UNIVERSITY OF BIRMINGHAM

Input-to-state representation in linear reservoirs dynamics

Verzelli, Pietro; Alippi, Cesare; Livi, Lorenzo; Tino, Peter

[Link to publication on Research at Birmingham portal](#)

INPUT-TO-STATE REPRESENTATION
IN LINEAR RESERVOIRS DYNAMICS

Pietro Verzelli*
Faculty of Informatics
(Università della Svizzera Italiana)

Cesare Alippi
Department of Electronics,
Information and bioengineering
(Politecnico di Milano)
Faculty of Informatics
(Università della Svizzera Italiana)

Lorenzo Livi
Departments of Computer Science and Mathematics
(University of Manitoba)
Department of Computer Science
(University of Exeter)

Peter Tiňo
School of Computer Science
(University of Birmingham)

February 15, 2021

ABSTRACT

Reservoir computing is a popular approach to design recurrent neural networks, due to its training simplicity and approximation performance. The recurrent part of these networks is not trained (e.g., via gradient descent), making them appealing for analytical studies by a large community of researchers with backgrounds spanning from dynamical systems to neuroscience. However, even in the simple linear case, the working principle of these networks is not fully understood and their design is usually driven by heuristics. A novel analysis of the dynamics of such networks is proposed, which allows the investigator to express the state evolution using the controllability matrix. Such a matrix encodes salient characteristics of the network dynamics; in particular, its rank represents an input-indepedent measure of the memory capacity of the network. Using the proposed approach, it is possible to compare different reservoir architectures and explain why a cyclic topology achieves favourable results as verified by practitioners.

***K*eywords** Reservoir Computing · Recurrent Neural Networks · Dynamical Systems

1 Introduction

Despite of being applied to a large variety of tasks, Recurrent Neural Networks (RNNs) are far from being fully understood and performance improvement is usually driven by heuristics. Understanding how the computation is conducted by the dynamics of the RNNs is an old question [1] which still remains unanswered, even though advances were recently achieved [2, 3]. The introduction of gating mechanisms (such as LSTM [4] and GRU [5]) dramatically improved the performance of RNNs, but the use of complex architectures makes the theoretical analysis harder [6, 7, 8]. Even for simple networks, we currently lack a sound framework to describe how the signal history is encoded in the state. Another relevant issue is the *memory-nonlinearity trade-off* [9, 10]. Memory capacity maximization does not necessarily imply performance (e.g., prediction) maximization [11]. In recent years, large efforts have been devoted to tackle these problems, by studying the dynamical systems behind RNNs [2, 12, 13, 3, 14, 15].

---

*Referring author : pietro.verzelli@usi.ch

Reservoir Computing (RC) is a computational paradigm developed independently by Jaeger [16, 17] (Echo State Networks (ESNs)) and Maas [18] (Liquid State Machines (LSMs)) and Tiňo [19] (Fractal Predicting Machines (FPMs)). The basic idea is to create a representation of the input signal using an untrained RNN, called *the reservoir*, and then to use a trainable *readout* layer to generate the network output. RC demonstrated its effectiveness in various tasks and has risen great interest in the physical computing community due to the underlying idea of *natural computation*. In particular, photonics [20] and neuromorphic computation [21] are commonly implemented using RC, but also a bucket of water [22] or road traffic [23] have been used as reservoirs. See [24] for a review.

The architecture simplicity makes RC prone to theoretical investigations [25, 26, 27, 15, 12, 28], which have mainly concentrated on questions about computational capabilities of whole classes of dynamical systems [26], while little has been understood about how specific setting of the dynamical system can influence the network computational property [15]. Some important theoretical results can be derived assuming that the reservoir dynamics are linear [12, 11, 15, 28, 29]. This assumption can be seen as a first-order approximation of the nonlinear system, sharing some – but not all – of its features [30].

In this work, we propose a novel analysis shedding light on how a linear RNN encodes input signals in its state. The analysis decouples the network state in two terms: the *controllability matrix* $\mathcal{C}$, which only depends on the reservoir and input weights vector, and the *network encoded input* $\mathbf{s}$, which depends on the reservoir weights and the input signal driving the system. Writing the system in this form, allows one to decouple the properties of the reservoir topology – encoded in $\mathcal{C}$ – from the specific input driving the network, encoded in $\mathbf{s}$. We analyze different reservoir topologies in terms of the nullspace of $\mathcal{C}$ and show that the rank of $\mathcal{C}$ is a measure for the richness of the representation of the input signal. More specifically, we show that the nullspace of $\mathcal{C}$ is linked to the memory forgetting capacity of the network. Based on these results, we demonstrate that a cyclic reservoir topology (forming a ring structure) is optimal. The claim is corroborated by empirical evidence.

The remaining of the paper is organized as follows. In Section 2 the RC paradigm is presented, along with the the various reservoir topologies studied in this work. Section 3 develops a representation of the network output in terms of its controllability matrix. Section 4 is devoted to the analysis of the input representation in the network's states, while Section 6 discusses the memory of the network in terms of the rank of the controllability matrix. In Section 6 some experiments are conducted to validate are claims. Finally, Section 7 draws the conclusion. The Appendices are dedicated to derivations.

2 Reservoir computing

RC was developed as a tool to explain the brain working principle [18] and as computational paradigm to avoid the complex and expensive training procedure of RNNs [16], e.g., based on backpropagation through time [31, 32]. RC training requires to randomly generate the weights of the recurrent layer called *reservoir*, which is tuned only at the hyper-parameter level (e.g., by searching for the best-performing spectral radius of the corresponding weight matrix). The reservoir reads the input signal through an input layer, resulting in an untrained representation of the input in the network's state. A *readout* layer is then trained to produce the system output. Common tasks involve time-series prediction [33, 34], simulation of dynamical systems [35], and time-series classification [36].

Let $\mathbf{x}_k \in \mathbb{R}^n$ be the *state* of a reservoir of dimension n at time k, $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ be its *reservoir connection matrix* and $\mathbf{w} \in \mathbb{R}^n$ its *input weights* vector. When considering left-infinite signal, the time-index k runs from 0 to $-\infty$, so that the driving *input signal* is $\mathbf{u} = (u_0, u_{-1}, u_{-2}, \dots)$, $u_{-k} \in \mathbb{R}$.

In the linear case, the reservoir evolves according to:

$$\mathbf{x}_k = \boldsymbol{W}\mathbf{x}_{k-1} + \mathbf{w}u_k \tag{1}$$

By recursively applying (1) we get

$$\mathbf{x}_0 = \boldsymbol{W}x_{-1} + \mathbf{w}u_0 = \boldsymbol{W}^2\mathbf{x}_{-2} + \boldsymbol{W}\mathbf{w}u_{-1} + \mathbf{w}u_0 \tag{2}$$

$$= \boldsymbol{W}^3\mathbf{x}_{-3} + \boldsymbol{W}^2\mathbf{w}u_{-2} + \boldsymbol{W}\mathbf{w}u_{-1} + \mathbf{w}u_0 \tag{3}$$

i.e.,

$$\mathbf{x}_0 = \sum_{k=0}^{\infty} \boldsymbol{W}^k \mathbf{w}u_{-k} \tag{4}$$

One usually relies on a (trained) linear readout $\mathbf{r} \in \mathbb{R}^n$ to generate the system output y_{-k}, so that at time 0

$$y_0 = \mathbf{r} \cdot \mathbf{x}_0 = \mathbf{r} \cdot \sum_{k=0}^{\infty} \boldsymbol{W}^k \mathbf{w}u_{-k} \tag{5}$$
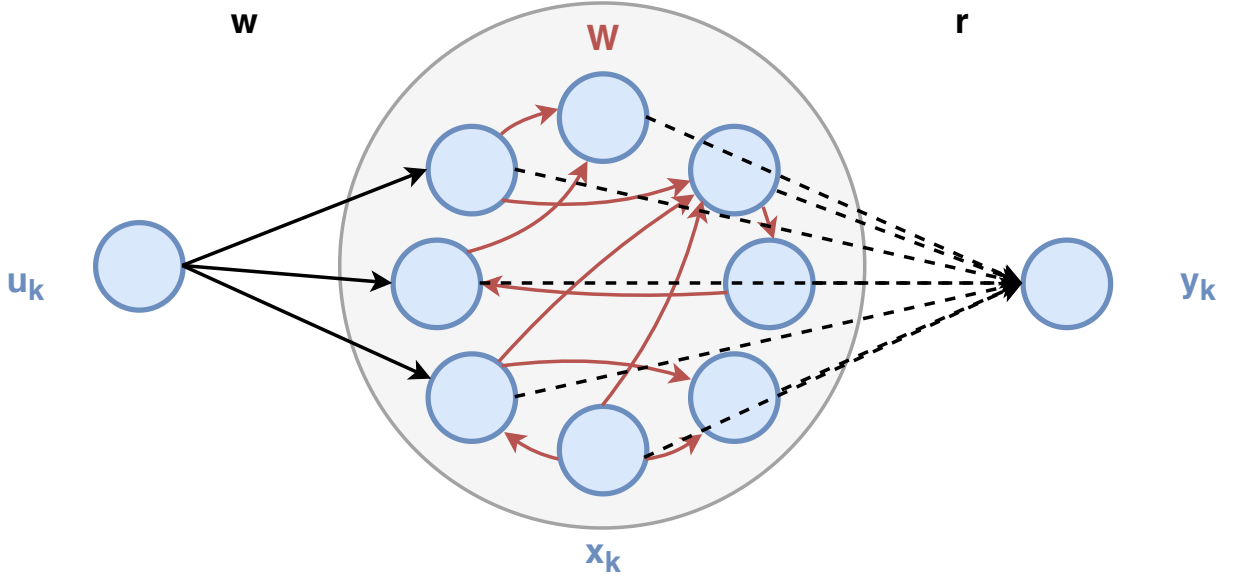
2

Figure 1: Schematic representation of a RC with a scalar input, a uni-dimensional output and $N = 8$ neurons in the reservoir. Fixed connections are drawn using solid lines while dashed lines represent learned connections.

