The target secants are not chosen for their proximity to $\mathbf{n}_i$; they are whatever the map happens to return at the same time indices. This is what makes the test directional.

A fiducial is discarded if fewer than four neighbours survive since the differentiability test we outline below needs at least two vectors on each side. The total number of strand-excluded neighbours that exist for a fiducial point are $K$. Once we have the selected these

$K$ neighbours, we split each of the $\mathbf{u}_\ell$ and $\mathbf{v}_l$ into fitting and testing halves. Given $K$ accepted neighbours, set $h = \lfloor K/2 \rfloor$ and define

$$\underbrace{\ell = 1, \ldots, h}_{\text{fitting half}}, \qquad \underbrace{\ell = h+1, \ldots, 2h}_{\text{testing half}}. \tag{46}$$

If $K$ is odd the final neighbour is unused.We now have four sets of secant vectors:

$$\mathbf{u}^{\text{fitting}}_{1\ldots h} \quad \mathbf{u}^{\text{test}}_{h\ldots 2h} \quad \mathbf{v}^{\text{fitting}}_{1\ldots h} \quad \mathbf{v}^{\text{test}}_{h\ldots 2h} \tag{47}$$

Before we talk about a derivative, we must confirm the target and the source are locally the same size, a linear map between spaces of different dimensions can never be invertible. The tool we employ therefore is the singular value decomposition (SVD). SVD provides the principal directions in which our vectors lie, and ranks it according to how much spread there is. For instance, if our vectors form a flat pancake sitting in 3D space, we get two large singular values and one near-zero. This indicates that the geometric structure is really two dimensional rather than three. Secants can be degenerate, they may all lie along a single direction, or the source and target neighbourhoods may simply not be geometrically comparable. Both cases are screened out before any map is fitted. For a matrix $Z$ whose rows are secants, let $s_1(Z) \geq s_2(Z) \geq \cdots \geq s_r(Z) \geq 0$ be its singular values, with $r = \min(\text{number of rows}, \text{number of columns})$ [42]. The local dimension is the smallest number of singular values needed to account for a fraction $\gamma$ of their total,

$$q(Z) = \min\left\{ q : \ \frac{\sum_{a=1}^{q} s_a(Z)}{\sum_{a=1}^{r} s_a(Z)} \ \geq \ \gamma \right\}, \qquad \gamma = 0.98. \tag{48}$$

$\gamma = 0.98$ is chosen on the suggestion of [13]. Because $Z$ and $Z^{\mathsf{T}}$ share singular values, whether secants are stored as rows or columns is immaterial here. Equation (48) is evaluated four times per fiducial, once for each half on each side:

$$q^{\text{fit}}_{\text{s}}, \quad q^{\text{fit}}_{\text{t}}, \quad q^{\text{test}}_{\text{s}}, \quad q^{\text{test}}_{\text{t}}, \tag{49}$$

where the subscript denotes source or target and the superscript denotes the fitting or testing half. The fiducial is discarded if any of the four is zero, which indicates a collapsed neighbourhood. Therefore for the given fiducial point and chosen secants we *must* ensure that:

$$q^{\text{fit}}_{\text{s}} = q^{\text{fit}}_{\text{t}} \qquad \text{and} \qquad q^{\text{test}}_{\text{s}} = q^{\text{test}}_{\text{t}}. \tag{50}$$

The rationale is that a differentiable map with a well-behaved derivative cannot change the local dimension of a neighbourhood; a mismatch signals that the neighbourhood has been collapsed or torn, so the linear fit would be meaningless even if it happened to score well [26]. Now that we have done the dimension consistency check, we fit a linear map on our fitting data outlined in equation 47. Now we collect the fitting secants as columns,

$$X_i = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_h \end{bmatrix} \in \mathbb{R}^{d_{\text{s}} \times h}, \qquad Y_i = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_h \end{bmatrix} \in \mathbb{R}^{d_{\text{t}} \times h}, \tag{51}$$

and take the least-squares solution of $Y_i \approx A_i X_i$,

$$A_i = Y_i X_i^{\mathsf{T}} \left( X_i X_i^{\mathsf{T}} \right)^{+}, \tag{52}$$

where $(\cdot)^+$ is the Moore–Penrose pseudoinverse [43]. The matrix $A_i$ is the numerical approximation of $D\phi(\mathbf{m}_i)$ appearing in Equation (44). Now this "estimate" of the linear map is cross-checked against the secants we deliberately did not use (indices $h...2h = \ell$), and for each one compute the predicted vectors against the real ones.

$$\widehat{\mathbf{v}}_{h+\ell} = A_i\, \mathbf{u}_{h+\ell}, \qquad \ell = 1, \ldots, h, \tag{53}$$

and compared with the target secant that was actually observed. The root-mean-square prediction error at fiducial $i$ is

$$E_i = \sqrt{\frac{1}{h} \sum_{\ell=1}^{h} \|\widehat{\mathbf{v}}_{h+\ell} - \mathbf{v}_{h+\ell}\|_2^2}. \tag{54}$$

This has the units of the target attractor, so it is not comparable across fiducials or across systems. It is therefore normalised by the typical length of the secants it was trying to predict,

$$L_i = \frac{1}{h} \sum_{\ell=1}^{h} \|\mathbf{v}_{h+\ell}\|_2\,, \tag{55}$$

giving the dimensionless local differentiability error

$$D_i = \frac{E_i}{L_i}. \tag{56}$$

Fiducials with $L_i = 0$, meaning the held-out target secants all vanish, are discarded because the ratio is undefined.

**The directional statistic:** Up to $N_f$ fiducial time indices are drawn without replacement, the margins guarantee that a full Theiler window exists on both sides of every fiducial. Writing $\mathscr{F}$ for this sample and $\mathscr{F}^{\mathrm{sel}} \subseteq \mathscr{F}$ for the fiducials that pass the neighbour-count, dimension-matching and normalisation checks, the statistic is the plain average

$$\boxed{D_{\mathscr{M}\to\mathscr{N}} = \frac{1}{\left|\mathscr{F}^{\mathrm{sel}}\right|} \sum_{i \in \mathscr{F}^{\mathrm{sel}}} D_i} \tag{57}$$

The statistic is asymmetric by construction, since neighbourhoods are always sought in the source space, so $D_{\mathscr{M}\to\mathscr{N}}$ and $D_{\mathscr{N}\to\mathscr{M}}$ are computed separately.

**Reading the number:** A small $D_{\mathscr{M}\to\mathscr{N}}$ means a matrix fitted on one set of neighbours also predicts the displacements of neighbours it never saw, which is what a locally differentiable map looks like. A large value means the local linear approximation fails to generalise. A value near or above unity therefore indicates that the fitted map carries no usable information about the local geometry.

**Degenerate outcomes:** Two failure modes are distinguished. If no sampled fiducial can be evaluated at all, because too few valid neighbours are available anywhere,the computation is aborted and the user is directed to lengthen the series, increase $N_f$, lower $K$, or shrink the Theiler window. If fiducials can be evaluated but none satisfies the dimension-matching condition of equation (50), the statistic is returned as $+\infty$, recording that the map failed the geometric precondition everywhere.

### 4.3.2 Algorithm

**Algorithm 2** Differentiability statistic from $\mathscr{M}$ to $\mathscr{N}$

**Require:** Time-paired series $\{(\mathbf{m}_i, \mathbf{n}_i)\}_{i=1}^{T}$; fiducial budget $N_f$; neighbour count $P$; Theiler window $w$; threshold $\gamma = 0.98$
**Ensure:** $D_{\mathscr{M}\to\mathscr{N}}$

Increment $P$ by one if it is odd, and build a KD-tree on $\mathscr{M}$
Draw $\mathscr{F}$: up to $N_f$ time indices, without replacement, from $\{w, \dots, T - w - 1\}$
$\mathscr{D} \leftarrow \emptyset$
**for all** fiducial indices $i \in \mathscr{F}$ **do**
    Retrieve up to $P$ neighbours of $\mathbf{m}_i$ after Theiler and strand exclusion
    **if** fewer than four neighbours survive **then**
        **continue**
    **end if**
    Form the paired secants $\mathbf{u}_\ell, \mathbf{v}_\ell$ by Equation (45)
    Split them into a fitting half and a testing half by Equation (46)
    Compute the four local dimensions using Equation (48)
    **if** any dimension is zero, or Equation (50) fails **then**
        **continue**
    **end if**
    Fit $A_i$ on the fitting half using Equation (52)
    Predict the testing half by Equation (53) and form $E_i$, $L_i$
    **if** $L_i > 0$ **then**
        Append $D_i = E_i/L_i$ to $\mathscr{D}$
    **end if**
**end for**
**if** no fiducial could be evaluated **then**
    **abort**: insufficient neighbours throughout
**else if** $\mathscr{D} = \emptyset$ **then**
    **return** $+\infty$
**end if**
**return** the mean of $\mathscr{D}$

### 4.3.3 Note on Numerical Implementation

The code for measuring topological statistics takes the drive space with the dimensions $T \times N_D$ and the reservoir to be $T \times N_R$ however, while calculating $\tilde{\mathbf{r}}(t)$ for a given hyperparameter subset, we may expect the presence of numerical instabilities and divergence of the reservoir states. The same might be said for the simulation of trajectories for a random initial condition. Therefore to avoid the presence of these numerical inconsistencies, we manually remove the time pairs in the generated $\tilde{\mathbf{r}}(t)$ and simultaneously in $\mathbf{x}(t)$ which is the drive. This leaves us with clean trajectories in both spaces to compare. We also recommend to implement this cleaning only on an ad-hoc basis since for most trajectories except those containing nonlinearities higher than an order of 5, the code was stable numerically.

# 5 Finding the Minimum Nonlinearity

In Figure 11 we found that the prediction quality of an attractor with a nonlinearity order of $\eta_a$ is highly correlated with the minimal nonlinearity $\eta_R$ for the minimal RC architecture. We found that when the nonlinearity of the attractor matches that of the reservoir system, the prediction performance is maximised for the minimal RC architecture. This phenomenon of matching nonlinearity can be viewed as an analogy: When the model's nonlinearity is too low ($\eta_R < \eta_a$), the reservoir cannot fully encode the nonlinear structure of the attractor, leading to a large $\mathscr{D} \to \mathscr{R}$ continuity radius (many drive-space neighbors map to distant reservoir states) and a small $\mathscr{R} \to \mathscr{D}$ radius (the reservoir's limited expressiveness compresses information). When $\eta_R$ is too high, the reservoir introduces spurious nonlinear distortions. At the correct nonlinearity, the mapping is optimally bicontinuous: The reservoir faithfully represents the attractor's topology. Therefore, we hypothesize that the peak in this performance is correlated with the minima of the difference between $\eta_R$ and $\eta_D$ which we call $\Delta_\eta$. We compute the statistic in both directions: $\mathscr{D} \to \mathscr{R}$ and $\mathscr{R} \to \mathscr{D}$. Both must yield a small $\varepsilon^*$ for the mapping to qualify as a homeomorphism. The key quantity we study is the asymmetry,

$$\Delta_\eta = \left| \frac{\varepsilon^*(\mathscr{D} \to \mathscr{R})}{\sigma_{\mathscr{R}}} - \frac{\varepsilon^*(\mathscr{R} \to \mathscr{D})}{\sigma_{\mathscr{D}}} \right|, \tag{58}$$

which measures how far the bidirectional mapping departs from a homeomorphism. When $\Delta_\eta$ is minimised, the mapping is as close to being invertible and continuous in both directions as the finite data allow. In this regard, we hypothesize that for both architectures, minimal RC, and classical RC, the $\Delta_\eta$ quantity should be extremised to match the attractor's minimal nonlinearity. For all synthetic systems, we integrate the trajectories using the explicit Runge–Kutta method of order 5(4) [44] with a step size of $\Delta t = 0.02$ (We use the parameters by [11] as standard values unless specified otherwise), discarding the first $10^4$ steps as transient behavior. The trajectories for the attractor were simulated for $8 \times 10^4$ steps out of which $4.5 \times 10^4$ reservoir states are used for training. We also use 500 datapoints to synchronise the reservoir and a regularization parameter for the ridge regression set at $\beta = 10^{-6}$. For reservoir computing, we use the SCAN software package [19]. For each chaotic system, we sweep the nonlinearity parameter $\eta$ in a minimal RC architecture over a range that includes the known nonlinearity of the system, using fractional exponents with a denominator of 50. We also try changing the readout exponent of a classical RC system. Usually a range between $1 - 5$ is a good range. Each value of $\mathbf{r}^\eta$ is computed as follows:

$$r^\eta = \left(r^d\right)^{\frac{1}{50}}, \quad \text{i.e.} \quad r^2 = \left(r^{100}\right)^{\frac{1}{50}}, \tag{59}$$

where $d$ is referred to as the $\eta$-numerator. The denominator was fixed at 50 to maintain numerical stability as advised by Prosperino et al. [11]. At each value of $\eta$, we:

1. Construct the classical RC setting/minimal RC with the specified $\eta$ and drive it with the input time series to obtain reservoir states. For the case of classical RC, we introduce the nonlinearity in the readout.

2. Compute the continuity statistic for the D→R mapping by finding nearest neighbors in drive space and measuring the corresponding distances in reservoir space.
3. Compute the continuity statistic for the R→D.
4. Record the absolute difference $\Delta_\eta = |\varepsilon^*(\mathrm{D} \to \mathrm{R})/\sigma_R - \varepsilon^*(\mathrm{R} \to \mathrm{D})/\sigma_D|$.

We repeat each experiment across multiple realizations and report the mean and standard deviation. The analysis for the minimum nonlinearity was done on three attractors: the Lorenz attractor, the Halvorsen attractor, and the Thomas attractor. Our results are divided between two architectures: the classical RC with a linear activation function helps determine the minimal nonlinearity required for the system, and then we use sub-reservoirs generated from the minimal RC architecture to determine which coordinates of the attractor are best embedded (to determine the potential underlying equations of the attractor itself).

## 5.1 Classical RC

We take the case of a linearly evolved RC for this section, with a random network as the reservoir. Consider an RC system with $n = 150$, $\rho = 0.99$ and a linear activation function. We evolve $\mathbf{r}(t)$ and augment it to give $\tilde{\mathbf{r}}(t) = [\mathbf{r}, \mathbf{r}^\eta](t)$. We assume that the number of nodes are sufficient for a full embedding and the spectral radius allows for maximal information exchange. Given this setting, we collect data for multiple initial conditions of the attractor and then calculate $\Delta_\eta$. The error bars are represented by shaded areas corresponding to the change in the value of continuity statistic based on initial conditions. The uncertainty in the minimal non linearity is denoted by:

$$\eta^+ = |\mathrm{maxima}(R \to D) - \mathrm{maxima}(D \to R)|\,, \tag{60}$$
$$\eta^- = |\mathrm{minima}(R \to D) - \mathrm{minima}(D \to R)|\,, \tag{61}$$
$$\text{Positive error} = |\eta^+ - \eta| \tag{62}$$
$$\text{Negative error} = |\eta^- - \eta| \tag{63}$$
$$\tag{64}$$

In the equation above, the maxima and the minima refer to the values of the error bars obtained from varying initial conditions. Therefore, the error in the nonlinearity is:

$$\pm\eta = \max\left(\text{Positive error, Negative error}\right). \tag{65}$$

### 5.1.1 Synthetic Systems

We begin with the Lorenz system, whose quadratic nonlinearity ($\eta = 2$) is well established. Figure 16 shows the results. The left panel displays the D→R continuity (red) decreasing monotonically with $\eta$, while the R→D continuity (blue) increases. The two curves cross near $\eta = 2.0$, the known nonlinearity of the system. The right panel shows $\Delta_\eta$, which displays a clear V-shaped minimum at $\eta = 2.00$, precisely matching the quadratic nonlinearity of the Lorenz equations.

