Optimal reservoir computers for forecasting systems of nonlinear dynamics

Pauliina T. Kärkkäinen¹ and Riku P. Linna¹
Department of Computer Science, Aalto University, P.O. Box 15400, FI-00076 Aalto, Finland

Prediction and analysis of systems of nonlinear dynamics is crucial in many applications. Here, we study characteristics and optimization of reservoir computing, a machine learning technique that has gained attention as a suitable method for this task. By systematically applying Bayesian optimization on reservoirs we show that reservoirs of low connectivity perform better than or as well as those of high connectivity in forecasting noiseless Lorenz and coupled Wilson-Cowan systems. We also show that, unexpectedly, computationally effective reservoirs of unconnected nodes (RUN) outperform reservoirs of linked network topologies in predicting these systems. In the presence of noise, reservoirs of linked nodes perform only slightly better than RUNs. In contrast to previously reported results, we find that the topology of linked reservoirs has no significance in the performance of system prediction. Based on our findings, we give a procedure for designing optimal reservoir computers (RC) for forecasting dynamical systems. This work paves way for computationally effective RCs applicable to real-time prediction of signals measured on systems of nonlinear dynamics such as EEG or MEG signals measured on a brain.

Reservoir computers (RCs) have shown promise in forecasting systems of nonlinear dynamics. Their applicability and performance boils down to finding optimal networks used as reservoirs. We determine the most important factors for designing an optimal RC. We find that the topology of reservoirs of equal connectivity does not affect RC performance. We show that RCs of very low connectivity perform as well or better than their high-connectivity counterparts. The most unexpected finding is that reservoirs of unconnected nodes outperform connected reservoirs in forecasting noiseless systems. In the presence of noise their performance is almost equal to that of connected RCs. This finding is highly important for application of computationally effective RCs in real-time prediction of signals from systems of nonlinear dynamics, an important example being simultaneous stimulation and measurement of brain signals for therapy.

I. INTRODUCTION

Analysis and prediction of dynamical processes based on time dependent signals and time series data is a ubiquitous challenge. Application areas range from predicting tsunamis or avalanches from acoustic or earth’s crust displacement measurements to medicine and everyday control systems. In neurology, analysis of measured time dependent EEG and MEG signals are at the core of understanding and predicting brain function. Specifically, a reliable and fast prediction method of signals reflecting the nonlinear brain dynamics during measurement in order to deliver transcranial magnetic stimulation (TMS) at the right time on the right brain region would have a significant impact on treatment of brain disorders.

Machine learning (ML) techniques are being applied to increasing number of fields, including learning and prediction of dynamical systems. In particular, reservoir computing, a fairly straightforward ML method where a reservoir consisting of a large number of nodes is driven by time dependent signals, has shown promise in reproducing the climate of chaotic systems. In this context, learning the correct climate ensures that the right kind of dynamics will be reproduced, even if the time series data generated by the reservoir computer (RC) might deviate from the driving data (signal) from the actual dynamical system. Emphasizing the climate over exact time prediction makes sense in chaotic systems that by definition do not allow for accurate prediction over long times. Also, some potential applications, such as cryptography, are not so much concerned with accurate generation of time series data pertinent to the dynamical system but rely on sufficient reproduction of the attractor properties. However, in many applications prediction of actual signals would be valuable.

A number of modifications to basic reservoir computing have been proposed. Conceptually, the hybrid model where time integration of a relevant dynamical model is used in combination with RC to make forecasting more accurate would seem promising for incorporating knowledge about the dynamical system in signal prediction. For increased computational efficiency, a modified reservoir computer (RC) was recently introduced by Gauthier et al..

If the available degrees of freedom, or equivalently, capacity for signal representation is taken as the main criterion
FIG. 1. High-level representation of a reservoir computer (RC). In the prediction phase the output of the reservoir is fed back to the input and the reservoir runs autonomously.

for designing RCs, then large and/or structurally complicated reservoirs are seen as the best option for improved quality. While this criterion is sound as such, it would be incorrect to take complexity of reservoir topology as a measure for a reservoir’s performance in dynamical system forecasting. As pointed out by Jaeger et al., sparse interconnectivity within the reservoir of a recurring network is beneficial in that it lets the reservoir decompose into many loosely coupled subsystems, establishing a richly structured reservoir of excitable dynamics. Also the fact that finite training time has an effect on RCs that are found optimal is often overlooked. Moreover, for some specific applications very simple reservoir implementations may be optimal. Computational efficiency in order to facilitate fast training and forecasting is of high priority in designing RCs. For this reason alone reservoirs should be as small and simple as possible. As we show in the present paper, very simple RCs are optimal also in other aspects.

Some very accurate predictions of the signals obtained from chaotic systems have been reported. However, these impressive results were obtained using a single optimal RC and as such are not representative of that particular topology class of RCs. It is fair to say that generally evaluations of RCs for predicting systems of chaotic dynamics have been made using a very low number of RCs, or even a single RC, which has led to some misleading conclusions. This deficiency was first addressed by Haluszczynski et al. who used quite large statistics in their recent study.

A judicious design of RCs for the analysis and prediction of systems of nonlinear dynamics boils down to finding optimal reservoirs. Although there are some promising systematic approaches, a comprehensive theoretical understanding of reservoir computing is still missing, as is typical for ML methods. Consequently, optimization needs to be done by experimentation, as was done by Griffith et al. who used Bayesian hyperparameter optimization.