In the seminal paper [16], training is based on a simple least-square regression. More sophisticated techniques where later introduced, including some forms of regularization [37] and online-training procedures [38]. For simplicity, we have only described the case in which the input and the output are uni-dimensional, but the proposed approach can be easily generalized to the multidimensional case. In Fig. 1, a schematic representation of the RC architecture is depicted.

Even though a reservoir layer does not require a proper training procedure, hyper-parameters tuning must be carried out in to improve performance. The most studied hyper-parameter is the Spectral Radius (SR) $\rho(\boldsymbol{W})$ [39, 40, 41] which is the largest eigenvalue in magnitude of $\boldsymbol{W}$, related to the Maximum Singular Value (MSV) $\sigma_{\max}(\boldsymbol{W})$. Other hyper-parameters refer to the input and output scaling factors, the sparsity degree of $\boldsymbol{W}$ and the input bias. Moreover, the update equation (1) is usually chosen to be non-linear, i.e., $\mathbf{x}_k = \phi(\boldsymbol{W}\mathbf{x}_{k-1} + \mathbf{w}u_k)$ and different choices of the nonlinear transfer function ϕ can be explored [42].

Many studies were devoted to understand how these hyper-parameters affect the dynamics of the network and its computational capabilities. In particular, it appears that the hyper-parameter space can be divided into a region where the dynamics are "regular" (meaning that they are stable with respect to the inputs driving the system) and another one where they are "disordered" (meaning that they are unstable and do not provide a representation for the input) [43]. The narrow region separating these two regimes is known in the literature as Edge of Chaos or Edge of Criticality (EoC) [44, 45, 46] and appears to be common to a large variety of complex systems beyond RNNs [47, 48, 49].

In the following, we introduce different reservoir architectures that are commonly found in the literature.

2.1 Delay line

In a *delay line* each neuron is connected the subsequent one to form a chain-like architecture, so that the reservoir connection matrix $\boldsymbol{W}_{\mathrm{d}}$ reads:

$$W_{\mathrm{d},ij} = \delta_{i,j-1} \tag{6}$$

where δ is the *Kronecker delta*. Note that the last neuron in the chain is not connected to the first one. Moreover, the input weights vector is $\mathbf{w}_{\mathrm{d}} = (1, 0, 0, \ldots, 0)$, meaning that the input enters the network only through the first neuron of the chain. Mathematically, such a model setting corresponds to the n-th order AR model, which is a really popular and studied tool in time series analysis and system identification [50].
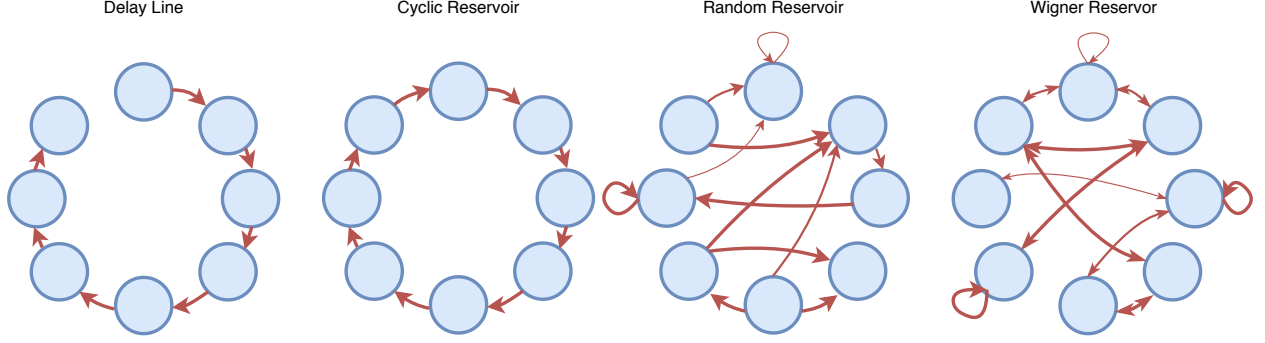
3

Figure 2: The different architectures discussed in this work. From left to right: delay line, cyclic, random and Wigner topology. The thickness of the arrow accounts for the strength of the connection. Notice that it has the same value for all the connections in both the delay line and the cyclic reservoir, while it varies for the other two. Note the presence of self loops in random and Wigner architectures. Also consider that the Wigner reservoir has symmetric connections (double-headed arrows).

2.2 Cyclic reservoir

A reservoir is said to be cyclic when each neuron is connected to another one, in a way that they form a ring. The reservoir matrix of a cyclic reservoir has the form:

$$W_{c,ij} = \delta_{i,j-1} \tag{7}$$

where, with an abuse of notation, $\delta_{0,-1} := \delta_{0,n-1}$. From the product of Kroenecker deltas, it follows that:

$$W_{c,ij}^2 = \sum_k W_{c,ik} W_{c,kj} = \sum_k \delta_{i,k-1} \delta_{k,j-1} = \delta_{i,j-2} \tag{8}$$

as well for higher powers.

2.3 Random reservoir

In a random reservoir the entries of the reservoir matrix $\boldsymbol{W}_r$ are independent random variables. In the sequel, we consider the generic ij component of the matrix to be drawn from a Gaussian distribution.

$$W_{r,ij} \sim \mathcal{N}\left(0, \frac{\rho^2}{n}\right) \tag{9}$$

With this choice, the expected value for the SR $\langle \rho(\boldsymbol{W}_r) \rangle = \rho$ [13] and MSV $\langle \sigma_{\max}(\boldsymbol{W}_r) \rangle = 2\rho$ [51] .

2.4 Wigner reservoir

The diagonal elements are distributed as in (9), i.e., $W_{w,ii} \sim \mathcal{N}(0, \rho_1^2/n)$, while off-diagonal elements follow:

$$W_{w,ij} = W_{r,ji} \sim \mathcal{N}\left(0, \frac{\rho_2^2}{n}\right), \quad i \neq j \tag{10}$$

Wigner matrices are symmetric. In this work, we will always set $2\rho_1 = \rho_2 = \rho$. Notably, this leads to $\langle \rho(\boldsymbol{W}_r) \rangle = \langle \sigma_{\max}(\boldsymbol{W}_r) \rangle = \rho$.

3 Controllability matrix and network encoded input

Here we develop a representation for the network state evolution based on the Cayley-Hamilton (CH) theorem [52]. The CH theorem states that every real square matrix satisfies its own characteristic equation, implying that

$$\boldsymbol{W}^n = \varphi_{n-1}\boldsymbol{W}^{n-1} + \varphi_{n-2}\boldsymbol{W}^{n-2} + \cdots + \varphi_1\boldsymbol{W} + \varphi_0\boldsymbol{I} \tag{11}$$

where the φ_i are the negated coefficient of the characteristic polynomial (see Appendix A for details). Accordingly, any power of matrix $\boldsymbol{W}$ can be written as a linear combination of the first $n-1$ powers, where n is the matrix order (and also the size of the reservoir):

$$\boldsymbol{W}^k = \sum_{j=0}^{n-1} \phi_j^{(k)} \boldsymbol{W}^j \tag{12}$$

where the apex k denotes the fact that the n coefficients are expansion coefficients of the k-th power of $\boldsymbol{W}$. In Appendix B we also show how the coefficients $\phi_j^{(k)}$ can be written in terms of φ_j in (11).

By inserting (12) in (4), we obtain:

$$\mathbf{x}_0 = \sum_{k=0}^{\infty} \sum_{j=0}^{n-1} \phi_j^{(k)} \boldsymbol{W}^j \mathbf{w} u_{-k} \tag{13}$$

$$= \sum_{j=0}^{n-1} \boldsymbol{W}^j \mathbf{w} \sum_{k=0}^{\infty} \phi_j^{(k)} u_{-k} = \sum_{j=0}^{n-1} \boldsymbol{W}^j \mathbf{w} s_j \tag{14}$$

where

$$s_j := \sum_{k=0}^{\infty} \phi_j^{(k)} u_{-k} \tag{15}$$

is what we call the *network encoded input*. It is useful to interpret $\mathbf{s} = (s_0, s_1, \ldots, s_{n-1})$ as a vector with n components, which "encodes" the left-infinite input signal $\mathbf{u}$ in the spatial representation provided by the network. In order for the s_j term to exist, the sum in (15) must converge; we will discuss this issue in the next section. We emphasize the fact that the sum over j (the dimensionality of our system) is a *finite* sum with n terms, as opposed to the infinite sum over the k (the time index).