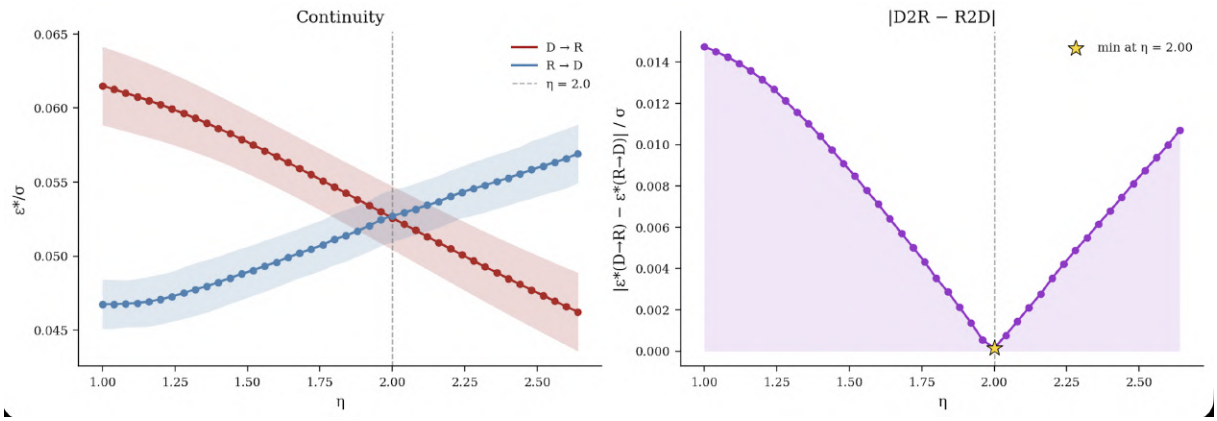



Figure 16: Two-way continuity for the Lorenz system $\rho^* = 0.90$.Left: $\varepsilon^*/\sigma$ for D→R (red) and R→D (blue) as a function of $\eta$. Shaded regions indicate one standard deviation. Right: $\Delta_\eta = |\varepsilon^*(\mathrm{D} \to \mathrm{R}) - \varepsilon^*(\mathrm{R} \to \mathrm{D})|/\sigma$. The minimum (star) occurs at $\eta = 2.00$, matching the quadratic nonlinearity. The dashed line marks $\eta = 2.0$.

From the graphs we infer $\eta^+ = 2.08, \eta^- = 1.88$ which give:

$$\boxed{\eta_{\text{Lorenz}} = 2.00 \pm 0.12.} \tag{66}$$

Now to confirm whether this result was a direct consequence of the topology or not, we use a $p$-value test. The test follows the method of surrogate data [45]: A *surrogate* is an artificial time series constructed to be consistent with a stated null hypothesis while being stripped of the property under investigation. It is obtained by randomising the measured signal in a way that preserves everything the null hypothesis is entitled to explain and destroys everything it is not, so that any statistic which separates the data from its surrogates cannot be accounted for by the null. Here the null hypothesis is that the drive is nothing more than a linearly correlated stochastic process, and the randomisation is the Fourier-transform (FT) surrogate: Each component of the Lorenz training trajectory is mapped to the frequency domain, its Fourier amplitudes $|\tilde{x}_k|$ are retained exactly. Then we randomize the Fourier phases as $\hat{x}_k \mapsto |\hat{x}_k| e^{i\varphi_k}$ with $\varphi_k \sim \mathscr{U}[0, 2\pi)$ i.i.d. for $1 \leq k < \lceil N/2 \rceil$, $\varphi_0 = \varphi_{N/2} = 0$, and $\varphi_{N-k} = -\varphi_k$ otherwise, preserving the power spectrum while destroying nonlinear structure. For even $N$, the surrogate is real-valued, and the signal is returned to the time domain by an inverse transform. Because the amplitude spectrum is left untouched, the power spectrum and hence—by the Wiener–Khinchin theorem [46]—the autocorrelation function of the surrogate are identical to those of the Lorenz signal; what is destroyed is the phase coherence that binds the Fourier modes into a deterministic trajectory. The surrogate is therefore a signal with the linear statistics of the Lorenz system but with no strange attractor, and consequently no attractor topology. Each of the $N_s = 500$ surrogate realisations is then driven through exactly the same pipeline as the true data: it feeds a reservoir of identical size ($N = 120$), spectral radius ($\rho = 0.99$), input matrix, network seed and regularisation, and the two continuity statistics $\varepsilon_{D\to R}(\eta)$ and $\varepsilon_{R\to D}(\eta)$ are recomputed over the same grid $\eta \in [1.00, 5.04]$, so that the surrogate minimiser $\eta_s^\star = \arg\min_\eta |\varepsilon_{D\to R}(\eta) - \varepsilon_{R\to D}(\eta)|$ is extracted precisely as before.

The 500 values of $\eta_s^\star$ constitute the empirical null distribution against which the observed $\eta^\star$ is assessed, and the Monte-Carlo $p$-value is

$$p \;=\; \frac{1 + \#\{\, s : |\eta_s^\star - \bar{\eta}_s^\star| \geq |\eta^\star - \bar{\eta}_s^\star| \,\}}{1 + N_s}, \tag{67}$$

where the $+1$ in numerator and denominator guarantees the correct type-I error rate for a test based on a finite ensemble [47]. Under the null the minimiser is tightly concentrated, $\eta_s^\star = 2.63 \pm 0.12$ (mean $\pm$ s.d.), confined to the interval $[2.36, 2.96]$ with 95% of its mass in $[2.40, 2.86]$. A small $p$ thus states that no reservoir driven by a signal carrying the Lorenz power spectrum but no Lorenz topology reproduces the observed balance point, so that $\eta^\star$ cannot be an artefact of the second-order statistics of the drive, nor of the reservoir architecture, which were held fixed throughout; it must instead be attributed to the topology of the attractor itself.

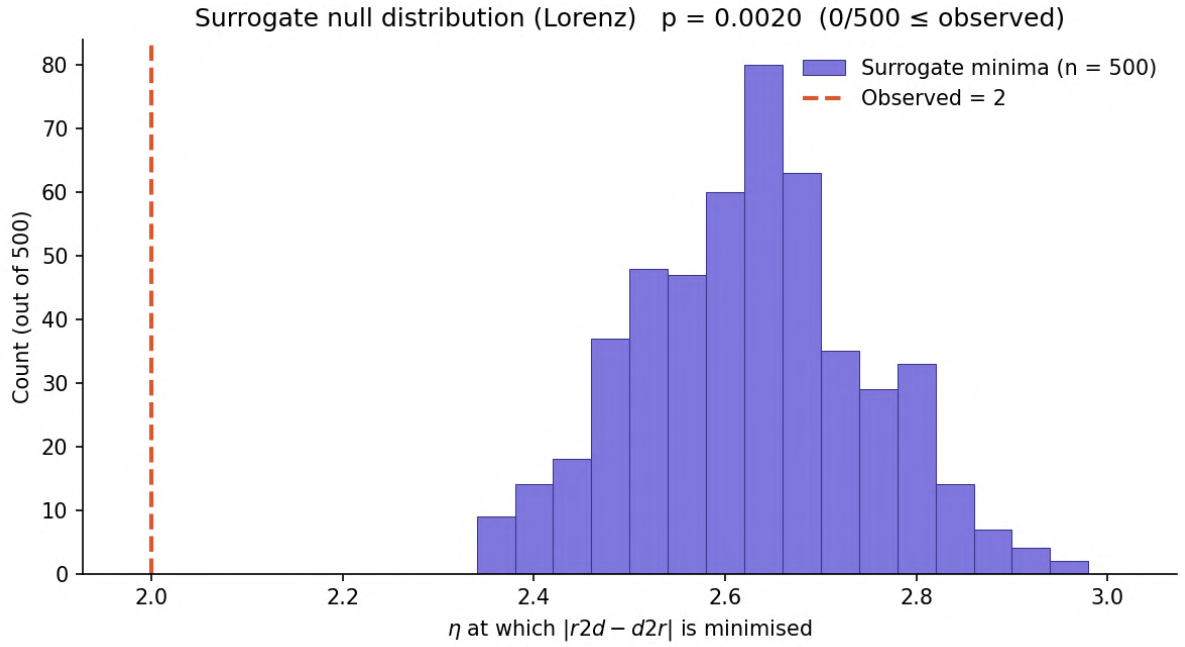



Figure 17: As seen in the graph, the surrogates show much higher nonlinearities than the true attractor would show, and hence it is safe to assume that it is indeed the attractor nonlinearity that gives rise to the shape of the curve.

The surrogate systems indicate that there is a bias in a single directional mapping, which means that the attractor looks more distorted than it actually is as a consequence of making surrogates. This is the reason the non linearity is higher than the true non linearity of the system. Now we can also see the nonlinearity of other systems. For the classical Halvorsen system, we observe the same qualitative pattern. The best embedding occurs at

$$\boxed{\eta_{\text{Halvorsen}} = 2.12 \pm 0.24}, \tag{68}$$

which is close to the actual nonlinearity of 2. The V-shaped minimum is robust and clearly identifiable in both cases.

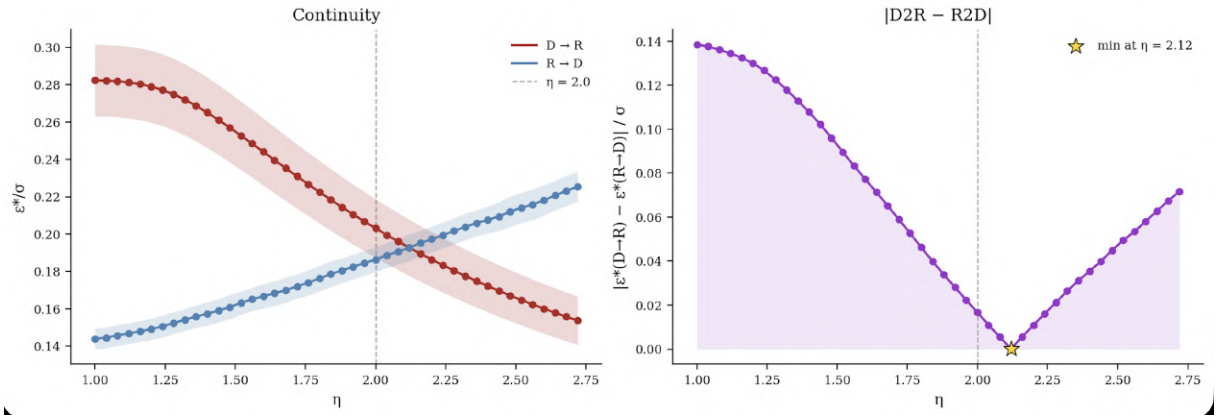



Figure 18: Classical RC setup for the Halvorsen system gives a nonlinearity of $\eta = 2.12$.

The Thomas system presents an interesting case because its nonlinearity resides in the sine function, whose first nonlinear Taylor term is cubic. For the classical RC setup, we find the nonlinearity resides at

$$\boxed{\eta_{\text{Thomas}} = 3.08 \pm 0.24} \tag{69}$$

Figure 19 which is very near to the actual nonlinearity of 3. This means that the reservoir states require slightly more than cubic data to embed the attractor. This is evident since the sine function is an infinite series.

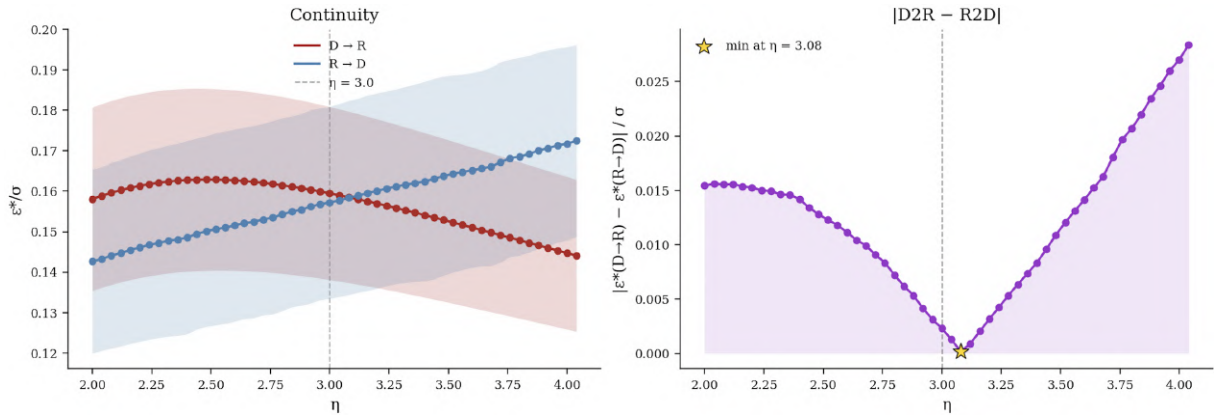



Figure 19: The Thomas attractor's continuity statistics for classical RC setup assuming a Taylor expansion of the sine function.

This indicates that the reservoir has the capacity to embed the attractor, and the matching of $\eta$ makes it possible for the attractor to be unfolded appropriate to its scale.

### 5.1.2 Real Datasets

We apply the continuity analysis to three financial time series: the MSCI World Index, the S&P 500, the STOXX Europe 600. All use $\rho^* = 0.99$ and $N = 150$ nodes. We found the best match in the SP500 data and MSCI world data with the exponents outlined in the paper by Prosperino et al. [11].

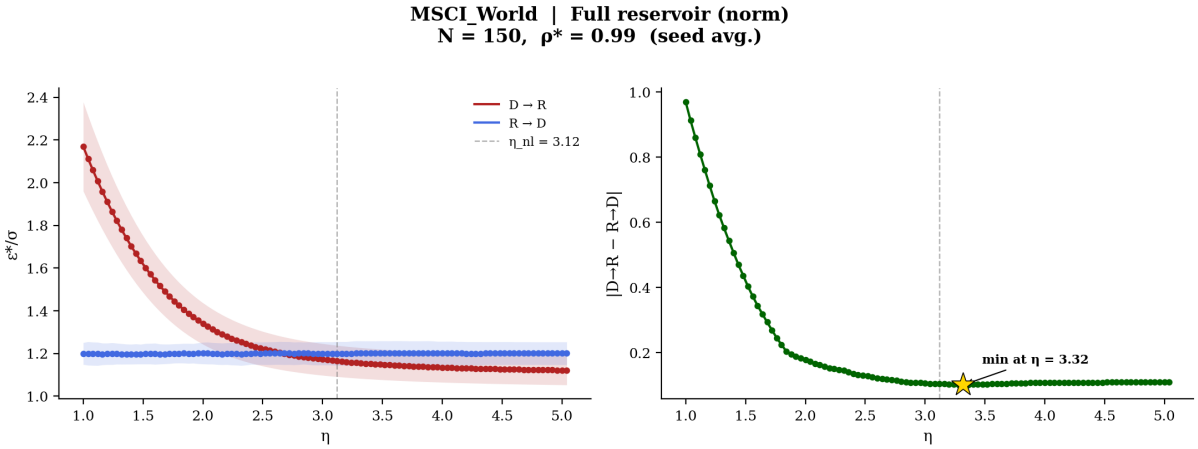



Figure 20: The regular RC continuity shows a minima at 3.32 for the MSCI World data

For the **MSCI World Index**, using the regular RC setup, the $\Delta(\eta)$ minimum occurs at $\eta = 3.32$ compared to The prediction-based estimate from Prosperino *et al.* which was $\eta = 3.12$. From the perspective of the S&P500 dataset, the continuity approach yields an $\eta = 1.52$ which is very close to the estimate from Prosperino *et al.* who found $\eta = 1.64$,

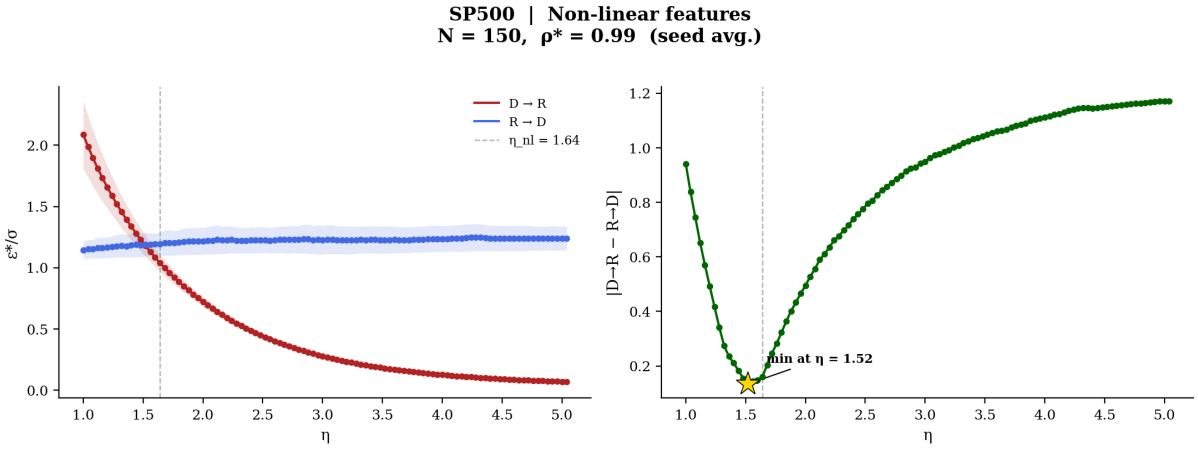



Figure 21: For the S&P500 data the best embedding occurs at $\eta = 1.52$. For this dataset, only the nonlinear part of the reservoir was taken (since the higher dimensional full reservoir always erased the nonlinearity signal)

### 5.1.3 AMOC climate data

Prosperino *et al.* [11] reported that their prediction-based method failed to detect nonlinearity in the AMOC dataset [48]. We apply our continuity statistic to the same data. Interestingly, we find an almost quadratic nonlinearity in the dataset with a minimum at $\eta = 2.2$.These results suggest that the continuity statistic may be more sensitive than the prediction-based approach for detecting nonlinear structure in short, noisy time series. The AMOC result is particularly noteworthy because it provides evidence of nonlinear structure in climate data where the prediction-based method was unable to draw conclusions, though we caution that the short length of the AMOC time series warrants careful interpretation.