We set out to determine judicious guidelines for implementing optimal RCs for forecasting nonlinear dynamics. Using a reasonable number of sample RCs with selected topologies, we investigate how reliably optimal hyperparameters can be determined. We use two systems of very different dynamical characteristic as our test bench: the Lorenz system in chaotic, intermittent chaos, and periodic regimes, and a coupled Wilson-Cowan (cW-C) system, a model for neuronal dynamics, in aperiodic (chaotic), quasiperiodic, and periodic regimes. Through systematically applying Bayesian optimization on reservoir hyperparameters we show that low-connectivity RCs perform, in fact, better or at least as well as their high-connectivity counterparts, which in the light of the statement in referred to above makes sense. More unexpectedly, we find that reservoirs of unconnected nodes (RUN) are the optimal choice when there is no measurement noise and that only in the presence of noise higher connectivity is beneficial.

This paper is organized as follows: In Sec. II we describe the operation of RCs. Sec. III describes the main features of the dynamical systems we use. The implementation of the used RCs and how they are evaluated in forecasting are outlined in Secs. IV and V respectively. Results are reported and analyzed in Sec. VI. Here, we first determine the main factors that should be taken into account in optimization regarding reservoir topology, data representation, and the size of the reservoir in Secs. VI. A and VI. B respectively. We then re-evaluate the performance of RUNs in determining Lyapunov exponents of the chaotic Lorenz system in Sec. VI. C. In Sec. VI. D we determine how unconnected reservoirs perform compared with connected reservoirs in the presence of noise. To assess how general our findings are, we compare RC forecasting of Lorenz and cW-C systems in Sec. VI. E. Finally, we summarize and outline our conclusions in Sec. VII.

II. RESERVOIR COMPUTER

RC comprises an input layer, an artificial neural network, and an output layer. The links between the nodes of the network have a direction, that is, each link establishes a connection either to or from a node, see Fig. 1. We use the number of connections to a node, in-degree $k$, as the measure of network connectivity. In the initial training of RC
the time-series data \( u(t) \), in applications typically either stored or obtained from real-time measurement, are fed to the reservoir network.

The connections from the input to the \( D_r \) nodes of the network are determined by the input matrix \( W_{in} \). The time evolution of the reservoir state \( r(t) \) is determined by

\[
\dot{r}(t) = -\gamma r(t) + \gamma \tanh[\lambda r(t) + W_{in}u(t)],
\]

where \( \lambda \) and \( r(t) \) are the reservoir’s adjacency and state matrices, respectively. \( \gamma \) defines the natural rate, or the inverse time scale, of the reservoir dynamics. Dimensions are: \( |u| = d, |r| = N, |\lambda| = N \times N, \) and \( |W_{in}| = N \times d \). The input matrix \( W_{in} \) and the topology of the reservoir via \( \lambda \) are determined before training and remain fixed after this.

The reservoir output is fed to the output layer that transforms reservoir output to \( W_{out}\tilde{r}(t) \). Here, \( \tilde{r}(t) = f_{out} r(t) \) is a transformation that removes unwanted symmetries in the reservoir. The symmetry in the combined system of the reservoir and Lorenz dynamics may deteriorate prediction. In order to break this symmetry, we use the form of \( f_{out} \) that transforms the reservoir node values as

\[
\tilde{r}_i(t) = \begin{cases} r_i(t) & \text{if } i \leq N/2, \\ r_i(t)^2 & \text{if } i > N/2. \end{cases}
\]

In the supervised training stage, \( W_{out} \) is adjusted such that \( W_{out}\tilde{r}(t) \) approximates the output \( u(t) \) of the dynamical system that is known for a time interval \( t \in [0, T_{\text{train}}] \), where \( T_{\text{train}} \) is the training time. This is done by ridge regression that minimizes the quantity

\[
\sum_{t=0}^{T_{\text{train}}} |u(t) - W_{out}\tilde{r}(t)|^2 + \mu ||W_{out}||^2,
\]

where \( \mu \) is the ridge regularization parameter. After this stage \( W_{out} \) remains fixed.

Forecasting is done by using the trained RC. At this stage the output \( W_{out}\tilde{r}(t) \) is continuously fed back to the input and RC runs autonomously:

\[
\dot{r}(t) = -\gamma r(t) + \gamma \tanh[\lambda r(t) + W_{in}W_{out}\tilde{r}(t)].
\]

In what follows, we often refer to the numerical form of the forecasting:

\[
\begin{align*}
    r(t + \Delta t) &= (1 - \beta) r(t) + \\
    &+ \beta \tanh[\lambda r(t) + W_{in}W_{out}\tilde{r}(t)].
\end{align*}
\]

Here, \( \Delta t \) is the time step. \( \beta = \gamma \Delta t \) is the leakage rate. Unlike the term implies, \( \beta \in [0, 1] \) is dimensionless. It is seen in Eq. [5] that the value of \( \beta \) determines how large a portion of the past state of the reservoir is directly repeated in the present state. For \( \beta = 1 \) this effect is at minimum and the present state is determined only through neural-network-type evolution by the second term in Eq. [5].

### III. DYNAMICAL SYSTEMS

We mainly use the Lorenz system in the chaotic regime for optimization and evaluation of RCs in forecasting. In order to gain understanding on the contributions of the dynamical system and the method to the resulting error we use this system also in the periodic and in intermittently chaotic regimes. For comparison and to evaluate how general our findings on the Lorenz system are we evaluate forecasting of a system consisting of two reciprocally coupled Wilson-Cowan (cW-C) models in three regimes: aperiodic (chaotic), quasiperiodic, and periodic. Wilson-Cowan oscillators that are commonly used as neuronal models serve the purpose of addressing the challenges in using RCs for prediction and analysis in this context. Each dynamical system is used to produce data \( u(t) \) extending over time \( t \in [0, T] \). The data extending over \( t \in [0, T_{\text{train}}] \), where \( T_{\text{train}} < T \) is used for training the RC and and the remaining data \( u(t) \), where \( t \in (T_{\text{train}}, T] \) is used for evaluating the precision of forecasting.
A. The Lorenz system