Inspired by well-known tools from control theory [53], we define the *controllability matrix* of the reservoir as

$$\mathcal{C} = [\mathbf{w} \quad \boldsymbol{W}\mathbf{w} \quad \boldsymbol{W}^2\mathbf{w} \quad \ldots \quad \boldsymbol{W}^{n-1}\mathbf{w}] \tag{16}$$

Then, the state-update equation (4) becomes

$$\mathbf{x}_0 = \mathcal{C}\mathbf{s} \tag{17}$$

and the output (5) can then be expressed as:

$$y_0 = \mathbf{r}\mathcal{C}\mathbf{s} \tag{18}$$

i.e., the readout filters the input according to controllability matrix.

4 How the network encodes the input signal

From (18), we see that the possibility for the readout to produce the correct output (i.e., the output that solves the task at hand depends on two distinct elements: the controllability matrix $\mathcal{C}$ (function of $\boldsymbol{W}$ and $\mathbf{w}$) and the network encoded input $\mathbf{s}$ (which depends on $\boldsymbol{W}$ and $\mathbf{u}$).

In Appendix B we show that the coefficients $\phi_i^{(k+1)}$ of (12) can be recursively expressed in terms of $\phi_i^{(k)}$ as:

$$\begin{bmatrix} \phi_0^{(k+1)} \\ \phi_1^{(k+1)} \\ \vdots \\ \phi_{n-2}^{(k+1)} \\ \phi_{n-1}^{(k+1)} \end{bmatrix} = \boldsymbol{M} \begin{bmatrix} \phi_0^{(k)} \\ \phi_1^{(k)} \\ \vdots \\ \phi_{n-2}^{(k)} \\ \phi_{n-1}^{(k)} \end{bmatrix} \tag{19}$$

where $\boldsymbol{M}$ is the Frobenius companion matrix of $\boldsymbol{W}$ (see Apprendix B for details). Note that the characteristic polynomial of $\boldsymbol{M}$ is that of $\boldsymbol{W}$; as such, the two matrices share the same eigenvalues. Thus, the series (15) converges, for bounded inputs, when $\boldsymbol{W}$ has a spectral radius smaller than 1.[2]

---

[2]Note that most theoretical results (see e.g.,[16, 26, 15]) require the MSV to be smaller than one, which is a stricter condition (as SR≤MSV); it follows that our analysis can be applied to a larger class of reservoirs.

The **s** vector can be written as:

$$
\begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-2} \\ s_{n-1} \end{bmatrix} = \begin{bmatrix} \sum_{k=0}^{\infty} \phi_0^{(k)} u_{-k} \\ \sum_{k=0}^{\infty} \phi_1^{(k)} u_{-k} \\ \vdots \\ \sum_{k=0}^{\infty} \phi_{n-2}^{(k)} u_{-k} \\ \sum_{k=0}^{\infty} \phi_{n-1}^{(k)} u_{-k} \end{bmatrix}
\tag{20}
$$

In Appendix A we show that for $k < n$, $\phi_j^{(k)} = \delta_{kj}$ holds. We also note that terms corresponding to time-step $k = n$ follow from (11). This means that (4) can be written as:

$$
\begin{bmatrix} s_0 \\ s_1 \\ \dots \\ s_{n-2} \\ s_{n-1} \end{bmatrix} = \begin{bmatrix} u_0 \\ u_{-1} \\ \dots \\ u_{-(n-2)} \\ u_{-(n-1)} \end{bmatrix} + \begin{bmatrix} u_{-n}\varphi_0 \\ u_{-n}\varphi_1 \\ \dots \\ u_{-n}\varphi_{n-2} \\ u_{-n}\varphi_{n-1} \end{bmatrix}
$$
$$
+ \begin{bmatrix} \sum_{k=n+1}^{\infty} \phi_0^{(k)} u_{-k} \\ \sum_{k=n+1}^{\infty} \phi_1^{(k)} u_{-k} \\ \dots \\ \sum_{k=n+1}^{\infty} \phi_{n-2}^{(k)} u_{-k} \\ \sum_{k=n+1}^{\infty} \phi_{n-1}^{(k)} u_{-k} \end{bmatrix}
\tag{21}
$$

All other terms in (21) corresponding to time steps $k > n$ can be computed according to (12).

This procedure shows that, in general, the inputs from 0 to $n - 1$ steps back in time will *always* appear in their original form, and the cross-contribution starts only from u_{-n} backwards in time. We will make use of this result to analytically examine the properties of different networks topologies. Moreover, by deriving the expression for the $\phi_i^{(k)}$ we can study how the network is able to recall its past inputs. In general, if the $\phi_i^{(k)}$'s are large then the network will not be able to recall the inputs, since the input u_{-j} can only be read through $s_j = u_{-j} + \sum_{k=n}^{\infty} \phi_j^{(k)} u_{-k}$. It follows that having large expansion coefficients $\phi_j^{(k)}$ prevents the network from being able to recall its past inputs. We will show in the next section that, when we can derive an analytical expression for the $\phi_i^{(k)}$, it is possible to anticipate how the network recalls its past inputs. Note that, as implied by (18), for a linear network this is deeply related to its expressive power, since the network output is basically a linear combination of past inputs. The inputs accessibility to the readout is also due to $\mathcal{C}$, which is a property of the network only, as it does not depend on any input signal. A detailed discussion about the relation between the network properties and the rank of the controllability matrix $\mathcal{C}$ was recently presented in [54]. There, the authors prove that the memory capacity for linear reservoirs equals the rank of $\mathcal{C}$. In the following, we analyze the different architectures described in Section 2 discussing the properties of their controllability matrices.

In the random reservoir case (Subsection 2.3, the property of $\mathcal{C}$ can be studied by considering the expected values of the norm of its columns, which describes how the system accesses past inputs. Let us consider the n-by-n matrix $\boldsymbol{W}_r$ in (9) and a vector with n components $\mathbf{w} = \{w_j\} \sim \mathcal{N}(0, \frac{1}{n})$. Since w_j are generated independently, the expected value of the squared norm of the random vector $\mathbf{w}$ is $l(\mathbf{w}) = n\langle w_i^2 \rangle$. We drop the r in $\boldsymbol{W}$, to simplify the notation. We now study $\mathbf{z} := \boldsymbol{W}\mathbf{w}$ and obtain:

$$
\langle z_i^2 \rangle = \langle (\boldsymbol{W}\mathbf{w})_i^2 \rangle = \langle (\sum_j W_{ij} w_j)^2 \rangle = n\langle W_{ij}^2 \rangle \langle w_i^2 \rangle
\tag{22}
$$

where the last equality follows from the independence of the zero-mean entries of $\boldsymbol{W}$ and $\mathbf{w}$. Now, by the way we constructed $\boldsymbol{W}$ and $\mathbf{w}$, we see that $\langle W_{ij}^2 \rangle = \rho^2/n$ and that $\langle w_i^2 \rangle = 1/n$. This results in:

$$
\langle z_i^2 \rangle = n\langle W_{ij}^2 \rangle \langle w^2 \rangle = n\frac{\rho^2}{n}\frac{1}{n} = \frac{\rho^2}{n}
\tag{23}
$$

This means that $l(\mathbf{z}) = n\langle z_i^2 \rangle = \rho^2$ and that the standard deviation is $\sqrt{\langle z_i^2 \rangle} = \frac{\rho}{\sqrt{n}}$.

From the above the first column of $\mathcal{C}$ has euclidean norm $\|\mathbf{w}\| = 1$, the second ρ, the third ρ^2; the last one $\rho^{(n-1)}$.

6

Since ρ must be smaller than 1, the components of last columns of $\mathcal{C}$ shrink quickly. This fact explains the shading observed in the column of $\mathcal{C}$ for the random case of Fig. 3 and Fig. 4. For the Wigner case the effect if emphasized by the correlations introduced by the symmetry of $\boldsymbol{W}_\mathrm{w}$. The controllability matrix $\mathcal{C}$ for the delay line and the cyclic reservoir can instead be described in exact terms (see Appendices C and D). A sample of each case in provided in Fig. 3 and Fig. 4 for $n = 100$ and $n = 1000$, respectively. For the delay line a complete analysis of the network output can be carried. As shown in Appendix C, we can write that:

$$\mathbf{y}_0 = \mathbf{r} \cdot \boldsymbol{I} \cdot \mathbf{s}_\mathrm{d} = \sum_{i=0}^{n-1} r_i u_{-i} \tag{24}$$

where $\boldsymbol{I}$ is the *identity matrix*. This is, as one would expect, simply a regression model of order n.

For the cyclic reservoir architecture (Subsection 2.2) we show in Appendix D that defined the i-time permuted input weights vector as:

$$\mathbf{w}^{(\mathbf{i})} := \boldsymbol{W}_\mathrm{c}^i \mathbf{w} \tag{25}$$

then, the output of the cyclic reservoir at time zero y_0 is:

$$y_0 = \mathbf{r} \cdot \tilde{\boldsymbol{C}}_\mathrm{c} \cdot \tilde{\mathbf{s}} \tag{26}$$

where

$$\tilde{s}_j = \sum_{p=0}^{\infty} \rho^{j+pn} u_{-j+pn} \tag{27}$$

$$\tilde{\boldsymbol{C}}_\mathrm{c} = \begin{bmatrix} \mathbf{w} & \mathbf{w}^{(1)} & \mathbf{w}^{(2)} & \dots \mathbf{w}^{(n-1)} \end{bmatrix} \tag{28}$$

The fact that, as suggested in [27], $\mathbf{w}$ should be non-periodic for the network to work at its best, is now evident. In fact, if $\mathbf{w}$ is periodic, it means that some columns of $\hat{\boldsymbol{C}}_\mathrm{c}$ are linearly related and, therefore, the rank degenerates, as supported by theoretical arguments in [15].