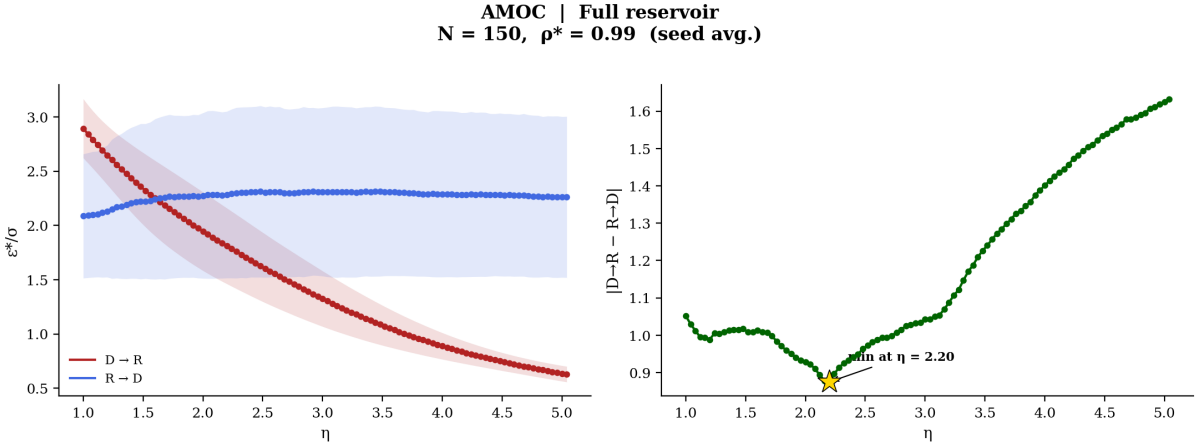



Figure 22: Two-way continuity for AMOC data gives minimum at $\eta = 2.2$.

The large error bars allude to the idea that this architecture is very sensitive to the adjacency matrix of the reservoir and therefore indicates that the AMOC data may not have a directly inferable non linearity. In fact, the error bars also make the data linear, which again makes the results unreliable. The error bars go from $\eta^- = 2.16, \eta^+ = 3.36$ and the nonlinearity is:

$$\boxed{\eta_{\text{AMOC}} = 2.20 \pm 1.16} \tag{70}$$

Note: There are several things that need to be said here. Primarily, predicting the nonlinearity of the dataset for real life datasets in no capacity involves being able to accurately predict their future. There still remains immense difficulty in being able to predict their curves. The number of data points in each data are different and hence the statistic may have higher sensitivity. Additionally, the results for STOXX 600 data are not shown. They did not converge, indicating that the data may not be sufficiently processed or that it may genuinely have a nonlinearity above 5. In this analysis we assume that there exists underlying attractors, and 150 nodes are enough to embed this assumed attractor.

## 5.2 Minimal RC

The advantage of the Minimal RC architecture is the fact that we can seperate coordinates and create subreservoirs. The block size was fixed at $b = 3$ with a spectral radius of $\rho = 0.101$. The following subreservoirs were constructed to check the coordinate spaces within which the best embedding occurs:

- the *full reservoir* $\tilde{\boldsymbol{r}}$,
- only the *linear part* $\boldsymbol{r}$,
- only the *nonlinear part* $\boldsymbol{r}^{\eta}$,
- the *base coordinates* $(x, x^{\eta}, y, y^{\eta}, \ldots)$,
- the *sum features* $(x{+}y, x{+}z, y{+}z$, and their powers$)$,
- the *sums without* the three-variable sum $(x{+}y{+}z)$.

This decomposition reveals which structural elements of the reservoir are responsible for encoding the data's nonlinearity.

### 5.2.1 Lorenz System

Now to probe the system further one can ask, can we gain additional information about the underlying equations of the attractor? To answer this question we take aid of the minRC architecture outlined above. We divide the reservoir states into subreservoirs, each corresponding to a different combination of the base coordinates. In the case of the Lorenz attractor we see the results outlined in Figure 23. The Lorenz attractor, compared to the other attractors outlined in the study, gives less information in its subreservoirs. For instance, from the subreservoir analysis, we can infer that the all terms with nonlinearity are important, but the main terms that actually give rise to the attractor are its $x_1x_3$ and $x_2x_1$, which should embed the "two-sums" features rather than all. This is to say that the subdivisions are not exact in matching the underlying equations, but prescriptive.

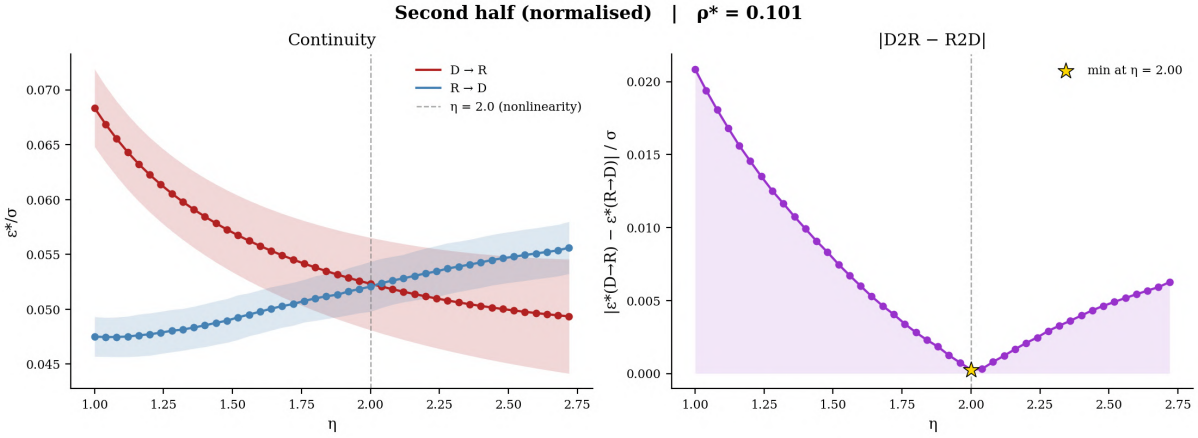



Figure 23: The subreserve with just the non linear parts $\mathbf{r}^{\eta}$ that embeds the Lorenz system the best is the subreservoir that which has only nonlinear terms.

The nonlinear part only with the error bars gives us the nonlinearity at $\eta = 2.00 \pm 0.25$.

### 5.2.2 Halvorsen System

If we look at the subreservoir setup, we notice that the best embedding occurs with the full reservoir as well as the reservoir with the base coordinates both of which occur at non linearities of 1.96 and 1.84 respectively. This indicates that the base coordinates $x_n, x_n^2$ hold most importance. This matches the underlying structure of the Halvorsen equation which is only constructed out of combining $x_n$ and $x_n^2$ terms. Using base coordinates, the $\Delta_\eta$ minimum occurs at $\eta = 1.96$, close to the true nonlinearity of 2 (Fig. 24). With the full reservoir, the minimum shifts slightly to $\eta = 1.84$, which we attribute to the additional sum features introducing mixed effects. The V-shaped minimum is robust and clearly identifiable in both cases.

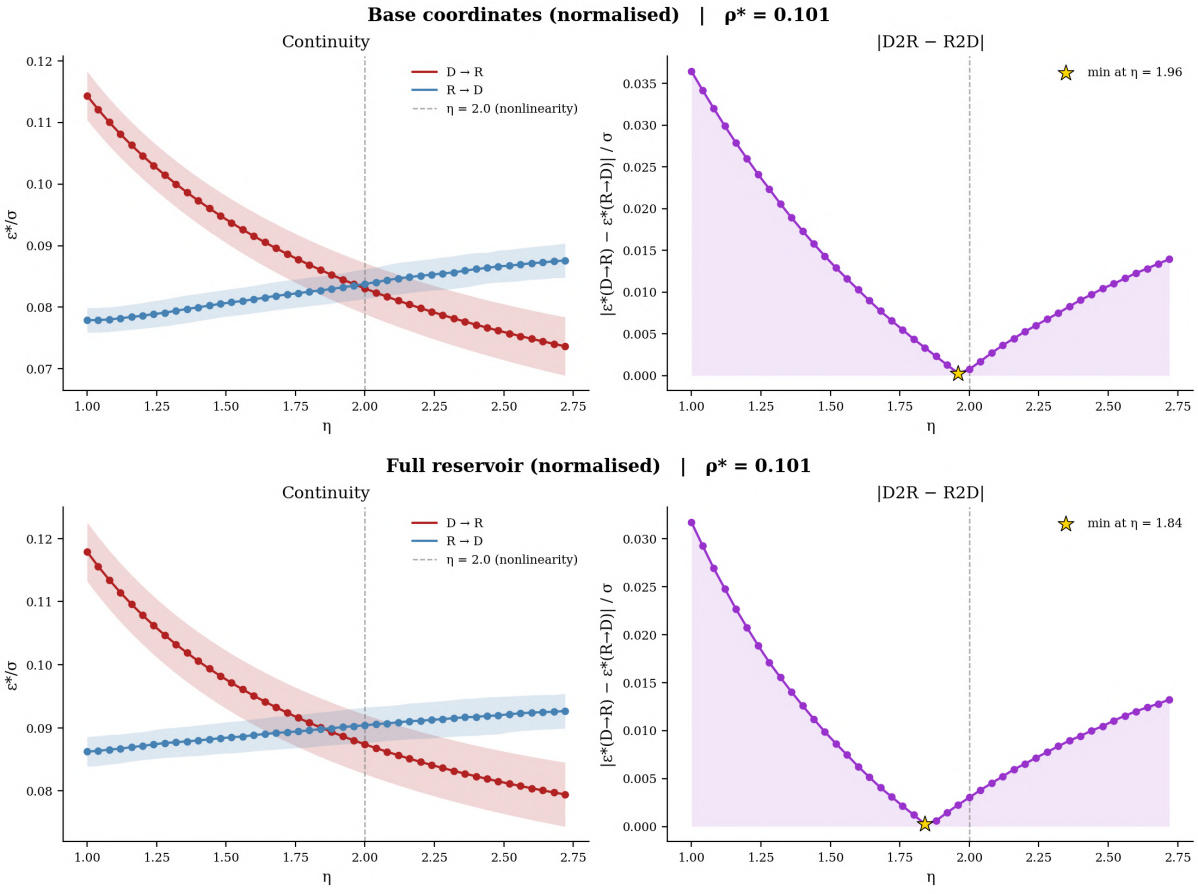



Figure 24: MinRC shows best embedding occurs for the base coordinates for the Halvorsen system.

The errror bars suggest that the base coordinates give a nonlinearity of $1.96 \pm 0.16$ and for the full reservoir, $1.84 \pm 0.25$.

### 5.2.3 Thomas System

The Thomas attractor equations are majorly influenced by summation terms of two base coordinates such as $x_1 + x_3^3$ combinations. This is why we expect the best embedding to occur in the base coordinates and the sum coordinates. Our results corroborate this idea with $\eta = 3.2$ in the two-sum subset and $\eta = 2.84$ in the base coordinate section as indicated by Figure 25. We note that the Thomas system's D→R curve is nearly flat in the base-coordinates configuration, while R→D shows a sharp increase beyond $\eta \approx 3$. This is consistent with the weak nonlinearity of the Thomas system (largest Lyapunov exponent $\lambda \approx 0.01$), which makes the drive-to-reservoir mapping relatively insensitive to $\eta$. It is also important to note that as we move higher in the exponent of the system, it becomes harder to isolate the attractor using just the sub divisions we indicate since $(x + y)^4$ has much more mixing between different coordinates than $(x + y)^2$.

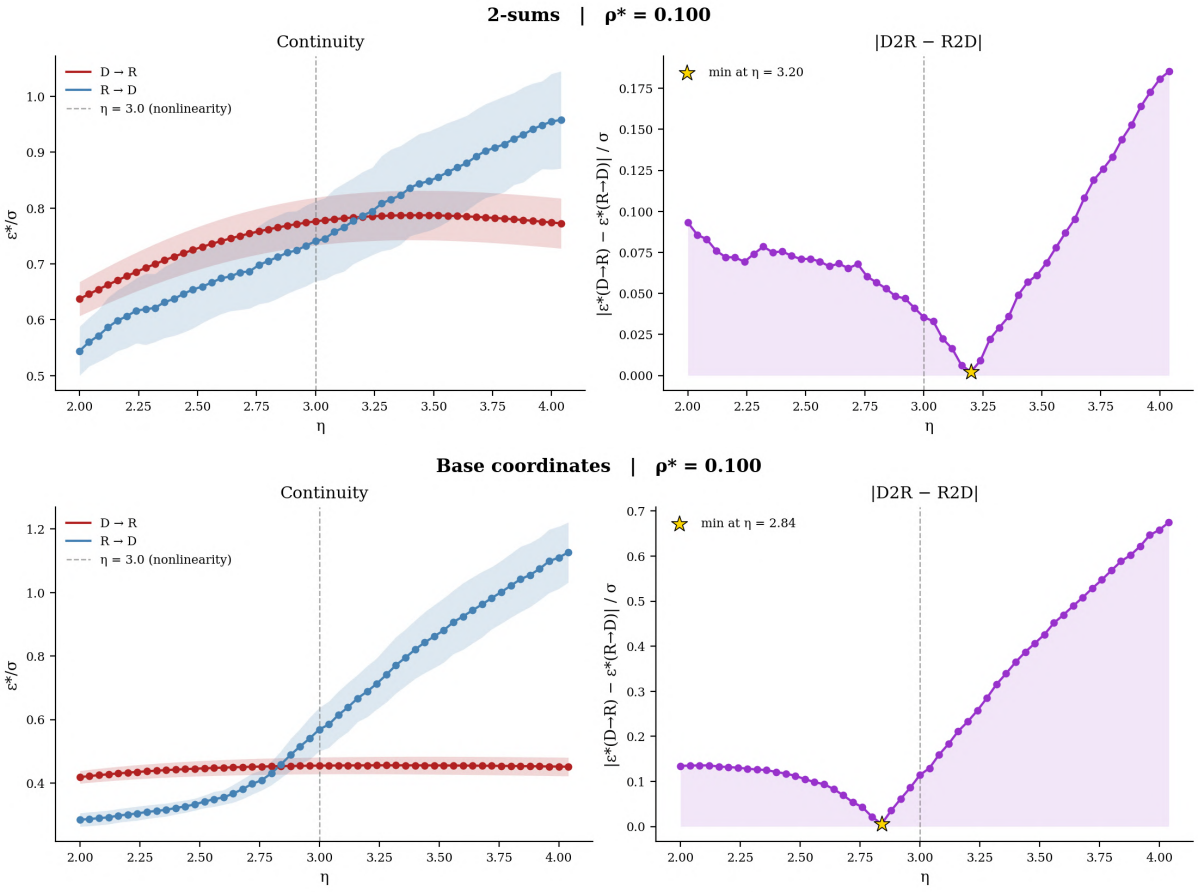



Figure 25: MinRC shows best embedding occurs for the base coordinates as well as the sum coordinates for the Thomas system.