The three-dimensional dynamical system, introduced by Edward Lorenz in 1963,\(^\text{21}\), is defined as

\[
\begin{align*}
\dot{x} &= \sigma(y - x) \\
\dot{y} &= rx - y - xz \\
\dot{z} &= xy - bz.
\end{align*}
\]

(6)

Originally Lorenz chose the parameter values \(\sigma = 10\) and \(b = 8/3\) and by varying the parameter \(r\) brought the system in different dynamical regimes such as dissipative, periodic, and, most notably, chaotic. Lorenz discovered his standard (oscillation) \((r = 28)\), intermittent chaos \((r = 100)\), and periodic (oscillation) \((r = 147)\) regimes. We use the inverse of the largest positive Lyapunov exponent of the standard chaotic Lorenz system, \(\lambda = 0.9056\), as our as our time reference.

B. The system of coupled Wilson-Cowan models

The dynamical system of two reciprocally connected Wilson-Cowan models, indexed \(l \in \{1, 2\}\), is defined as\(^\text{22}\)

\[
\begin{align*}
\tau_e \frac{dE_1(t)}{dr} &= -E_1(t) + (k_e - E_1(t)) \cdot S_e(c_1E_1(t)) \\
&\quad - c_2I_1(t) + P + \alpha E_2(t)) \\
\tau_i \frac{dI_1(t)}{dr} &= -I_1(t) + (k_i - I_1(t)) \cdot S_i(c_3E_1(t)) \\
&\quad - c_4I_1(t) \\
\tau_e \frac{dE_2(t)}{dr} &= -E_2(t) + (k_e - E_2(t)) \cdot S_e(c_1E_2(t)) \\
&\quad - c_2I_2(t) + P' + \alpha E_1(t)) \\
\tau_i \frac{dI_2(t)}{dr} &= -I_2(t) + (k_i - I_2(t)) \cdot S_i(c_3E_2(t)) \\
&\quad - c_4I_2(t)
\end{align*}
\]

(7)

where \(\tau_e\) and \(\tau_i\) are time constants of excitatory and inhibitory neurons, respectively, \(E_l(t)\) and \(I_l(t)\) are average activities of groups of these neurons at time \(t\), \(c_1, c_2, c_3,\) and \(c_4\) are coupling strengths between and within neural groups, and \(k_e\) and \(k_i\) are constants. The coupling strength of the two Wilson-Cowan models is defined by \(\alpha\). \(S_e\) and \(S_i\) are sigmoidal functions defined as

\[
\begin{align*}
S_e(x) &\equiv \frac{1}{1 + \exp(-a_e(x - \theta_e))} - \frac{1}{1 + \exp(a_e\theta_e)} \\
S_i(x) &\equiv \frac{1}{1 + \exp(-a_i(x - \theta_i))} - \frac{1}{1 + \exp(a_i\theta_i)}
\end{align*}
\]

where parameter values \(\theta_e = 4.0\), \(a_e = 1.3\), \(\theta_i = 3.7\), and \(a_i = 2.0\) are used. cW-C models were run with parameter values \(P = 1.09\), \(P' = 1.06\), \(\tau_e = \tau_i = 1.0\), \(k_e = k_i = 1.0\), \(c_1 = 16.0\), \(c_2 = 12.0\), \(c_3 = 15.0\), and \(c_4 = 3.0\). RC forecasting was evaluated for three different regimes: aperiodic (chaotic) \((\alpha = 1.3)\), quasiperiodic \((\alpha = 1.9)\), and periodic \((\alpha = 4)\).

IV. RC IMPLEMENTATION

In designing a RC, the first thing to determine is the topology of the reservoir, that is, the connections of its nodes and also connection strengths by the adjacency matrix \(A\). The input connections are defined by \(W_{in}\).

Judiciously selected low-connectivity reservoirs were found to perform as well as high-connectivity reservoirs in dynamical system forecasting\(^\text{19}\). This performance is crucially dependent on reservoir optimization. In finding the
optimal RC we test the performance of five network topologies also used by Griffith et al.\textsuperscript{19}, a random Erdös-Rényi (ER) network ($k \in [1, 5]$), a network including a single cycle ($k = 1$), a network including no cycles ($k = 1$), a single cycle including all the nodes ($k = 1$), and a single line including all the nodes ($k = 0$). The elements of adjacency matrices $A$ for all networks are generated as random variables drawn from the standard normal distribution. The input matrix $W_{in}$ is determined such that connections are first established with a given probability, after which the strength for each existing connection is a number chosen randomly from a standard normal distribution. The elements of $A$ and $W_{in}$ are scaled such that a given spectral radii $\rho_r$ and $\rho_{in}$ are obtained.

The important part of finding hyperparameter values that optimize RC performance in forecasting is done by Bayesian optimization that is well suited for target functions involving randomness. Optimal values are found for five hyperparameters, namely spectral radius $\rho_r$, probability of connecting an element of $W_{in}$ to RC $\sigma$, spectral radius of $W_{in}$ $\rho_{in}$, leakage rate $\beta$, and regression parameter $\mu$. Here, we make close comparison to the study by Griffith et al.\textsuperscript{19}. We corroborate their main finding that reservoirs of low connectivity perform unexpectedly well. However, our results differ from theirs in some important aspects when optimization is done using sufficient statistics.

After having decided on the topology of the reservoir network, an optimization procedure to determine values for the selected hyperparameters is performed, see Fig. 2. After this, we either use these optimized reservoirs as such or train more reservoirs using the hyperparameter values obtained from optimization. In the latter case, $W_{out}$ is determined and fixed for each reservoir in the training stage. The trained RCs are then ready to be used for forecasting based on time-series data.

