

Interpreting recurrent neural networks behaviour via excitable network attractors

Andrea Ceni^{*1}, Peter Ashwin^{†2}, and Lorenzo Livi^{‡§3,1}

¹Department of Computer Science, University of Exeter, Exeter EX4 4QF, UK

²Department of Mathematics, University of Exeter, Exeter EX4 4QF, UK

³Departments of Computer Science and Mathematics, University of Manitoba, Winnipeg, MB R3T 2N2, Canada

February 9, 2019

Abstract

Introduction: Machine learning provides fundamental tools both for scientific research and for the development of technologies with significant impact on society. It provides methods that facilitate the discovery of regularities in data and that give predictions without explicit knowledge of the rules governing a system. However, a price is paid for exploiting such flexibility: machine learning methods are typically black-boxes where it is difficult to fully understand what the machine is doing or how it is operating. This poses constraints on the applicability and explainability of such methods. **Methods:** Our research aims to open the black-box of recurrent neural networks, an important family of neural networks used for processing sequential data. We propose a novel methodology that provides a mechanistic interpretation of behaviour when solving a computational task. Our methodology uses mathematical constructs called excitable network attractors, which are invariant sets in phase space composed of stable attractors and excitable connections between them. **Results and Discussion:** As the behaviour of recurrent neural networks depends both on training and on inputs to the system, we introduce an algorithm to extract network attractors directly from the trajectory of a neural network while solving tasks. Simulations conducted on a controlled benchmark task confirm the relevance of these attractors for interpreting the behaviour of recurrent neural networks, at least for tasks that involve learning a finite number of stable states and transitions between them.

Keywords— Recurrent neural networks; Dynamical systems; Network attractors; Bifurcations.

1 Introduction

Artificial recurrent neural networks (RNNs) are widely used to solve tasks involving temporal data, e.g., speech [18] and handwriting recognition [45], audio classification [26, 53] or time series forecasting [8]. RNNs are characterised by the presence of feedback connections in a hidden layer, which allows generating a state-space representation that equips the network with short-term memory capability. RNNs are universal approximators of dynamical systems [14, 19] meaning that, given enough neurons in the hidden layer, it is possible to fine tune the weights to achieve any desired level of accuracy. Nevertheless, training via back-propagation through time is difficult due to the vanishing/exploding gradient problem [23, 44]. This has led to the development of new and faster techniques for training RNNs, including a different paradigm known

*ac860@exeter.ac.uk

†p.ashwin@exeter.ac.uk

‡lorenz.livi@gmail.com

§Corresponding author

as reservoir computing [32, 33]. Echo state networks (ESNs) [21, 30] constitute an important example of reservoir computing, where a recurrent layer (called a reservoir) is composed of a large number of neurons with randomly initialised connections that are not fine-tuned via gradient-based optimisation mechanisms. The main idea behind ESNs is to exploit the rich dynamics generated by the reservoir with an output layer, the read-out that is optimised to solve a specific task.

1.1 Problem statement and research hypothesis

The high-dimensional and non-linear nature of RNNs complicates interpretability of their internal dynamics, which are characterised by complex, input-dependent spatio-temporal patterns of activity [48, 56]. This poses constraints on understanding the behaviour of RNNs: they are usually viewed as black-boxes from which it is hard to extract useful knowledge about their inner workings. As highlighted by recent research efforts [10, 24, 41], similar interpretability issues affect many other machine learning methods. Furthermore, an increasing societal need to develop accountability and explainability of decision making by AI [17] is driving the development of methodologies for explaining the behaviour of such methods.

Our aim here is to develop effective models that capture the essential dynamical behaviour of RNNs on computational tasks as input-driven responses of a dynamical system, while neglecting microscopic details of the RNN dynamics in phase space (i.e., the space of all possible neuron activations). To this end, we hypothesise that RNNs can undertake computations by exploiting (i) transient dynamical regimes and (ii) excitable connections to switch between different stable attractors, depending on input and current state. The RNN behaviour depends on the task at hand: for example, long transients are mostly exploited in time series forecasting problems, while switching between attractors is mostly exploited in classification problems or tasks requiring to learn a finite set of memory states. These mechanisms are not mutually exclusive and can be exploited synergistically to realise more complex computations.

1.2 Contribution and paper organisation

In order to test our hypothesis, we develop a theoretical framework based on network attractors of dynamical systems. Intuitively, an attractor is a subset of values, in the space of all variables used to describe the state of a dynamical system (i.e. the phase space), where the state will asymptotically converge for “usual” choices of initial conditions. Network attractors are special kinds of attractor which, roughly speaking, look like directed graphs where nodes correspond to local attractors and directed edges corresponds to particular trajectories connecting those local attractors. Since in this paper local attractors of interest are stable fixed points, depending on the context we will often use the term attractor meaning a stable fixed point. Stable fixed points are the simplest possible attractors, which are points in phase space that represent steady-state solutions where all variables of the dynamical system assume constant values. However, there exist fixed points that are only partially stable, which are called saddle points, or are totally unstable, which are called repellers. According to the nature of those particular trajectories connecting local attractors it is possible to consider both heteroclinic network attractors composed of heteroclinic connections between saddles (i.e. partially stable fixed points), and excitable network attractors (ENAs) composed by excitable connections between stable attractors [5, 61]. Heteroclinic network attractors have already been suggested as models to describe transient computation [46, 47], here, conversely, we aim attention at excitable network attractors. Particularly, we focused on ENAs between stable fixed points [5], although ENAs can be theoretically conceived between any kind of attractor: limit cycles, limit tori or strange attractors. More precisely, we show how to construct ENAs describing RNN behaviour for tasks that involve switching between a finite set of attractors. Interestingly, from a directed graph (representing the underlying network attractor of a dynamical system) it is possible to reverse engineer [4] a set of ordinary differential equations ruling, in our case, the RNN behaviour. This, in turn, opens the way to a more analytic description of how RNNs solve computational tasks.

In this paper, we focus on the flip-flop task with two bits [56], since it provides a controlled workbench to test our hypothesis. The input to discrete-time RNN is assumed to be a discrete temporal sequence with values from $\{+1, 0, -1\}$. In general, we consider discrete time k varying input, denoted as $\mathbf{u}[k]$, and output,

denoted as $\mathbf{z}[k]$, of a system related by

$$(\mathbf{x}[k], \mathbf{z}[k]) = \mathbf{R}(\mathbf{u}[k], \mathbf{x}[k - 1]),$$

where \mathbf{R} represents an RNN with evolving internal state, namely $\mathbf{x}[k]$, and connection weights that are optimised during the training phase. We consider response to inputs where, independently for any k , $\mathbf{u}[k]$ is $(0, 0)$ with probability $1 - p$ or otherwise chosen to be one of the four inputs $(\pm 1, 0)$ or $(0, \pm 1)$ with equal probability. This generates a sequence of inputs that remain at $(0, 0)$ for an exponentially distributed period of time, but occasionally takes one of the four values $(0, \pm 1)$, $(\pm 1, 0)$. The target output to be learned by the RNN is a two-dimensional vector $\mathbf{z}[k]$ that can assume four possible configurations: $(1, 1)$, $(1, -1)$, $(-1, 1)$, and $(-1, -1)$. The system (1.2) is considered to successfully accomplish the task if the network is able to reliably and accurately reproduce all 20 possible actions described in Table 1.

Table 1: The two-bit flip-flop task. Depending on the output $\mathbf{z}[k - 1]$ and input $\mathbf{u}[k]$, we expect the output $\mathbf{z}[k]$ to be as given in the table, where N.C. indicates “not change” with regard to the current output value.

Outputs \ Inputs	$(0, 0)$	$(1, 0)$	$(-1, 0)$	$(0, 1)$	$(0, -1)$
$(1, 1)$	N.C.	N.C.	$(-1, 1)$	N.C.	$(1, -1)$
$(1, -1)$	N.C.	N.C.	$(-1, -1)$	$(1, 1)$	N.C.
$(-1, 1)$	N.C.	$(1, 1)$	N.C.	N.C.	$(-1, -1)$
$(-1, -1)$	N.C.	$(1, -1)$	N.C.	$(-1, 1)$	N.C.

Here we take into account RNNs that are ESNs trained with a perturbation matrix obtained by injecting the output into the ESN dynamics. This choice does not limit the validity of our hypothesis: we claim that our results are general and describe the behaviour of all discrete-time, input-driven RNNs [35]. The contributions of this paper can be summarised as follows:

- We provide a theoretical framework to describe the behaviour of RNNs by means of ENAs. The proposed theoretical framework is general and covers several types of computational tasks. However, here we present a specific instance of such a framework that is suitable to describe RNN behaviour on tasks requiring to learn a finite set of memory states;
- By means of bifurcation analysis, we show how to manually design (low-dimensional) ESNs that give rise to ENAs able to solve the flip-flop task with any number of bits. This allows us to justify our choice of modelling framework and its validity in the context of RNNs;
- As ESNs are driven by inputs and subject to training via perturbation matrices, bifurcation analysis alone is not sufficient to explain their behaviour. In fact, inputs and perturbation matrices can affect ESN behaviour in a non-trivial way. To this end, we introduce an algorithm to extract the ENA describing the ESN behaviour for the task at hand. The algorithm analyses an ESN trajectory and constructs a directed graph encoding the underlying ENA on which the dynamics takes place. The vertices (nodes) of such a graph are associated with the fixed points of the dynamics and directed edges describe excitable connections between them. We apply this algorithm to an ESN trained to solve the flip-flop task and show that the resulting ENA is able to explain how ESNs perform computations in a detailed and mechanistic way;
- We define a suitable excitability threshold for this high-dimensional, non-linear dynamical system driven by inputs. We propose a method for computing this excitability threshold that accounts for inputs and can directly be applied to trajectories generated by a neural network while solving a task;
- Our simulations suggest three important findings. First, as already noted by recent research [1, 56], the dynamics of high-dimensional RNNs takes place in a much lower-dimensional phase space region that is determined by the structure introduced with training and inputs. Here, we observe that the

dynamics is indeed low-dimensional, but highlight the fact that additional dimensions are used occasionally to switch between stable states according to control inputs. This suggests that, for example, a simplistic use of methods based on explained variance to reduce dimensions needs to be avoided. Second, we show how ENA models describing RNN behaviour can be exploited to provide a mechanistic interpretation of errors occurring during the computation undertaken by RNNs. Finally, we show how excitability thresholds of the extracted ENAs allow us to assess the robustness of RNNs to noise-induced perturbations.

The remainder of this paper is organised as follows. Section 2 introduces the essential background material needed in this paper. Section 3 shows how to design low-dimensional ESN models that give rise to ENAs able to solve the flip-flop task. In Section 4 we propose an algorithm for automatically extracting ENAs directly from an ESN trajectory generated while solving a task. In Section 5, we present and discuss the results of the simulations. Finally, Section 6 draws conclusions and points to future research directions. We include three appendices. The first one, Appendix A, introduces notions of linear stability that will be useful throughout the paper. Appendix B discuss bifurcations of fixed points in low-dimensional ESN maps. Appendix C provides details about the procedure used to determine fixed points.

2 Background

2.1 Echo State Networks

We consider a specific system of the form (1.2), corresponding to a discrete-time ESN state-update and related output:

$$\mathbf{x}[k] = \phi(\mathbf{W}_r \mathbf{x}[k-1] + \mathbf{W}_{in} \mathbf{u}[k]), \quad (1)$$

$$\mathbf{z}[k] = \mathbf{W}_o \mathbf{x}[k]. \quad (2)$$

$\mathbf{x}[k] \in \mathbb{R}^{N_r}$ is the state, $\mathbf{u}[k] \in \mathbb{R}^{N_i}$ and $\mathbf{z}[k] \in \mathbb{R}^{N_o}$ denote input and output, respectively. The activation function $\phi(\cdot)$ is applied component-wise; without loss of generality, we consider $\phi = \tanh : \mathbb{R} \rightarrow (-1, 1)$. It is worth mentioning [44] that (1) is often written $\mathbf{x}[k] = \mathbf{W}_r \phi(\mathbf{x}[k-1]) + \mathbf{W}_{in} \mathbf{u}[k]$. However, Miller and Fumarola [38] proved that the two formulations are equivalent up to a change of coordinates; they produce the same discrete-time dynamics. In this paper, we build on (1) and consider a network of discrete-time leaky-integrator neurons [22] of the form:

$$\mathbf{x}[k] = (1 - \alpha)\mathbf{x}[k-1] + \alpha\phi(\mathbf{W}_r \mathbf{x}[k-1] + \mathbf{W}_{in} \mathbf{u}[k] + \boldsymbol{\epsilon}). \quad (3)$$

Here, $\alpha \in (0, 1]$ is called a leak rate and explicitly sets the time-scale of the ESN [57]. The $\boldsymbol{\epsilon}$ term represents additive white Gaussian noise with spherical covariance matrix and unit standard deviation.

The reservoir $\mathbf{W}_r \in \mathbb{R}^{N_r \times N_r}$ and input $\mathbf{W}_{in} \in \mathbb{R}^{N_r \times N_i}$ matrices are usually random with i.i.d. entries drawn from uniform or Gaussian distributions [31, 32]. However, in the literature it is possible to find reservoirs with different connection patterns, including deterministic topologies [51] and those exploiting the norm-preserving property of orthogonal matrices [37]. Relevant hyperparameters directly affecting ESN performance include the number of neurons and sparseness of their connections, the spectral radius of the reservoir matrix, and leak rate α [9, 29]. The so-called echo-state property (ESP) [15, 34, 63] guarantees the existence and uniqueness of a global attracting trajectory for any input sequence in a compact set. The ESP, although useful in some tasks like forecasting tasks, is in practice difficult to verify and it is formulated only for ESNs of the form shown in Eq. 1. Therefore, it is not suitable to ESN models and tasks we discuss here; see the following subsection.

2.2 Training ESNs with low-rank perturbation matrices

Training of RNNs is typically implemented by means of stochastic gradient descent or variations of thereof [52]. Learning long-term dependencies in RNNs with gradient descent is known to be problematic, as a

consequence of the so-called vanishing/exploding gradient problem [44]. To this end, different approaches have been proposed that can be summarised in two categories: (i) methods using gating mechanisms (such as Long Short-Term Memory [20] and Gated Recurrent Unit [12] networks) and (ii) those based on unitary matrices and constant-slope activations [3]. On the other hand, training of the recurrent layer in ESNs is typically realised by perturbing a randomly-initialised reservoir with a low-rank, deterministic matrix. This is conventionally accomplished by feeding back the ESN output to the recurrent layer [27, 49, 50, 60, 62] or, as Mastrogiuseppe and Ostojic [36] recently proposed, by designing the reservoir directly as $\mathbf{W}_r = \mathbf{X} + \mathbf{D}$, where \mathbf{X} is a random matrix and \mathbf{D} is a deterministic, low-rank matrix encoding the task of interest.

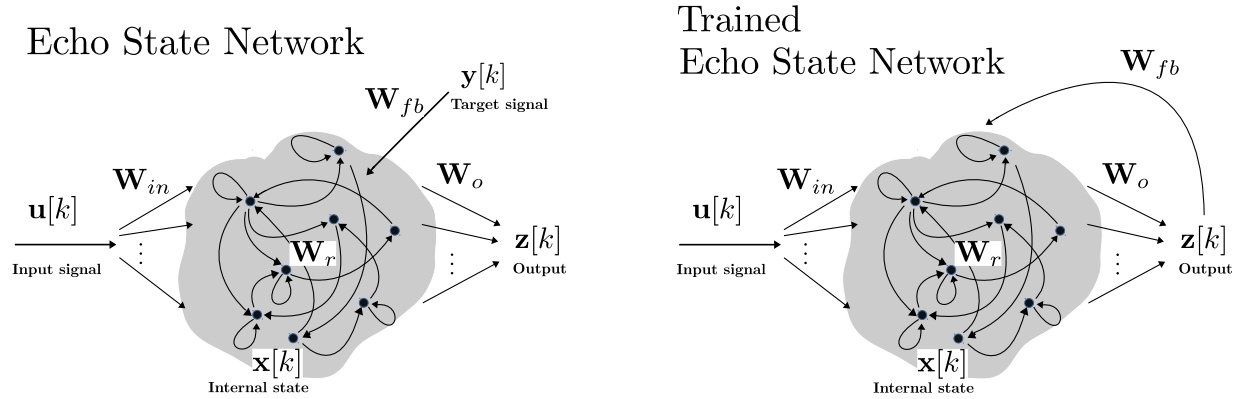


Figure 1: **Left:** Illustration of an ESN in the open-loop phase (4). Training is supervised by means of the target signal to learn \mathbf{y} . **Right:** Illustration of an ESN (5) after the optimisation of the read-out matrix. Feeding back the output signal into the reservoir (closed loop system) allows the system to self-sustain dynamics, which is also driven by the input signal.