From the error bars we find that for the 2 sums the nonlinearity is $3.20 \pm 0.30$,whereas for the base coordinates only the error is smaller at $2.84 \pm 0.08$.

## 5.3 Usage Prescription

Before we assume that this method is a full proof method to extract the nonlinearity of a system, I must clarify certain caveats. Primarily, unlike the prediction based approach, this method is very sensitive to certain hyperparameters such as the spectral radius, and the adjacency matrix. Therefore, it requires averaging over large values before making any valuable conclusions. We prescribe this method as a way to explain why RC works rather than a hyperparameter tuning method for the nonlinearity. At the same time, the method warrants a deeper study and proposes a need for a better methodology. The nonlinearities classified correctly were definite consequences of the embedding phenomena.

We prescribe using the method in [11] to confirm the nonlinearity suggested by this method. This method we suggest has more heuristic elements that can be fine tuned further and be made more scientifically robust.

# 6 Multifunctionality

The idea of multifunctionality finds its roots in neuroscience [49]. In the brain, the neuron is the fundamental unit of computation, and scientists showed that the same neuron can learn multiple behavioral patterns and perform tasks together [50, 51]. The Miller Lab [49] has shown that cortical neurons can be multifunctional. A neuron does not respond only to one variable, such as the stimulus or the task rule. Instead, it may respond to a nonlinear combination of several variables, such as the stimulus, context, memory, and required action. This creates a high-dimensional representation of the task. The benefit is that different task conditions become easier to distinguish, and a simple downstream readout can extract the appropriate response.

In dynamical systems context, this behaviour is categorised as "multistability", the ability of containing multiple stable systems within the same trained reservoir [52]. There are two modes in which multifunctionality has been realised in RC architectures. In [31] the reservoir state itself has a symmetry due to which we find the presence of multiple mirror attractors in the reservoir that were not trained for. Contrary to this, we can also embed multifunctionality by design to predict multiple strange attractor trajectories under the same reservoir. This is possible exactly because of the reservoir being a higher dimensional space (as argued by Miller's Lab for the human brain). These RC architectures find their realisation in [10, 53].

Using a single reservoir for simulating multiple attractors is quite hyperparameter-intensive, requiring a sensitive adjustment to achieve [54]. Therefore there exists a significant need for a diagnostic tool (or statistic) that can aid in hyperparameter selection.

In this chapter we develop a novel framework that uses the continuity statistic as a diagnostic measure to tune the offset parameters.

## 6.1 Architectures

A multifunctional reservoir is not neccesarily reservoir that has learned two tasks. In our architectural choice, it is a reservoir that has learned *one* map, evaluated on two disjoint regions of its own state space. Consider the case of wanting to use the same reservoir for two attractors. For $K = 2$, Let

$$\boldsymbol{x}_t^{(k)} \in \mathbb{R}^D, \qquad k = 1, \ldots, K, \tag{71}$$

denote a trajectory from the $k$th dynamical system. Each system is standardized independently,

$$\widetilde{\mathbf{x}}_{t,j}^{(k)} = \frac{\mathbf{x}_{t,j}^{(k)} - \mu_j^{(k)}}{\sigma_j^{(k)}}, \tag{72}$$

In the equation 72, $\sigma_j^{(k)}$ is the standard deviation of the training data (drive). The standardization must occur before training due to the fact that these attractors may operate on different scales with their centers also being on different coordinates. As a direct consequence of normalising two different drive systems we may have a situation where two points from two different attractors may be seperated in their true coordinate space but overlap post normalising.

$$\tilde{x}_A(t) \approx \tilde{x}_B(t) \quad \text{but} \quad x_A(t) \not\approx x_B(t) \tag{73}$$

This overlapping of attractors may translate to overlapping singular states in the reservoir space where the nonlinear readout cannot distinguish the two attractors. Therefore there exists the need for an offset parameter for a multifunctional reservoir. The central idea is to assign a fixed vector $\boldsymbol{c}_k \in \mathbb{R}^N$ to each system, where $N$ is the reservoir dimension. The vector $\boldsymbol{c}_k$ is injected at every reservoir update. It therefore acts as a recurring signal that shifts the embedded $k$th attractor in reservoir space and avoids overlap. The training weights and the readout still remains shared. For the minimal RC system, we use the transformation,

$$\boxed{\mathbf{r}_k(t+1) = \mathbf{A}\mathbf{r}_k(t) + \mathbf{W}_{\text{in}}\tilde{\mathbf{x}}(t) + \mathbf{c}_k} \tag{74}$$

For each attractor we seperately generate the reservoir states and then vertically stack them to obtain,

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}_1 \\ \boldsymbol{R}_2 \\ \vdots \\ \boldsymbol{R}_K \end{bmatrix}, \qquad \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y}_1 \\ \boldsymbol{Y}_2 \\ \vdots \\ \boldsymbol{Y}_K \end{bmatrix}. \qquad \boldsymbol{Y}_k = \begin{bmatrix} (\widetilde{\boldsymbol{x}}_{s+1}^{(k)})^{\mathsf{T}} \\ (\widetilde{\boldsymbol{x}}_{s+2}^{(k)})^{\mathsf{T}} \\ \vdots \\ (\widetilde{\boldsymbol{x}}_{T_k-1}^{(k)})^{\mathsf{T}} \end{bmatrix} \tag{75}$$

A single matrix $\boldsymbol{W}_{\text{out}}$ is obtained by ridge regression,

$$\boldsymbol{W}_{\text{out}} = \arg\min_{\boldsymbol{W}} \left\{ \sum_{k=1}^{K} \left\| \boldsymbol{Y}_k - \boldsymbol{R}_k \boldsymbol{W}^{\mathsf{T}} \right\|_F^2 + \lambda \|\boldsymbol{W}\|_F^2 \right\}, \tag{76}$$

$\boldsymbol{W}$ is the dummy variable which looks at different values of $\boldsymbol{W}_{\text{out}}$ and gives the most optimised one. We find the optimised version to be:

$$\boldsymbol{W}_{\text{out}} = \boldsymbol{Y}^{\mathsf{T}} \boldsymbol{R} \left( \boldsymbol{R}^{\mathsf{T}} \boldsymbol{R} + \lambda \boldsymbol{I} \right)^{-1}. \tag{77}$$

This vertical stacking is the defining training operation.

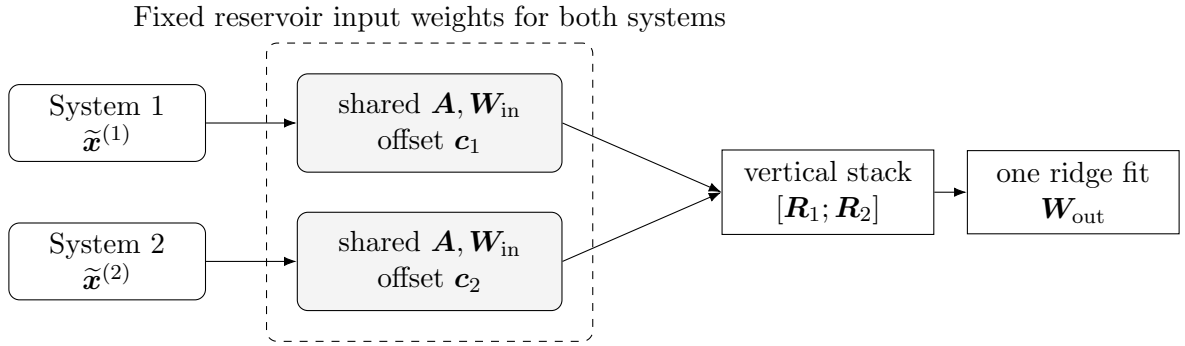



Figure 26: Two-system multifunctional training. Each trajectory is driven through the same reservoir weights but with a different persistent offset. The resulting feature matrices are vertically stacked and one shared readout is fitted. Each system is later predicted separately by selecting its offset and synchronization data.

To show the role of the two offsets in the system we take the equations for the reservoir update,

$$\boldsymbol{r}_1(t+1) = \boldsymbol{A}_{\text{m}}\boldsymbol{r}_1(t) + \boldsymbol{W}_{\text{in}}\widetilde{\boldsymbol{x}}^{(1)}(t) + a\mathbf{1}_N, \tag{78}$$

$$\boldsymbol{r}_2(t+1) = \boldsymbol{A}_{\text{m}}\boldsymbol{r}_2(t) + \boldsymbol{W}_{\text{in}}\widetilde{\boldsymbol{x}}^{(2)}(t) + b\mathbf{1}_N. \tag{79}$$

The subscript $m$ denotes the fact that we are discussing minimal reservoir computing. Note that the two input streams $\widetilde{\boldsymbol{x}}^{(1)}$ and $\widetilde{\boldsymbol{x}}^{(2)}$ are *different* (Lorenz and Rössler, say), so we cannot simply subtract Equation (79) from Equation (78) and hope the inputs disappear. For each system, let $\boldsymbol{s}_k(t)$ denote the state of the *same* reservoir, driven by the *same* input, but with the offset switched off.

$$\boldsymbol{s}_k(t) = \boldsymbol{A}_{\text{m}}\boldsymbol{r}_k(t) + \boldsymbol{W}_{\text{in}}\widetilde{\boldsymbol{x}}^{(k)}(t)$$

Now we define the displacement caused by the offset,

$$\boldsymbol{h}_k(t) := \boldsymbol{r}_k(t) - \boldsymbol{s}_k(t). \tag{80}$$

Subtracting the two update rules, the input term appears identically in both and cancels:

$$\boldsymbol{h}_k(t+1) = \underbrace{\boldsymbol{A}_{\text{m}}\boldsymbol{r}_k(t) + \boldsymbol{W}_{\text{in}}\widetilde{\boldsymbol{x}}^{(k)}(t) + \boldsymbol{c}_k}_{\boldsymbol{r}_k(t+1)} - \underbrace{\left[\boldsymbol{A}_{\text{m}}\boldsymbol{s}_k(t) + \boldsymbol{W}_{\text{in}}\widetilde{\boldsymbol{x}}^{(k)}(t)\right]}_{\boldsymbol{s}_k(t+1)} = \boldsymbol{A}_{\text{m}}\boldsymbol{h}_k(t) + \boldsymbol{c}_k, \tag{81}$$

with $\boldsymbol{h}_k(0) = \mathbf{0}$. Nothing about the driving signal survives in Eq. (81): $\boldsymbol{h}_k$ is precisely the part of the reservoir state that the offset is responsible for. Set $\boldsymbol{\Delta}(t) := \boldsymbol{h}_1(t) - \boldsymbol{h}_2(t)$, the offset-induced separation at time $t$. Subtracting Equation (81) for $k = 2$ from the same equation for $k = 1$, and using $\boldsymbol{c}_1 - \boldsymbol{c}_2 = (a - b)\mathbf{1}_N$,

$$\boxed{\boldsymbol{\Delta}(t+1) = \boldsymbol{A}_{\text{m}}\boldsymbol{\Delta}(t) + (a-b)\mathbf{1}_N, \qquad \boldsymbol{\Delta}(0) = \mathbf{0}.} \tag{82}$$

The whole problem is now this single recursion with a single constant forcing term. Starting from $\boldsymbol{\Delta}(0) = \mathbf{0}$ and applying Equation (82) repeatedly,

$$\begin{aligned}
\boldsymbol{\Delta}(1) &= (a-b)\mathbf{1}_N, \\
\boldsymbol{\Delta}(2) &= \boldsymbol{A}_{\mathrm{m}}(a-b)\mathbf{1}_N + (a-b)\mathbf{1}_N = (a-b)\big(\boldsymbol{I} + \boldsymbol{A}_{\mathrm{m}}\big)\mathbf{1}_N, \\
\boldsymbol{\Delta}(3) &= \boldsymbol{A}_{\mathrm{m}}(a-b)\big(\boldsymbol{I} + \boldsymbol{A}_{\mathrm{m}}\big)\mathbf{1}_N + (a-b)\mathbf{1}_N = (a-b)\big(\boldsymbol{I} + \boldsymbol{A}_{\mathrm{m}} + \boldsymbol{A}_{\mathrm{m}}^2\big)\mathbf{1}_N,
\end{aligned} \tag{83}$$

so that

$$\boldsymbol{\Delta}(t) = (a-b)\sum_{s=0}^{t-1} \boldsymbol{A}_{\mathrm{m}}^s \mathbf{1}_N. \tag{84}$$

Every row of the minimal reservoir matrix sums to $\rho$, since $\boldsymbol{A}_{\mathrm{m}} = \rho\boldsymbol{P}$ with $\boldsymbol{P}$ a permutation matrix. Therefore

$$\big(\boldsymbol{A}_{\mathrm{m}}\mathbf{1}_N\big)_i = \sum_{j=1}^{N}\big(\boldsymbol{A}_{\mathrm{m}}\big)_{ij} = \rho \quad \text{for every } i, \qquad \text{that is} \qquad \boldsymbol{A}_{\mathrm{m}}\mathbf{1}_N = \rho\mathbf{1}_N. \tag{85}$$

Applying this repeatedly,

$$\boldsymbol{A}_{\mathrm{m}}^2\mathbf{1}_N = \boldsymbol{A}_{\mathrm{m}}\big(\rho\mathbf{1}_N\big) = \rho\big(\boldsymbol{A}_{\mathrm{m}}\mathbf{1}_N\big) = \rho^2\mathbf{1}_N, \qquad \text{and in general} \qquad \boldsymbol{A}_{\mathrm{m}}^s\mathbf{1}_N = \rho^s\mathbf{1}_N. \tag{86}$$

Substituting into Equation (84) turns a sum of matrices into a sum of scalars,

$$\boldsymbol{\Delta}(t) = (a-b)\big(1 + \rho + \rho^2 + \cdots + \rho^{t-1}\big)\mathbf{1}_N. \tag{87}$$

Since this is a geometric progression with ratio $\rho$ we can use the summation of the series. This gives the exact separation at any finite time,

$$\boldsymbol{\Delta}(t) = (a-b)\,\frac{1-\rho^t}{1-\rho}\,\mathbf{1}_N. \tag{88}$$

Since $\rho < 1$ we have $\rho^t \to 0$ as $t \to \infty$, and therefore

$$\boxed{\Delta\boldsymbol{r}_{\mathrm{offset}} = \lim_{t\to\infty}\boldsymbol{\Delta}(t) = \frac{a-b}{1-\rho}\,\mathbf{1}_N} \tag{89}$$

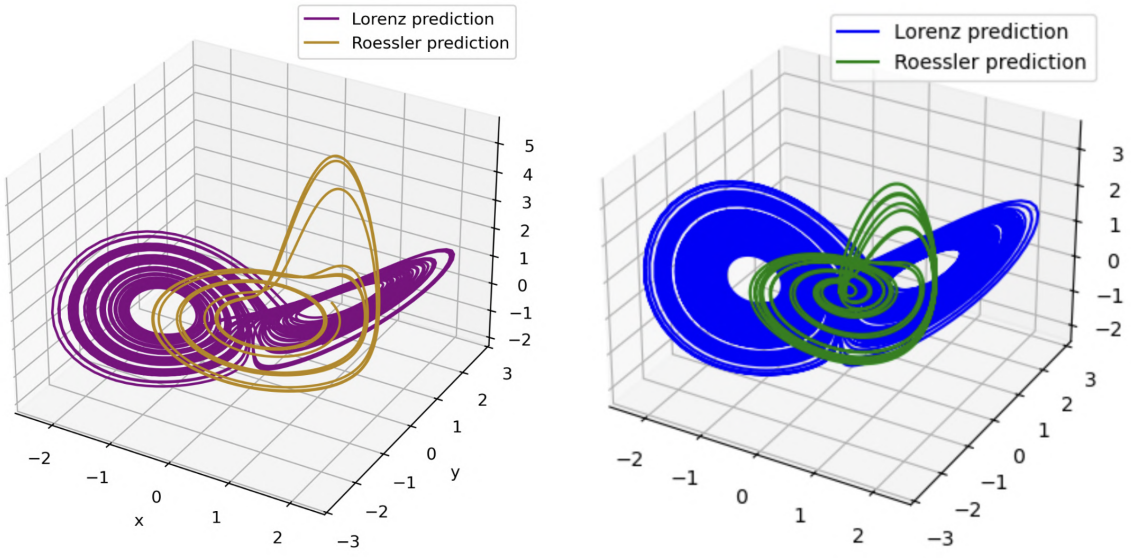
This proves that the offsets are responsible for separating the reservoir states, allowing the possibility of a multifunctional reservoir.