Note that u_{-j} is only readable through the term $s_j = u_{-j} + \rho^n u_{-(j+n)} + \dots$ and, in order to do that, it must hold that $u_{-j} \gg \rho^n u_{-(j+n)}$. This may suggest to choose small spectral radii, but the smaller the spectral radius, the faster the decay of the memory, since $\hat{s}_j = \rho^j s_j$. This confirms previous intuitions [27], stating that by choosing a small spectral radius, the network preserves an accurate representation of recent inputs, at the expense of losing the ability to recall remote ones. Conversely, if one sets a large spectral radius (i.e., close to 1) the network will be able to (partially) recall inputs from the past, but its memory of more recent inputs will decrease.

5 The nullspace of C and the network memory

By (18) one can understand how the rank of the controllability matrix $\mathcal{C}$ is associated with the degrees of freedom (the effective number of parameters used by the model to solve the task at hand) that can be exploited by the readout (i.e., the "complexity" of the model).

Note from Figures 3 and 4 that the cyclic reservoir always has the highest rank of $\mathcal{C}$, while the Wigner the lowest. The difference increases with the number of neurons.[3]

The fact that $\mathcal{C}$ is not full-rank is linked to the presence of the *nullspace*.[4] This means that there are some network encoded inputs $\mathbf{s}$ which are mapped to $\mathbf{0}$ by $\mathcal{C}$ (17) and hence are indistinguishable by the readout perspective. In Fig. 5, we plot the rank of $\mathcal{C}$ as a function of the reservoir dimension n. In the experiments using Wigner and cyclic reservoirs, the spectral radius ρ and the maximum singular value σ_max coincide and their values are set to 0.995 (Fig. 5a) and 0.9 (Fig. 5b). For the random reservoir, ρ and σ_max are distinct, so we design an experiment where the spectral radius is fixed and another one where the maximum singular value is set (we remind the reader that $\langle \rho \rangle = \frac{1}{2} \langle \sigma_\mathrm{max} \rangle$). But what is the shape of the basis of this nullspace? We show its basis in two cases (see Fig. 6). The controllability matrix obtained with a cyclic reservoir does not have a nullspace for such a value of ρ, since $\mathcal{C}$ is full-rank. Note that, in order to interpret each vector in Fig. 6 as a time series, one must consider the last inputs seen as the ones closer to the origin. Given this interpretation, we clearly see how the memory is linked to the rank of $\mathcal{C}$: the reservoirs' ability to recall past inputs depend on the rank of $\mathcal{C}$ as inputs which only differ in the far-away past are mapped to the same final state.

---

[3]Note that this is coherent with the findings in [15], since the $\boldsymbol{Q}$ defined in that work is simply $Q = \mathcal{C}^\top \mathcal{C}$ and the number of motifs is related to the rank of $\boldsymbol{Q}$ (and so, of $\mathcal{C}$).

[4]What we call the nullspace is practically the *effective* nullspace detected up to the numerical precision, computed using the Numpy dedicated function [55].
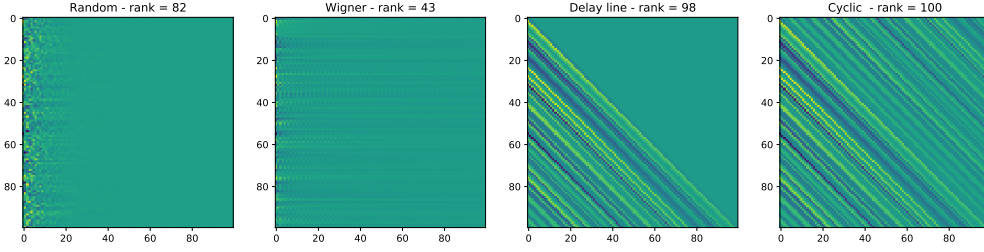
Figure 3: The controllability matrix and its rank for different architectures. The Spectral Radius is $\rho = 0.99$ and the reservoirs has $N = 100$ neurons. The four architectures share the same randomly-generated $\mathbf{w}$.
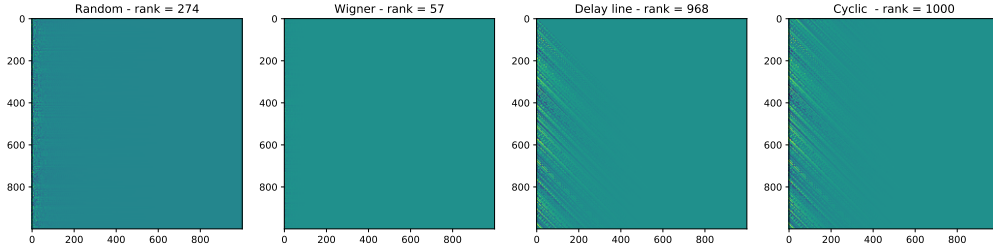

Figure 4: The controllability matrix and its rank for different architectures. The Spectral Radius is $\rho = 0.99$ and the reservoirs has $N = 1000$ neurons. The four architectures share the same randomly-generated $\mathbf{w}$.

Let us consider an example. Let $\mathbf{s}_1$ and $\mathbf{s}_2$ be two network encoded inputs, which differ only in the last $n - m$ elements. Denote by $\mathbf{x}_0^i$ the final state of the system after being fed by the signal $\mathbf{s}_i$. We can then write $\mathbf{s}_2 = \mathbf{s}_1 + \mathbf{d}$, where $\mathbf{d}$ encodes the difference between the two representations. We see that the first m elements of $\mathbf{d}$ are null. So, we can write:

$$\mathbf{x}_0^2 = \mathcal{C}\mathbf{s}_2 = \mathcal{C}(\mathbf{s}_1 + \mathbf{d}) = \mathcal{C}\mathbf{s}_1 + \mathcal{C}\mathbf{d} = \mathcal{C}\mathbf{s}_1 + \mathbf{0} = \mathbf{x}_0^1 \tag{29}$$

since $\mathbf{d}$ lives in the nullspace of $\mathcal{C}$. This results in the network not being able to distinguish between the two signals.

6   Memory curves

In order to validate our theoretical claims, we designed networks asked to remember a random i.i.d. input. We generate inputs of length T and split them in a *training set* ranging in $(0, t_0)$ and a *test set* ranging in (t_0, T). In the task under consideration, the network is trained to reproduce past input (a white noise signal) at a given past-horizon τ, so that $y_k \equiv u_{k-\tau}$. The input signal $\mathbf{u}$ is chosen to be Gaussian i.i.d. white noise, $u_k \sim \mathcal{N}(0, 1)$. With choice $T = 1500$ and $t_0 = 1000$, the training set contains $L_{\text{train}} = 1000$ sample and the test set $L_{\text{test}} = 500$. The readout is finally configured through a least means square procedure and its accuracy is then evaluated on the test set as $\gamma = \max\{1 - \text{NRMSE}, 0\}$, where the Normalized Root Mean Squared Error (NRMSE):

$$\text{NRMSE} := \sqrt{\frac{\sum_{k=t_0}^{T} (y_k - \hat{y}_k)^2}{\sum_{k=t_0}^{T} (y_k - \overline{y})^2}} \tag{30}$$

y_k denotes the system output at time k, $\overline{y} := \frac{1}{T} \sum_{k=t_0}^{T} y_k$ is its average and $\hat{y}_k$ stands for the predicted output.

In Figure 7 the *memory curves*, introduced in [16], are plotted for four different architectures. The Random and the Wigner architectures appear to have a short memory. Their accuracy dramatically decreases as τ grows. The behavior of the cyclic reservoir appears to be radically different. As described in [27], the performance does not decrease gradually, but remains almost constant for some time and then abruptly decreases. The drop in performance occurs when $\tau = n$,