V. RC EVALUATION

During forecasting we evaluate the performance of RC over time $T_{eval}$ by the root-mean-square error averaged over $P$ start times

$$\varepsilon = \left\{ \frac{1}{P} \sum_{i=1}^{P} \varepsilon_i^2 \right\}^{1/2},$$

where $\varepsilon_i^2 = \frac{\Delta T_{eval}}{T_{eval}} \sum_{t=t_i}^{t_i+T_{eval}} |u(t) - W_{out}\tilde{r}(t)|^2$. In the Lorenz system $T_{eval} = 1/\lambda$, that is, one Lyapunov period. In a chaotic system the $P$ start times are points in the attractor, so $\varepsilon$ is calculated as an average over separate trajectories. We both minimize $\varepsilon$ in determining the optimal hyperparameter values and use it as a figure of merit for the evaluation of RC performance in forecasting. Using $\varepsilon$ as a figure of merit instead of $\varepsilon_i$ better captures how well the climate, the attractor, is reproduced by RC.\textsuperscript{19} For consistency, we use $\varepsilon$ also for systems that are not chaotic, despite the start times lying on the same trajectory in such cases.

A more direct figure of merit for forecasting is the valid time $T_v$ defined as the elapsed time before the normalized error $E(t) = \frac{||u - W_{out}\tilde{r}(t)||}{||u||}$ exceeds some value $f \in [0, 1]$. We use $f = 0.4$. Here $||\cdot||$ denotes the $L_2$-norm. As will be seen in Sec. \textsuperscript{VIE} using $\varepsilon$ and $T_v$ aids in distinguishing contributions of the dynamical system and the RC method to the error in forecasting.
We first performed optimization using the same ranges for hyperparameters as Griffith et al., see the first row, labeled R, in Table I. One hundred iterations of Bayesian optimization were performed. Networks of sizes $D = 100$ were generated using parameter values obtained from the optimization. In order to evaluate the validity of the optimization procedure we run it for 20 times for each RC topology. Each optimization produces a different realization of a given topology where the input connections and internal link strengths differ, which results in the shown distributions of all hyperparameter values, see Fig. 3. After this the median value of each hyperparameter was used for generating the 200 RCs that we tested. This is a major difference to the procedure by Griffith et al. who found optimal parameter values by a single run of the optimization algorithm and then used this set of values for generating RCs. We also include the ridge regularization parameter $\mu$ in our Bayesian optimization, unlike Griffith et al. who optimize $\mu \in [10^{-5}, 10^5]$ separately by leave-one-out cross validation.

We first normalize the variance of the data $u(t)$ as was done in\cite{Giffith2019}. The normalized quantity for each component $i \in [1, D]$, where $D$ is the dimension of the system, is obtained as $\bar{u}_i(t) = [u_i(t) - \bar{u}_i] / \text{Var}(u_i)$. Here $\bar{u}_i$ is the mean and $\text{Var}(u_i)$ is the variance of component $u_i$ computed over the training time. We inspect data normalization in RCs later in this section.

Fig. 3 shows distributions of optimal values of hyperparameters. The medians of these optimal values are shown in Table I. As can be seen in the distribution of optimal values of $k$ for the ER networks, the most probable optimal RC will have $k = 1$. Based on optimal hyperparameter values found for one RC it was previously found that very low-connectivity reservoirs may perform as well as reservoirs of higher connectivity\cite{Giffith2019}. Our finding based on optimal hyperparameter values determined by larger statistics is stronger: very low-connectivity ($k = 1$) RCs perform better than higher-connectivity RCs of ER topology.

The $k$-distribution for ER networks also displays the difficulty in determining optimal hyperparameter values: the median of this $k$-distribution is $k = 2$, which clearly is not a good choice as the optimal value.

The distributions of optimal values are seen to be wide. Consequently, determining optimal hyperparameter values by a single optimization procedure, as was done in\cite{Giffith2019}, is not sufficient. Applying the set of hyperparameter values obtained this way to a different RC belonging to the same topology class does not guarantee optimal performance. This is seen in Fig. 4, where distributions of averaged short time prediction errors $\epsilon$ (see Eq. (8)) are shown for the different topologies. The vertical lines show $\text{Med}(\epsilon)$, the median values for measured errors. The first column shows distributions of $\epsilon$ for RCs with hyperparameter values that in this section.

Fig. 3 shows distributions of optimal values of hyperparameters. The medians of these optimal values are shown in Table I. As can be seen in the distribution of optimal values of $k$ for the ER networks, the most probable optimal RC will have $k = 1$. Based on optimal hyperparameter values found for one RC it was previously found that very low-connectivity reservoirs may perform as well as reservoirs of higher connectivity\cite{Giffith2019}. Our finding based on optimal hyperparameter values determined by larger statistics is stronger: very low-connectivity ($k = 1$) RCs perform better than higher-connectivity RCs of ER topology.

The $k$-distribution for ER networks also displays the difficulty in determining optimal hyperparameter values: the median of this $k$-distribution is $k = 2$, which clearly is not a good choice as the optimal value.