The two driven attractors are not exact translations of each other because their input trajectories differ. Nevertheless, Equation (89) shows exactly how two scalars give multifunctional reservoirs. In the case of Normal RC we obtain two options for $r_k$:

$$\begin{aligned}
\mathbf{r}_k(t+1) &= \tanh\left(\mathbf{A}\mathbf{r}_k(t) + \mathbf{W}_{\mathrm{in}}\tilde{\mathbf{x}}(t)\right) + \mathbf{c}_k \\
\mathbf{r}_k(t+1) &= \tanh\left(\mathbf{A}\mathbf{r}_k(t) + \mathbf{W}_{\mathrm{in}}\tilde{\mathbf{x}}(t) + \mathbf{c}_k\right)
\end{aligned}$$

Both these equations remain valid options for our multifunctional toolkit, however, the use of scalar offsets within the hyperbolic function leads to a saturation for high offset values, making it less interpretable. Due to the presence of the nonlinearity, the seperation parameters for optimal multifunctionality remain unintuitive and heuristic.

The results for learning and predicting two attractors (Lorenz and Rössler) using the two architectures are shown in the figure below. The activation offset in the regular RC architecture is taken to be outside the hyperbolic tangent function in this version.




***Regular RC*** ***Minimal RC***

Figure 27: The classical RC has 100 nodes with 20000 training datapoints. It has set offsets for Lorenz and Rössler as $[-0.9, 1.5]$ respectively. The spectral radius is $\rho = 0.9$. For minimal reservoir computer, we use a block size 15, target spectral radius of 0.2, cubic input and reservoir-state nonlinearities, trained on $10,000$ samples per system with attractor labels -1 (Lorenz) and +1 (Rössler) and the ridge of $\beta = 10^{-1}$.

## 6.2 Inclusion of three attractors

In the setup described above, we emphasize on using scalar values as offsets. This means that this architecture is only attuned to predicting two attractors. The architecture can only predict two attractors because a scalar offset implies that we are placing both attractors on the same line. Adding a third attractor and assigning a scalar value to this would indicate that we have three attractors that are co-linearly embedded. They all become a part of a one-dimensional subspace that spans $\mathbf{1}_N$. Experimentation with multiple hyperparameters showed that this one-dimensional parameterization for three attractors was able to at most learn two out of the three attractors reliably. After two attractors were separated, the third remained insufficiently distinguishable to the shared ridge readout. Therefore, we propose a new scheme to predict three attractors. The added advantage of using this scheme is the fact that we only need to minimise two hyperparameters instead of three separate scalars. The main concept behind multifunctionality is the fact that we can push the centres of the attractors in the reservoir space far from one another so that a single ridge regression can fit and learn multiple attractors.

A caveat of this phenomenon is the idea that the reservoir itself must be big enough to be able to embed these many separated attractors. If we solely rely on a scalar offset, then despite the size of the reservoir (its dimensions), we are only pushing two attractors in opposite ends. Instead, we could do the other obvious thing and push three attractors in three different directions randomly. We push these in random directions since the reservoir is assumed to be homogeneous without any preferred direction that favours learning. If we use this approach, we can certainly get a multifunctional reservoir. However, this approach requires the choice of three independent vectors, and we cannot take advantage of the fact that one line has the ability to learn two attractors. Leveraging this idea, we propose to create a triangle within the reservoir space and push the three attractors in three different vertices of this triangle. We draw a random $N \times 3$ Gaussian matrix and orthonormalise it,

$$\mathbf{UR} = \mathtt{qr}\big(\mathcal{N}(0,1)^{N\times 3}\big), \qquad \hat{\vec{u}} = \mathbf{U}_{:,1}, \quad \hat{\vec{v}} = \mathbf{U}_{:,2}, \qquad \hat{\vec{u}} \perp \hat{\vec{v}}, \;\; \|\hat{\vec{u}}\| = \|\hat{\vec{v}}\| = 1, \tag{90}$$

In this equation `qr` refers to the module in Numpy on Python that allows for the selection and orthonormalization of random directions. The $N$ refers to the dimension of the reservoir itself. Once we have sampled these directions, we place the three offsets at the vertices of an isosceles triangle:

$$\boxed{\mathbf{o}_1 = +\tfrac{d}{2}\,\hat{\vec{u}}, \qquad \mathbf{o}_2 = -\tfrac{d}{2}\,\hat{\vec{u}}, \qquad \mathbf{o}_3 = \; p\,\hat{\vec{v}}.} \tag{91}$$

Doing this reduces our free parameters from three arbitrary directions to two. The two free parameters are the *base* $d$ (separation of the first two attractors) and the *height* $p$ (how far the third one leaves the line). The pairwise separations of the offsets are

$$\|\mathbf{o}_1 - \mathbf{o}_2\| = d, \qquad \|\mathbf{o}_1 - \mathbf{o}_3\| = \|\mathbf{o}_2 - \mathbf{o}_3\| = \sqrt{\tfrac{d^2}{4} + p^2}. \tag{92}$$

This is visualised as follows,

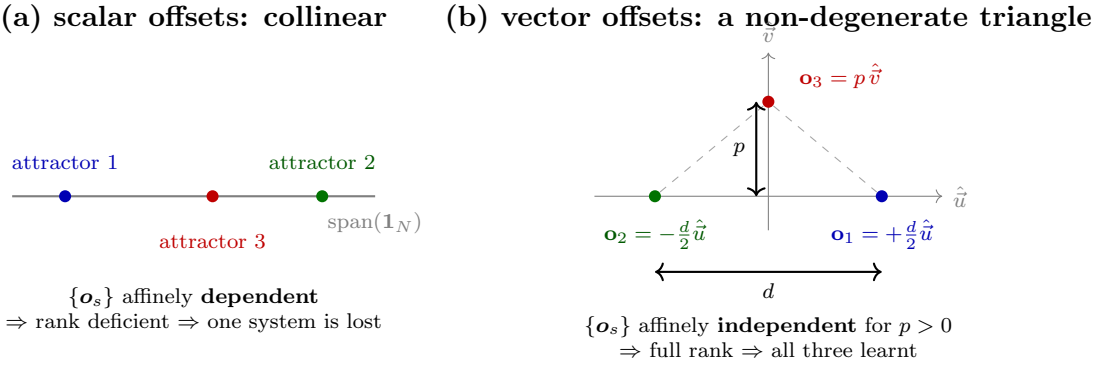



Figure 28: Scalar offsets can only slide the reservoir image along $\mathbf{1}_N$, so three attractors are forced to be collinear. Placing the third offset along a direction $\hat{\vec{v}} \perp \hat{\vec{u}}$ restores independence, giving space for the third attractor. The presentation's "perpendicular direction to the two" is this $\hat{\vec{v}}$.

Consequently, we predict three attractors now, Lorenz, Rucklidge, and Rössler. For the figure below we use the minimal RC architecture. The block size is taken to be 15 with a spectral radius of 0.3 and a readout of $[\mathbf{r}, \mathbf{r}^2, \mathbf{r}^3]$. We train for $20,000$ steps and predict the next $10,000$. The regularisation is taken to be $\beta = 10^{-2}$. With $d = 1.5$ and $p = 0.15$ we achieve multifunctionality in figure 29,

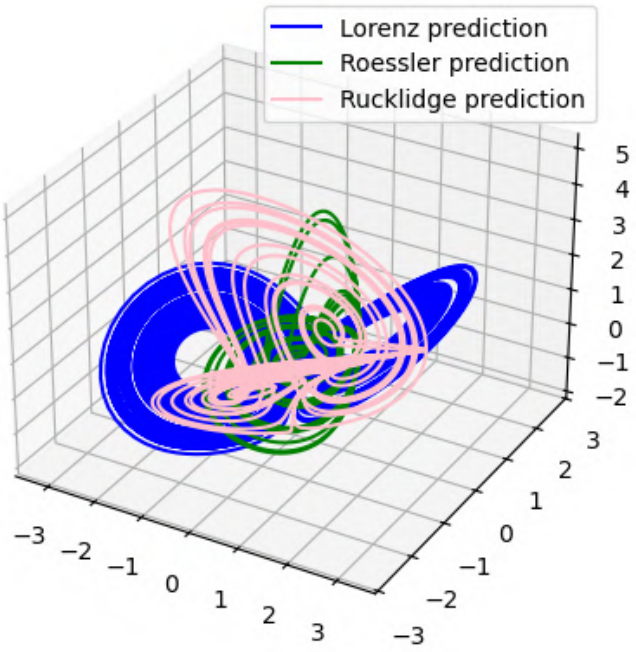



Figure 29: Multifunctional minimal RC with three learnt attractors. We see that even a small push in the third direction is enough to provide a prediction that individually distinguishes each attractor.

Even parameters like $d = 1.5, p = 0.15$ work. These parameters makes the triangle very flat — the height is a tenth of the base and the minimum pairwise separation is $\approx 0.86$. The reason for such a tiny separation to suffice is breaking the exact degeneracy (in direction), not the size of the break. For classical RC, $p$ is bounded above by saturation — too large and the offset pushes the units into the flat tails of the hyperbolic tangent function, killing the input-driven dynamics.

The three attractor multifunctionality architecture also works for the special case where we can construct an equilateral triangle where the third vector becomes $\frac{\sqrt{3}d}{2}$. In this case, a single hyperparameter dictates the learning ability of the three attractors. However, we already get the presence of multifunctionality without going this far in the $p$ parameter. Therefore, this case of an equilateral triangle is redundant.

## 6.3 Continuity in Multifunctionality

The continuity statistic defined in chapter Four, and also in the work of Pecora et al. [13] provides an indirect measure of how points close by in a given space (say, the drive), map to points in another space (say, the reservoir). As discussed earlier, systems must be normalized before the reservoir is trained. The normalization provided in Equation 72 squeezes the drive between $(0, 1)$. Consequently, two points on distinct systems that operate on different length scales can act like neighbours. This phenomenon is shown in Equation 73. This overlap in the case of multifunctionality is a bad sign. Overlapping attractors during the training phase imply that the ridge regression cannot be fit in a way that can distinguish the two. Therefore, we introduce a fixed offset term that seperates the two attractors individually and pushes their centers away from one another in the reservoir space. If we considered the case of a stacked drive, such that,

$$\mathbf{X}_{\text{train}} = [\mathbf{x}_{\text{Attractor 1}}, \mathbf{x}_{\text{Attractor 2}}]^T \tag{93}$$

This drive without the offset terms would give low values of the continuity radius $\varepsilon^*/\sigma_{std}$ between the drive and the reservoir $(D \to R)$ since points in the neighbourhood are all operating at the same length scale between $(0, 1)$. However, when independent offsets are added to each attractor, the points that used to overlap one another and be classified as neighbours are by construction made to move further appart. Therefore the diagnostic $\varepsilon^*/\sigma_{std}$ sees an increase in its continuity radius. We therefore invert the usual reading of the statistic: Rather than treating a *small* $\varepsilon(D \to R)$ as a sign of continuity, we *maximise* it as a sign of multifunctionality. Accordingly, we also collect and vertically stack the reservoir states and evaluate the continuity statistic on the stacked data, where it now reports how well the reservoir states of the two attractors are separated: Since we want their images pushed apart, a *larger* $\varepsilon(D \Rightarrow R)$ is the signature we seek.

> It is essential to note that the largest offsets do not always lead to the greatest degree of multifunctionality. The offsets interact with the drive in a non-trivial way. They are added to every coordinate, and the nonlinear features makes the relationship less transparent still.

The usage of the continuity statistic over other metrics can be justified using three aspects. Primarily, the continuity statistic also offers the reverse direction $(R \to D)$ mapping as another independent direction. Therefore, the same quantity of $\varepsilon^*/\sigma_{std}$ in the reverse direction can be minimised and checked. The two statistics are genuinely complementary: At zero offset it is $R \to D$ that fails, while at excessive offset it is $D \Rightarrow R$ that collapses.

Another reason to use these statistics is the fact that offset alone cannot inflate the values. In other words, the highest offset does not imply the highest continuity radius. The continuity statistic itself is normalised by the size of the reservoir cloud $\sigma_{std}$. Therefore, the ratio rises only when $\varepsilon^*$ grows *faster* than the cloud radius, that is, only when neighbourhoods which remain genuinely ambiguous in drive space are split across the two reservoir clusters. The third aspect that makes the statistic more robust to use is its mitigation of outliers.

In our statistic, a large number of fiducial points are chosen across the two attractors, and the binomial probability test makes sure that we are looking at a set of neighboring points that move rather than individual transformations. A stray point, or a single anomalous excursion, will not move the statistic. The Theiler window and the strand exclusion of Kennel reinforce this by ensuring the $n_\delta$ neighbours are separate recurrences rather than consecutive samples of one orbit segment. This means that the continuity statistic can exactly probe the movement of overlapping points in the drive to seperate clusters in the reservoir.

We also note that the statistic calculated on the stacked drive is guaranteed to give a minumum on the diagonal line where the two offsets, $\mathbf{c}_1 = \mathbf{c}_2$ for the minimal RC architecture with a linear activation. This statistic therefore can map the failure of the ridge exactly.

## 6.4 Results

We use the continuity statistics and the differentiability statistics in the context of multifunctional reservoirs for the classical RC and the minimal RC architectures. We try to measure the multifunctionality in the Lorenz-Rössler pairs for the minimal RC architecture along with the Rucklidge-Chua attractors to compare. Similarly, for the three system approach we collect data for the minimal reservoir trying to learn the Lorenz-Roessler-Rucklidge triplet. We apply the same principles to the classical RC architecture where we look at the Lorenz-Rössler pair, and also look at the Rucklidge-Complex Butterfly pair for multifunctionality. For all these trajectories we integrate using $\Delta t = 10^{-2}$.