8
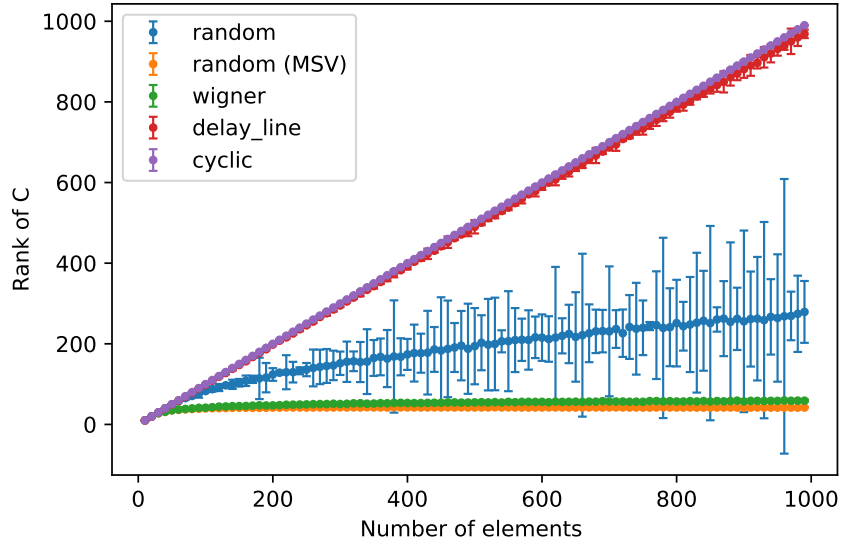
(a)
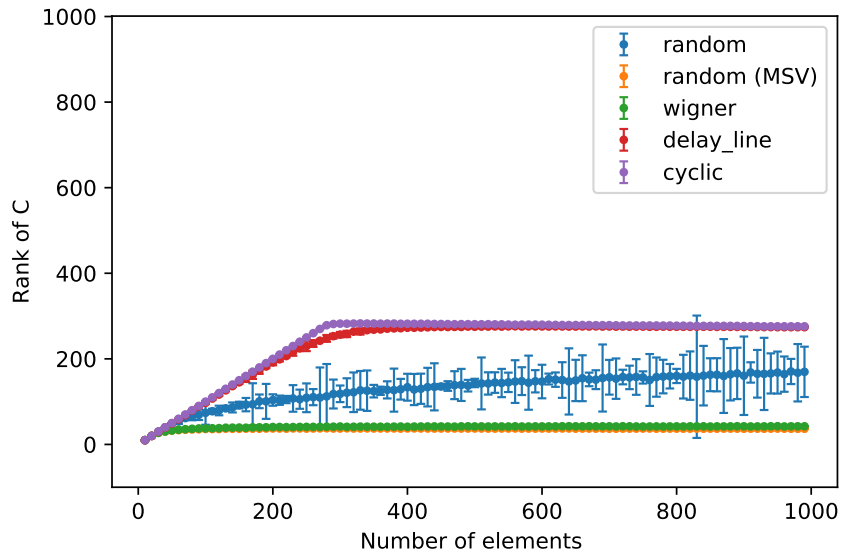
(b)

Figure 5: Ranks of the controllability matrix $\mathcal{C}$ as a function of the reservoir dimension n, for $\rho = 0.995$ (a) and $\rho = 0.9$ (b). Note the saturation of the delay line and the cyclic reservoir, which happens, because ρ is not close enough to 1 for its powers to be numerically distinguishable from zero.

n being the number of neurons in the reservoir. This is coherent with the theory we developed and with the findings in [56]. Note that in [57, Section 3] a similar shape for the memory curve is obtained by using an "almost unitary" reservoir matrix, where all singular values equal a constant $C < 1$. We note that also the cyclic reservoir $\boldsymbol{W}_c$ shares this feature, explaining why the results are similar.

We comment on how the SR affects the performance: when the SR is close to one, the accuracy in the reconstruction of $\mathbf{u}(t - \tau)$ is lower for recent inputs samples (i.e., smaller τ) but higher for distant in time ones. In other words, choosing a large ρ allows the network to better remember the distant past, at the price of compromising its ability to remember
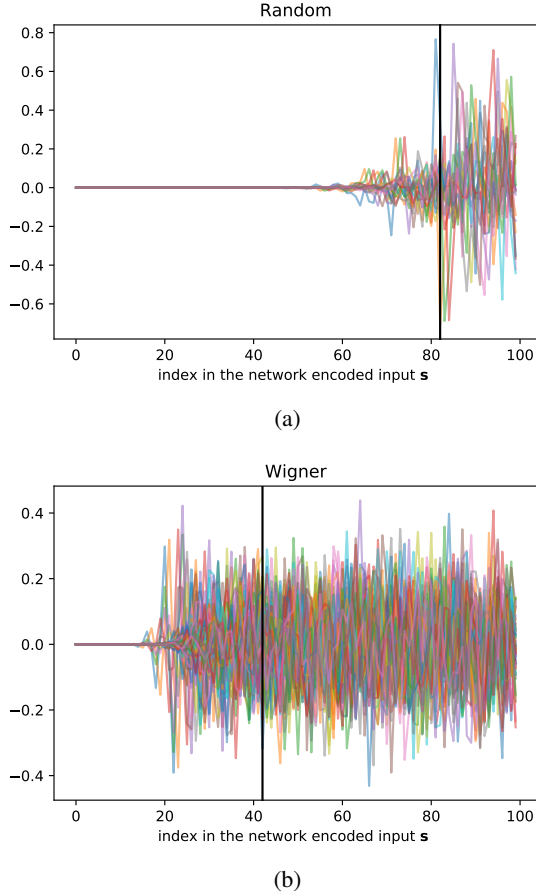
9

(a)

(b)

Figure 6: Nullspace basis for a Random (a) and a Wigner (b) reservoir matrices. Each curve represents a basis vector of the nullspace of $\mathcal{C}$, where the x-axis accounts for the vector components. In both cases, we set the spectral radius to $\rho = 0.99$, while the reservoirs size is $n = 100$. The vertical black lines represent the ranks of $\mathcal{C}$, i.e., the dimensions of the image spaces.

the recent one. This is a direct implication of (27) and (21): a larger SR amplifies the contribution of past inputs over the more recent ones, since the input reproducibility property is controlled by ρ^{j+pn} (see Appendix B for more details).

We investigate the impact of the spectral radius on memory capacity in Figure 8, where the accuracy γ of the three architectures in recalling a past input (at various τ) is plotted as function of the SR. According to our predictions, a larger SR is required to correctly recall inputs that are further in past (but for which $\tau < N$), since the SR controls the magnitude of the $\phi_j^{(k)}$, i.e., the permanence of u_k on the state. We notice that the Random and the Wigner architectures show a similar behavior, with the former displaying a superior performance than the latter. Instead, the Cyclic network has the same behavior as the SR increases, but displays an abrupt fall as it approaches 1. This is coherent with the theory we developed, since for $\rho \approx 1$ the powers of ρ in (27) do not shrink towards zero fast enough to forget remote inputs with the consequence that the network state will be an unreadable superposition of all the past outputs

7   Conclusions

In this paper, we proposed a methodology for explaining how linear reservoirs encode inputs in their internal states. Theoretic findings allow to express the system state in terms of the *controllability matrix* $\mathcal{C}$ and the *network encoded input* s. The matrix properties of $\mathcal{C}$ allow us to compare different connectivity patterns for the reservoir in a quantitative way by. Results show that reservoirs with a cyclic topology give the richest possible encoding of input signals, yet they also offer one the most parsimonious reservoir parametrization. To the best of our knowledge, our contribution pioneers the rigorous study on how specific coupling patterns for the recurrent layer and individual setting of the dynamical
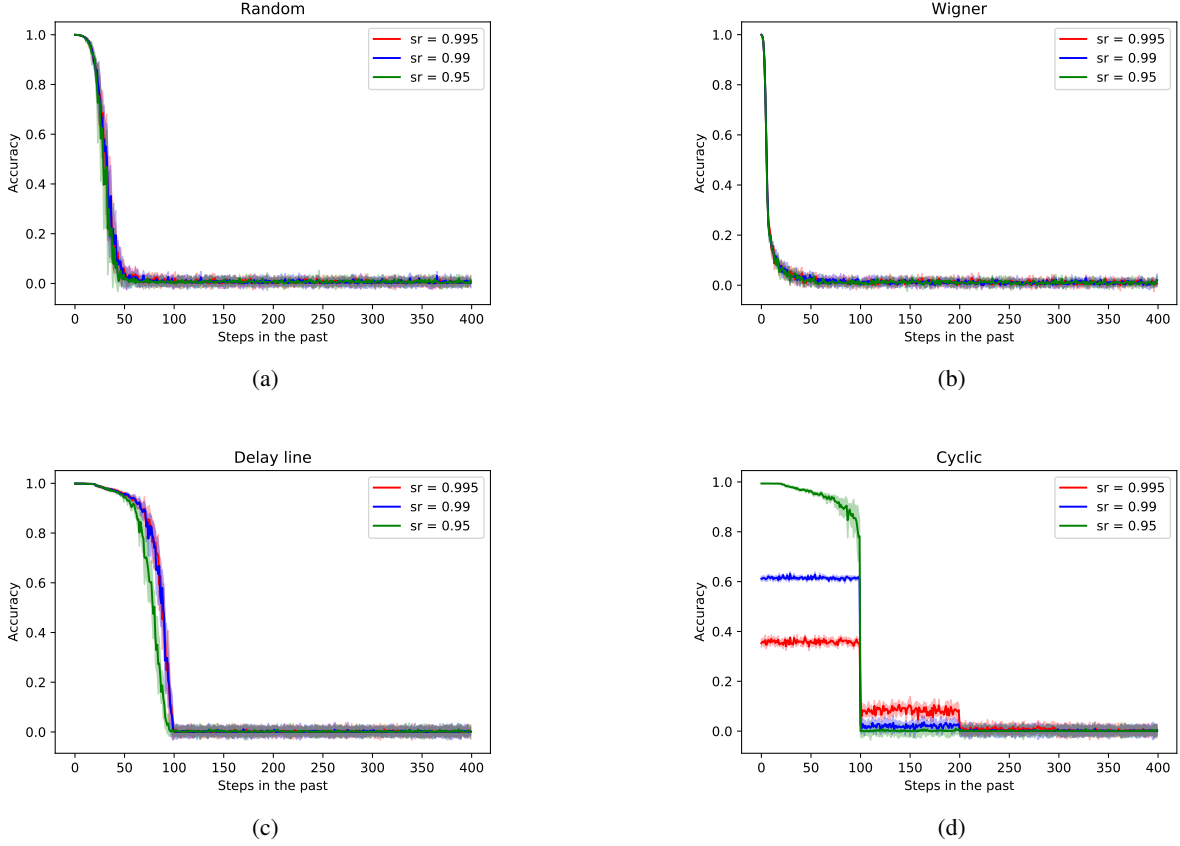
Figure 7: Memory curves for the random, Wigner, delay line and cyclic reservoir with different values of ρ, for a reservoir of $N = 100$ neurons. Plotted values are averages over 10 different realizations, with the shaded area accounting for a standard deviation: a side effect of plotting data in this way is that the value may be negative even if the accuracy is defined as a positive quantity. Note that having a high ability to reconstruct recent inputs ($k < N$) compromises the capacity to remember the more distant ones.

system influence computational properties (e.g., memory), providing deeper insights about phenomena that so far have been observed only empirically in the literature.