The distributions of optimal values are seen to be wide. Consequently, determining optimal hyperparameter values by a single optimization procedure, as was done in\cite{Giffith2019}, is not sufficient. Applying the set of hyperparameter values obtained this way to a different RC belonging to the same topology class does not guarantee optimal performance. This is seen in Fig. 4, where distributions of averaged short time prediction errors $\epsilon$ (see Eq. (8)) are shown for the different topologies. The vertical lines show $\text{Med}(\epsilon)$, the median values for measured errors. The first column shows distributions of $\epsilon$ for RCs with hyperparameter values that in this section.

| $\rho$  | $p_{in}$ | $p_{in}$ | $\beta$ | $\log \mu$ | $k$ |
|---------|---------|---------|---------|-----------|-----|
| R [0.3, 1.5] | [0.0, 1.0] | [0.3, 1.5] | [0.07, 0.11] | [-5.5] | [1, 5] |
| A [0.1, 1.5] | [0.1, 1.0] | [0.1, 1.5] | [0.05, 1.0] | [-5.0] | [1, 5] |
| B [0.0, 1.5] | [0.0, 1.0] | [0.0, 1.5] | [0.05, 1.0] | [-5.0] | [1, 5] |

TABLE I. Hyperparameter optimization ranges for RCs with ER networks. $\rho$, $p_{in}$, $p_{in}$, $\beta$, $\log \mu$, and $k$ are spectral density, input connection probability, spectral radius of the input matrix, leakage rate, logarithm of the ridge regularization parameter, and in-degree of the reservoir nodes, respectively.

VI. RESULTS

Optimizing the RC consists of two parts: finding the optimal topology and hyperparameter values for the task. The general reservoir structure is chosen first, after which hyperparameter values are determined using Bayesian optimization, see Fig. 2. When the in-degree $k$ is one of these parameters, hyperparameter optimization also optimizes the topology of the reservoir although its structure is predetermined. Data representation turned out to have a significant effect on RC’s performance in forecasting, so this, too, can be regarded as part of RC optimization.

A. Optimization, reservoir topology, and data representation

We first normalize the variance of the data $u(t)$ as was done in\cite{Giffith2019}. The normalized quantity for each component $i \in [1, D]$, where $D$ is the dimension of the system, is obtained as $\bar{u}_i(t) = [u_i(t) - \bar{u}_i] / \text{Var}(u_i)$. Here $\bar{u}_i$ is the mean and $\text{Var}(u_i)$ is the variance of component $u_i$ computed over the training time. We inspect data normalization in RCs later in this section.

Fig. 3 shows distributions of optimal values of hyperparameters. The medians of these optimal values are shown in Table I. As can be seen in the distribution of optimal values of $k$ for the ER networks, the most probable optimal RC will have $k = 1$. Based on optimal hyperparameter values found for one RC it was previously found that very low-connectivity reservoirs may perform as well as reservoirs of higher connectivity\cite{Giffith2019}. Our finding based on optimal hyperparameter values determined by larger statistics is stronger: very low-connectivity ($k = 1$) RCs perform better than higher-connectivity RCs of ER topology.

The $k$-distribution for ER networks also displays the difficulty in determining optimal hyperparameter values: the median of this $k$-distribution is $k = 2$, which clearly is not a good choice as the optimal value.

The distributions of optimal values are seen to be wide. Consequently, determining optimal hyperparameter values by a single optimization procedure, as was done in\cite{Giffith2019}, is not sufficient. Applying the set of hyperparameter values obtained this way to a different RC belonging to the same topology class does not guarantee optimal performance. This is seen in Fig. 4, where distributions of averaged short time prediction errors $\epsilon$ (see Eq. (8)) are shown for the different topologies. The vertical lines show $\text{Med}(\epsilon)$, the median values for measured errors. The first column shows distributions of $\epsilon$ for RCs with hyperparameter values that in this section.

The error distributions in the first column of Fig. 4 are similar to those reported in\cite{Giffith2019}. Using different reservoir topologies seems to result in different median values $\text{Med}(\epsilon)$. These differences are small except for the reservoir consisting of a single cycle, which appears to be the poorest choice for forecasting. Here the error distribution is the widest and skewed, so the probability for errors considerably larger than the median value is much higher than for other RC topologies.
FIG. 3. Optimized parameters for each topology. Topologies from top down are an ER network where $k \in [1, 5]$, a network where $k = 1$ with one cycle, a network where $k = 1$ with no cycles (tree), a network consisting of a single cycle, and a network consisting of a single line. Optimization ranges $R$ (see Table I). Distributions were found fairly similar when optimization was run for different sets of 20 reservoirs.

| Typology          | $\varepsilon$ | $T_v$ | $\rho$ | $\rho_m$ | $\beta$ | $\log \mu$ |
|-------------------|---------------|------|--------|----------|---------|-----------|
| ER                | 0.02          | 4.78 | 0.30   | 0.64     | 0.32    | 0.08      | $-5$      |
| cycle included    | 0.03          | 4.41 | 0.30   | 0.72     | 0.30    | 0.09      | $-5$      |
| single tree       | 0.03          | 4.27 | 0.33   | 0.63     | 0.30    | 0.09      | $-5$      |
| single cycle      | 0.03          | 4.40 | 0.30   | 0.78     | 0.30    | 0.07      | $-5$      |
| single line       | 0.03          | 4.79 | 0.30   | 0.79     | 0.30    | 0.08      | $-5$      |

TABLE II. Median values of figures of merit, $\varepsilon$ and $T_v$, and optimized hyperparameters for each topology. $k = 1$ for all but ER, for which the median value $k = 2$. Error normalized data (EN) and optimization range $R$ (see Table I).

In contrast to the error distributions shown in the first column of Fig. 3, in the second column there are no discernible differences between different reservoir topologies. Also $\text{Med}(\varepsilon)$ are equal for all topologies. We also checked that the resulting distributions did not change appreciably by using ten times longer training times for both sets of hyperparameter values. In other words, in the first column differences of the median values for different topologies result from insufficient optimization statistics.

We conclude that the variation in distributions and $\text{Med}(\varepsilon)$ in the first column for different reservoir topologies is a consequence of the hyperparameter values deviating too much from the true optimal values for individual RCs. In contrast, choosing median values by running Bayesian optimization for 20 times gives consistent error estimates and similar distributions for all topologies. The important conclusion is that for reservoirs of very low connectivity $k = 1$ there is no optimal topology for time-series forecasting.