In this paper, we use a supervised learning algorithm which exploit the feedback of the ESN output as a mechanism for training the recurrent layer. The state-update equation (3) takes then the following form:

$$\mathbf{x}[k] = (1 - \alpha)\mathbf{x}[k - 1] + \alpha\phi(\mathbf{W}_r\mathbf{x}[k - 1] + \mathbf{W}_{in}\mathbf{u}[k] + \mathbf{W}_{fb}\mathbf{y}[k - 1] + \epsilon), \quad (4)$$

where $\mathbf{W}_{fb} \in \mathbb{R}^{N_r \times N_o}$ is a matrix with i.i.d. random coefficients usually drawn from a uniform or Gaussian distribution. Depending on whether training is performed batch or online, $\mathbf{y}[k - 1]$ in (4) takes the form of either the target signal or the output produced by the ESN (2), respectively. In batch mode, it is possible to clearly distinguish two main phases, see Fig. 1. First, the reservoir is fed with the target signal (*teacher forcing*) \mathbf{y} that we want the machine to learn, and this “open-loop phase” corresponds to the training session. Second, you replace the target signal with the generated output signal \mathbf{z} , and this “closed-loop phase” corresponds to the test phase of the trained ESN. An analysis of the stability of the transition from open to closed loop can be found in [50]. Specifically, in the open-loop phase we collect in columns of a matrix $\mathbf{X} \in \mathbb{R}^{N_r \times N}$ all the internal states $\mathbf{x}[k]$ of the ESN in response to the target $\mathbf{y}[k - 1]$ and input $\mathbf{u}[k]$ signals, for $k = 1, \dots, N$, being N the length of the training session. Now, since we want the output to be the target, we just have to solve the linear system $\mathbf{Y} = \mathbf{W}_o\mathbf{X}$, where $\mathbf{Y} \in \mathbb{R}^{N_o \times N}$ is the matrix with k th column the target $\mathbf{y}[k]$. Once optimised the read-out matrix $\mathbf{W}_o \in \mathbb{R}^{N_o \times N_r}$ for the task, by imposing $\mathbf{y}[k] = \mathbf{z}[k]$ and expanding in Eq. 4 with (2), we obtain:

$$\begin{aligned} \mathbf{x}[k] &= (1 - \alpha)\mathbf{x}[k - 1] + \alpha\phi(\mathbf{W}_r\mathbf{x}[k - 1] + \mathbf{W}_{in}\mathbf{u}[k] + \mathbf{W}_{fb}\mathbf{W}_o\mathbf{x}[k - 1] + \epsilon) = \\ &= (1 - \alpha)\mathbf{x}[k - 1] + \alpha\phi(\mathbf{M}\mathbf{x}[k - 1] + \mathbf{W}_{in}\mathbf{u}[k] + \epsilon), \end{aligned} \quad (5)$$

where we denoted with

$$\mathbf{M} := \mathbf{W}_r + \mathbf{W}_{fb}\mathbf{W}_o, \quad (6)$$

the “trained reservoir” obtained by means of matrix $\mathbf{W}_{fb}\mathbf{W}_o \in \mathbb{R}^{N_r \times N_r}$, which is low-rank $N_o \ll N_r$. Inputs $\mathbf{u}[k]$ in Eq. 5 play the role of control inputs and are typically constant or impulsive signals. The ESN read-out matrix \mathbf{W}_o is conventionally determined by solving a regularised least-squares problem. However, we note that also online training schemes have been developed, e.g., the FORCE learning algorithm originally introduced in [55] and further extended by De Pasquale et al. [13]. During the test phase, regardless of the adopted training mechanism, state-update of ESNs is described by Eq. 5. In this paper, we consider batch training via regularised least-squares (ridge regression) and analyse the trajectory generated by (5) during the test phase. It is worth nothing that our theoretical framework does not rely on a particular training method or a particular RNN architecture. However complicated it may be a RNN model, after the training session the trained neural network can be described by a noisy nonautonomous dynamical system, which in our case is represented by Eq. 5. We have chosen to work with ESN because, to our knowledge, it is the simplest RNN model to handle and implement. For this reason, terms ESN and RNN are used interchangeably in this paper.

2.3 Network attractors

Many common dynamical systems encountered in Nature are dissipative [11, 54]. In such systems, the absence of any conservative law brings the system to evolve on a attracting set of dimension strictly less than the original phase space dimension; such a set (or a particular subset of it) is commonly defined as *attractor*. A formal definition can be found in [40], where the concept of attractor has been investigated thoroughly. Furthermore, it is called *basin of attraction* of an attractor the set of all initial conditions for which the system evolves toward the attractor. Attractors convey crucial information about the behaviour of the dynamical systems which have generated them.

Here, we consider the following noise-free discrete-time dynamical system with inputs:

$$\mathbf{x}[k] = \mathbf{G}(\mathbf{x}[k-1], \mathbf{u}[k]), \quad (7)$$

where $\mathbf{G} : \mathbb{R}^{N_r} \times \mathbb{R}^{N_i} \rightarrow \mathbb{R}^{N_r}$ is related to Eq. 5 as follows:

$$\mathbf{G}(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} (1-\alpha)x_1 + \alpha\phi(\mathbf{M}_{(1)} \cdot \mathbf{x} + (\mathbf{W}_{in})_{(1)} \cdot \mathbf{u}) \\ (1-\alpha)x_2 + \alpha\phi(\mathbf{M}_{(2)} \cdot \mathbf{x} + (\mathbf{W}_{in})_{(2)} \cdot \mathbf{u}) \\ \vdots \\ (1-\alpha)x_{N_r} + \alpha\phi(\mathbf{M}_{(N_r)} \cdot \mathbf{x} + (\mathbf{W}_{in})_{(N_r)} \cdot \mathbf{u}) \end{pmatrix}, \quad (8)$$

where is \mathbf{M} the trained reservoir matrix (6), the subscript (i) denotes the i th rows of a matrix and \cdot the usual dot product. As mentioned before, the input signal for the flip-flop task is null most of the time. Hence, it is worth studying the autonomous dynamics of

$$\mathbf{x}[k] = \mathbf{F}(\mathbf{x}[k-1]) = \mathbf{G}(\mathbf{x}[k-1], \mathbf{0}). \quad (9)$$

Fixed points $\mathbf{p} \in \mathbb{R}^{N_r}$ are solutions of $\mathbf{F}(\mathbf{p}) = \mathbf{p}$. Related to the notion of attractor is the notion of *limit set* [11]. The ω -limit set of a point \mathbf{x}_0 under the iterated map (9) is defined by

$$\omega_{\mathbf{F}}(\mathbf{x}_0) := \bigcap_{n \in \mathbb{N}} \overline{\{\mathbf{F}^h(\mathbf{x}_0) \mid h > n\}}, \quad (10)$$

which is the set of cluster points of the forward trajectory $\{\mathbf{F}^h(\mathbf{x}_0)\}_{h \in \mathbb{N}}$. For a given fixed point \mathbf{p} , its basin of attraction is formed by all points $\mathbf{x} \in \mathbb{R}^{N_r}$ such that $\omega_{\mathbf{F}}(\mathbf{x}) = \{\mathbf{p}\}$. If such a fixed point is stable, then there exists a neighbourhood of \mathbf{p} whose ω -limit set corresponds to this fixed point.

More complex attractors consisting of networks of invariant sets in phase space have been proposed in the literature [42, 61]. Such models found renewed interest in neuroscience [39, 59] and other fields of research, as they provide a fundamental tool to describe dynamic processes occurring on transients that explore excitable connections. More relevant to our paper, we focus on networks of stable fixed points that are connected by

excitable connections. Following [5, 6], we say that there exists an excitable connection for amplitude $\delta > 0$ from stable fixed point \mathbf{p}_i to \mathbf{p}_j whenever

$$B_\delta(\mathbf{p}_i) \cap W^s(\mathbf{p}_j) \neq \emptyset, \quad (11)$$

where $B_\delta(\mathbf{p}_i)$ stands for the closed ball centred on \mathbf{p}_i with radius $\delta > 0$ and $W^s(\mathbf{p}_j) = \{\mathbf{x} \in \mathbb{R}^{N_r} \mid \omega_{\mathbf{F}}(\mathbf{x}) = \{\mathbf{p}_j\}\}$ denotes the basin of attraction¹ of the fixed point \mathbf{p}_j . We say an excitable connection from \mathbf{p}_i to \mathbf{p}_j has threshold $\delta_{th}(\mathbf{p}_i, \mathbf{p}_j)$, where

$$\delta_{th}(\mathbf{p}_i, \mathbf{p}_j) := \inf\{\delta > 0 : B_\delta(\mathbf{p}_i) \cap W^s(\mathbf{p}_j) \neq \emptyset\}. \quad (12)$$

The relation (12) can be informally interpreted as the fact that the fixed point \mathbf{p}_i is $\delta_{th}(\mathbf{p}_i, \mathbf{p}_j)$ away from the basin of \mathbf{p}_j ; see Fig. 2 for a visual explanation.

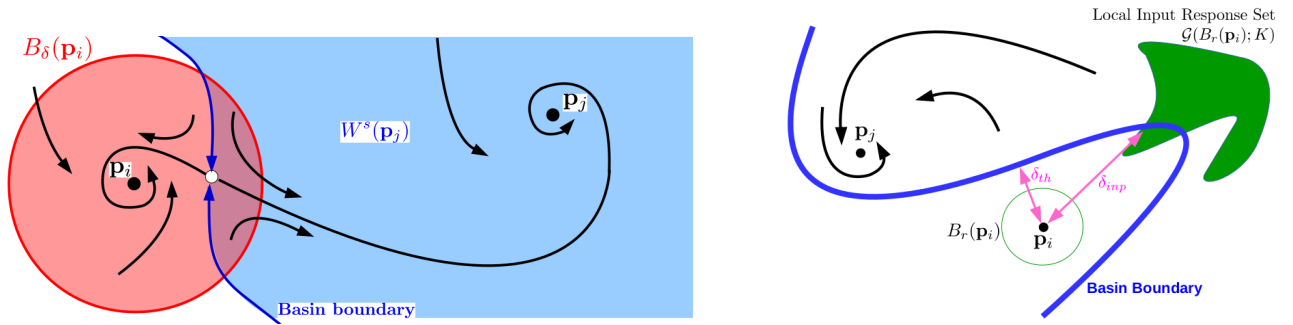


Figure 2: **Left:** Depiction of an excitable connection from \mathbf{p}_i to \mathbf{p}_j . The red area is a closed ball centred on \mathbf{p}_i of radius δ . The blue area represents the stable manifold of \mathbf{p}_j , i.e., its basin of attraction. The open circle represents a saddle whose stable manifold (blue curves) denotes the boundary of the basin of attraction of \mathbf{p}_j . Some points of $B_\delta(\mathbf{p}_i)$ converge to \mathbf{p}_i itself, while those points beyond the basin boundary converge towards \mathbf{p}_j , as suggested by the black arrows. **Right:** Representation of the activation of an excitable connection through the action of the input which allows to accomplish the switch from the stable point \mathbf{p}_i to the stable point \mathbf{p}_j . Firstly, the internal state of the RNN stands nearby the stable point \mathbf{p}_i , in the neighbourhood $B_r(\mathbf{p}_i)$. Then, a control input $\mathbf{u}[k+1] \in K$ drives the current internal state $\mathbf{x}[k]$ to the next state $\mathbf{x}[k+1]$ inside the local input response set, represented as the green subregion. Finally, if the state falls beyond the basin boundary then the internal state converges to the stable point \mathbf{p}_j . Excitability threshold $\delta_{th}(\mathbf{p}_i, \mathbf{p}_j)$, in (12), is computed along the direction where the distance is shortest in order to escape from the basin of attraction of \mathbf{p}_i and converge to \mathbf{p}_j . Nevertheless, input can potentially drive the dynamics towards alternative dimensions for the purpose of achieving the switch from \mathbf{p}_i to \mathbf{p}_j . Input-driven excitability threshold $\delta_{inp}(\mathbf{p}_i, \mathbf{p}_j)$, in (15), is computed considering the subspace exploited by the input to solve the task.

A set $X_{exc} \subset \mathbb{R}^{N_r}$ is called an ENA for amplitude $\delta > 0$ if there exists a collection of fixed points $\{\mathbf{p}_i\}_{i=1}^M$, such that

$$X_{exc}(\{\mathbf{p}_i\}_{i=1}^M, \delta) := \bigcup_{\substack{i,j=1 \\ i \neq j}}^M \{\mathbf{F}^h(B_\delta(\mathbf{p}_i))\}_{h \geq 0} \cap W^s(\mathbf{p}_j), \quad (13)$$

where $\mathbf{F}(B_\delta(\mathbf{p}_i)) := \cup_{\mathbf{x} \in B_\delta(\mathbf{p}_i)} \mathbf{F}(\mathbf{x})$ is based on Eq. 9 (see [5, 6]). The autonomous dynamics on such a set converges to one of the stable fixed points; hence, external inputs are necessary to get interesting dynamics. If the external input is large enough, then the state will escape from the current basin of attraction and switch to a different one, until another sufficiently large input will lead to another change of basin.

¹The basin of attraction corresponds to the stable manifold whenever \mathbf{p}_j is hyperbolic. This has interior if \mathbf{p}_j is an attractor.

The excitability threshold (12) is defined as the (Euclidean) distance between a given stable fixed point, \mathbf{p}_i , and the basin of attraction of another fixed point, say \mathbf{p}_j . Such a quantity measures the minimum distance in phase space necessary to escape from the basin of \mathbf{p}_i and converge towards \mathbf{p}_j . Nevertheless, if the dynamical system is high dimensional, then there will be a large number of possible escaping directions that could be exploited by inputs. Therefore, excitability thresholds (12) alone may not be representative of nonautonomous systems driven by inputs. For this purpose, in order to take into account the action of inputs on the dynamics, we introduce the notion of *input-driven excitability threshold* of an excitable connection. Considering K as a compact subspace of \mathbb{R}^{N_i} , we define $\mathcal{G}(\mathbf{x}_0; K) := \bigcup_{\mathbf{u} \in K} \mathbf{G}(\mathbf{x}_0, \mathbf{u})$, where \mathbf{G} is the function in (8) defining the nonautonomous dynamical system which describes the trained neural network. The set $\mathcal{G}(\mathbf{x}_0; K)$ contains all states reachable from \mathbf{x}_0 under the action of input values $\mathbf{u} \in K$. Importantly, the presence of noise has the effect to perturb away the internal state from the exact location of the stable point in the deterministic counterpart of the dynamics. Therefore, instead of $\mathcal{G}(\mathbf{p}_i; K)$ we rather observe the following set, that we called *local input response set*,

$$\mathcal{G}(B_r(\mathbf{p}_i); K) := \bigcup_{\mathbf{x} \in B_r(\mathbf{p}_i)} \mathcal{G}(\mathbf{x}; K), \quad (14)$$

where the radius r can be shrunk according to the amplitude of the noise. The continuity of \mathbf{G} implies that if $r \rightarrow 0^+$ then the monotonically decreasing sequence of sets $\{\mathcal{G}(B_r(\mathbf{p}_i); K)\}_{r \geq 0}$ converges to $\mathcal{G}(\mathbf{p}_i; K)$ regardless of $K \subset \mathbb{R}^{N_i}$. We note that $\mathcal{G}(B_r(\mathbf{p}_i); K)$, as a subspace of \mathbb{R}^{N_r} , is compact if $K \subset \mathbb{R}^{N_i}$ is compact. If K represents all possible inputs of a particular task², then we can drop it and denote (14) simply as $\mathcal{G}(B_r(\mathbf{p}_i))$. Therefore, the subregion of the phase space, represented by the local input response set $\mathcal{G}(B_r(\mathbf{p}_i))$, encodes the action exercised by the input when the internal state of the RNN is nearby the stable point \mathbf{p}_i . We say an excitable connection from \mathbf{p}_i to \mathbf{p}_j has input-driven excitability threshold $\delta_{\text{inp}}(\mathbf{p}_i, \mathbf{p}_j)$, where

$$\delta_{\text{inp}}(\mathbf{p}_i, \mathbf{p}_j) := \inf\{\delta > 0 : \mathcal{G}(\mathbf{p}_i) \cap B_\delta(\mathbf{p}_i) \cap W^s(\mathbf{p}_j) \neq \emptyset\}. \quad (15)$$

Excitability threshold in Eq. 15 has a similar meaning to the one defined in (12), although it considers only the subspace exploited by the action of inputs nearby \mathbf{p}_i , where the excitable connection starts from, see Fig. 2. Finally, from the fact that $\mathcal{G}(\mathbf{p}_i) \cap B_\delta(\mathbf{p}_i) \cap W^s(\mathbf{p}_j) \subseteq B_\delta(\mathbf{p}_i) \cap W^s(\mathbf{p}_j)$, it follows that $\delta_{\text{th}}(\mathbf{p}_i, \mathbf{p}_j) \leq \delta_{\text{inp}}(\mathbf{p}_i, \mathbf{p}_j)$ holds for all fixed points.

3 Designing low-dimensional ESNs to solve flip-flop tasks

In this section, starting from ESN models, we show how to manually design ENAs that realise computations needed for the flip-flop task. This step allows us to justify the choice of the modelling framework adopted here and its validity in the context of RNNs. For this purpose, we rely on bifurcation theory; see Appendix B for technical notions. The concept of bifurcation refers to a formal mathematical definition [28] which describes a relevant change in phase space topology of a dynamical system. Whenever, varying the parameters of the model, such qualitative changes appear, then it's used to say that the system has undergone a bifurcation. For this reason, the notion of bifurcation is strictly related with the concept of training in machine learning and neural networks. Training the recurrent layer of RNNs corresponds to shaping their phase space topology, possibly inducing bifurcations that lead to a qualitative change in behaviour. In particular, we argue that adding a low-rank perturbation matrix to the reservoir (6) induces bifurcations, leading to the creation of attracting regions in ESN phase space useful to solve the task at hand. Notably, this framework does not exclude others more sophisticated adaptive training algorithms. Indeed, from this perspective they would correspond to a sequence of suitable bifurcations for the RNN in order to solve the task, instead of a single shot in batch training. Let us assume the origin as the only global attractor for the autonomous system $\mathbf{x}[k] = (1 - \alpha)\mathbf{x}[k - 1] + \alpha\phi(\mathbf{W}_r\mathbf{x}[k - 1])$ associated with (4). Then, the bifurcation induced by (6) leads to

²For example, in the flip-flop task we have $K = \{(0, 0), (1, 0), (-1, 0), (0, 1), (0, -1)\}$

a transition from such a trivial dynamic to one where the origin becomes unstable³ and repels trajectories towards other attracting regions in phase space, where the nonautonomous dynamics actually takes place.