For the minimal RC architectures, we simulate a trajectory of 65,000 steps with discarding 5000 transient steps. We use 45,000 of these points to generate reservoir states (however, for differentiability we only take 25000 points for stability of the state matrix), 5,000 points are used to synchronize the reservoir and 5,000 steps are predicted to measure the prediction quality.

For the classical RC, we simulate a trajectory of only $40,000$ points with discarding 2,000 transient points. We train on $20,000$ points and use 500 points to synchronize the reservoir. Then we predict for the next 5,000 points.

The architectures have the specifications outlined in table 2.
We measure the continuity statistic using $12,000$ fiducial points of the joint trajectories. The parameters obtained from continuity are compared with the forecast horizon plots with the definition discussed in equation 3. We now take a harmonic mean of the two systems instead of the normal mean since it is plausible that these attractor pairs show a big difference in their performance. Therefore we measure:

$$T_D = \frac{2 \times T_A \times T_B}{T_A + T_B}. \tag{94}$$

Table 2: Reservoir-computer architecture and readout specifications.

| Property | Normal RC | Minimal RC (NMRC) |
|---|---|---|
| Reservoir nodes | 100 | block size $b = 15$ |
| Spectral radius | 0.9 | 0.2 |
| Mean in-degree | 6.0 | — |
| State nonlinearity | $\tanh(x)$ | Linear |
| Readout features | $[\mathbf{r},\, \mathbf{r}^2]$ | $[\mathbf{r},\, \mathbf{r}^2\, \mathbf{r}^3]$ |
| Readout fit | ridge regression | ridge regression |
| Regularisation $\lambda$ | $10^{-6}$ | $10^{-1}$ |

All the results are averaged over ten initial condition realisations. For the minimal RC reservoir trying to learn the Lorenz and the Rössler system shown in figure 30, we find that the prediction performance is maximised in the second and the fourth quadrants of the graph. We find the emergence of two quarter circles, each centered around the edge of these quadrants. There also exists a valley in the form of $y = x$ diagonal where the performance is minimised, as expected from identical offsets.

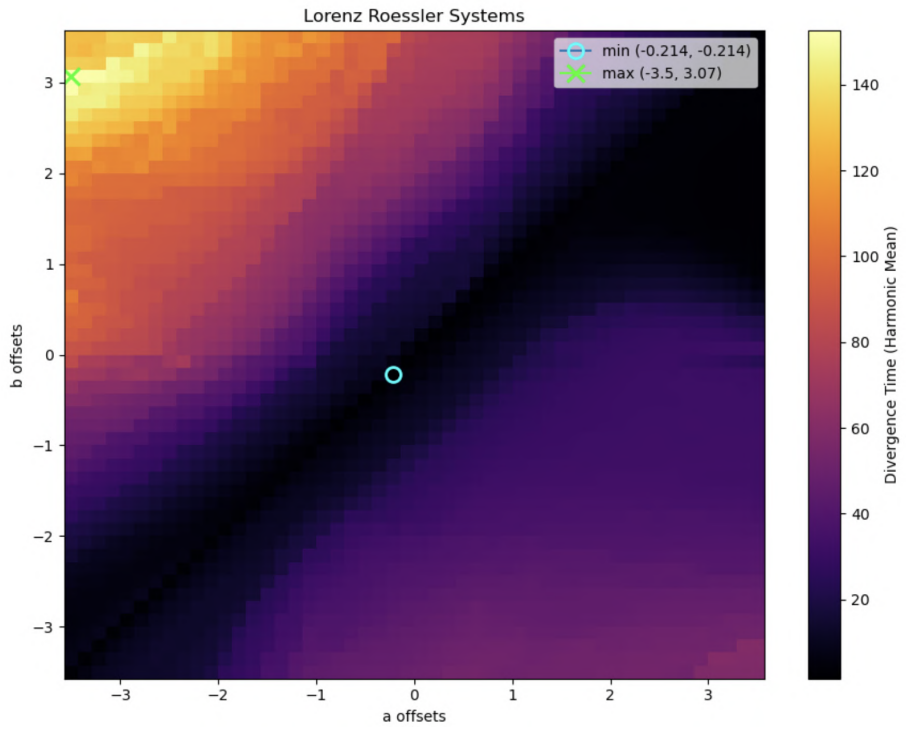



Figure 30: The divergence time data for the Lorenz Rössler system indicates the optimisation of performance for the minimal reservoir in the second and the fourth quadrant

The hyperparameters suggested by the continuity statistic allude to similar regions where the divergence time also peaks. The failure of the ridge on the diagonal is also clearly emphasised in the figure 31.

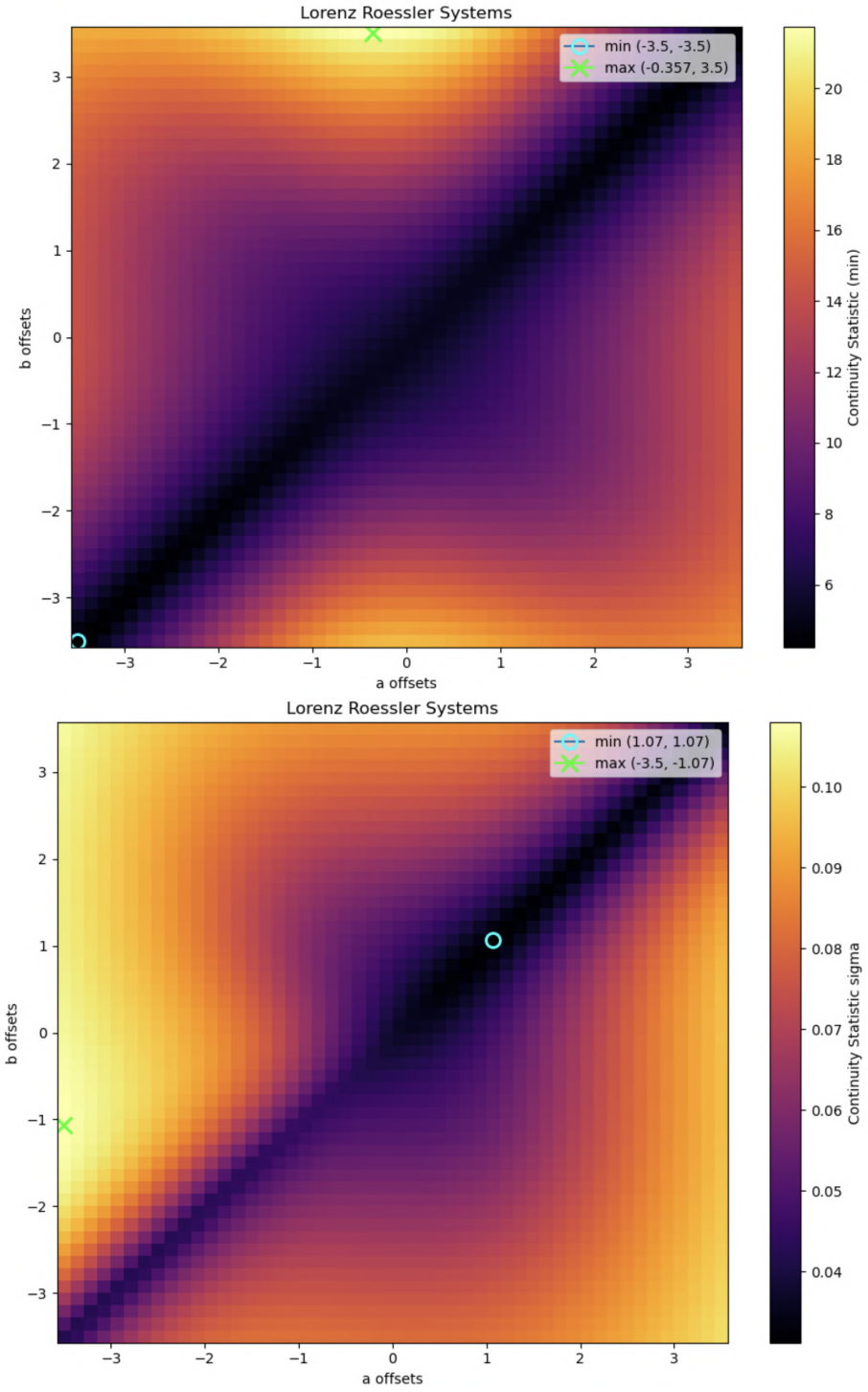



Figure 31: Both statistics give results that are in the bright zones of the divergence time heatmap, and the clear failure of the ridge is seen in the contrasting $y = x$ line. In this graph, $\varepsilon_{min}$ is more useful than $\varepsilon_\sigma$ normalisation.

The parameters indiacted by the continuity statistic indeed happens to fall in the region where the reservoir is multifunctional and has a good predictive performance.

This same behaviour can also be inferred from the differentiability statistic that is shown in figure 32. The differentiability statistic shows bright patches in the first and the fourth quadrant correlating with the divergence time performance. These zones are even smaller than the continuity statistics.

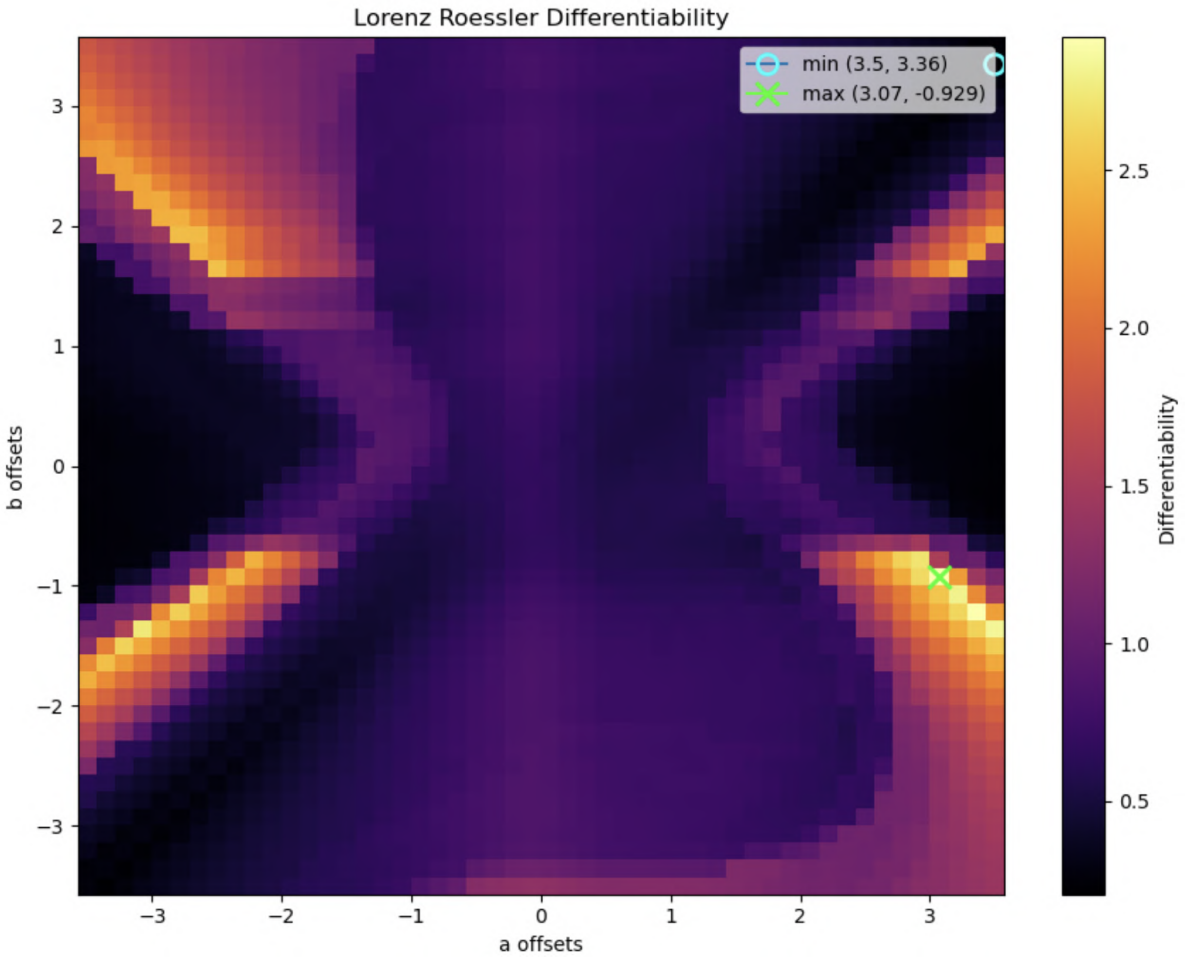



Figure 32: The differentiability statistic maxima again highlights the zones which contain parameters where multifunctionality works. These zones are even more restrictive than the continuity statistic zones

To compare this with another system, we look at the Rucklidge-Chua pair and how the continuity statistic appropriately predicts the zones where the reservoir is multifunctional.

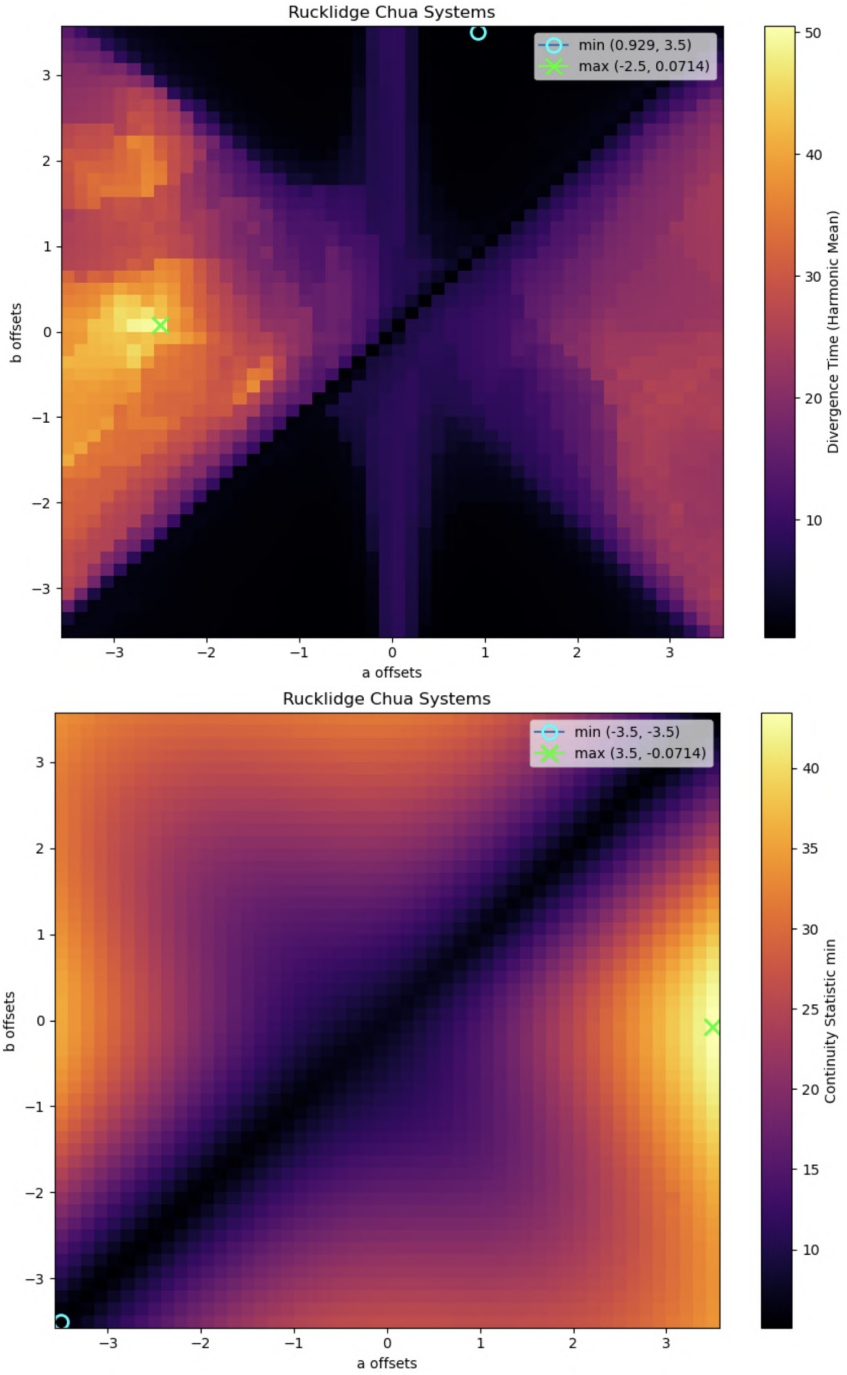



Figure 33: As infered from the graphs, the continuity statistic may not provide the best configuration, but is always able to provide an offset configuration where the reservoir shows a decent multifunctional performance

The use of the statistic on another pair ensures that the continuity statistic is always able to provide a configuration of offsets that show multifunctional behaviour. From the Rucklidge-Chua pair in figure 33 we also infer that the statistic is not sensitive enough in its current configuration to give the best value.