A striking characteristic in Fig. 3 is that the most probable optimal values of many of the hyperparameters lie close to the minimum or maximum limit of the range used for optimization. This would suggest that the used optimization ranges are too narrow. This, indeed, proves to be the case.

So far, each component $u_i(t)$ of the time-series data was transformed to $\tilde{u}_i(t) = [u_i(t) - \bar{u}_i]/\text{Var}(u_i)$ and used in this form in RC. This data normalization is commonly used in gradient-descent based learning for faster convergence. It normalizes the variation of the data and so the error that is used as a figure of merit $\varepsilon$, see Eq. (8). However, it does not preserve the relative magnitudes of $u_i(t)$ within the reservoir. Intuitively, this is an unwanted characteristic. While we do want to scale $u_i(t)$ such that magnitudes vary within a reasonable range, it would seem natural that the data by which the network learns to trace $u(t)$ preserves interrelated magnitudes of its components.

We test our intuition by comparing RCs with ER networks using data as is, $u(t)$, variance-normalized data $\tilde{u}(t)$, and data normalized as $\hat{u}(t) \in [0, 1]: \hat{u}(t) = [u(t) - u_{\text{min}}(t)]/[u(t)_{\text{max}} - u(t)_{\text{min}}]$, where $u(t)_{\text{min}}$ and $u(t)_{\text{max}}$ are the minimum and maximum of all components $u_i(t)$ and $t \in [0, T_{\text{train}}]$. Hence, unlike when using $\tilde{u}_i(t)$, we normalize the data instead of fluctuations by using $\hat{u}(t)$.

The first optimizations resulted in many of the optimal hyperparameter values lying at the lower ends of the available set of ranges labeled $R$. Hence, concurrently with investigating the optimal data representation we allow for larger hyperparameter ranges, except for $\mu$ that never obtained values greater than one. Here, we apply two sets of ranges,
FIG. 4. Distributions of averaged short time prediction errors $\epsilon$ from 200 simulations. These are visualized as a Gaussian kernel density estimation in $\log_{10} \epsilon$ with bandwidth of 0.1. The first column: hyperparameters the same as in 19. The second column: hyperparameters median values from 20 Bayesian optimization runs.

| Data | $\rho$ | $k$ | $\rho_m$ | $\rho_u$ | $\beta$ | $\log \mu$ | $\epsilon$ | $T_v$ |
|------|--------|-----|----------|----------|---------|-------------|----------|------|
| OR   | 0.10   | 2   | 0.76     | 0.10     | 0.18    | -4          | 0.02     | 4.67 |
| EN   | 0.10   | 3   | 0.79     | 0.36     | 0.09    | -5          | 0.05     | 4.32 |
| DN   | 0.74   | 2   | 0.77     | 1.28     | 0.39    | -5          | 0.02     | 5.20 |
| OR   | 0.36   | 2   | 0.45     | 0.06     | 0.23    | -4          | 0.03     | 4.45 |
| EN   | 0.44   | 3   | 0.77     | 0.29     | 0.18    | -4          | 0.08     | 3.50 |
| DN   | 1.01   | 1   | 0.64     | 1.31     | 0.35    | -5          | 0.02     | 5.15 |

TABLE III. ER reservoirs. Optimal hyperparameter values and figures of merit, $\epsilon$ and $T_v$, using different data transformations in 200 RCs: OR - original data, EN - data transformed such that error is normalized, and DN - data normalized. The first three rows are for the set A of hyperparameter optimization ranges and the following three rows for the set B (see Table I). $D_r = 100$. The narrower ranges of A results in slightly better performance due to faster optimization.

labeled A and B, wider than the set R, see Table I. Importantly, in set B, $\rho = 0$ and, consequently, reservoirs of unconnected nodes (RUN) are allowed.

Wider ranges require longer optimization. Accordingly, for the sets A and B Bayesian optimization was run for 200 and 500 iterations, respectively. As before, optimization procedure was repeated 20 times and medians of the results were used as hyperparameter values in generating the reservoirs. Here, reservoir size $D_r = 100$ was used. Figures of merit, $\epsilon$ and $T_v$, obtained for 200 RCs using optimal hyperparameter values for raw data and the data transformed in the ways described above are shown in Table III.

The error-normalizing transformation (EN) of the data is seen to result in the poorest performing RCs: errors are the largest and valid times the shortest. Normalizing the data (DN) improves the performance of RCs compared to using original raw data (OR). In other words, scaling the data so that it varies within $[0, 1]$ and at the same time preserves relative magnitudes of the components seems to be the optimal form of representing data in reservoir computing. As EN also scales the data so that it varies within limits close to unity but results in poorer RC performance than using raw data (OR), we conclude that of scaling and preserving the relative magnitudes the latter is more important. Since the best forecasting was obtained using DN, we normalize the data in all our RC implementations including and from Sec. VI B onward.

For the optimization ranges B where a reservoir of unconnected nodes is allowed, optimization ended up with $\rho = 0$ in 3 out of 20 cases for raw data (OR) and 1 out of 20 cases for normalized data (DN). This is an even more surprising outcome than the previous finding that a reservoir of very low connectivity may perform better than one of higher connectivity. To verify (or disprove) this outcome we include reservoirs of unconnected nodes in the following simulations.

To make sure that the found optimal reservoirs of very low connectivity are not just optima within the hyperparameter ranges we chose, we also ran optimization of ER reservoirs using the wider range of $k \in [1, 10]$ and several ranges
for $\rho$. We found that even when allowing for larger $k$, optimization tends to result in minimal $k$ values, which only vary slightly with the minimum value allowed for $\rho$. Decreasing this minimum value leads to slightly more connected topologies. So, the optimal hyper parameter values tend to fall within the ranges used in this paper and within these smaller ranges optimal $k$ values again tend toward the lower end of the range.