In Sec. 3.1, we provide a low-dimensional example of an ESN that is able to solve the flip-flop task; the reservoir of such ESN is formed by two neurons. In Sec. 3.2, instead, we design an ESN model with a reservoir formed by $2k$ neurons which is able to solve the flip-flop task with k bits. For both examples we show the underlying ENA ruling their dynamics.

3.1 A minimal-dimension example to solve the two-bit flip-flop task

In order to solve the k -bit flip-flop task, one needs to learn 2^k memory states (stable fixed points) and the related switching patterns dictated by control inputs. Here, we show how to design an ESN with two neurons in the recurrent layer, giving rise to an ENA able to solve the flip-flop task with $k = 2$ bits. According to the analysis of Appendix B, we know that it is possible to obtain, with k neurons, up to 3^k fixed points, 2^k of which are stable, $3^k - 2^k - 1$ are saddles, and 1 (the origin) is a repeller. Therefore, we set the trained reservoir weights (6) according to condition (41). In particular,

$$\mathbf{M}(b) := \omega_r \begin{bmatrix} 1 & b \\ b & 1 \end{bmatrix} \quad (16)$$

with $\omega_r = 3$ scaling the reservoir weights, and b acting as tuning parameter. As shown in Fig. 3, for every $0 \leq b < 0.47$, the autonomous system $\mathbf{x}[k] = \tanh(\mathbf{M}(b)\mathbf{x}[k-1])$ has four stable attractors located near the vertices of the invariant square $[-1, 1]^2$.

The input is injected into the ESN via the following weight matrix, $\mathbf{W}_{in} = \omega_{in}\mathbf{I}_2$, i.e., a scaled 2×2 identity matrix, setting the scaling factor (called a hyperparameter in machine learning) to $\omega_{in} = 6$. Let us set the remaining parameters in Eq. 5 as $\epsilon = 0$ and $\alpha = 1$, and let the ESN output (2) be the identity mapping. Let us assume that, at time k , the system is in state $\mathbf{p} = (p_1, p_2)$, which is close to one of the four attractors, i.e., $\mathbf{p}_1 \approx (1, 1)$, $\mathbf{p}_2 \approx (-1, 1)$, $\mathbf{p}_3 \approx (-1, -1)$, $\mathbf{p}_4 \approx (1, -1)$. The possible, non-null inputs at time-step k are $\mathbf{u}[k] \in \{(1, 0), (0, 1), (-1, 0), (0, -1)\}$. The action of the input pulse is encoded in a vector $\Delta \in [-1, 1]^2$, representing the difference between the state before and after the occurrence of such a pulse, whose components are

$$\Delta_j \approx \tanh(3p_j + 6u_j) - \tanh(3p_j), \quad j = 1, 2, \quad (17)$$

considering $\omega_r = 3$, $\omega_{in} = 6$ and, for the sake of simplicity, $b = 0$, which corresponds to the case of two independent neurons. Hence, the switching mechanism between attractors is ruled by the signs of both p_j and u_j ; the former encoding the position and the latter the direction where to move. Therefore, the state changes if and only if p_j and u_j assume different signs for some $j \in \{1, 2\}$. In fact, if they have the same sign, then (17) is null because $\tanh(\text{sgn}(p_j)[3 + 6]) \approx \tanh(\text{sgn}(p_j)3)$ due to saturating activation functions. While, if they have different sign, then (17) becomes close to -1 or 1 , according to $\text{sgn}(p_j)$, because $\tanh(\text{sgn}(p_j)[3 - 6]) = -\tanh(\text{sgn}(p_j)3)$. Clearly, it is not necessary that $|\omega_r - \omega_{in}| = \omega_r$ holds, as in our set up. Although we assumed $b = 0$ for clarity of explanation, the conclusion is the same for each $b \in [0, 0.47)$. However, when the parameter approaches the bifurcation value $b \approx 0.47$, the escaping dynamics from certain fixed points become significantly slower.

Given a set of stable fixed points, a $\delta > 0$ gives rise to a specific ENA; see definition of ENA in (13). For instance, a large δ will potentially activate all excitable connections between the attractors. However, in order to properly solve the flip-flop task, some of the connections need not to be active, namely, the diagonal connections between \mathbf{p}_1 and \mathbf{p}_3 , and between \mathbf{p}_2 and \mathbf{p}_4 . Due to symmetry, there are only three different excitability thresholds that an excitable connection can have, that is, $\delta_{th}(\mathbf{p}_2, \mathbf{p}_1)$, $\delta_{th}(\mathbf{p}_1, \mathbf{p}_2)$, $\delta_{th}(\mathbf{p}_1, \mathbf{p}_3)$. In the particular configuration shown in Fig. 3, it holds that $\delta_{th}(\mathbf{p}_2, \mathbf{p}_1) < \delta_{th}(\mathbf{p}_1, \mathbf{p}_2) < \delta_{th}(\mathbf{p}_1, \mathbf{p}_3)$. Therefore, the underlying ENA supporting the nonautonomous ESN dynamics is defined as $X_{\text{exc}}(\{\mathbf{p}_i\}_{i=1}^4, \delta)$, for every $\delta_{th}(\mathbf{p}_1, \mathbf{p}_2) < \delta < \delta_{th}(\mathbf{p}_1, \mathbf{p}_3)$. We note that $\delta_{th}(\mathbf{p}_1, \mathbf{p}_3) \approx \sqrt{2}$, regardless of $b \in [0, 0.47)$. We can reduce the size of the basin of attraction of \mathbf{p}_2 and \mathbf{p}_4 by increasing the value of b , which in turn reduces the

³It could, in principle, remain stable with other attractors appearing elsewhere, an example can be found in [63].

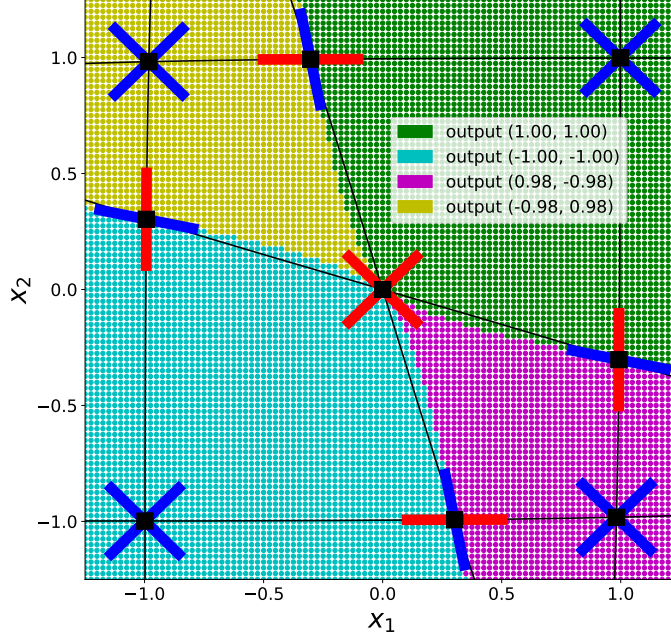


Figure 3: Basins of attraction, nullclines, fixed points and corresponding eigenvectors of the two-dimensional system defined by (16), with $b = 0.2$. Black curves represent the nullclines (see Appendix B) whose reciprocal intersections determine fixed points (square black points). Red segments indicate eigenvectors associated with real eigenvalues larger than one. Blue line segments represent eigenvectors of real eigenvalues in $(0, 1)$. Particularly important are blue lines of saddles, which represent local linear approximations of the boundary of the basins of attraction. The whole phase space is divided in four basins of attraction associated to the four stable points. Boundaries between these basins coincide with the four heteroclinic trajectories connecting the origin with the saddles. The legend shows output values produced at every attractor, which, in this specific example, correspond with the internal state, i.e., the phase space coordinates of the attractors. Points on the plane have been coloured according with the attractor on which they converged to after 100 steps.

excitability threshold of the corresponding connections, until at $b \approx 0.47$ a bifurcation occurs making them disappear⁴. However, the basins of attraction of the other two stable points, \mathbf{p}_1 and \mathbf{p}_3 , become bigger ($\delta_{th}(\mathbf{p}_1, \mathbf{p}_2) \approx 2 - \delta_{th}(\mathbf{p}_2, \mathbf{p}_1)$), so increasing their excitability thresholds. Therefore, when the ESN state is close to \mathbf{p}_1 (or \mathbf{p}_3), the input needs to be very precise to throw back the state to the narrow basins of \mathbf{p}_2 and \mathbf{p}_4 . Moreover, it must have a large amplitude compared to what is needed to escape from such narrow basins. Nevertheless, setting ω_{in} large enough and depending on b (until a value of $\omega_{in} = 6$, which is enough for every $0 < b < 0.47$), the system is able to reproduce the flip-flop dynamics without errors.

Unfortunately, it is not possible to design a two-dimensional ESN for the two-bit flip-flop where the excitability of each attractor can be tuned by changing the location of the nearby saddles. Nevertheless, as we will show in the next section, this becomes possible by including two additional dimensions in the model.

3.2 A $2k$ -dimensional model for k -bit flip-flop tasks

In this section, we propose a $2k$ -dimensional model able to solve k -bit flip-flop tasks. For the sake of clarity, but without loss of generality, we show the $k = 2$ bit case. The key insight is to model the switching dynamics

⁴A bifurcation where both saddles collide with the corresponding stable points \mathbf{p}_2 and \mathbf{p}_4 , simultaneously annihilating all six fixed points.

between two fixed points in a two-dimensional space and then build a model formed by two independent systems with two-dimensional switching dynamics.

Unlike the previous example, here we want to tune the excitability of all connections. This can be obtained by imposing the condition in (40) and tuning the reservoir weights to change the position of the saddles. Therefore, we define

$$\mathbf{B} := \begin{bmatrix} 1.1 & 4 \\ -s & 4 \end{bmatrix}, \quad (18)$$

where s is a real parameter. Hence, for $0 \leq s < 2.15$, the autonomous dynamical system defined by $\mathbf{x}[k] = \tanh(\mathbf{B}\mathbf{x}[k-1])$ has one repeller (the origin), two attractors, namely $\mathbf{p}_1 \approx (1, 1)$, $\mathbf{p}_2 \approx (-1, -1)$, and two saddles. By increasing s within the $[0, 2.15)$ interval, we can make the saddles closer to the respective stable points, see Fig. 4, hence decreasing the excitability thresholds of these attractors, until a saddle-node bifurcation occurs approximately at $s = 2.15$, annihilating attractors with saddles.

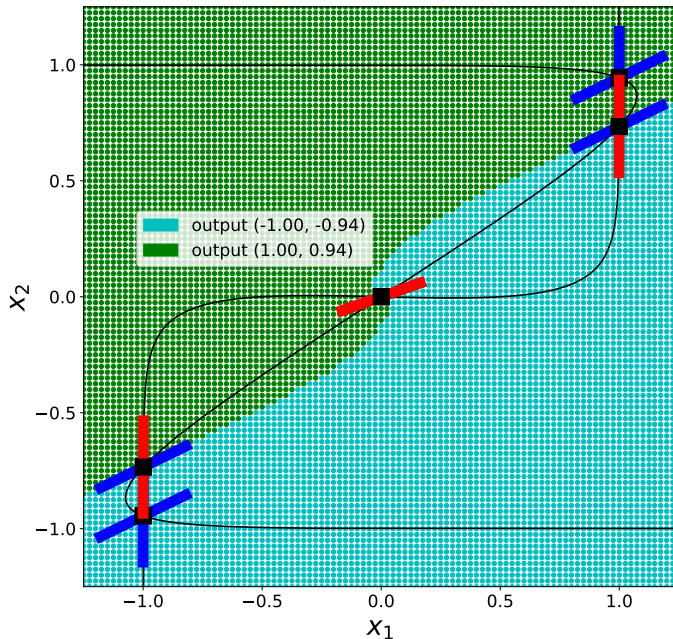


Figure 4: Basins of attraction, nullclines, fixed points and eigenvectors of the two-dimensional autonomous system having (18) as reservoir matrix, with $s = 2$. Black curves represent the nullclines (see Appendix B) whose reciprocal intersections determine fixed points (square black points). Red segments indicate real eigenvectors associated with real eigenvalues larger than one. Blue line segments represent real eigenvectors of real eigenvalues in $(0, 1)$. The origin has complex conjugated eigenvalues lying outside the unit circle and a pair of eigenvectors; in the figure, only the real-valued one is depicted. Points of the plane have been coloured according with the attractor on which they converged after 100 steps.

Let us define the input matrix as follows:

$$\mathbf{W}_{in} := \omega_{in} \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},$$

where ω_{in} is a positive real parameter. For instance, setting $s = 2$ and $\omega_{in} = 1$, the two-dimensional dynamical system defined by $\mathbf{x}[k] = \tanh(\mathbf{B}\mathbf{x}[k-1] + \mathbf{W}_{in}\mathbf{u}[k])$ is able to accomplish the switching mechanism between \mathbf{p}_1 and \mathbf{p}_2 according to inputs $(1, 0)$, $(-1, 0)$, and ignore other inputs, i.e., $(0, 1)$, $(0, -1)$. Of course, replacing zeros in \mathbf{W}_{in} with relatively small values (compared to ω_{in}) does not change the results.

The complete system able to solve the two-bit flip-flop dynamics can be obtained by defining the reservoir as the following four-dimensional block diagonal matrix,

$$\mathbf{M} := \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}, \quad (19)$$

where $\mathbf{0}$ denotes a matrix with all zeros. Starting from a given initial condition, for every $0 \leq s < 2.15$, the state of the four-dimensional autonomous dynamical system $\mathbf{x}[k] = \tanh(\mathbf{M}\mathbf{x}[k-1])$ converges to one of these four attractors $(\mathbf{p}_1, \mathbf{p}_1)$, $(\mathbf{p}_1, \mathbf{p}_2)$, $(\mathbf{p}_2, \mathbf{p}_1)$, $(\mathbf{p}_2, \mathbf{p}_2)$. In order to produce a two-dimensional output (2) suitable for the task at hand, we define $\mathbf{W}_o := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$, which basically corresponds to a projection onto the first and third component, i.e., $(x_1, x_2, x_3, x_4) \mapsto (x_1, x_3)$. Finally, we define the input matrix as

$$\mathbf{W}_{in} := \omega_{in} \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

We considered the case of a 4-dimensional system composed by two independent two-dimensional systems. Nevertheless, the dynamics remains qualitatively the same even if the coupling between these two-dimensional systems is weak, i.e., if the zero matrices in (19) are replaced by matrices with relatively small (absolute) values. With those definitions in mind and, as before, $\epsilon = 0$ and $\alpha = 1$, the ESN ruled by (5) and (2) correctly implements the flip-flop task with two bits.

As the dynamics of systems (x_1, x_2) and (x_3, x_4) are independent from each other, the set of fixed points of the overall dynamics turns out to be the Cartesian product of the set of fixed points of (x_1, x_2) and the fixed points of (x_3, x_4) system. This gives rise to a large number of fixed points: there are 4 stable points, 8 saddles with 1 unstable directions, 8 saddles with 2 unstable directions, 4 saddles with 3 unstable directions and 1 repeller. However, the set of fixed points where the nonautonomous flip-flop dynamics take place is formed by 4 stable fixed points and 8 saddles with only one unstable direction; see Sec. 5.1 for a detailed example. Every stable fixed point is close to a pair of saddles and, due to symmetry, they are all at the same distance $\delta_{th}(\mathbf{p}_1, \mathbf{p}_2)$. This quantity defines the excitability thresholds of the connections needed in the flip-flop task, and therefore implicitly defines the ENA ruling the behaviour of the four-dimensional ESN.

4 Extracting ENAs from the ESN trajectory

In this section, we describe the proposed algorithm to extract an ENA from an ESN trajectory. The proposed algorithm, which is schematically illustrated in Fig. 5, takes the trained reservoir matrix \mathbf{M} (6) and a trajectory of the nonautonomous ESN (5) as input and produces a weighted directed graph, representing the ENA describing the ESN behaviour, as output. The algorithm is composed of two main steps. In Sec. 4.1, we describe the procedure to find fixed points of the ESN dynamics (corresponding to the vertices of the graph). Successively, in Sec. 4.2, we show how to determine the excitable connections and related thresholds between stable fixed points (corresponding to the weighted directed edges).