A   Cayley-Hamilton Theorem

The CH Theorem allows one to describe the n-th power of a matrix in term of the first $n-1$-powers (including the zero power, which is the identity).

Let $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ be a square matrix. Its characteristic polynomial is defined as:

$$\det\left(\lambda \boldsymbol{I} - \boldsymbol{W}\right) = 0 \quad \rightarrow \quad \lambda^n + \alpha_{n-1}\lambda^{n-1} + \cdots + \alpha_1\lambda + \alpha_0 = 0 \tag{31}$$

where λ is an eigenvalue of $\boldsymbol{W}$ and the α_k are the coefficients of the characteristic polynomial.

Theorem 1 (Cayley-Hamilton) *Every real square matrix satisfies its characteristic equation*

$$\boldsymbol{W}^n + \alpha_{n-1}\boldsymbol{W}^{n-1} + \cdots + \alpha_1\boldsymbol{W}^1 + \alpha_0\boldsymbol{I} = \boldsymbol{0}. \tag{32}$$

Accordingly, it is possible to show that the n-th power of the matrix can be represented as a *linear combination* of its lower powers:

$$\boldsymbol{W}^n = -\alpha_{n-1}\boldsymbol{W}^{n-1} - \cdots - \alpha_1\boldsymbol{W}^1 - \alpha_0\boldsymbol{I} \tag{33}$$
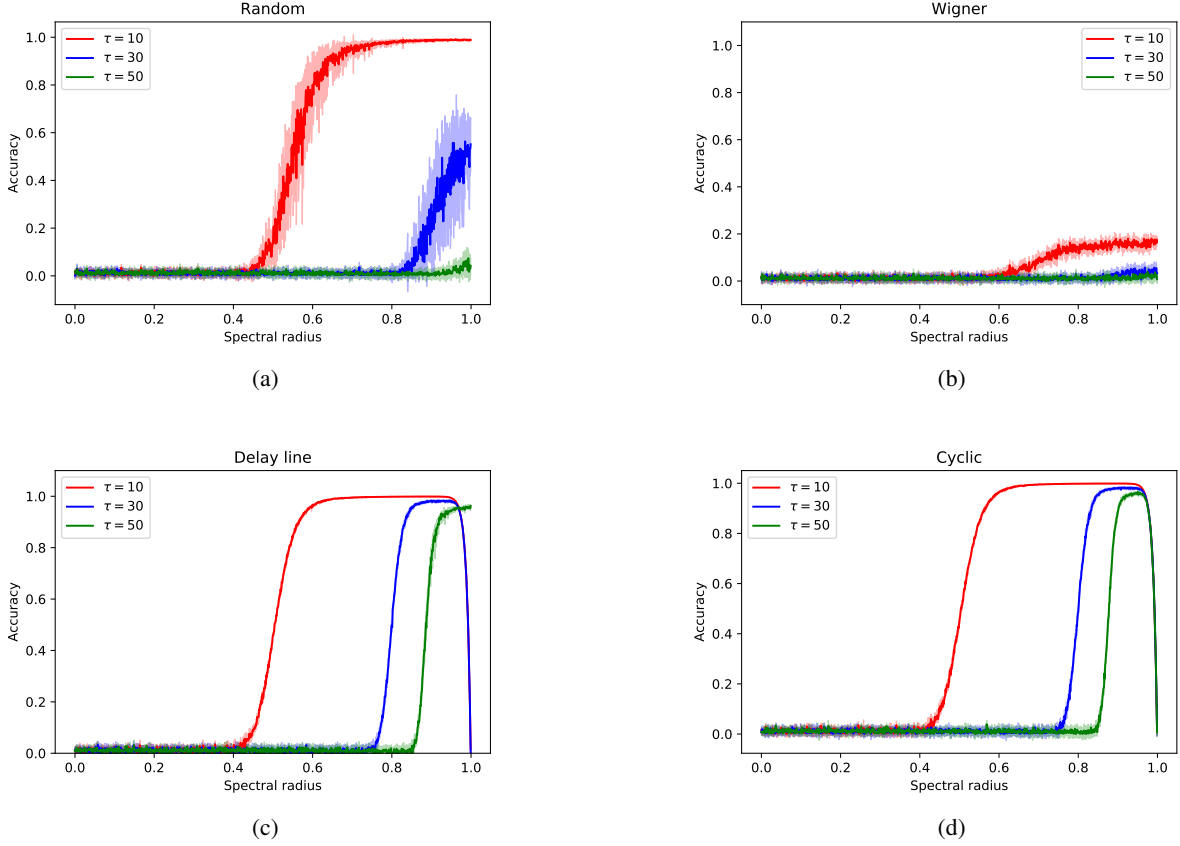
Figure 8: Accuracy in remembering an i.i.d. past-input as function of the spectral radius. All the networks have $N = 100$ neurons. Plotted values are averages over 10 different realizations, with the shaded area accounting for a standard deviation: a side effect of plotting data in this way is that the value may be negative even if the accuracy is defined as a positive quantity.

For matrix $\boldsymbol{W}$, Theorem 1 states:

$$\boldsymbol{W}^n = \varphi_{n-1}\boldsymbol{W}^{n-1} + \varphi_{n-2}\boldsymbol{W}^{n-2} + \cdots + \varphi_1\boldsymbol{W} + \varphi_0\boldsymbol{I} \tag{34}$$

Here, $\boldsymbol{W}$ is a $n \times n$ matrix, $\boldsymbol{I}$ is the $n \times n$ identity matrix, and $\varphi_k = -\alpha_k$ are the negated coefficients of the characteristic polynomial of $\boldsymbol{W}$. It holds true that

$$\boldsymbol{W}^m = \phi_{n-1}^{(m)}\boldsymbol{W}^{n-1} + \phi_{n-2}^{(m)}\boldsymbol{W}^{n-2} + \cdots + \phi_1^{(m)}\boldsymbol{W} + \phi_0^{(m)}\boldsymbol{I} \tag{35}$$

implying that any power $m \geq n$ of $\boldsymbol{W}$ can be specified by $\boldsymbol{W}$ and scalars $(\phi_{n-1}^{(m)}, \ldots \phi_0^{(m)})$. The apexes denote the fact that the n coefficients are those proper of the m-th power for the ϕ_j^m coefficients. Note that, for $m < n$, we have $(\phi_{n-1}^{(i)}, \ldots, \phi_0^{(i)}) = (0, \ldots, 0, 1, 0, \ldots, 0)$, where the only non-zero term is the m-th one, i.e.,

$$\phi_j^{(m)} = \delta_{mj} \quad \text{for} \quad m < n \tag{36}$$

Moreover, note that for $m = n$, $(\phi_{n-1}^{(m)}, \ldots, \phi_0^{(m)}) = (\varphi_{n-1}, \ldots, \varphi_0)$.

For each $m \geq n$, we can derive the scalars in recursive way by noting that:

$$W^{m+1} = W^m W \tag{37}$$

$$= (\phi_{n-1}^{(m)} W^{n-1} + \phi_{n-2}^{(m)} W^{n-2} + \cdots + \phi_1^{(m)} W + \phi_0^{(m)} I) W \tag{38}$$

$$= \phi_{n-1}^{(m)} W^n + \phi_{n-2}^{(m)} W^{n-1} + \cdots + \phi_1^{(m)} W^2 + \phi_0^{(m)} W \tag{39}$$

$$= \phi_{n-1}^{(m)} \left(\varphi_{n-1} W^{n-1} + \varphi_{n-2} W^{n-2} + \cdots + \varphi_1 W + \varphi_0 I \right)$$
$$+ \phi_{n-2}^{(m)} W^{n-1} + \cdots + \phi_1^{(m)} W^2 + \phi_0^{(m)} W \tag{40}$$

$$= \underbrace{(\varphi_{n-1} \phi_{n-1}^{(m)} + \phi_{n-2}^{(m)})}_{\phi_{n-1}^{(m+1)}} W^{n-1} + \underbrace{(\varphi_{n-2} \phi_{n-1}^{(m)} + \phi_{n-3}^{(m)})}_{\phi_{n-2}^{(m+1)}} W^{n-2}$$

$$+ \cdots + \underbrace{(\varphi_1 \phi_{n-1}^{(m)} + \phi_0^{(m)})}_{\phi_1^{(m+1)}} W + \underbrace{(\varphi_0 \phi_{n-1}^{(m)})}_{\phi_0^{(m+1)}} I \tag{41}$$

which implies

$$\begin{cases} \phi_0^{(m+1)} &= \varphi_0 \phi_{n-1}^{(m)} \\ \phi_1^{(m+1)} &= \varphi_1 \phi_{n-1}^{(m)} + \phi_0^{(m)}) \\ &\cdots \\ \phi_{n-2}^{(m+1)} &= \varphi_{n-2} \phi_{n-1}^{(m)} + \phi_{n-3}^{(m)} \\ \phi_{n-1}^{(m+1)} &= \varphi_{n-1} \phi_{n-1}^{(m)} + \phi_{n-2}^{(m)} \end{cases} \tag{42}$$

Eq. 42 can be thought as a linear system:

$$\begin{bmatrix} \phi_0^{(m+1)} \\ \phi_1^{(m+1)} \\ \vdots \\ \phi_{n-2}^{(m+1)} \\ \phi_{n-1}^{(m+1)} \end{bmatrix} = M \begin{bmatrix} \phi_0^{(m)} \\ \phi_1^{(m)} \\ \vdots \\ \phi_{n-2}^{(m)} \\ \phi_{n-1}^{(m)} \end{bmatrix} \tag{43}$$

where M is defined as:

$$M = \begin{bmatrix} 0 & \cdots\cdots & 0 & \varphi_0 \\ 1 & 0 & \cdots & 0 & \varphi_1 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ 0 & \cdots & 1 & 0 & \varphi_{n-2} \\ 0 & 0 & \cdots & 1 & \varphi_{n-1} \end{bmatrix} \tag{44}$$

Note that the characteristic polynomial of M is equal to the one of W, so that they also share the same eigenvalues. In fact, M is also know as the Frobenius companion matrix of W.