B. Optimization and network size

Similarly to the (wrong) expectation that highly connected reservoirs outperform those of low connectivity, one also expects that a larger reservoir always performs at least as well as a smaller one. As it turns out, the matter is not quite that straightforward. Again, optimization procedure plays a crucial role in the performance of reservoirs of different sizes. In what follows, we compare performance of RCs of different topologies and sizes.

Fig. 5 shows medians of valid times $T_v$ measured for 100 RCs of the five different topologies. The training was run either for $10^4$ (left column) or $10^5$ (right column) time steps of size $\Delta t = 10^{-2}$ for 100 RCs of each topology and sizes $D_r \in \{50, 100, 150, \ldots, 450\}$. RCs use hyperparameter values obtained as medians of optimization of 20 RCs of size $D_r^O = 100$ (top row) and $D_r^O = 300$ (bottom row). Regardless of $D_r^O$ used for optimization, for RCs that were trained for $10^4$ time steps $T_v$ increases with $D_r$ and saturates for sufficiently large reservoirs. This is the qualitative characteristics one would expect.

For the longer training of $10^5$ time steps and $D_r^O = 100$, $T_v$ is seen to have a maximum for RCs of this same size $D_r = D_r^O = 100$. This holds for all topologies except for the cycle. RCs are seemingly overfit during this longer training using $D_r^O = 100$. The cycle, on the other hand, is a topology whose optimum hyperparameter values are not as strongly dependent on the reservoir size as those for other topologies, since there are no rigid boundaries in the network, but in effect periodic boundary conditions apply. In accordance to this, the optimum hyperparameter values for the cycle RCs of sizes $D_r = 100$ and 300 were seen to be more similar than the rest of the RCs. Specifically, the regularization parameter obtains the same value $\mu = 10^{-4}$ for the cycle when using either $D_r^O = 100$ or 300 for optimization. Consequently, the cycle topology is not as sensitive to the size of RCs used in optimization.

The results for $D_r^O = 300$ and the longer training seem to corroborate the above interpretation. Since larger RCs require longer training, none of the connected reservoirs are overfit to the data and the expected increase and saturation of $T_v$ with increasing $D_r$ is obtained. RUNs are a special case. While they seem to be overfit to $D_r = 100$ when that is the size used for optimization, also RCs using hyperparameter values from optimizing RCs of $D_r^O = 300$ show maximum $T_v$ for $D_r < 300$. As overfitting can be to some extent tackled with regularization, we increased regularization from $\mu = 10^{-5}$ to $\mu = 10^{-4}$ in Eq. (3) for $D_r^O = 100$ and 300. The maxima are, indeed, seen to be removed by stronger regularization, which would corroborate that they are due to overfitting.

In summary, the maximum training time after which RCs become too specifically optimized for a certain reservoir size varies with the reservoir topology. If RCs of certain size are to be used in forecasting, then the good news is that RCs can be trained for this particular size within realistic times.

![Fig. 5](image-url) Medians of valid times $T_v$ vs reservoir size $D_r$. Left column: $T_{\text{train}} = 10000$ steps of size $\Delta t = 10^{-2}$. Right column: $T_{\text{train}} = 100000$ steps. Top row: Optimization of hyperparameters for RCs of size $D_r^O = 100$. Bottom row: Optimization of hyperparameters for RCs of size $D_r^O = 300$. RUN reg-4 is RUN with $\mu = 10^{-4}$. 
FIG. 6. Three largest Lyapunov exponents measured for 20 optimized RCs trained for 100 steps. Reservoirs from top down are: ER, \( k = 1 \) with cycle, and the unconnected network. The in-degree of ER-topo varies between \( k \in [1, 5] \) with mean 3.05. The Lorenz-system exponent values \( \lambda_1 = 0.91, \lambda_2 = 0.00, \) and \( \lambda_3 = -14.6 \) are shown by vertical red lines. Lyapunov exponents were calculated over a trajectory of 5000 time steps.

C. Network topologies and Lyapunov exponents

We compute Lyapunov exponents of RCs as described in \(^6\). The whole spectrum of these exponents \( \lambda_i \), where \( i \in [1, 2, \ldots, D_r] \), was calculated for 20 optimized RCs of different topologies, each of size \( D_r = 100 \). The algorithm gives the exponents in descending order. We report the three largest and most important ones as they correspond to the exponents of the true dynamical system that reservoirs try to mimick. The largest exponent \( \lambda_1 \) characterizes the dynamical system most straightforwardly: \( \lambda_1 = 0 \) and \( \lambda_1 < 0 \) indicate energy-conserving and dissipative dynamics, respectively, and \( \lambda_1 > 0 \) may indicate chaotic dynamics. In Fig. 6 we show distributions of the values of \( \lambda_1, \lambda_2, \) and \( \lambda_3 \) for three RC topologies that gave the most precise estimates: ER, reservoir with a cycle included (\( k = 1 \)), and the unconnected ensemble of nodes (RUN). We experimented with trajectories of different lengths and found that 5000 points was sufficient for a reasonable estimate of these Lyapunov exponents.