4.1 Finding fixed points of the dynamics

The optimisation algorithm we devised in order to find fixed points is based on Sussillo and Barak [56]; see [16] for an open-source Tensorflow toolbox for finding fixed points in arbitrary RNN architectures and [25] for an alternative method to identify fixed points. The key idea is to define a scalar function, whose minima correspond to fixed points of the ESN dynamics. We call velocity field the vector field $\mathbf{Q} : \mathbb{R}^{N_r} \rightarrow \mathbb{R}^{N_r}$, defined as

$$\mathbf{Q}(\mathbf{x}) := \mathbf{F}(\mathbf{x}) - \mathbf{x}, \quad (20)$$

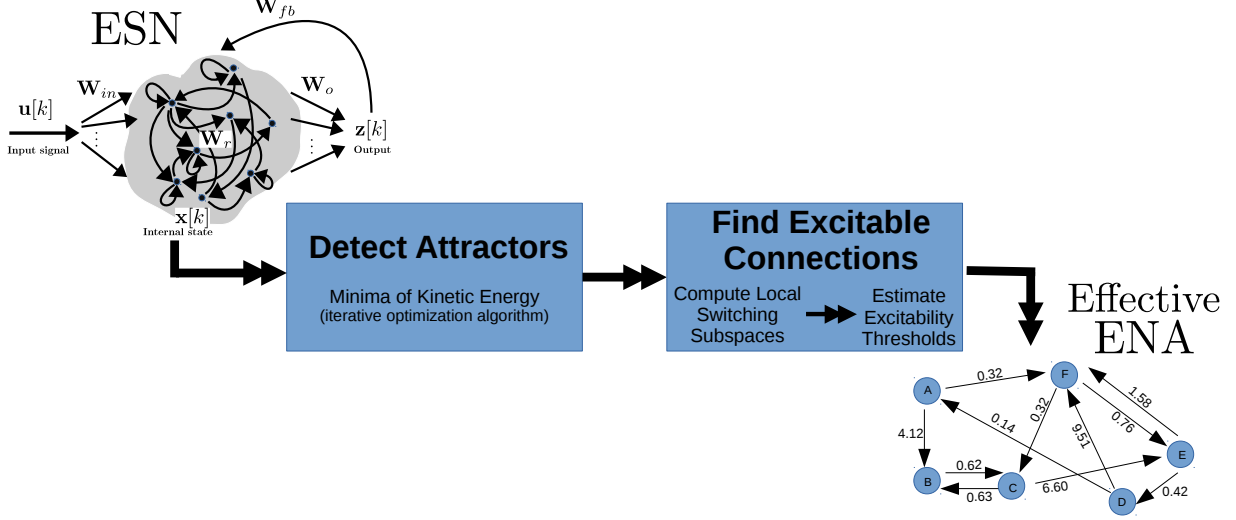


Figure 5: Illustration of the proposed method to extract ENAs from trajectories.

with \mathbf{F} being the map in Eq. 9. Therefore, the velocity field takes the following form:

$$\mathbf{Q}(\mathbf{x}) = \alpha [\tanh(\mathbf{M}_{(i)} \cdot \mathbf{x}) - \mathbf{x}], \quad (21)$$

where $\mathbf{Q}(\mathbf{x}[k])$ is the vector that needs to be added to the current state $\mathbf{x}[k]$ to obtain the next one. In fact,

$$\mathbf{x}[k+1] = \mathbf{F}(\mathbf{x}[k]) \iff \mathbf{x}[k+1] - \mathbf{x}[k] = \mathbf{F}(\mathbf{x}[k]) - \mathbf{x}[k] \iff \mathbf{x}[k+1] = \mathbf{x}[k] + \mathbf{Q}(\mathbf{x}[k]). \quad (22)$$

We define a “kinetic energy” of the autonomous system (9) as the following scalar function,

$$q(\mathbf{x}) := \frac{1}{2} \|\mathbf{Q}(\mathbf{x})\|^2. \quad (23)$$

By this definition, it holds that fixed points $\mathbf{x}^* \in \mathbb{R}^{N_r}$ satisfy $\mathbf{Q}(\mathbf{x}^*) = \mathbf{0}$ if and only if $q(\mathbf{x}^*) = 0$. Fixed points are hence identified as the global minima of (23). We used the well-known quasi-Newton algorithm called BFGS [43] to minimise (23). In order to speed-up the optimisation by several orders of magnitude, we explicitly provided the gradient of (23) to BFGS, which reads:

$$\nabla q(\mathbf{x}_0) = \mathbf{J}_{\mathbf{Q}}(\mathbf{x}_0)^T \mathbf{Q}(\mathbf{x}_0) = \alpha (\mathbf{D}(\mathbf{x}_0) \mathbf{M} - \mathbf{I}_{N_r})^T \mathbf{Q}(\mathbf{x}_0), \quad (24)$$

where \mathbf{I}_{N_r} is an $N_r \times N_r$ identity matrix, $\mathbf{J}_{\mathbf{Q}}(\mathbf{x}_0)$ denotes the Jacobian matrix of the velocity field (21) and $\mathbf{D}(\mathbf{x}_0)$ is a diagonal matrix defined in (29) of Appendix A, both evaluated on \mathbf{x}_0 .

The initial conditions for minimising (23) are determined by randomly sampling states from a trajectory of the nonautonomous ESN (5). The convergence of the BFGS algorithm depends on a tolerance. In fact, the algorithm might converge to similar solutions that, depending on chosen tolerance, are numerically different. As these solutions represent fixed points of the dynamics, we aggregate them in a meaningful way and return a non-redundant set of fixed points. For this purpose, as post-processing step, we run a clustering algorithm on the set of all solutions and return only cluster representatives; details are provided in Appendix C.

4.2 Determining excitable connections between attractors

Once stable fixed points have been identified, we determine the excitable connections between them. As discussed before (and in more detail in Appendix B), the ESN training (6) induces bifurcations that generate

new fixed points; and possibly also other attracting regions in phase space that, however, are not explicitly modelled in this work. The ESN is driven by control inputs that allow us to correctly perform switches between stable states. As a consequence, we first need to understand how such inputs affect the dynamics from a geometric point of view. This is done in Sec. 4.2.1 by introducing the notion of local switching subspace (LSS), which is strictly related to the notion of local input response set, Eq. 14, introduced in Sec. 2.3. Then, in Sec. 4.2.2 we describe how we determine all excitable connections that are relevant for the task under consideration and compute their excitability thresholds. The method we propose is based on a grid of points lying in the LSS, accounting for the input action on the dynamics. By simulating the autonomous dynamics with initial conditions taken from such a grid, we are able to approximate input-driven excitability thresholds (15) and also to quantify how likely it is that the RNN uses such connections while solving the task.

4.2.1 Local switching subspaces

Let us consider an ESN trajectory (5), $\mathbf{x}[0], \mathbf{x}[1], \mathbf{x}[2], \dots, \mathbf{x}[k], \dots$, obtained with inputs $\mathbf{u}[1], \mathbf{u}[2], \dots, \mathbf{u}[k+1], \dots$. Moreover, let us denote with $\mathcal{K} := \{k_i\}_{i \in \mathbb{N}}$ the set of indices for which $\mathbf{u}[k_i] \neq \mathbf{0}$. We define the pulse difference vector (PDV) as a vector containing the difference between pre- and post-input states, namely $\mathbf{x}[k_i] - \mathbf{x}[k_i - 1]$, $k_i \in \mathcal{K}$. The PDVs convey relevant information about the action of inputs on ESN state and therefore we will exploit such information to determine the excitable connections and related thresholds. We remind the reader that the number of non-zero inputs is controlled by the parameter p of the exponential distribution (see Eq. 1.2 and related discussion in the text), which in turn determines the (average) number of PDVs available for the following analysis.

In order to compute only those connections that are actually used by the ESN while solving the task, we need to focus on the action of inputs in the neighbourhood of each attractor. To this end, we consider an Euclidean ball $B_r(\mathbf{p})$ of radius $r \geq 0$ centred on an attractor \mathbf{p} , and call local PDVs of \mathbf{p} the finite set of PDVs that originate inside $B_r(\mathbf{p})$. Therefore, referring to Eq. 14, the local PDVs of \mathbf{p} is a collection of vectors originating in $B_r(\mathbf{p})$ and ending in $\mathcal{G}(B_r(\mathbf{p}))$. For each attractor \mathbf{p} , we perform principal components analysis of the related local PDVs and retain only $l \ll N_r$ principal components. The projection along such l basis vectors defines a vector space that we denote as $\mathbb{L}(\mathbf{p})$. We define the LSS of \mathbf{p} as the affine space composed by attaching the vector space $\mathbb{L}(\mathbf{p})$ over \mathbf{p} , and we denote it as $\mathbf{p} + \mathbb{L}(\mathbf{p})$; see Fig. 6, top panel.

4.2.2 Estimation of excitability thresholds

The idea is to sample the LSS of a stable point \mathbf{p}_i and describe it as a grid of points $G(\mathbf{p}_i)$. Then, we consider those points on the grid as initial conditions for the autonomous system (9), which is then iterated for a sufficiently large number of steps to ensure convergence to nearby attractors. Finally, tracing the evolution of these initial conditions allows us to estimate excitability thresholds and other relevant quantities.

The proposed algorithm for finding excitability thresholds is graphically illustrated in Fig. 6. The algorithm is based on a hypercube $H(\mathbf{p}_i)$ centred on attractor \mathbf{p}_i that is contained within the LSS $\mathbf{p}_i + \mathbb{L}(\mathbf{p}_i)$. The length of the hypercube is such that $H(\mathbf{p}_i)$ contains the projection of $\mathcal{G}(B_r(\mathbf{p}_i))$ on $\mathbf{p}_i + \mathbb{L}(\mathbf{p}_i)$. In order to estimate excitability thresholds, we consider a mesh with a pre-defined density of points on the hypercube $H(\mathbf{p}_i)$, thus obtaining a grid of points, $G(\mathbf{p}_i) = \{\mathbf{g}_j^i\}$. To simplify the notation, we use a single index j to enumerate points of the grid. Through the ω -limit set (10) of the grid $G(\mathbf{p}_i)$, that is:

$$\omega_{\mathbf{F}}(G(\mathbf{p}_i)) = \bigcup_j \omega_{\mathbf{F}}(\mathbf{g}_j^i),$$

we can define the following positive integer $c(\mathbf{p}_i) := |\omega_{\mathbf{F}}(G(\mathbf{p}_i))| - 1$, which counts the number of excitable connections originating from \mathbf{p}_i . Indeed, $\omega_{\mathbf{F}}(G(\mathbf{p}_i))$ is composed of a set of stable fixed points, $\{\mathbf{p}_{i_0}, \mathbf{p}_{i_1}, \mathbf{p}_{i_2}, \dots, \mathbf{p}_{i_{c(\mathbf{p}_i)}}\}$, determining the endpoints of the $c(\mathbf{p}_i)$ different excitable connections. Note that if the grid is sufficiently dense, then the attractor itself is always included in such a set of fixed points. For that reason, it must not be counted as an excitable connection. In what follows, we assume that the attractor is the one indexed by i_0 , i.e., $\mathbf{p}_{i_0} = \mathbf{p}_i$.

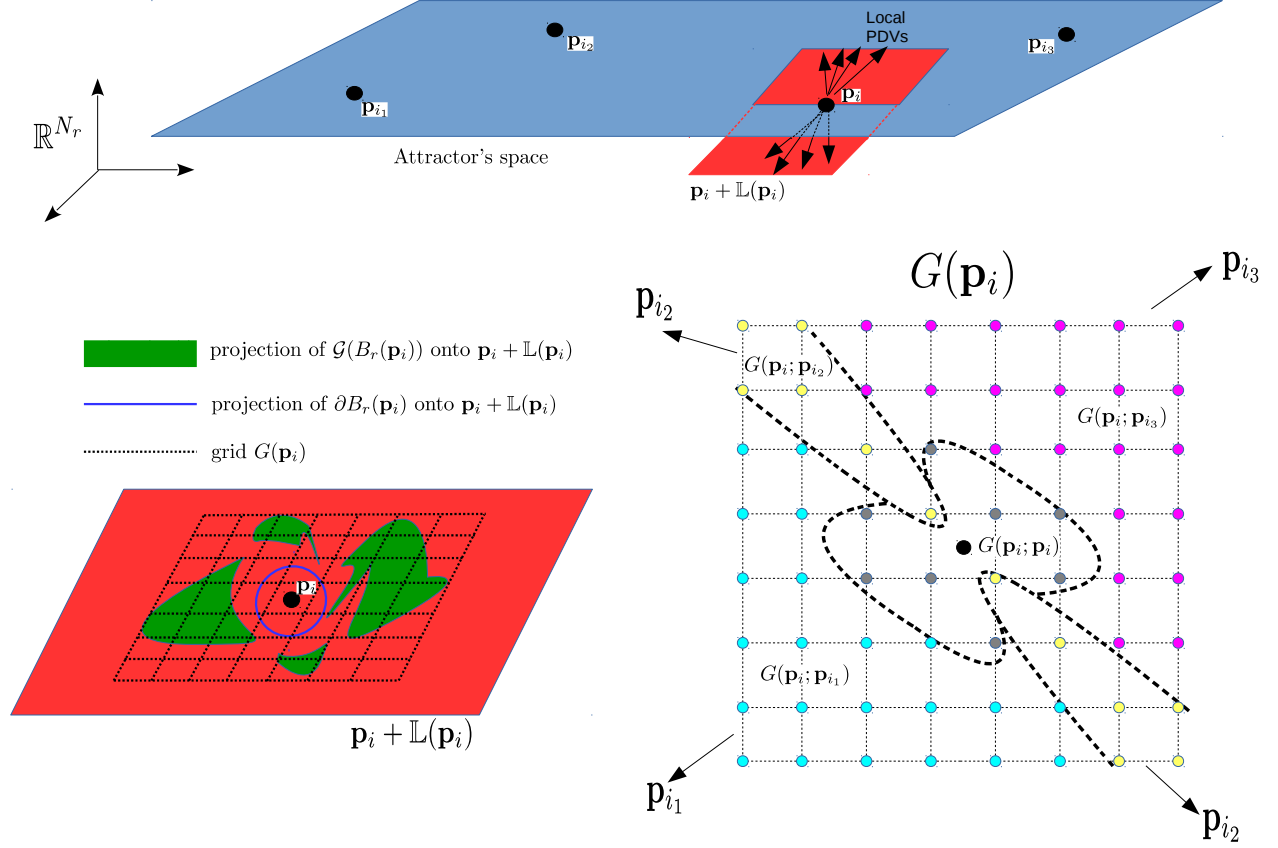


Figure 6: **Top:** in blue, the low-dimensional space containing the attractors; in red, a sketch of the LSS of attractor \mathbf{p}_i . **Bottom left:** representation of the local input response set $\mathcal{G}(B_r(\mathbf{p}_i))$ and the grid $G(\mathbf{p}_i)$ in the LSS of \mathbf{p}_i . **Bottom right:** illustration of the partition of a two-dimensional grid $G(\mathbf{p}_i)$. Every colour represents a subset (25). Dashed black lines track the boundary of the basins corresponding to those attractors reachable from \mathbf{p}_i through excitable connections, which can be enabled by inputs allowing to explore the hypercube centred on \mathbf{p}_i .

With these definitions in mind, we are ready to compute thresholds of excitable connections. To this end, let us denote with

$$\sigma_i : \{1, \dots, |G(\mathbf{p}_i)|\} \longrightarrow \{i_0, i_1, \dots, i_{c(\mathbf{p}_i)}\},$$

an indexing function such that $\mathbf{p}_{\sigma_i(j)} = \omega_{\mathbf{F}}(\mathbf{g}_j^i)$. Then, through the preimage $\sigma_i^{-1}(\cdot)$, we obtain a partition of points of the grid into the following subsets:

$$G(\mathbf{p}_i; \mathbf{p}_{i_t}) := \{\mathbf{g}_j^i \in G(\mathbf{p}_i) \mid j \in \sigma_i^{-1}(i_t)\}, \quad t = 0, 1, \dots, c(\mathbf{p}_i). \quad (25)$$

The points of the grid belonging to the subset defined by (25) are all destined to converge to \mathbf{p}_{i_t} . Therefore, we estimate the excitability threshold (15) of the connection from \mathbf{p}_i to \mathbf{p}_{i_t} as follows:

$$\tilde{\delta}_{\text{inp}}(\mathbf{p}_i, \mathbf{p}_{i_t}) = \min \{\|\mathbf{g}_j^i - \mathbf{p}_i\| \mid \mathbf{g}_j^i \in G(\mathbf{p}_i; \mathbf{p}_{i_t})\}, \quad t = 1, \dots, c(\mathbf{p}_i). \quad (26)$$

The excitability thresholds (26) represent geometric properties of the attractors and related basins in phase space learned through training. In order to determine the effective excitability (accounting for inputs)

of each connection outgoing from \mathbf{p}_i , we exploit the local topology of the LSS of \mathbf{p}_i by means of the grid partition (25) induced by $\sigma_i(\cdot)$. To this end, we define the ratio of initial conditions taken from the grid that converged to attractor \mathbf{p}_{i_t} as follows:

$$\nu_{i,i_t} := \frac{|G(\mathbf{p}_i; \mathbf{p}_{i_t})|}{|G(\mathbf{p}_i)| - |G(\mathbf{p}_i; \mathbf{p}_i)|} \in [0, 1]. \quad (27)$$

In the limit of infinite number of points in the grid, the quantity (27) converges to the ratio of volumes between the portion of the hypercube belonging to the basin of \mathbf{p}_{i_t} and the portion of the hypercube that does not belong to the basin of \mathbf{p}_i . Therefore, dense grids give ratios (27) providing accurate information about how the LSS is distributed between basins of attraction of stable fixed points. Finally, merging both (26) and (27) into a single expression, we define effective excitability of the connection from \mathbf{p}_i to \mathbf{p}_{i_t} as follows:

$$\beta_{i,i_t} := \frac{\nu_{i,i_t}}{\tilde{\delta}_{\text{inp}}(\mathbf{p}_i, \mathbf{p}_{i_t})}. \quad (28)$$

Note that this quantity takes into account both the distance between the attractor \mathbf{p}_i and the basin's boundary, and the volume of the basin itself. A low value for β_{i,i_t} indicates that it is difficult to activate the connection from \mathbf{p}_i to \mathbf{p}_{i_t} during the task. This may be due (i) to the small volume occupied by the basin of the attractor \mathbf{p}_{i_t} in the LSS of \mathbf{p}_i or (ii) to a very high excitability threshold associated with such a connection. On the other hand, $\beta_{i,i_t} \gg 1$ necessarily implies that such a connection has a low excitability threshold, since $\nu_{i,i_t} \in [0, 1]$. As a consequence, the distance between the basin of attraction of \mathbf{p}_{i_t} and \mathbf{p}_i is small, thus the connection can be easily activated during the task.