In the minimal RC case shown in figure 34, we can see the clear correlation between the performance of a multifunctional three-system (Lorenz Rössler Rucklidge triplet) and the continuity statistic.

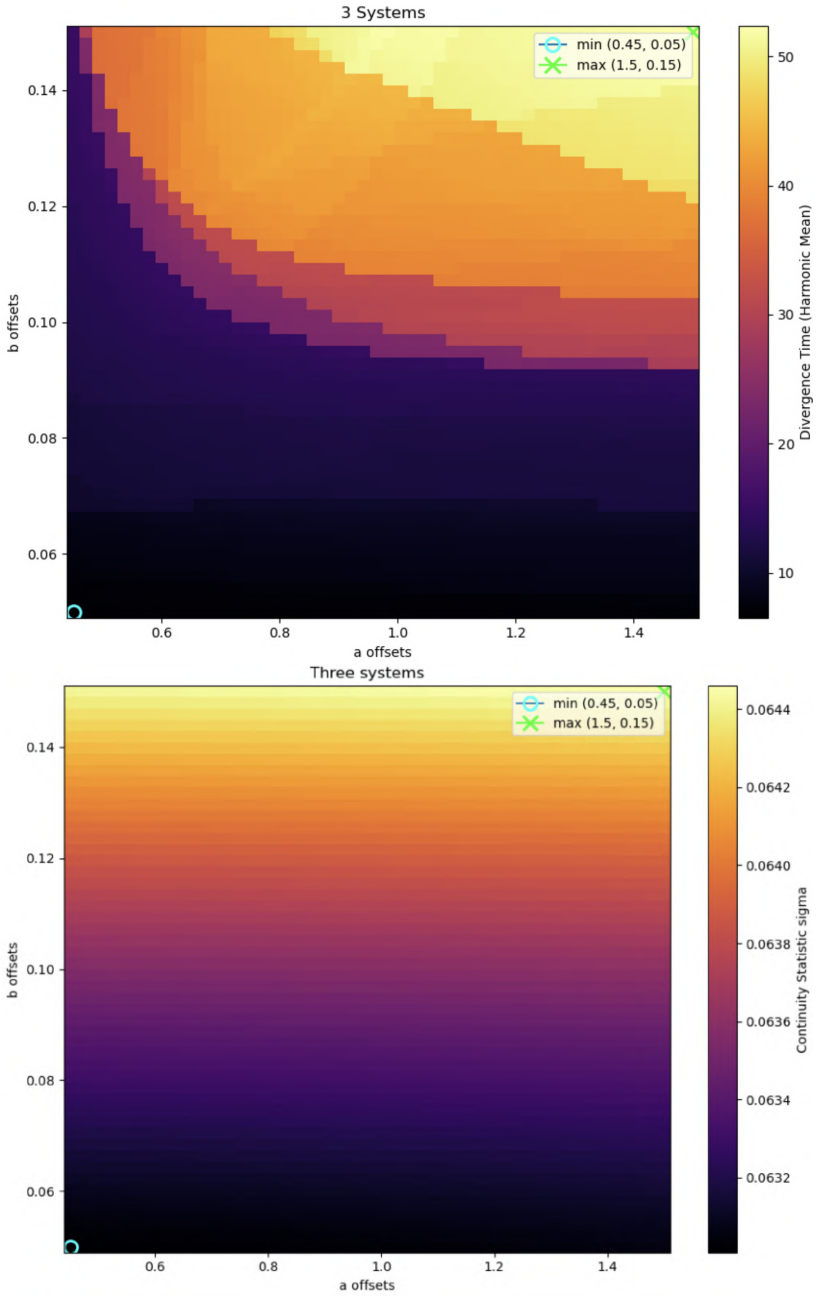



Figure 34: The gradient of performance matches the gradient of the $\varepsilon_\sigma$ statistic with correct identifications of points that work.

There is a clear correlation between the divergence time and the continuity of the three system minimal RC configuration 34. In this case the performance is maximised at the same point where the continuity statistic is also maximised. To test this statistic as a general statistic that can be used to optimise multifunctionality in any RC system.

We apply this to the classical RC architecture. The results for the Lorenz-Rössler pair are as shown in figure 35.

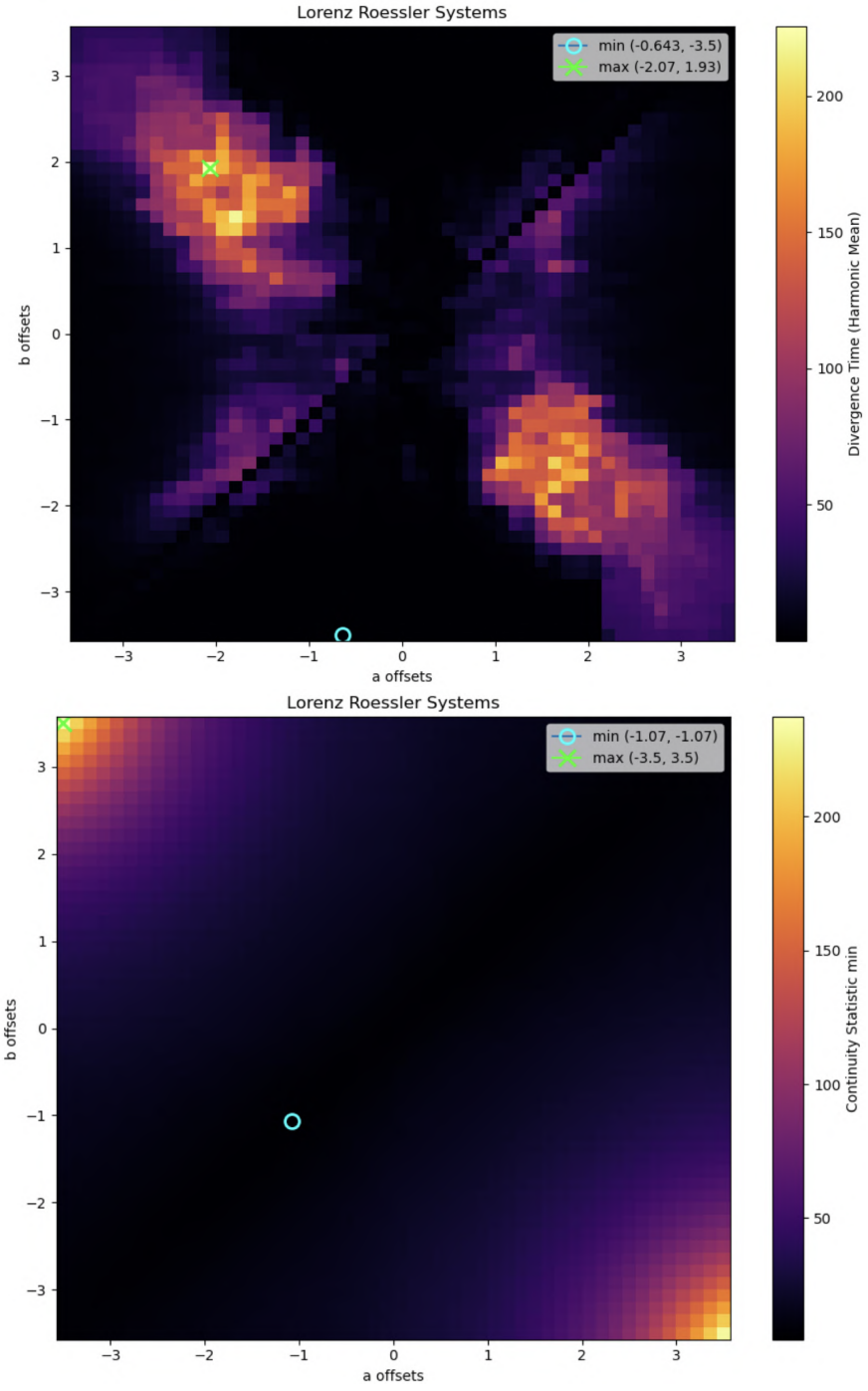



Figure 35: The hyperbolic tangent may show some signs of saturation but the statistic still identifies quadrants of optimal performance.

We see that while the introduction of the hyperbolic tangent activation function does show saturation and regions of good performance are much more restrictive than the actual regions, we still are able to get a multifunctional reservoir.

This can be better seen in the case of the Complex Butterfly-Rucklidge pair for the classical RC system shown in figure 36.

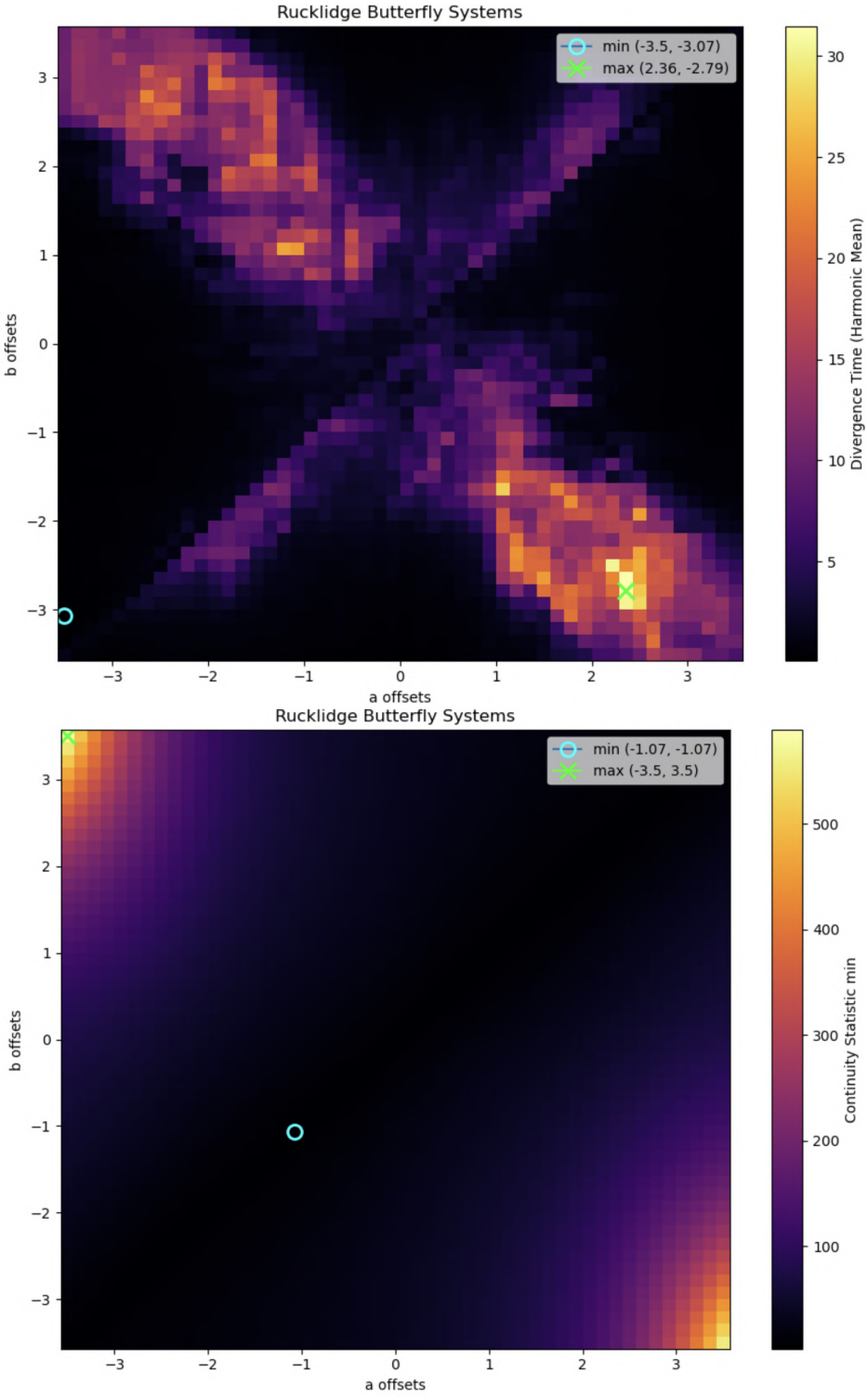



Figure 36: The performance again hints at the correct quadrants of multifunctional performance.

The results on a different system allude to the same idea: The continuity statistic can provide a configuration of offsets that clearly lie in the region of multifunctional reservoir. However, this configuration may not be the best, prediction-wise. Therefore, we propose the use of the continuity statistic as a means to choose a correct region of hyperparameters over a solely heuristic choice made by practitioners.

# 7 Conclusion

This thesis developed a novel approach for hyperparameter selection to find the minimal nonlinearity required in the data for embedding and the correct offsets required for a multifunctional reservoir using topological properties of the reservoir itself. The thesis claims that hyperparameters such as the readout nonlinearity $\eta$ for minimal RCs and the multifunctional offset parameters $a, b$ can be adjusted just through the study of the relationship between the reservoir and the drive. This approach is prediction-free and does not require generating $W_{out}$ via ridge regression. In this regard, this work also contributes to the field of explainable AI, arguing that the reservoir and the readout are two fundamental aspects that embed the attractor during training and significantly affect prediction performance. This thesis also reiterates the importance of the reservoir topology to its prediction. Additionally, the thesis proposes a novel minimal RC architecture that uses random perpendicular vectors to create a three-system multifunctional reservoir. Conclusively, this work is able to reverify the claims made by Pecora et al. [13] and apply it to the context Prosperino et al.'s work [11]. We also apply the statistics to the multifunctional architectures proposed by Flynn et al. [10]. This thesis is also one of the primary works on multifunctional hyperparameter tuning.

In this thesis we adequately contributed to the work of Prosperino et al. [11] who claimed that there is a correlation between the nonlinearity of the dynamical system and the nonlinearity of the generalised reservoir states. We showed that when the nonlinearity of the system matches that of the reservoir, the quantity $\Delta_\eta$ (the difference in continuity statistic in the two directions) is minimised. In other words, we showed that the smallest nonlinearity of the data ensured that the mapping was continuous in both directions. This method also showed congruence with the minimum nonlinearity predicted by Prosperino's method on unknown real-world data for the financial MSCI World and S&P500 indices. However, this was not always the case. The prediction-free approach to deduce nonlinearity has several shortcomings.

The major flaw in this method is the fact that the minima observed are highly sensitive to other hyperparameters of the drive, such as the spectral radius $\rho$ and the block size $b$. Since the spectral radius only scales the adjacency matrix,

$$\mathbf{A} \mapsto \frac{\rho^*}{\rho(\mathbf{A})}\mathbf{A}. \tag{95}$$

This implies that when we raise $\mathbf{r}^\eta$ we also exponentially amplify the effect of the spectral radius. This is why it is essential to keep it near one. However, even small numerical inconsistencies cause exponential differences. Another limitation with our method as a means to predict the nonlinearity of the system is the fact that the method still remains highly heuristic with its choice of sub-reservoirs. For example, we showed that for the Thomas system, the best embedding occurs in the sum coordinate subreservoir at 3.2. However, for the base coordinate the embedding occured best at 2.84. This huge variety in range makes the method a good approximator but not a reliable method of hunting for nonlinearities.