B   The network encoded input

The possibility for the readout to produce the correct output for the task at hand depends on two distinct elements: the controllability matrix $\mathcal{C}$ (which depends on W and $\mathbf{w}$) and the $\mathbf{s}$ vector (which depends on both W and the signal $\mathbf{u}$). Here we show how $\mathbf{s}$ is obtained from $\mathbf{u}$.

Under the assumption of bounded inputs $u_{-k} \in [-U, U], \forall k$, we see that

$$|s_j| = \left| \sum_{k=0}^{\infty} \phi_j^{(k)} u_{-k} \right| \leq U \sum_{k=0}^{\infty} \left| \phi_j^{(k)} \right|$$

allowing us to focus on the properties of the $\phi_j^{(k)}$.

13

These terms are the element of $\mathbf{s}$, which we rewrite as:

$$
\begin{bmatrix} s_0 \\ s_1 \\ \dots \\ s_{n-2} \\ s_{n-1} \end{bmatrix} = \begin{bmatrix} \sum_{k=0}^{\infty} \phi_0^{(k)} u_{-k} \\ \sum_{k=0}^{\infty} \phi_1^{(k)} u_{-k} \\ \dots \\ \sum_{k=0}^{\infty} \phi_{n-2}^{(k)} u_{-k} \\ \sum_{k=0}^{\infty} \phi_{n-1}^{(k)} u_{-k} \end{bmatrix}
\tag{45}
$$

In Appendix A we show that for $k < n$, $\phi_j^{(k)} = \delta_{kj}$ (Eq. 36). This implies that the first $n-1$ time steps are simply the inputs:

$$
\begin{bmatrix} s_0 \\ s_1 \\ \dots \\ s_{n-2} \\ s_{n-1} \end{bmatrix} = \begin{bmatrix} u_0 + \sum_{k=n}^{\infty} \phi_0^{(k)} u_{-k} \\ u_{-1} + \sum_{k=n}^{\infty} \phi_1^{(k)} u_{-k} \\ \dots \\ u_{-(n-2)} + \sum_{k=n}^{\infty} \phi_{n-2}^{(k)} u_{-k} \\ u_{-(n-1)} + \sum_{k=n}^{\infty} \phi_{n-1}^{(k)} u_{-k} \end{bmatrix}
$$
$$
= \begin{bmatrix} u_0 \\ u_{-1} \\ \dots \\ u_{-(n-2)} \\ u_{-(n-1)} \end{bmatrix} + \begin{bmatrix} \sum_{k=n}^{\infty} \phi_0^{(k)} u_{-k} \\ \sum_{k=n}^{\infty} \phi_1^{(k)} u_{-k} \\ \dots \\ \sum_{k=n}^{\infty} \phi_{n-2}^{(k)} u_{-k} \\ \sum_{k=n}^{\infty} \phi_{n-1}^{(k)} u_{-k} \end{bmatrix}
\tag{46}
$$

Then, we observe that the terms corresponding to time-step $k = n$ follow from Eq. 34:

$$
\begin{bmatrix} s_0 \\ s_1 \\ \dots \\ s_{n-2} \\ s_{n-1} \end{bmatrix} = \begin{bmatrix} u_0 \\ u_{-1} \\ \dots \\ u_{-(n-2)} \\ u_{-(n-1)} \end{bmatrix} + \begin{bmatrix} u_{-n}\varphi_0 \\ u_{-n}\varphi_1 \\ \dots \\ u_{-n}\varphi_{n-2} \\ u_{-n}\varphi_{n-1} \end{bmatrix} +
$$
$$
+ \begin{bmatrix} \sum_{k=n+1}^{\infty} \phi_0^{(k)} u_{-k} \\ \sum_{k=n+1}^{\infty} \phi_1^{(k)} u_{-k} \\ \dots \\ \sum_{k=n+1}^{\infty} \phi_{n-2}^{(k)} u_{-k} \\ \sum_{k=n+1}^{\infty} \phi_{n-1}^{(k)} u_{-k} \end{bmatrix}
\tag{47}
$$

successive terms corresponding to time steps $k > n$ can be computed by using (42). This procedure shows that, in general, the inputs from 0 to $n - 1$ time steps in the past will *always* appear in their original form, and the "mixing" will begin starting from the n-th time step in the past.

C   Delay line

It is easy to see that, applying $\boldsymbol{W}_{\mathrm{d}}$ to a vector $\mathbf{v} = (v_1, v_2, \dots, v_n)$ results in a vector

$$
\mathbf{v}' := \boldsymbol{W}_{\mathrm{d}ij}\mathbf{v} = (0, v_1, \dots, v_{n-1})
$$

and because of the associativity of the matrix product, we see that applying $\boldsymbol{W}_{\mathrm{d}ij}$ to a vector k times results in permuting the vector k times and the substituting the first k elements with the same number of 0s. So, the controllability matrix for the delay line is:

$$
\mathcal{C}_{\mathrm{d}} = [\mathbf{w}_{\mathrm{d}} \quad \boldsymbol{W}_{\mathrm{d}}\mathbf{w}_{\mathrm{d}} \quad \dots \quad \boldsymbol{W}_{\mathrm{d}}^{n-1}\mathbf{w}_{\mathrm{d}}]
\tag{48}
$$

which would be a lower diagonal matrix for a generic $\mathbf{v}$ but for $\mathbf{w}_{\mathrm{d}} = (1, 0, \dots, 0)$ is just the identity.

Now, consider the fact that

$$
\boldsymbol{W}_{\mathrm{d}}^{n} = \mathbf{0}
\tag{49}
$$

The CH theorem implies that any higher power will be null as well. So we simply have:

$$s_0 = u_0$$

$$s_1 = u_1$$

and so on, because all the $\phi_j^{(m)}$ for $m > n$ are null. If we define $\mathbf{s}_d := (u_0, u_{-1}, u_{-2}, \ldots, u_{-(n-1)})$:

$$\mathbf{y}_0 = \mathbf{r} \cdot \boldsymbol{\mathcal{C}}_d \cdot \mathbf{s}_d = \mathbf{r} \cdot \boldsymbol{I} \cdot \mathbf{s}_d = \sum_{i=0}^{n-1} r_i u_{-i} \tag{50}$$

which, as expected, is simply a regressive model of order n.

D   Cyclic reservoirs

The characteristic polynomial of $\boldsymbol{W}_c$ is $\lambda^n = 1$ so that the CH Theorem implies:

$$\boldsymbol{W}_c^n = \boldsymbol{I} \tag{51}$$

Meaning that, for all $m > n$,

$$\boldsymbol{W}_c^m = \sum_{j=0}^{n-1} \phi_j^{(m)} \boldsymbol{W}_c^j = \boldsymbol{W}_c^{\mu} \tag{52}$$

where $\mu := m \mod n$. Note that, in general:

$$(a\boldsymbol{W}_c)^m = a^m \boldsymbol{W}_c^{\mu} \tag{53}$$

So, if in our reservoir we fix $\boldsymbol{W} = \rho\boldsymbol{W}_c$ (where ρ is a parameter controlling the spectral radius) we obtain a number of simplifications. First of all, the elements of $\mathbf{s}$ assume a regular form. For example:

$$s_0 = u_0 + \rho^n u_{-n} + \rho^{2n} u_{-2n} + \ldots$$

$$s_1 = u_{-1} + \rho^n u_{-(n+1)} + \rho^{2n} u_{-(2n+1)} + \ldots$$

so that their general form is

$$s_j = \sum_{k=0}^{\infty} \phi_j^{(k)} u_{-k} = \sum_{p=0}^{\infty} \rho^{pn} u_{-j+pn} \tag{54}$$

Moreover, the controllability matrix $\boldsymbol{\mathcal{C}}_c$ assumes a simple form. If we define the i-time permuted input weight vector as:

$$\mathbf{w}^{(\mathbf{i})} := \boldsymbol{W}_c^i w \tag{55}$$

we obtain:

$$\boldsymbol{\mathcal{C}}_c = [\mathbf{w} \quad \rho\mathbf{w}^{(1)} \quad \rho^2\mathbf{w}^{(2)} \quad \ldots \quad \rho^{n-1}\mathbf{w}^{(n-1)}] \tag{56}$$

so that:

$$y_0 = (r_0, r_1, \cdots, r_{n-1})\boldsymbol{\mathcal{C}}_c \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_{n-1} \end{pmatrix} \tag{57}$$

The output can be written in compact form by defining:

$$\tilde{s}_j = \sum_{p=0}^{\infty} \rho^{j+pn} u_{-j+pn} \tag{58}$$

$$\tilde{\boldsymbol{\mathcal{C}}}_c = [\mathbf{w} \quad \mathbf{w}^{(1)} \quad \mathbf{w}^{(2)} \quad \ldots \mathbf{w}^{(n-1)}] \tag{59}$$

so that, finally:

$$y_0 = \mathbf{r}\tilde{\boldsymbol{\mathcal{C}}}_c\tilde{\mathbf{s}} \tag{60}$$

Acknowledgment

References

[1] M. Casey, "The dynamics of discrete-time computation, with application to recurrent neural networks and finite state machine extraction," *Neural computation*, vol. 8, no. 6, pp. 1135–1178, 1996.