Remarks on computational complexity There are three parameters controlling the complexity and, accordingly, the accuracy of the search in the grid: the dimension ζ_1 of the hypercube, the length ζ_2 of the hypercube's edge, and the number of points ζ_3 on each edge determining the density of the grid. ζ_3 and ζ_2 have a linear and polynomial impact on the computational complexity of the algorithm, respectively. However, ζ_1 is more critical as it increases exponentially the number of points in the grid and, accordingly, the overall complexity. In the simulations we always set $\zeta_1 = l$, that is, the dimension of the hypercube is equal to the dimension of the LSS, which in our case is always a low-dimensional subspace. More generally, the dimension of LSSs depends on the complexity of the inputs and their impact on the dynamics, which needs to be assessed on a case-by-case basis.

5 Simulations

In this section, we discuss the results of simulations, highlighting different aspects of our theoretical developments. In Sec. 5.1, in order to evaluate the correctness of the algorithm proposed in Sec. 4, we apply it to the manually-designed, low-dimensional ESN maps discussed in Sec. 3.1 and Sec. 3.2. Successively, in Sec. 5.2, we apply our method to high-dimensional trained ESNs. First, we show that, even though the ESN is high-dimensional, the dynamics is always low-dimensional due to the low-rank perturbation introduced during training. Second, we show the usefulness of ENAs for giving a mechanistic interpretation of prediction errors occurring during the task execution. Finally, in Sec. 5.3, we show how ENA models can be used to assess the robustness to noise of trained ESNs.

In all simulations, we control inputs with $p = 0.1$ and set the tolerance of the kinetic energy (23) for detecting minima to 10^{-6} .

5.1 Evaluation on manually designed low-dimensional ESNs

For the grid search algorithm, we used $\zeta_1 = 2$, $\zeta_2 = 4$, and $\zeta_3 = 223$.

Minimal-dimension model The LSS, determined as described in Sec. 4.2.1, corresponds to the whole 2D phase space. As a consequence, excitability thresholds (12) match the corresponding input-dependent counterparts (15). For each attractor, the Euclidean distances from the two closest saddles are consistent with the estimation of excitability thresholds provided by our method. As discussed in Sec. 3.1 and graphically represented in Fig. 7, bottom-centre panel, we find three excitability thresholds in the computed ENA: 0.49, 1.39, and 1.42. In Fig. 7, bottom-right panel, we show that undesired connections (i.e., connections corresponding to state transitions that are not defined in the flip-flop task) between fixed points $(0.97, -0.97)$ and $(-0.97, 0.97)$ have very low effective excitability values (28), indicating that it is unlikely to use such connections during the task execution. The low effective excitability is due to the small volume ratio (27) of the basins associated with the two attractors. On the other hand, larger thresholds characterise the undesired connections from $(-1, -1)$ to $(1, 1)$, reflecting the disproportion between basin volumes of $(1, 1)$, $(-1, -1)$ and $(0.97, -0.97)$, $(-0.97, 0.97)$.

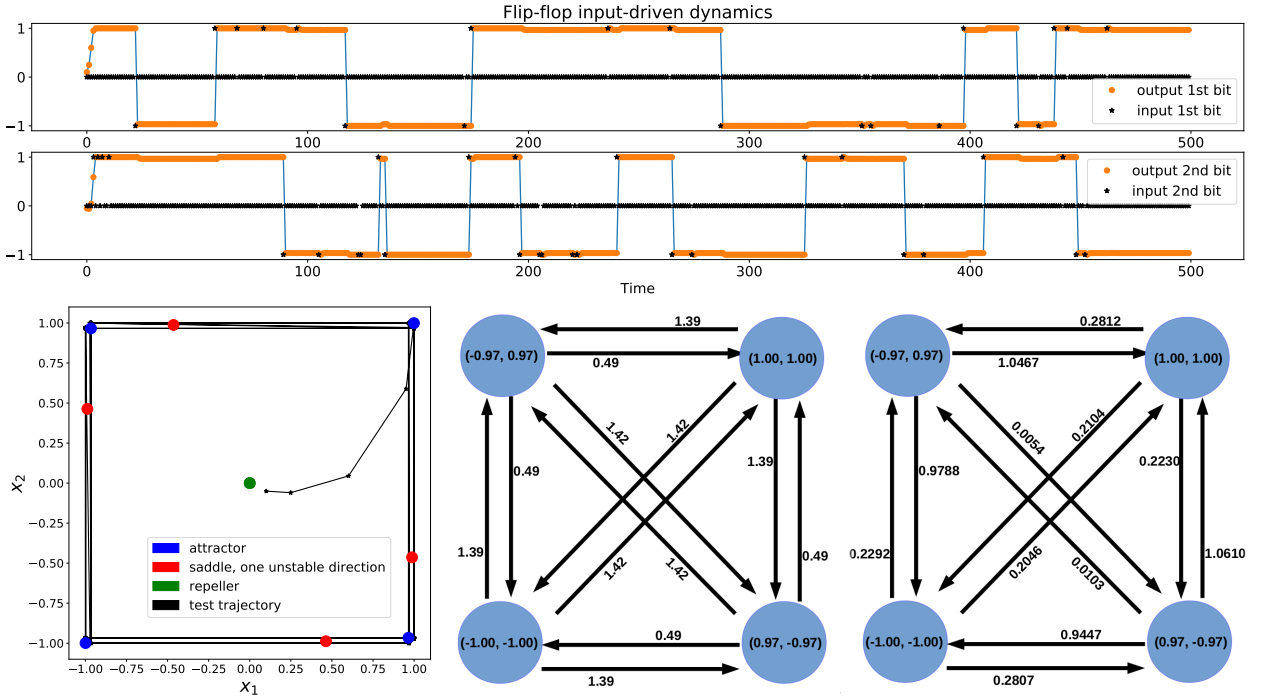


Figure 7: **Top:** output produced by the two-dimensional ESN defined in Sec. 3.1. **Bottom left:** fixed points found by the optimisation algorithm with 100 initial conditions. Length of the test trajectory was 1000 steps. **Bottom centre:** extracted ENA with edges weighted by excitability thresholds (26). **Bottom right:** extracted ENA with edges weighted according to (28). Node labels represent output values (2) produced on the attractors.

Four-dimensional model The first two principal components are related to all identified fixed points and produce a cumulative variance ratio larger than 0.96; Fig. 8 shows a trajectory of the map described in Sec. 3.2 and related fixed points projected on the plane spanned by the two principal components. For every attractor, the computed LSS is a two-dimensional plane; this is expected, since it is actually the plane depicted in Fig. 4. Furthermore, we note that none of these planes is aligned with the one where the attractors lie, stressing the importance of defining reference frameworks local to each attractor that take the action of inputs into account. Fig. 8, middle-right panel, shows how the input moves the states out of the plane, using additional dimensions for the switches. The computed ENA, weighted with excitability thresholds (26) and effective excitability (28), is shown in Fig. 8, bottom-left and bottom-right panels,

respectively. Symmetries of the dynamical system are clearly present in the resulting directed graphs. All desired connections are characterised by an excitability threshold of $\simeq 0.83$, while undesired ones have higher thresholds equal to $\simeq 1.19$. We note that the presence of undesired connections does not imply that the ESN actually used such connections during the task execution. To this end, the effective excitability thresholds (28), shown in the bottom-right panel of Fig. 8, provide us with a more realistic picture of the behaviour under the action of inputs. In fact, effective excitability thresholds are very low for undesired connections, implying that the LSS used by inputs is mostly occupied by basins corresponding to attractors adjacent to the end-point of desired connections.

5.2 Application of the proposed method to high-dimensional trained ESNs

From now on, all results are assumed to be obtained using ESNs (5) with a reservoir composed of 500 neurons. Training ESNs with different reservoir sizes suggested us that our theoretical framework does not depend on the number of neurons (results not reported), so our choice is mostly dictated by hardware and computing time considerations. For the grid search algorithm, we used $\zeta_1 = 3, \zeta_2 = 18,$ and $\zeta_3 = 12$. The state-update (5) is configured without leakage, $\alpha = 1$; the standard deviation of noise ϵ is set to 10^{-4} during training. Entries of matrices $\mathbf{W}_{in}, \mathbf{W}_{fb},$ and \mathbf{W}_r are i.i.d. drawn from a uniform distribution in $[-1, 1]$; the sparseness of \mathbf{W}_r is 95%. Moreover, the reservoir matrix was rescaled to obtain a spectral radius equal to 0.9. Finally, the training set length is always 50000 time-steps.

Low-dimensional dynamics Fig. 9, top panel, shows the output produced by an ESN achieving high prediction performance. The extracted ENA, shown in the bottom-right panel, reveals that undesired connections have excitability thresholds (15) significantly higher than those of desired connections. This means that, in the LSS of every stable point, basins corresponding to attractors adjacent to the end-point of undesired connections stand relatively far away from the stable point compared to basins of attractors adjacent to the end-point of desired connections. The fixed points of the dynamics lie in a two-dimensional subspace of the 500-dimensional phase space (the cumulative variance ratio is close 1). It is observed that trajectory spend most of the time on such plane, hence, based on principal components analysis of the trajectory, we can claim that the dynamics is bidimensional. Nevertheless, during the task execution, the trajectory is occasionally driven away from such a 2D plane by the inputs in order to achieve the switches between attractors, and this feature is crucial to understand how the trained neural network behaves for the purpose of solving the task. The cumulative variance ratio for the identification of the LSS of every attractor, as described in Sec. 4.2.1, revealed that the switching between stable fixed points takes place in a three-dimensional subspace of the phase space, highlighting that the overall dynamics of ESNs is low-dimensional after training. It is worth stressing that such LSSs are usually not aligned with the standard coordinate system of the original phase space and the subspaces where attractors lie, suggesting that inputs operate in phase space regions that are disjoint with respect to the low-dimensional linear subspace of the attractors; this is consistent with what is reported in [56].

Computation accuracy and spurious attractors Various measures of accuracy exist for quantifying the performance on prediction tasks. For instance, the mean-squared-error (MSE) is typically adopted in tasks involving continuous targets. The MSE is a real-valued scalar that informs us about how close the computed output is to the target one. However, it is not possible to infer additional insights by looking only at the MSE; for instance, it is not possible to provide a mechanistic description of errors in the computation, i.e., why and how they occur. Here, we show how the effective ENA extracted from the ESN trajectory can be used to describe how the computation takes place in phase space and, in particular, how to diagnose the nature of prediction errors.

The top panel in Fig. 10 shows the output produced by an ESN achieving low MSE of the order of 10^{-3} . The small errors observable around time-step 13200 can be explained by looking at the ENA model depicted in the bottom panels of Fig. 10, which is represented with excitability thresholds (26), left panel, and with effective excitability values (28), right panel. The directed graphs (whose topology is clearly identical) reveal

the presence of two extra stable fixed points in the ESN phase space, which are generated during training. Nodes of these graphs are coloured according to output values produced by the ESN. The activation of the related excitable connections brings the ESN to operate in the proximity of such superfluous states, producing inaccurate output values and hence explaining the origin of such errors. Notably, the directed edge – in the graph on the right-hand side – connecting the cyan with the yellow node has a relatively high value of effective excitability. This means that, in the LSS of the related attractor (cyan), whose output is $(-1.03, -1.07)$, it is relatively easy, starting nearby the attractor, to transition inside the basin of the other attractor (yellow). This, in turn, produces an output value equal to $(-0.84, -0.40)$, which is significantly different from the target output, i.e., $(-1, -1)$. The prediction error, visible to the naked eye in the top panel around time-step 13200, is indeed due to the activation of that excitable connection.

It is worth emphasizing that, both spurious attractors act as a sort of surrogates for the correct ones (i.e., those producing lower prediction errors). In fact, once the internal state switches to a spurious attractor, the ESN still behaves consistently. More precisely, spurious attractors are associated with higher effective excitability values on connections ending up in the correct attractors – see the bottom-right panel of Fig. 10. The discovery of such extra attractors and the unravelling of their roles in the ESN computation could not be inferred by looking only at the MSE or plotting the outputs produced by the ESN, hence stressing the importance of ENA models for describing the ESN behaviour.

5.3 Noise tolerance and effective excitability of ENAs

Here, we aim to provide a further example of the importance of ENAs for characterising the computation of ESNs. In particular, we ask the following question: given a set of ESNs trained on the same task and achieving the same performance during training, which one will be more robust to noise during test? Typical performance measures, such as MSE, cannot be used to answer such a question and provide useful insights. We show that an ENA model of the computation, weighted with effective excitability values (28), allows us to assess robustness to noise of ESNs.

As a perturbation, we consider the usual white Gaussian noise term ϵ in the state-update (5), corresponding to perturbations directly applied to all neuron pre-activation values. We stress that such perturbations are applied only during the test phase; training is implemented as described in the previous section. By increasing the noise standard deviation, the ESN trajectory gets increasingly perturbed neglecting the possibility to reach the proximity of attractors, hence affecting the accuracy of the resulting output values.

We note that: (i) ESNs yielding ENAs with higher effective excitability values are less robust to noise perturbations than ESNs giving rise to ENAs with low effective excitability values (see [6]), and (ii) ESNs producing ENAs with balanced edge weights on desired outgoing connections are more robust to noise perturbations than ENAs with unbalanced outgoing weights. Regarding (i), high excitability values imply the existence of connections with low excitability thresholds. That is, the basin of the attractor corresponding to the end-point of such a connection is very close to the attractor associated with the origin of the connection, resulting in unnecessary switches that induce errors. Concerning (ii), for a given attractor, unbalanced outgoing connection weights could be a symptom of unbalanced distribution of space between basins of attraction in the LSS or the existence of some basins significantly closer to the attractor than other basins. In the latter case, a reasoning similar to (i) applies. On the other hand, if there is indeed a disproportionate distribution of volumes between basins, then basins of attractors corresponding to connections possessing large volume ratios will fill most of the LSS, leaving little space for basins of those attractors related to connections with small volume ratios. Therefore, if the ESN state is close to an attractor with unbalanced outgoing connection weights, but with similar excitability thresholds, then under noise perturbations it is more likely to switch towards an attractor reachable through a connection with high effective excitability even when such a connection should not be activated. For these reasons, an ENA where each node is characterised by balanced weights on outgoing connections and very low effective excitability values on undesired connections, provides a prototypical example of ESNs that are robust to noise.

To provide a quantification of what has been discussed so far, we selected two ESNs that solve the two-bit flip-flop task with very high and comparable accuracy – MSE during training is $\simeq 10^{-4}$. Successively, we tested them on the same test series composed of 10^5 time-steps and recorded their MSEs by considering

different noise instances with increasing standard deviation. Directed graphs representing the extracted ENAs are shown in Fig. 11. Results for increasing noise standard deviation are divided in two columns: on the left column, we report results obtained by the ESN that is least tolerant to noise. The directed graph on the left-hand side of Fig. 11 shows the presence of two undesired connections. Furthermore, edge weights of desired connections are not balanced, especially those outgoing from attractors with output values $(1, 1)$ and $(-1, -1)$. Conversely, the directed graph on the right-hand side does not contain undesired connections and possess balanced outgoing weights. The absence of undesired connections indicates that, in the LSS of every attractor, the basins of the attractors corresponding to undesired connections do not exist or, alternatively, they are very small and hence not detectable with the grid density used in our simulations; therefore, they are not relevant for describing the ESN behaviour according with the numerical precision we considered in our simulations. Finally, we highlight that a performance breakdown for the ESN on the left-hand side panel is observed starting from noise standard deviation of 8×10^{-2} . On the other hand, the ESN on the right-hand side is significantly more robust to noise, denoting a performance breakdown for noise standard deviation of 1.4×10^{-1} .

6 Conclusions

In this paper, we presented a novel methodology for modeling and interpreting the behaviour of RNNs driven by inputs. In order to obtain a mechanistic model describing how RNNs solve tasks, we exploited the theoretical framework offered by excitable network attractors, which are defined as networks of stable fixed points connected by excitable connections. Here, we introduced a procedure to extract excitable network attractors directly from a trajectory generated by a trained RNN. Such a procedure is composed of two main steps: first, fixed points are computed by solving a non-linear optimisation problem, and successively excitable connections, with related thresholds, are determined by simulating the dynamics of the autonomous system.