We recommend using this method as a way to understand the topology of the ML architecture rather than a method of finding nonlinearities accurately.

With regards to the multifunctional reservoir, the continuity statistic especially $\varepsilon_{min}$ acted as a reliable hyperparameter selection quantity. We suggest practitioners to do a sweep of continuity statistics on their coarse-grained grid to get a rough estimate of the offset regions they should focus on, and then choosing parameters in the bright range for further multifunctional study. It is important to reiterate that our interpretation of the continuity statistic requires to maximize the $D \to R$ mapping rather than minimize it since it is now made into a tool that hunts for points in the same neighbourhood that get mapped and separated in the reservoir space so that the ridge regression can distinguish between them.

A shortcoming of this method is the fact that it works better for the minimal RC architecture than it does for the regular RC architecture simply because of the choice of the activation function. It is important to choose an offset range in the regular RC architecture that does not saturate the $\tanh(x)$ activation function to its ends. Therefore we proposed an alternate activation where we explicitly placed the offset after the function.

Another limitation of the multifunctional approach is that reservoir computing has two modes of failure in multifunctionality. The first is the fact that the $N$ attractors are not embedded correctly in the reservoir space, and the second is that there is the failure or numerical divergence of the ridge regression. Both these independently influence the performance. Our method can identify regions in which there is the embedding failure but not necessarily look at the failure in optimisation parameters for the ridge regression. This is the reason we can give a wide area where the reservoir works but not predict the exact coordinates of peaking performance.
Several future directions seem worth pursuing with our approach. Primarily, it would be interesting to look at other architectures such as the Next Generation Reservoir Computing (NGRC) developed by [55] or look at the architectures where the reservoir still remains a black box. The other direction that is important to focus on is the use of topological statistics in parameter-aware Reservoir Computing [56]. The idea would be to ask, can we identify bifurcations in parameter space of attractors just through looking at the topologies of the reservoir? In other words, we ask, do reservoir topologies hold information of bifurcations?

In the context our current work, a natural extension would include an in-depth study of the continuity statistics through mathematical tools. Possibly an explicit calculation showing why the Taken's theorem works in this case at all. Additionally, studying systems that are hard to predict, such as mixed systems (that combine the input of two attractors) and deducing topological reasons on why they are so hard to predict in the first place would be a good use of the continuity statistics.

Conclusively, our work remains an interdisciplinary study that can be used across different contexts to find fundamental relations in datasets. We attempt to make AI architectures less opaque and create a future that is more interpretable. We try to reduce the burden on our colleagues who spend valuable computational resources on hyperparameter optimization.

# References

[1] Christopher M. Danforth. Chaos in an atmosphere hanging on a wall. URL http://mpe.dimacs.rutgers.edu/2013/03/17/chaos-in-an-atmosphere-hanging-on-a-wall/.

[2] J. P. Eckmann and D. Ruelle. Ergodic theory of chaos and strange attractors. *Rev. Mod. Phys.*, 57:617–656, Jul 1985. doi: 10.1103/RevModPhys.57.617. URL https://link.aps.org/doi/10.1103/RevModPhys.57.617.

[3] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German national research center for information technology gmd technical report*, 148(34):13, 2001.

[4] Floris Takens. Detecting strange attractors in turbulence. In David Rand and Lai-Sang Young, editors, *Dynamical Systems and Turbulence, Warwick 1980*, pages 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg. ISBN 978-3-540-38945-3.

[5] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. doi: 10.1109/72.279181.

[6] Jaideep Pathak, Zhixin Lu, Brian R. Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12):121102, 12 2017. ISSN 1054-1500. doi: 10.1063/1.5010300. URL https://doi.org/10.1063/1.5010300.

[7] Zhixin Lu, Brian R. Hunt, and Edward Ott. Attractor reconstruction by machine learning. *Chaos*, 28 6:061104, 2018. URL https://api.semanticscholar.org/CorpusID:49647172.

[8] Mantas Lukoševičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009. ISSN 1574-0137. doi: https://doi.org/10.1016/j.cosrev.2009.03.005. URL https://www.sciencedirect.com/science/article/pii/S1574013709000173.

[9] Giuseppe Alessio D'Inverno and Jonathan Dong. Comparison of reservoir computing topologies using the recurrent kernel approach. *Neurocomputing*, 611:128679, 2025. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2024.128679. URL https://www.sciencedirect.com/science/article/pii/S0925231224014504.

[10] Andrew Flynn, Vassilios A. Tsachouridis, and Andreas Amann. Multifunctionality in a reservoir computer. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(1):013125, 01 2021. ISSN 1054-1500. doi: 10.1063/5.0019974. URL https://doi.org/10.1063/5.0019974.

[11] Davide Prosperino, Haochun Ma, and Christoph Räth. Tailored minimal reservoir computing: on the bidirectional connection between nonlinearities in the reservoir and in data. *Chaos*, 35 9, 2025. URL https://api.semanticscholar.org/CorpusID:281101827.

[12] Haochun Ma, Davide Prosperino, and Christoph Räth. A novel approach to minimal reservoir computing. *Scientific Reports*, 13(1):12970, Aug 2023. ISSN 2045-2322. doi: 10.1038/s41598-023-39886-w. URL https://doi.org/10.1038/s41598-023-39886-w.

[13] Louis Pecora and Thomas Carroll. Statistics for differential topological properties between datasets with an application to reservoir computers. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 35(7):073153, 07 2025. ISSN 1054-1500. doi: 10.1063/5.0269914. URL https://doi.org/10.1063/5.0269914.

[14] Alan Wolf, Jack Swift, Harry L. Swinney, and John A. Vastano. Determining lyapunov exponents from a time series. *Physica D: Nonlinear Phenomena*, 16:285–317, 1985. URL https://api.semanticscholar.org/CorpusID:14411384.

[15] S.H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. CRC Press, 2018. ISBN 9780429961113. URL https://books.google.de/books?id=A0paDwAAQBAJ.

[16] Alexander Haluszczynski and Christoph Räth. Good and bad predictions: Assessing and improving the replication of chaotic attractors by means of reservoir computing. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29 (10):103143, 10 2019. ISSN 1054-1500. doi: 10.1063/1.5118725. URL https://doi.org/10.1063/1.5118725.

[17] Michael T Rosenstein, James J Collins, and Carlo J De Luca. A practical method for calculating largest lyapunov exponents from a small data set. *Physica D: Nonlinear Phenomena*, 65(1-2):117–134, 1993.

[18] Edward N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, 20(2):130–148, March 1963. doi: 10.1175/1520-0469(1963)020⟨0130:DNF⟩2.0.CO;2.

[19] S. Baur, T. Nakano, D. Duncan, F. Fischbach, A. Haluszczynski, M. Klatt, D. Köglmayr, H. Ma, D. Prosperino, and C. Räth. SCAN: A versatile implementation of reservoir computing methods. https://github.com/DLR-KI/scan, 2023. Unpublished.

[20] Julien Clinton Sprott. *Elegant Chaos*. WORLD SCIENTIFIC, 2010. doi: 10.1142/7183. URL https://www.worldscientific.com/doi/abs/10.1142/7183.

[21] R. THOMAS. Deterministic chaos seen in terms of feedback circuits: Analysis, synthesis, "labyrinth chaos". *International Journal of Bifurcation and Chaos*, 09 (10):1889–1905, 1999. doi: 10.1142/S0218127499001383. URL https://doi.org/10.1142/S0218127499001383.

[22] O.E. Rössler. An equation for continuous chaos. *Physics Letters A*, 57(5):397–398, 1976. ISSN 0375-9601. doi: https://doi.org/10.1016/0375-9601(76)90101-8. URL https://www.sciencedirect.com/science/article/pii/0375960176901018.

[23] Alastair M. Rucklidge. Chaos in magnetoconvection. *Nonlinearity*, 7(6):1565–1591, 1994.

[24] T. Matsumoto. A chaotic attractor from chua's circuit. *IEEE Transactions on Circuits and Systems*, 31(12):1055–1058, 1984. doi: 10.1109/TCS.1984.1085459.

[25] Simin Yu and Wallace K.S. Tang. Tetrapterous butterfly attractors in modified lorenz systems. *Chaos, Solitons & Fractals*, 41(4):1740–1749, 2009. ISSN 0960-0779. doi: https://doi.org/10.1016/j.chaos.2008.07.023. URL https://www.sciencedirect.com/science/article/pii/S0960077908003226.

[26] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2019.03.005. URL https://www.sciencedirect.com/science/article/pii/S0893608019300784.

[27] T. L. Carroll and L. M. Pecora. Network structure effects in reservoir computers. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(8):083130, August 2019. doi: 10.1063/1.5097686. URL https://doi.org.

[28] Haochun Ma, Davide Prosperino, and Christoph Räth. A novel approach to minimal reservoir computing. *Scientific Reports*, 13(1):12970, Aug 2023. ISSN 2045-2322. doi: 10.1038/s41598-023-39886-w. URL https://doi.org/10.1038/s41598-023-39886-w.

[29] Haochun Ma. *Causal Decomposition of Complex Systems & Prediction of Chaos using Machine Learning*. Dissertation, Ludwig-Maximilians-Universität, 2024. URL https://elib.dlr.de/202134/.

[30] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14:2531–2560, 2002. URL https://api.semanticscholar.org/CorpusID:1045112.

[31] Joschka Herteux and Christoph Räth. Breaking symmetries of the reservoir equations in echo state networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(12):123142, 12 2020. ISSN 1054-1500. doi: 10.1063/5.0028993. URL https://doi.org/10.1063/5.0028993.

[32] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on

perturbations. *Neural Computation*, 14(11):2531–2560, 11 2002. ISSN 0899-7667. doi: 10.1162/089976602760407955. URL https://doi.org/10.1162/089976602760407955.

[33] Lyudmila Grigoryeva and Juan-Pablo Ortega. Echo state networks are universal. *Neural Networks*, 108:495–508, December 2018. ISSN 0893-6080. doi: 10.1016/j.neunet.2018.08.025.

[34] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12(1):69–82, 1970.

[35] M. H. Eggar. D. r. j. chillingworth, differential topology with a view to applications (pitman, 1976), £8·50. *Proceedings of the Edinburgh Mathematical Society*, 21(1): 89–89, 1978. doi: 10.1017/S0013091500015947.

[36] Andrea Ceni, Peter Ashwin, Lorenzo Livi, and Claire Postlethwaite. The echo index and multistability in input-driven recurrent neural networks. *Physica D: Nonlinear Phenomena*, 412:132609, 2020. ISSN 0167-2789. doi: https://doi.org/10.1016/j.physd.2020.132609. URL https://www.sciencedirect.com/science/article/pii/S0167278920300427.

[37] Louis M. Pecora, Thomas L. Carroll, and James F. Heagy. Statistics for mathematical properties of maps between time series embeddings. *Physical Review E*, 52(4): 3420–3439, 1995.

[38] Louis M. Pecora, Linda Moniz, Jonathan Nichols, and Thomas L. Carroll. A unified approach to attractor reconstruction. *Chaos*, 17(1):013110, 2007.

[39] James Theiler. Spurious dimension from correlation algorithms applied to limited time-series data. *Physical Review A*, 34(3):2427–2432, 1986.

[40] Matthew B. Kennel and Henry D. I. Abarbanel. False neighbors and false strands: A reliable minimum embedding dimension algorithm. *Physical Review E*, 66(2): 026209, 2002.

[41] Michael Spivak. *Calculus on Manifolds: A Modern Approach to Classical Theorems of Advanced Calculus*. Mathematics Monograph Series. Benjamin/Cummings Publishing Company, New York, 1965.

[42] NumPy Developers. numpy.linalg.svd. NumPy Reference Manual, version 2.5. URL https://numpy.org/doc/2.5/reference/generated/numpy.linalg.svd.html. Accessed 12 July 2026.

[43] Roger Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955. doi: 10.1017/S0305004100030401.

[44] J.R. Dormand and P.J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980. ISSN

0377-0427. doi: https://doi.org/10.1016/0771-050X(80)90013-3. URL https://www.sciencedirect.com/science/article/pii/0771050X80900133.

[45] James Theiler, Stephen Eubank, André Longtin, Bryan Galdrikian, and J. Doyne Farmer. Testing for nonlinearity in time series: The method of surrogate data. *Physica D: Nonlinear Phenomena*, 58(1–4):77–94, 1992. doi: 10.1016/0167-2789(92)90102-S.

[46] Eric W. Weisstein. Wiener-Khinchin Theorem. From *MathWorld*—A Wolfram Web Resource. URL https://mathworld.wolfram.com/Wiener-KhinchinTheorem.html. Accessed: 2026-08-28.

[47] Bernard V. North, David Curtis, and Pak C. Sham. A note on the calculation of empirical P values from Monte Carlo procedures. *The American Journal of Human Genetics*, 71(2):439–441, 2002. doi: 10.1086/341527.

[48] Peter Ditlevsen and Susanne Ditlevsen. Warning of a forthcoming collapse of the Atlantic meridional overturning circulation. *Nature Communications*, 14(1):4254, July 2023. ISSN 2041-1723. doi: 10.1038/s41467-023-39810-w. URL https://doi.org/10.1038/s41467-023-39810-w.

[49] Mattia Rigotti, Omri Barak, Melissa R. Warden, Xiao-Jing Wang, Nathaniel D. Daw, Earl K. Miller, and Stefano Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, 497(7451):585–590, May 2013. doi: 10.1038/nature12160.

[50] Peter A. Getting. Emerging principles governing the operation of neural networks. *Annual Review of Neuroscience*, 12:185–204, 1989. doi: 10.1146/annurev.ne.12.030189.001153.

[51] Patsy S Dickinson. Interactions among neural networks for behavior. *Current Opinion in Neurobiology*, 5(6):792–798, 1995. ISSN 0959-4388. doi: https://doi.org/10.1016/0959-4388(95)80108-1. URL https://www.sciencedirect.com/science/article/pii/0959438895801081.

[52] Alexander N. Pisarchik and Ulrike Feudel. Control of multistability. *Physics Reports*, 540(4):167–218, 2014. ISSN 0370-1573. doi: https://doi.org/10.1016/j.physrep.2014.02.007. URL https://www.sciencedirect.com/science/article/pii/S0370157314000453. Control of multistability.

[53] Zhixin Lu and Danielle S. Bassett. Invertible generalized synchronization: A putative mechanism for implicit learning in neural systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(6):063133, 06 2020. ISSN 1054-1500. doi: 10.1063/5.0004344. URL https://doi.org/10.1063/5.0004344.

[54] Andrew Flynn, Vassilios A. Tsachouridis, and Andreas Amann. Seeing double with a multifunctional reservoir computer. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(11):113115, 11 2023. ISSN 1054-1500. doi: 10.1063/5.0157648. URL https://doi.org/10.1063/5.0157648.

[55] Daniel J. Gauthier, Erik Bollt, Aaron Griffith, and Wendson A. S. Barbosa. Next generation reservoir computing. *Nature Communications*, 12(1):5564, 2021. doi: 10.1038/s41467-021-25801-2. URL https://doi.org/10.1038/s41467-021-25801-2.

[56] Dishant Sisodia and Sarika Jalan. Dynamical analysis of a parameter-aware reservoir computer. *Phys. Rev. E*, 110:034211, Sep 2024. doi: 10.1103/PhysRevE.110.034211. URL https://link.aps.org/doi/10.1103/PhysRevE.110.034211.

# Declaration of authorship

I hereby declare that this thesis is my own work, and that I have not used any sources and aids other than those stated in the thesis. I am aware that the thesis in digital form can be examined for the use of unauthorized aid and to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Munich, 01/09/2026



Spandan Pandya