[2] D. Sussillo and O. Barak, "Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks," *Neural computation*, vol. 25, no. 3, pp. 626–649, 2013.

[3] A. Ceni, P. Ashwin, and L. Livi, "Interpreting recurrent neural networks behaviour via excitable network attractors," *Cognitive Computation*, pp. 1–27, 2019.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[6] C. Tallec and Y. Ollivier, "Can recurrent neural networks warp time?" *arXiv preprint arXiv:1804.11188*, 2018.

[7] J. Van Der Westhuizen and J. Lasenby, "The unreasonable effectiveness of the forget gate," *arXiv preprint arXiv:1804.04849*, 2018.

[8] I. D. Jordan, P. A. Sokol, and I. M. Park, "Gated recurrent units viewed through the lens of continuous time dynamical systems," *arXiv preprint arXiv:1906.01005*, 2019.

[9] M. Inubushi and K. Yoshimura, "Reservoir computing beyond memory-nonlinearity trade-off," *Scientific Reports*, vol. 7, no. 1, p. 10199, 2017.

[10] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, "Information processing capacity of dynamical systems," *Scientific reports*, vol. 2, no. 1, pp. 1–7, 2012.

[11] S. Marzen, "Difference between memory and prediction in linear recurrent networks," *Physical Review E*, vol. 96, no. 3, p. 032308, 2017.

[12] A. Goudarzi, S. Marzen, P. Banda, G. Feldman, C. Teuscher, and D. Stefanovic, "Memory and information processing in recurrent neural networks," *arXiv preprint arXiv:1604.06929*, 2016.

[13] A. Rivkind and O. Barak, "Local dynamics in trained recurrent neural networks," *Physical review letters*, vol. 118, no. 25, p. 258101, 2017.

[14] F. Mastrogiuseppe and S. Ostojic, "A geometrical analysis of global stability in trained feedback networks," *Neural computation*, vol. 31, no. 6, pp. 1139–1182, 2019.

[15] P. Tiňo, "Dynamical systems as temporal feature spaces." *Journal of Machine Learning Research*, vol. 21, no. 44, pp. 1–42, 2020.

[16] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," 2001.

[17] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *science*, vol. 304, no. 5667, pp. 78–80, 2004.

[18] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[19] P. Tino and G. Dorffner, "Predicting the future of discrete sequences from fractal representations of the past," *Machine Learning*, vol. 45, no. 2, pp. 187–217, 2001.

[20] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman, "Experimental demonstration of reservoir computing on a silicon photonics chip," *Nature Communications*, vol. 5, no. 1, pp. 1–6, 2014.

[21] A. Katumba, M. Freiberger, F. Laporte, A. Lugnan, S. Sackesyn, C. Ma, J. Dambre, and P. Bienstman, "Neuromorphic computing based on silicon photonics and reservoir computing," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 24, no. 6, pp. 1–10, Nov 2018.

[22] C. Fernando and S. Sojakka, "Pattern recognition in a bucket," in *European conference on artificial life*. Springer, 2003, pp. 588–597.

[23] H. Ando and H. Chang, "Road traffic reservoir computing," *arXiv preprint arXiv:1912.00554*, 2019.

[24] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: A review," *Neural Networks*, 2019.

[25] L. Gonon, L. Grigoryeva, and J.-P. Ortega, "Risk bounds for reservoir computing," *arXiv preprint arXiv:1910.13886*, 2019.

[26] L. Grigoryeva and J.-P. Ortega, "Echo state networks are universal," *Neural Networks*, vol. 108, pp. 495–508, 2018.

[27] A. Rodan and P. Tino, "Minimum complexity echo state network," *IEEE transactions on neural networks*, vol. 22, no. 1, pp. 131–144, 2010.

[28] S. Ganguli, D. Huh, and H. Sompolinsky, "Memory traces in dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 105, no. 48, pp. 18 970–18 975, 2008.

[29] M. Hermans and B. Schrauwen, "Memory in linear recurrent neural networks in continuous time," *Neural Networks*, vol. 23, no. 3, pp. 341–355, 2010.

[30] E. Bollt, "On explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrasts to VAR and DMD," *arXiv preprint arXiv:2008.06530*, 2020.

[31] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[32] 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*, vol. 28, Atlanta, Georgia, USA, 2013, pp. 1310–1318.

[33] F. M. Bianchi, E. De Santis, A. Rizzi, and A. Sadeghian, "Short-term electric load forecasting using echo state networks and pca decomposition," *Ieee Access*, vol. 3, pp. 1931–1943, 2015.

[34] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, "An overview and comparative analysis of recurrent neural networks for short term load forecasting," *arXiv preprint arXiv:1705.04378*, 2017.

[35] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach," *Physical review letters*, vol. 120, no. 2, p. 024102, 2018.

[36] A. Prater, "Spatiotemporal signal classification via principal components of reservoir states," *Neural Networks*, vol. 91, pp. 66–75, 2017.

[37] R. F. Reinhart and J. J. Steil, "Regularization and stability in reservoir networks with output feedback," *Neurocomputing*, vol. 90, pp. 96–105, 2012.

[38] D. Sussillo and L. F. Abbott, "Generating coherent patterns of activity from chaotic neural networks," *Neuron*, vol. 63, no. 4, pp. 544–557, 2009.

[39] S. Basterrech, "Empirical analysis of the necessary and sufficient conditions of the echo state property," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 888–896.

[40] J. Jiang and Y.-C. Lai, "Model-free prediction of spatiotemporal dynamical systems with recurrent neural networks: Role of network spectral radius," *Physical Review Research*, vol. 1, no. 3, p. 033056, 2019.

[41] K. Caluwaerts, F. Wyffels, S. Dieleman, and B. Schrauwen, "The spectral radius remains a valid indicator of the echo state property for large reservoirs," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2013, pp. 1–6.

[42] P. Verzelli, C. Alippi, and L. Livi, "Echo state networks with self-normalizing activations on the hyper-sphere," *Scientific Reports*, vol. 9, p. 13887, 2019.

[43] I. B. Yildiz, H. Jaeger, and S. J. Kiebel, "Re-visiting the echo state property," *Neural networks*, vol. 35, pp. 1–9, 2012.

[44] 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*, vol. 29, no. 3, pp. 706–717, Mar. 2018.

[45] S. Valverde, S. Ohse, M. Turalska, B. J. West, and J. Garcia-Ojalvo, "Structural determinants of criticality in biological networks," *Frontiers in Physiology*, vol. 6, p. 127, 2015.

[46] P. Verzelli, L. Livi, and C. Alippi, "A characterization of the edge of criticality in binary echo state networks," in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2018, pp. 1–6.

[47] C. G. Langton, "Computation at the edge of chaos: phase transitions and emergent computation," *Physica D: Nonlinear Phenomena*, vol. 42, no. 1-3, pp. 12–37, 1990.

[48] L. Cocchi, L. L. Gollo, A. Zalesky, and M. Breakspear, "Criticality in the brain: A synthesis of neurobiology, models and cognition," *Progress in neurobiology*, vol. 158, pp. 132–152, 2017.

[49] M. Prokopenko, J. T. Lizier, O. Obst, and X. R. Wang, "Relating fisher information to order parameters," *Physical Review E*, vol. 84, no. 4, p. 041116, 2011.

[50] S. Bittanti, *Model identification and data analysis*. Wiley Online Library, 2019.

[51] M. Rudelson and R. Vershynin, "Non-asymptotic theory of random matrices: extreme singular values," in *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*. World Scientific, 2010, pp. 1576–1602.

[52] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*. Princeton, NJ, USA: Princeton University Press, 2009.

[53] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. Springer Science & Business Media, 2013, vol. 6.

[54] L. Gonon, L. Grigoryeva, and J.-P. Ortega, "Memory and forecasting capacities of nonlinear recurrent networks," *Physica D: Nonlinear Phenomena*, vol. 414, p. 132721, 2020.

[55] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006, vol. 1.

[56] A. Rodan and P. Tiňo, "Simple deterministically constructed cycle reservoirs with regular jumps," *Neural computation*, vol. 24, no. 7, pp. 1822–1852, 2012.

[57] H. Jaeger, *Short term memory in echo state networks*. GMD-Forschungszentrum Informationstechnik, 2001, vol. 5.