We validated our theoretical developments by considering echo state networks trained on the flip-flop task, a simple yet relevant benchmark that consists of learning a prescribed number of stable states and related switching patterns guided by control inputs. Simulation results provided several interesting insights on how RNNs solve tasks and highlighted the usefulness of excitable network attractors in describing how RNN undertake computations. Here, we trained echo state networks by means of ridge regression. Our results (not shown here) denote that the regularisation parameter has a direct impact on the number of attracting regions in phase space generated through training: using too low values produces under-regularised models with a large number of attractors. An interesting future perspective consists of studying the impact of different training mechanisms (e.g., via FORCE learning or similar online approaches) on the resulting network attractor.

We believe that the proposed modelling framework based on excitable network attractors will be suitable to describe the RNN behaviour for all tasks requiring the learning of a number of attractors (which need not be stable fixed points) and related switching patterns. To this end, in future we will also examine classification tasks. A related next step includes the possibility to handle inputs that are not instantaneous pulses, opening the way to more interesting case studies of practical relevance. As mentioned in the introductory sections network attractors can in principle be constructed between any type of invariant sets, including limit cycles and strange attractors. Our focus on fixed points was mainly dictated by the fact that computing fixed points from a trajectory is significantly easier than computing limit cycles, for instance. Obviously, fixed points are not powerful enough to accurately model all possible behaviours in phase space. Therefore, an interesting future perspective consists in extending our modeling framework to handle networks composed of heterogeneous attractors, such as a network in phase space connecting fixed points and limit cycles. In turn, this will allow modeling more complex RNN behaviour. Finally, future directions include embedding the directed graph representing the excitable network attractor extracted from the trajectory in a new phase space, thus producing a set of ordinary differential equations [4] describing the RNN behaviour on the task under consideration.

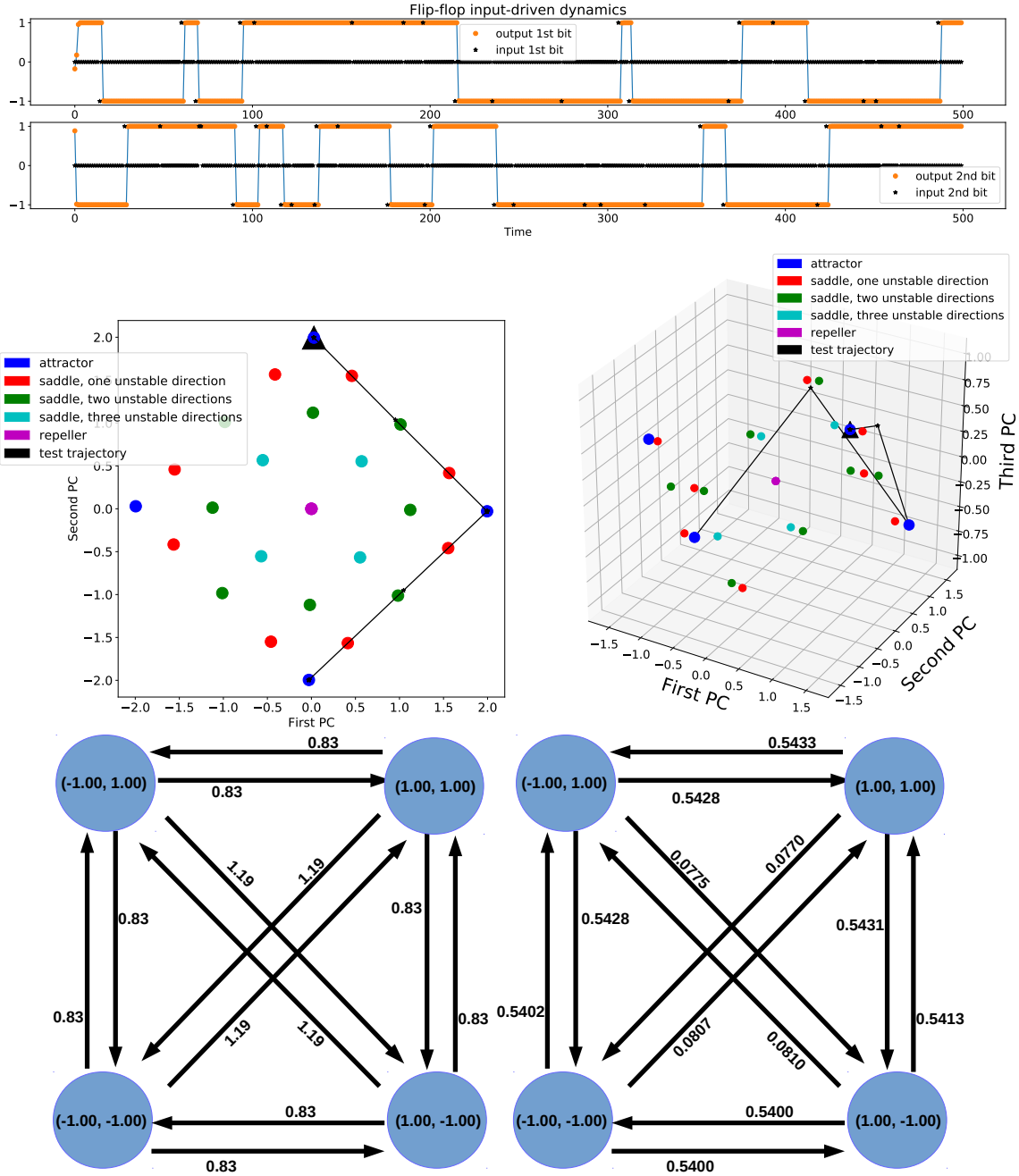


Figure 8: **Top:** output of the four-dimensional ESN defined in Sec. 3.1. **Centre left:** fixed points found by the optimisation algorithm with 200 initial conditions, depicted on the space spanned by the first two principal components. In black, it is shown a trajectory starting on the attractor (marked by a black triangle); to improve readability, the trajectory is limited to two switches only. **Centre right:** fixed points and trajectory depicted in the centre-left picture are embedded in the space spanned by the three principal components. **Bottom left:** ENA with edge weights representing excitability thresholds (26). **Bottom right:** ENA with edge weights representing the effective excitability of connections (28). Node labels represent output values produced by the ESN on the attractors.

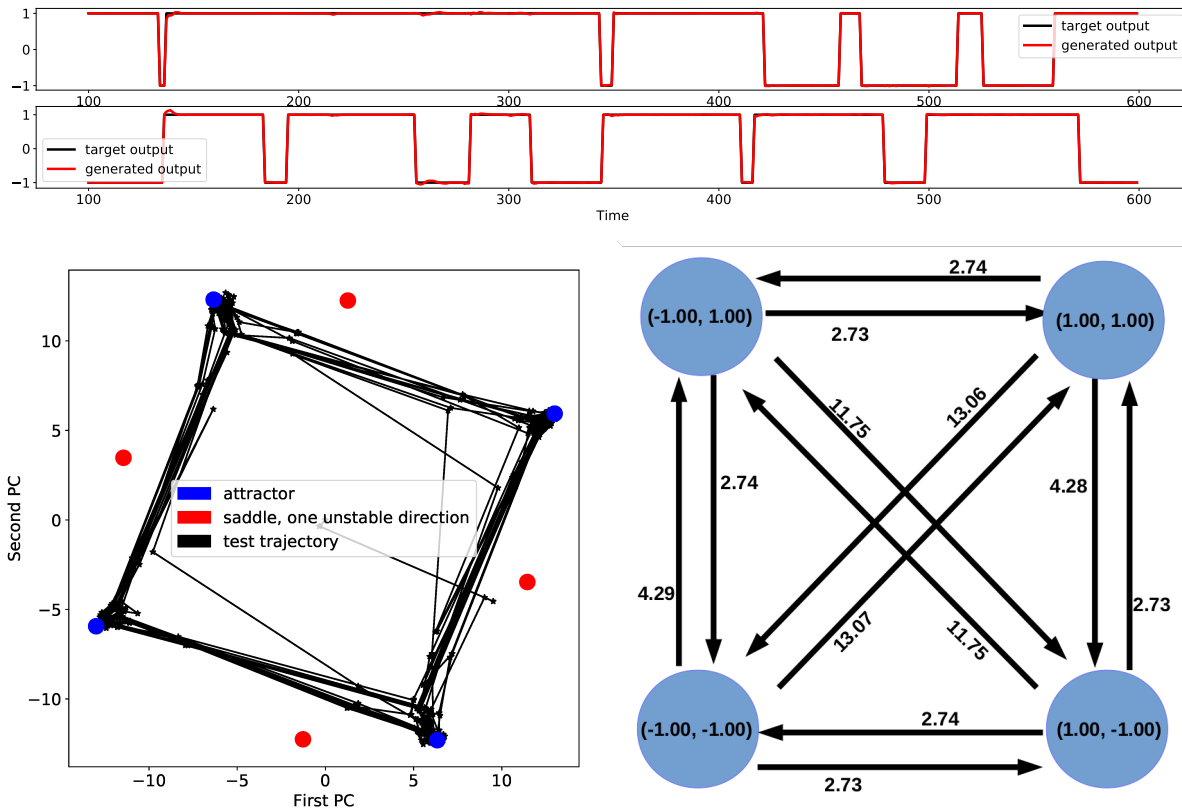


Figure 9: **Top:** output produced by a trained ESN and target output. **Bottom left:** fixed point found by means of the optimisation algorithm with 500 initial conditions. The plane in the picture represents the space spanned by the first two principal components determined over all 500 solutions returned by the optimisation algorithm. **Bottom right:** extracted ENA with edge weights representing excitability thresholds (26). Node labels represent output values produced by the ESN (2) on the attractors.

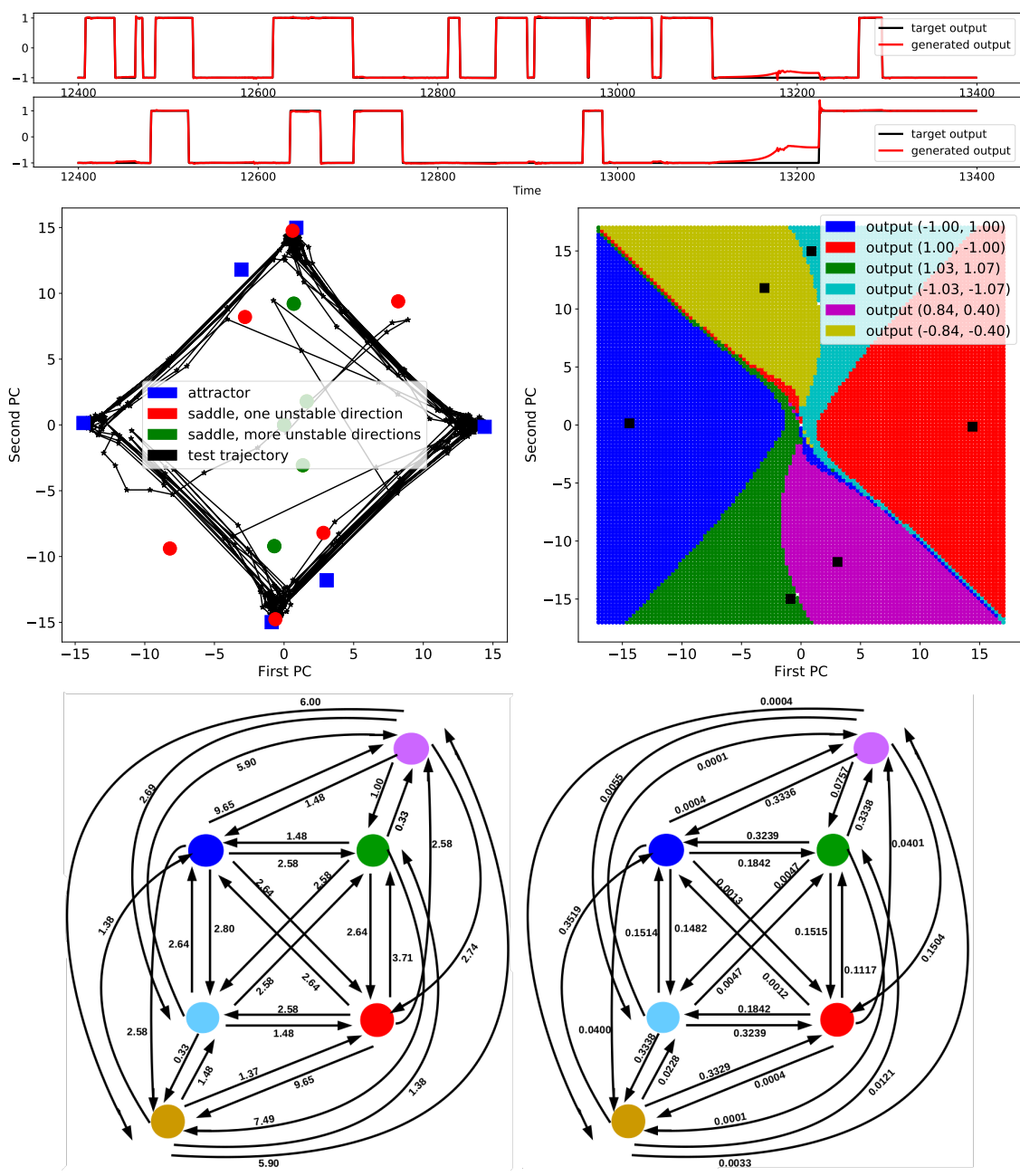


Figure 10: **Top:** output of the trained ESN during an incorrect switch around step 13200. **Center left:** fixed points found by the optimisation algorithm plotted in their two-dimensional subspace together with the trajectory. **Center right:** attractor plane divided by basins of attraction of every stable fixed point. The figure is drawn assuming a transient of 300 time-steps. Every basin is coloured depending on the attractor where the ESN converges to without applying any input. **Bottom left:** extracted ENA weighted with estimation of input-driven excitability thresholds (26). **Bottom right:** extracted ENA weighted with effective excitability value (28). Nodes are coloured according to the colours used in the center-right figure.

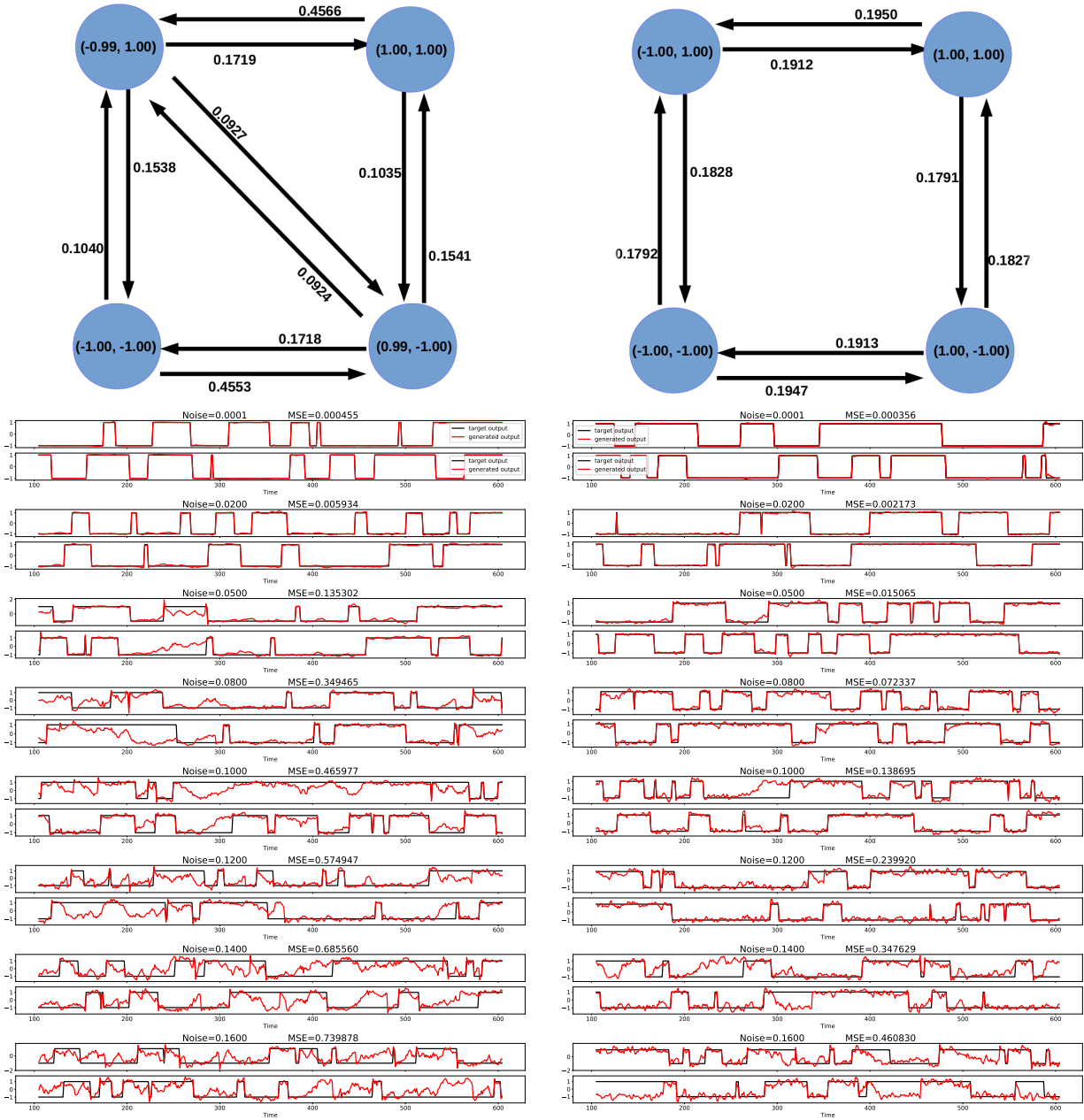


Figure 11: **Left:** results corresponding to a trained ESN giving rise to an ENA with undesired connections and unbalanced outgoing weights (28). **Right:** results obtained with a trained ESN giving rise to an ENA with balanced outgoing weights for all excitable connections. From top to bottom, in both columns, ESN outputs and related MSEs obtained with increasing noise standard deviation: 10^{-4} , $2 \cdot 10^{-2}$, $5 \cdot 10^{-2}$, $8 \cdot 10^{-2}$, 10^{-1} , $1.2 \cdot 10^{-1}$, $1.4 \cdot 10^{-1}$, and $1.6 \cdot 10^{-1}$.

Appendix A Linear stability analysis

By considering the case where the neural network is autonomous, we follow Eq. 9 and write the related map as $\mathbf{x}[k] = \mathbf{F}(\mathbf{x}[k-1]) = \mathbf{G}(\mathbf{x}[k-1], \mathbf{0})$. The i th component of the ESN state, $x_i, i = 1, \dots, N_r$, evolves according to the scalar map $F_i(\mathbf{x}) = (1 - \alpha)x_i + \alpha \tanh(\mathbf{M}_{(i)} \cdot \mathbf{x})$. By noting that $\frac{\partial F_i}{\partial x_j}(\mathbf{x}_0) = (1 - \alpha)\delta_{ij} + \alpha(1 - \tanh^2(\mathbf{M}_{(i)} \cdot \mathbf{x}_0))m_{ij}$, where δ_{ij} is the Kronecker delta, it is possible to write the Jacobian matrix evaluated onto \mathbf{x}_0 as $\mathbf{J}_{\mathbf{F}}(\mathbf{x}_0) = (1 - \alpha)\mathbf{I}_{N_r} + \alpha\mathbf{D}(\mathbf{x}_0)\mathbf{M}$, where

$$\mathbf{D}(\mathbf{x}_0) = \begin{bmatrix} 1 - \tanh^2(\mathbf{M}_{(1)} \cdot \mathbf{x}_0) & 0 & \dots & 0 \\ 0 & 1 - \tanh^2(\mathbf{M}_{(2)} \cdot \mathbf{x}_0) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 - \tanh^2(\mathbf{M}_{(N_r)} \cdot \mathbf{x}_0) \end{bmatrix} \quad (29)$$

is an $N_r \times N_r$ diagonal matrix representing the squashing action of $\tanh(\cdot)$ along the saturating components of \mathbf{x}_0 . By linearizing the network state-update (5) around a given fixed point \mathbf{x}^* , we obtain the linear system $\delta\mathbf{x}[k+1] = \mathbf{J}_{\mathbf{F}}(\mathbf{x}^*)\delta\mathbf{x}[k]$, where $\delta\mathbf{x}[k] = \mathbf{x}[k] - \mathbf{x}^*$. It is known [54] that if a fixed point \mathbf{x}^* is hyperbolic (i.e., $\mathbf{J}_{\mathbf{F}}(\mathbf{x}^*)$ has no eigenvalues on the unit circle in the complex plane), then the linear approximation provides a bona-fide characterisation of the non-linear behaviour around that fixed point. Therefore, the (linear) stability of \mathbf{x}^* is completely determined by the spectral radius of $\mathbf{J}_{\mathbf{F}}(\mathbf{x}^*)$. If all eigenvalues of $\mathbf{J}_{\mathbf{F}}(\mathbf{x}^*)$ are inside the unit circle, then \mathbf{x}^* is a stable fixed point; on the other hand, if even just one eigenvalues has norm larger than one, then the linearized map is expanding along the corresponding eigenvectors and \mathbf{x}^* is called a saddle⁵. We conclude observing that holds $x_i^* = \tanh(\mathbf{M}_{(i)} \cdot \mathbf{x}^*) \iff \alpha x_i^* = \alpha \tanh(\mathbf{M}_{(i)} \cdot \mathbf{x}^*) \iff x_i^* + \alpha x_i^* = x_i^* + \alpha \tanh(\mathbf{M}_{(i)} \cdot \mathbf{x}^*) \iff x_i^* = (1 - \alpha)x_i^* + \alpha \tanh(\mathbf{M}_{(i)} \cdot \mathbf{x}^*)$, hence the number of fixed points and their positions in phase space do not change by varying $\alpha \in (0, 1]$. However, their linear stability properties are directly affected by α .

Appendix B Bifurcation of fixed points in ESNs

In what follows, we perform a bifurcation analysis of one-dimensional ESN map and provide some sufficient conditions to design two-dimensional ESNs with a desired number of fixed points. Particularly, we focus on *fold bifurcations*⁶ of low-dimensional ESN maps, which constitute the only codimension-1 bifurcation that generate new fixed points; as discussed by Tiño et al. [58], it is the usual mechanism for creating new attractive fixed point in RNNs. Moreover, as shown by Beer [7], some fold bifurcations are responsible for reducing the dimensions where the actual dynamics takes place, dividing the RNN parameter space in different regions of effective dimensionality.

B.1 ESN maps in one dimension

Here, we consider the following one-dimensional map,

$$x[k] = \tanh(mx[k-1] + w), \quad (30)$$

where $(m, w) \in \mathbb{R}^2$ are the bifurcation parameters; we simplify the state-update in Eq. 5 and set leak rate $\alpha = 1$, $\epsilon = 0$, and remove explicit reference to inputs. Nevertheless, studying the map (30) is still useful to obtain insights about high-dimensional input-driven ESN dynamics. In fact, in the high-dimensional case (5), the activation function of the j th neuron is ruled by

$$x_j[k] = \tanh\left(m_{jj}x_j[k-1] + \sum_{s \neq j} m_{js}x_s[k-1] + (\mathbf{W}_{in})_{(j)} \cdot \mathbf{u}[k] + \varepsilon_j\right). \quad (31)$$

⁵If every eigenvalue has norm greater than one, then the fixed point is a repeller.

⁶We refer to Kuznetsov [28] for the terminology; fold bifurcation is also known as *saddle-node*, *limit point* or *turning point*.

Hence, the parameter w in (30) can be interpreted as the weighted sum of all incoming neurons, plus input and noise terms. Fixed points of the map $F(x) = \tanh(mx + w)$ are the solutions x^* of the equation $Q(x^*) = 0$, where

$$Q(x) := F(x) - x = \tanh(mx + w) - x. \quad (32)$$

Unfortunately, it is not possible to find a closed-form expression for the fixed points. However, it is possible to state that: (i) for every $(m, w) \in \mathbb{R}^2$, there exists at least one fixed point; (ii) there can be one, two, or three fixed points. The proof of these statements follow straightforwardly from the fact that $\lim_{x \rightarrow \pm\infty} Q(x) = \mp\infty$ and because, if $m < 1$ then Q is monotonic. Otherwise, there exist two critical points, $x_l < x_r$, namely

$$x_{l,r} = \frac{1}{m} \left[\pm \tanh^{-1} \left(\sqrt{\frac{m-1}{m}} \right) - w \right], \quad (33)$$

such that the function $Q(x)$ folds. Moreover, since $Q(x^*) = 0 \iff x^* = \tanh(mx^* + w) \in (-1, 1)$ for every $(m, w) \in \mathbb{R}^2$, all fixed points lie in the open interval $(-1, 1)$. Critical points in (33) are solutions to $Q'(x) = 0$, or equivalently to $F'(x) = 1$.

Let us assume $m > 0$. We observe a fold bifurcation whenever a critical point assumes the zero value. The condition $Q(x_{l,r}) = 0$ gives rise to the following parametrization of the fold bifurcation curve,

$$w_{\pm}(m) := \pm \left[m \sqrt{\frac{m-1}{m}} - \tanh^{-1} \left(\sqrt{\frac{m-1}{m}} \right) \right], \quad m \in [1, +\infty), \quad (34)$$

which possesses two symmetric branches ending on a cusp; see Fig. 12 for an illustration. Crossing that curve in parameter space towards the region containing the semi-axis of $m > 1$, a new fixed point is formed (x_l or x_r , depending on the branch) and splits in a pair of fixed points, one stable and one unstable⁷. This curve delimits the boundary between a dynamic regime where two stable points coexist with an unstable one, and a regime where only one fixed point exists. Due to symmetry, in the $m < 0$ case, there are two bifurcation branches identifying a *flip bifurcation*, which is analytically described by the curve $w_{\pm}(-m)$ with $m \leq -1$. Crossing that curve towards the region containing the semi-axis of $m < -1$, a stable fixed point loses stability and gives rise to a 2-periodic attracting trajectory surrounding it. We note that the flip bifurcation is detrimental for the flip-flop task considered here, as it implies autonomous jumps corresponding to undesired RNN behaviours.

B.2 ESN maps in two dimensions

Here, we consider a two-neuron reservoir. We denote the activation functions of these neurons as x and y , so that the ESN state evolution defines a trajectory $(x[k], y[k]) \in [-1, 1]^2, k = 1, 2, \dots$, ruled by:

$$\begin{aligned} x[k] &= \tanh(ax[k-1] + by[k-1]), \\ y[k] &= \tanh(cx[k-1] + dy[k-1]). \end{aligned} \quad (35)$$

It is known that a fixed point can undergo only fold or flip bifurcations in one-dimensional, discrete-time dynamical systems [28]. Nevertheless, considering two or more dimensions, also *Neimark-Sacker* bifurcations could occur, where a stable point loses stability and an invariant curve surrounding it is created. These are the only possible codimension-1 bifurcations of a fixed point for a discrete-time dynamical system. Among them, only the fold bifurcation can generate new fixed points. The origin is always a fixed point of (9). Considering Eq. 35, the Jacobian matrix evaluated on the origin reads

$$\mathbf{W}_r = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

Therefore, training ESNs via (6) implies a qualitative change of the dynamics around the origin if some eigenvalue crosses the unit circle, i.e., fixed point $(0, 0)$ bifurcates. However, adding a low-rank matrix to

⁷The particular case of crossing that curve through the cusp gives rise to a pitchfork bifurcation of the origin.

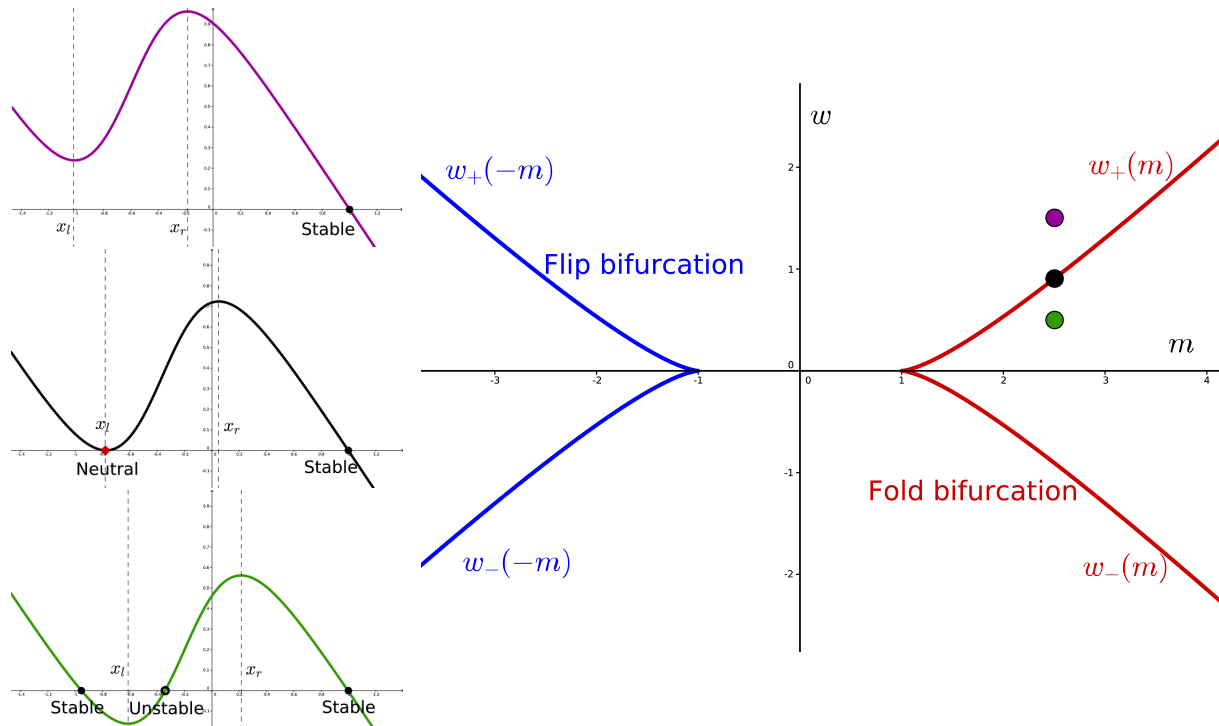


Figure 12: **Left:** three different folding configuration of the function $Q(x)$ in (32) with $m = 2.5$; purple $w = 1.5$, black $w \approx 0.9$, green $w = 0.5$. **Right:** bifurcation diagram of fixed points in one-dimensional ESN.

the reservoir can induce global bifurcations far away from the origin as well; hence, we cannot rely on the local bifurcation of the origin to deduce the global attractor structure after training. As a counterexample, suppose $\lambda_1, \lambda_2 > 1$. The diagonal matrix $\text{diag}(\lambda_1, \lambda_2)$ and the upper triangular matrix $\begin{pmatrix} \lambda_1 & \gamma \\ 0 & \lambda_2 \end{pmatrix}$ share the same spectrum $\{\lambda_1, \lambda_2\}$. Nevertheless, if the coupling is strong enough, that is, $|\gamma| > \frac{w_+(\lambda_1)}{y^*}$, where $w_+(\lambda_1)$ is the function (34) evaluated on λ_1 and y^* is the positive stable solution of $y[k] = \tanh(\lambda_2 y[k - 1])$, then the fold bifurcation curve is crossed, making the dynamics for x variable trivial. As a consequence of this, the upper triangular matrix induces dynamics with just two attractors (plus two saddles and the origin is a repeller), while the diagonal matrix induces four attractors (plus four saddles and the repeller).

In order to count the number of fixed points of the map (35), we can draw its nullclines. Defining function $N_{\alpha,\beta}(\eta) := \frac{1}{\beta}[-\alpha\eta + \tanh^{-1}(\eta)]$, the nullclines are given by

$$\begin{aligned} y &= N_{a,b}(x), \\ x &= N_{d,c}(y), \end{aligned} \quad (36)$$

and they represent the locus of points where x -dynamic / y -dynamic is stationary. As a consequence, the solutions of the algebraic nonlinear system (36) coincide with the set of fixed points. In the last part of this subsection, we show some sufficient conditions to control the number of fixed points assuming $a, d > 1$, i.e., when both nullclines are folding. We refer to Fig. 13 for a graphical illustration.

- Case $bc \geq 0$.

The first one is a condition on the nullclines intersecting the origin, that is,

$$|N'_{a,b}(0)| < |N'_{d,c}(0)|^{-1}, \text{ i.e. } (1-a)(1-d) < bc \implies \text{there are exactly 3 fixed points.} \quad (37)$$

On the other hand, when $(1-a)(1-d) \geq bc$, there could exist 5, 7 or 9 fixed points. Critical points of the function $N_{\alpha,\beta}(\eta)$ are $\eta_{\pm} = \pm\sqrt{\frac{\alpha-1}{\alpha}}$. Therefore, if $(1-a)(1-d) \geq bc$ holds then

$$\left|N_{a,b}\left(\sqrt{\frac{a-1}{a}}\right)\right| < \sqrt{\frac{d-1}{d}} \text{ or } \left|N_{d,c}\left(\sqrt{\frac{d-1}{d}}\right)\right| < \sqrt{\frac{a-1}{a}} \implies \text{there are exactly 5 fixed points.} \quad (38)$$

Fig. 13 depicts a nullcline configuration where there are 5 intersections⁸. From that configuration, we can make the humps of $y = N_{a,b}(x)$ more pronounced by increasing the $\frac{a}{b}$ ratio, until a fold bifurcation occurs, giving rise to a new couple of pair of fixed points. The case of 7 fixed points is obtained in the particular setting of the fold bifurcation.

- Case $bc < 0$.

$$\left|N_{a,b}\left(\sqrt{\frac{a-1}{a}}\right)\right| < \sqrt{\frac{d-1}{d}} \text{ and } \left|N_{d,c}\left(\sqrt{\frac{d-1}{d}}\right)\right| < \sqrt{\frac{a-1}{a}} \implies \text{there is exactly 1 fixed point.} \quad (39)$$

From the above configuration, increasing the d/c ratio makes stretch the humps of $N_{d,c}$ until a fold bifurcation occurs producing exactly three fixed points. Beyond the fold bifurcation, a new pair of fixed points is generated.

$$\left|N_{a,b}\left(\sqrt{\frac{a-1}{a}}\right)\right| < \sqrt{\frac{d-1}{d}} \text{ and } \left|N_{d,c}\left(\sqrt{\frac{d-1}{d}}\right)\right| > 1 \implies \text{there are exactly 5 fixed points.}^9 \quad (40)$$

In both cases, a simple sufficient condition to ensure the existence of 9 fixed points, see Fig. 13, is

$$\left|N_{a,b}\left(\sqrt{\frac{a-1}{a}}\right)\right| > 1 \text{ and } \left|N_{d,c}\left(\sqrt{\frac{d-1}{d}}\right)\right| > 1 \implies \text{there are exactly 9 fixed points.} \quad (41)$$

The maximum number of nullcline intersections is 9, setting the maximum number of fixed points that can be generated in a two-dimensional ESN map.

⁸Note that it holds $N_{\alpha,\beta}\left(\pm\sqrt{\frac{\alpha-1}{\alpha}}\right) = -\frac{1}{\beta}w_{\pm}(\alpha)$.

⁹Due to symmetry, this condition holds also when inverting the role of $N_{d,c}$, $N_{a,b}$ and consequently (a, b) , (d, c) .

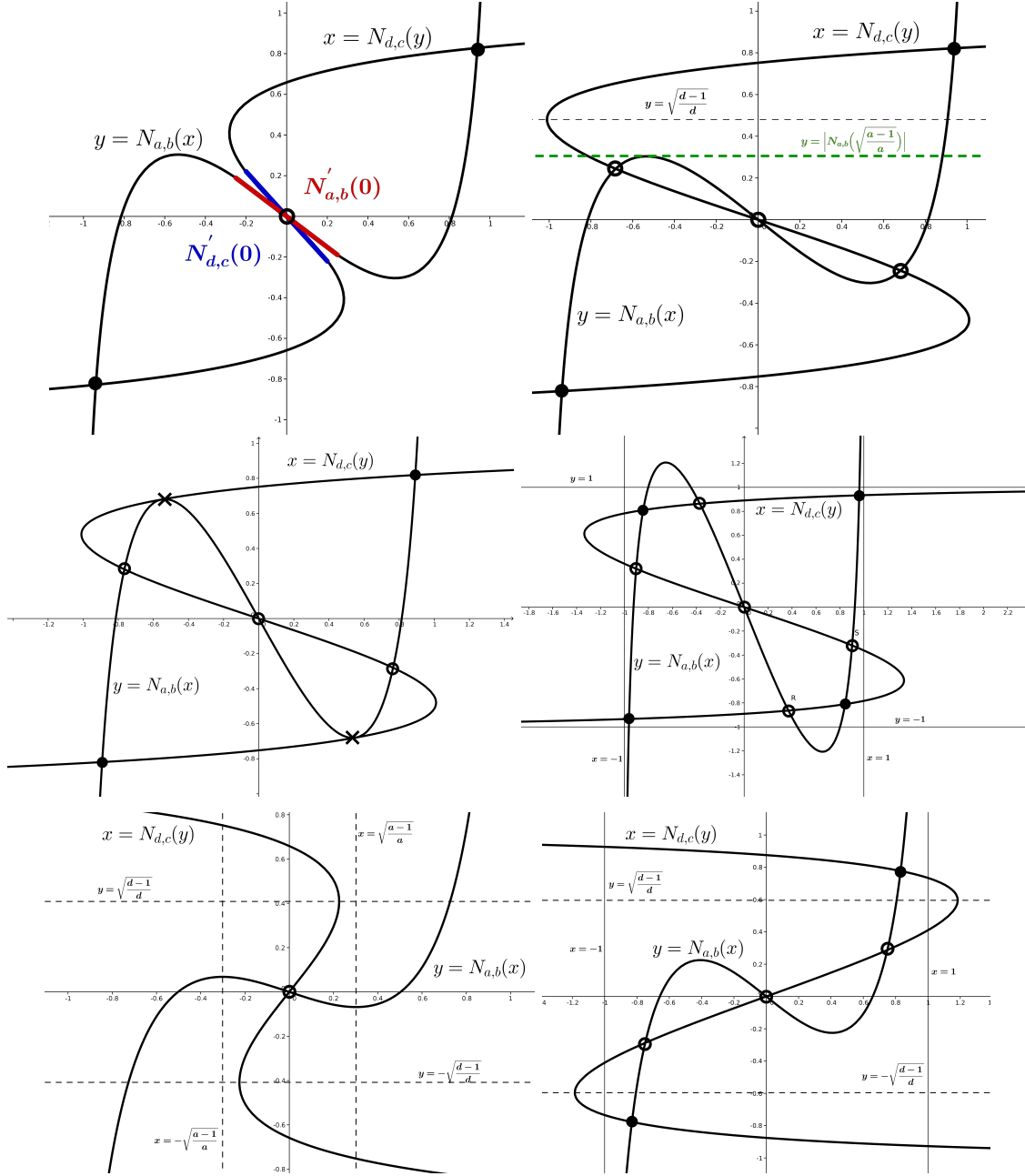


Figure 13: In all panels, filled points denote stable attractors, while circles denote saddles or repellers. **Top left:** geometric representation of the sufficient condition (37); nullcline slopes on the origin are highlighted with different colours. **Top right:** illustration of the condition (38); the black dashed line acts as an upper bound ensuring that the maximum height of the peak of the x -nullcline (represented by the green dashed line) does not cross further the y -nullcline. **Centre left:** special case of 7 fixed points. Crosses indicate neutral fixed points where fold bifurcations take place. **Centre right:** an example of nullclines configuration holding the condition (41) in the case of $bc > 0$. Both curves come out of the invariant square $[-1, 1]^2$ with their humps, that guarantees 9 intersections, i.e., 9 fixed points. **Bottom left:** depiction of the sufficient condition (39); the origin is the only fixed point and it is unstable. **Bottom right:** illustration of the sufficient condition (40).

Appendix C Aggregation of fixed points via clustering

The optimization procedure described in Sec. 4.1 provides a number of solutions equal to the number of initial conditions (assuming all initial conditions converge); however, such solutions might be similar up to a prescribed numerical precision. In order to reduce the set of all solutions to a few effective fixed points, we run the k -means clustering algorithm and retain only the elements belonging to the final clusters minimizing the distance w.r.t. the cluster centroids. Instead of aggregating all solutions at once, we first group such solutions according to their linear stability properties as indicated by the spectrum of the Jacobian matrix of the autonomous map (9). In particular, we form the group of linear stable fixed points, the group with one unstable direction and so on. Finally, for each group, we run k -means with parameter k identified according to the minimum of the Davies-Bouldin index [2].

Compliance with Ethical Standards

Conflict of Interest Andrea Ceni, Peter Ashwin, and Lorenzo Livi declare that they have no conflict of interest.

Funding We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. LL gratefully acknowledges partial support of the Canada Research Chairs program. PA gratefully acknowledges partial support of the EPSRC Centre for Predictive Modelling in Healthcare via grant EP/N014391/1.

Human and Animal Rights This article does not contain any studies with human participants or animals performed by any of the authors.

References

- [1] J. Aljadeff, D. Renfrew, M. Veu e, and T. O. Sharpee. Low-dimensional dynamics of structured random networks. *Physical Review E*, 93(2):022302, 2016. doi: 10.1103/PhysRevE.93.022302.
- [2] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. P erez, and I. Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013. doi: 10.1016/j.patcog.2012.07.021.
- [3] M. Arjovsky, A. Shah, and Y. Bengio. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, pages 1120–1128, New York, USA, June 2016.
- [4] P. Ashwin and C. Postlethwaite. On designing heteroclinic networks from graphs. *Physica D: Nonlinear Phenomena*, 265: 26–39, 2013. doi: 10.1016/j.physd.2013.09.006.
- [5] P. Ashwin and C. Postlethwaite. Designing heteroclinic and excitable networks in phase space using two populations of coupled cells. *Journal of Nonlinear Science*, 26(2):345–364, 2016. doi: 10.1007/s00332-015-9277-2.
- [6] P. Ashwin and C. Postlethwaite. Sensitive finite state computations using a distributed network with a noisy network attractor. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–12, 2018. doi: 10.1109/TNNLS.2018.2813404.
- [7] R. D. Beer. Parameter space structure of continuous-time recurrent neural networks. *Neural Computation*, 18(12):3009–3051, 2006. doi: 10.1162/neco.2006.18.12.3009.
- [8] F. M. Bianchi, S. Scardapane, A. Uncini, A. Rizzi, and A. Sadeghian. Prediction of telephone calls load using echo state network with exogenous variables. *Neural Networks*, 71:204–213, 2015. doi: 10.1016/j.neunet.2015.08.010.
- [9] F. M. Bianchi, L. Livi, and C. Alippi. Investigating echo state networks dynamics by means of recurrence analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 29(2):427–439, 2018. doi: 10.1109/TNNLS.2016.2630802.
- [10] D. Castelvechhi. Can we open the black box of AI? *Nature News*, 538(7623):20, 2016.
- [11] M. Cencini, F. Cecconi, and A. Vulpiani. *Chaos: From Simple Models to Complex Systems*. World Scientific, Singapore, 2010.
- [12] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [13] B. De Pasquale, C. J. Cueva, K. Rajan, G. S. Escola, and L. F. Abbott. full-FORCE: A target-based method for training recurrent networks. *PLoS ONE*, 13(2):1–18, 2 2018. doi: 10.1371/journal.pone.0191527.
- [14] K. Funahashi and Y. Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6(6):801–806, 1993.

- [15] C. Gallicchio and A. Micheli. Echo state property of deep reservoir computing networks. *Cognitive Computation*, 9(3): 337–350, Jun. 2017. ISSN 1866-9964. doi: 10.1007/s12559-017-9461-9.
- [16] M. D. Golub and D. Sussillo. Fixedpointfinder: A tensorflow toolbox for identifying and characterizing fixed points in recurrent neural networks. *The Journal of Open Source Software*, 3:1003, 2018. doi: 10.21105/joss.01003.
- [17] B. Goodman and S. Flaxman. European union regulations on algorithmic decision-making and a “right to explanation”. *arXiv preprint arXiv:1606.08813*, 2016.
- [18] A. Graves, A.-R. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, Vancouver, BC, Canada, May 2013. IEEE.
- [19] B. Hammer. On the approximation capability of recurrent neural networks. *Neurocomputing*, 31(1):107–123, 2000. doi: 10.1016/S0925-2312(99)00174-5.
- [20] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [21] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004. doi: 10.1126/science.1091277.
- [22] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007. doi: 10.1016/j.neunet.2007.04.016.
- [23] S. Kanai, Y. Fujiwara, and S. Iwamura. Preventing gradient explosions in gated recurrent units. In *Advances in Neural Information Processing Systems*, pages 435–444, 2017.
- [24] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2318–2331, Oct. 2017. ISSN 1041-4347. doi: 10.1109/TKDE.2017.2720168.
- [25] G. E. Katz and J. A. Reggia. Using directional fibers to locate fixed points of recurrent neural networks. *IEEE transactions on neural networks and learning systems*, 29(8):3636–3646, 2018.
- [26] L. Keuninckx, J. Danckaert, and G. Van der Sande. Real-time audio processing with a cascade of discrete-time delay line-based reservoir computers. *Cognitive Computation*, 9(3):315–326, 2017. doi: 10.1007/s12559-017-9457-5.
- [27] D. Koryakin, J. Lohmann, and M. V. Butz. Balanced echo state networks. *Neural Networks*, 36:35–45, 2012. doi: 10.1016/j.neunet.2012.08.008.
- [28] Y. A. Kuznetsov. *Elements of Applied Bifurcation Theory*, volume 112. Springer Science & Business Media, Berlin, Germany, 2013.
- [29] L. Livi, F. M. Bianchi, and C. Alippi. Determination of the edge of criticality in echo state networks through Fisher information maximization. *IEEE Transactions on Neural Networks and Learning Systems*, 29(3):706–717, Mar. 2018. doi: 10.1109/TNNLS.2016.2644268.
- [30] S. Løkse, F. M. Bianchi, and R. Jenssen. Training echo state networks with regularization through dimensionality reduction. *Cognitive Computation*, 9(3):364–378, 2017. doi: 10.1007/s12559-017-9450-z.
- [31] M. Lukoševičius. *A Practical Guide to Applying Echo State Networks*, pages 659–686. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. doi: 10.1007/978-3-642-35289-8_36.
- [32] M. Lukoševičius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009. doi: 10.1016/j.cosrev.2009.03.005.
- [33] W. Maass, P. Joshi, and E. D. Sontag. Computational aspects of feedback in neural circuits. *PLoS Computational Biology*, 3(1):e165, 2007. doi: 10.1371/journal.pcbi.0020165.eor.
- [34] G. Manjunath and H. Jaeger. Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks. *Neural Computation*, 25(3):671–696, 2013. doi: 10.1162/NECO.a.00411.
- [35] V. Mante, D. Sussillo, K. V. Shenoy, and W. T. Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, 2013. doi: 10.1038/nature12742.
- [36] F. Mastrogiuseppe and S. Ostojic. Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 2018. ISSN 0896-6273. doi: 10.1016/j.neuron.2018.07.003.
- [37] N. M. Mayer and Y.-H. Yu. Orthogonal echo state networks and stochastic evaluations of likelihoods. *Cognitive Computation*, 9(3):379–390, 2017. doi: 10.1007/s12559-017-9466-4.
- [38] K. D. Miller and F. Fumarola. Mathematical equivalence of two common forms of firing rate models of neural networks. *Neural computation*, 24(1):25–31, 2012.
- [39] P. Miller. Itinerancy between attractor states in neural systems. *Current Opinion in Neurobiology*, 40:14–22, 2016. doi: 10.1016/j.conb.2016.05.005.
- [40] J. Milnor. On the concept of attractor. In *The Theory of Chaotic Attractors*, pages 243–264. Springer, 1985.
- [41] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, Feb. 2017. doi: 10.1016/j.dsp.2017.10.011.
- [42] F. S. Neves, M. Voit, and M. Timme. Noise-constrained switching times for heteroclinic computing. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(3):033107, 2017. doi: 10.1063/1.4977552.
- [43] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, Berlin, Germany, 2006.
- [44] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 1310–1318, Atlanta, Georgia, USA, 2013.
- [45] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *14th International Conference on Frontiers in Handwriting Recognition*, pages 285–290, Crete Island, Greece, Sep. 2014. doi: 10.1109/ICFHR.2014.55.

- [46] M. Rabinovich, A. Volkovskii, P. Lecanda, R. Huerta, H. Abarbanel, and G. Laurent. Dynamical encoding by networks of competing neuron groups: winnerless competition. *Physical Review Letters*, 87(6):068102, 2001.
- [47] M. Rabinovich, R. Huerta, and G. Laurent. Transient dynamics for neural processing. *Science*, 321(5885):48–50, 2008. doi: 10.1126/science.1155564.
- [48] K. Rajan, L. F. Abbott, and H. Sompolinsky. Stimulus-dependent suppression of chaos in recurrent neural networks. *Physical Review E*, 82(1):011903, 2010. doi: 10.1103/PhysRevE.82.011903.
- [49] R. F. Reinhart and J. J. Steil. Regularization and stability in reservoir networks with output feedback. *Neurocomputing*, 90:96–105, 2012. doi: 10.1016/j.neucom.2012.01.032.
- [50] A. Rivkind and O. Barak. Local dynamics in trained recurrent neural networks. *Physical Review Letters*, 118:258101, Jun. 2017. doi: 10.1103/PhysRevLett.118.258101.
- [51] A. Rodan and P. Tiño. Simple deterministically constructed cycle reservoirs with regular jumps. *Neural Computation*, 24(7):1822–1852, 2012. doi: 10.1162/NECO_a.00297.
- [52] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- [53] S. Scardapane and A. Uncini. Semi-supervised echo state networks for audio classification. *Cognitive Computation*, 9(1): 125–135, 2017. doi: 10.1007/s12559-016-9439-z.
- [54] S. H. Strogatz. *Nonlinear Dynamics and Chaos*. Hachette, UK, 2014.
- [55] D. Sussillo and L. F. Abbott. Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4): 544–557, 2009. doi: 10.1016/j.neuron.2009.07.018.
- [56] D. Sussillo and O. Barak. Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Computation*, 25(3):626–649, 2013. doi: 10.1162/NECO_a.00409.
- [57] C. Tallec and Y. Ollivier. Can recurrent neural networks warp time? In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=SJcKhk-Ab>.
- [58] P. Tiño, B. G. Horne, and C. L. Giles. Attractive periodic sets in discrete-time recurrent networks (with emphasis on fixed-point stability and bifurcations in two-neuron networks). *Neural Computation*, 13(6):1379–1414, 2001.
- [59] I. Tsuda. Chaotic itinerancy and its roles in cognitive neurodynamics. *Current Opinion in Neurobiology*, 31:67–71, 2015. doi: 10.1016/j.conb.2014.08.011.
- [60] P. Vincent-Lamarre, G. Lajoie, and J.-P. Thivierge. Driving reservoir models with oscillations: a solution to the extreme structural sensitivity of chaotic networks. *Journal of Computational Neuroscience*, pages 1–18, 2016. doi: 10.1007/s10827-016-0619-3.
- [61] O. Weinberger and P. Ashwin. From coupled networks of systems to networks of states in phase space. *Discrete & Continuous Dynamical Systems - B*, 23:2043, 2018. ISSN 1531-3492. doi: 10.3934/dcdsb.2018193.
- [62] F. wyffels, J. Li, T. Waegeman, B. Schrauwen, and H. Jaeger. Frequency modulation of large oscillatory neural networks. *Biological Cybernetics*, 108(2):145–157, 2014. doi: 10.1007/s00422-013-0584-0.
- [63] I. B. Yildiz, H. Jaeger, and S. J. Kiebel. Re-visiting the echo state property. *Neural Networks*, 35:1–9, 2012. doi: 10.1016/j.neunet.2012.07.005.