Lyapunov exponents \( \lambda_1 \) and \( \lambda_3 \) are seen to be estimated with the highest precision by RUNs. For \( \lambda_2 \), all topologies give fairly good estimates. This means that the climate of the Lorenz system is best reproduced by RUNs. The estimate of the most important exponent \( \lambda_1 \) obtained with RUNs is clearly better than estimates obtained with other RCs. The difference to RUNs’ advantage is even clearer for \( \lambda_3 \). Distributions of the values are very narrow around the correct values. This is related to the small number of degrees of freedom of RUNs compared to those of connected topologies: the far simpler unconnected system can be learned and described by fewer variables. Indeed, for each RUN, all \( \lambda_i \), where \( i \in [4, 5, 6, \ldots, 100] \), have a constant very large negative value (ranging from \(-100\) to \(-3500\)), whereas they have much larger values, typically almost of the order of \( \lambda_3 \), for the other topologies.

Summarizing, of the evaluated RCs the computationally most effective RC of unconnected nodes also reproduces the climate of the Lorenz system with clearly the best precision.

D. Forecasting in the presence of noise

The reservoir of unconnected nodes (RUN) was seen to outperform the other reservoirs used for forecasting in our study. Nonlinear dynamics, particularly chaos, per se is a research field where noise is rarely included. Since our motivation here is to lay the basis for further development of forecasting signals measured in real-world systems exhibiting nonlinear dynamics, we want to assess how this simplest of reservoirs performs in the presence of noise.

Figure 7 shows medians of \( \varepsilon \) and \( T_v \) over 50 forecast procedures vs the magnitude of the added white Gaussian noise for ER reservoirs and RUNs. The noise was added to the original normalized training data. Thus the noise is observational and inherent to the measurement. The measured \( \varepsilon \) is seen to increase and \( T_v \) to decrease with increasing noise for both RCs, as expected.

In the absence of noise, \( \varepsilon \) is slightly smaller and \( T_v \) slightly larger for the RUNs, in keeping with the results presented above. This, however, changes in the presence of noise. Even noise of the very small magnitude \( \mathcal{N}(0, 10^{-6}) \) is
sufficient to reverse the performance of the two RCs. Noise deteriorates the performance of RUNs slightly more strongly than that of RCs with an ER network. This outcome makes sense in that adding noise to the data considerably increases the number of degrees of freedom in the dynamical system to be learned by a RC. Since a reservoir of linked nodes has significantly more degrees of freedom than one with the same number of unconnected nodes, it can be expected to learn to replicate the noisy data better.

Decreasing the leakage rate to $\beta = 0.05$ improves the performance of RCs of both topologies in the presence of noise, since this makes the reservoir’s present state more strongly dependent on its previous state - see Eq. (5) - thus increasing correlation within the reservoir and, in effect, increasing the signal-to-noise ratio. Consequently, when the data is obtained by measurement of a system of nonlinear dynamics, such as brain, it is important that RC be optimized, when possible, in the presence of realistic noise, not for idealized model data.

**FIG. 7.** $\varepsilon$ and $T_v$ obtained from running 50 RCs of ER ($k = 2$) and unconnected networks (RUN) of size $D_r = 100$ in the presence of white Gaussian noise $N(0, n)$ added to the data (observational noise). Medians of $\varepsilon$ (top row) and $T_v$ (bottom row) are plotted vs the standard deviation of noise $\sqrt{n} \in \{0, 0.001, 0.01, 0.1\}$. The small noise magnitudes correspond approximately to signal-to-noise ratios 54, 34, and 14 dB. Hyperparameter values for RCs are at their optimal values. Left column: leakage rate $\beta$ at optimized values. Right column: leakage rate $\beta = 0.05$.

### E. Forecasting Lorenz and coupled Wilson-Cowan systems in different regimes

Since our motivation is to determine a workable procedure for optimizing computationally effective RCs for forecasting a general nonlinear system, we implement RC forecasting of the Lorenz system and a system of coupled Wilson-Cowan (cW-C) models (see Sec. III B) in different regimes. cW-C has been used to model neuronal dynamics. It exhibits characteristics that are very different from those of the Lorenz system and thus suits well for evaluating how generally applicable our findings are to forecasting dynamical systems of varied characteristics.

We optimize 20 ER reservoirs and RUNs that we use in forecasting data from the Lorenz systems in chaotic ($r = 28$), intermittent chaos ($r = 100$), and periodic ($r = 147$) regimes, and cW-C systems in aperiodic (i.e. chaotic) ($\alpha = 1.3$), quasiperiodic ($\alpha = 1.9$), and periodic ($\alpha = 4$) regimes. To make regularization for the two systems comparable, they are allowed the same numbers, $10^4$ and $10^5$, of training steps. The step lengths are $\Delta t = 10^{-2}$ and $10^{-1}$ giving short and long training times $T_{\text{train}} = 100$ and 1000 for the Lorenz system and $T_{\text{train}} = 1000$ and 10000 for the cW-C system. The longer times in connection with the cW-C system reflect its longer characteristic times.

Medians of the measured figures of merit are shown in Table IV. It is seen that forecasting the Lorenz system shows consistent results: for both $T_{\text{train}}$, the valid times $T_v$ increase when going from chaotic through intermittent chaos to periodic systems. For $T_{\text{train}} = 100$, the error $\varepsilon$ differs for systems in different regimes. However, when we apply the longer $T_{\text{train}} = 1000$, $\varepsilon$ decreases and reaches a minimum value of $\varepsilon \approx 0.02$ for the Lorenz systems in all regimes. This is the limit of precision we can achieve by training this particular set of RCs.

The outcome of forecasting the cW-C system appears to be less consistent when $T_{\text{train}} = 1000$ is applied. The smallest $T_v$ is obtained for systems in the aperiodic regime, as expected. However, $T_v$ measured for systems in the periodic regime is smaller than for systems in the quasiperiodic regime despite the measured $\varepsilon$ for these regimes being of the same order. (The only exception occurs for ER reservoirs and $T_{\text{train}} = 10000$.) Applying the longer
| System | RC   | $\epsilon^s$ | $T_{sv}^s$ | $\epsilon^l$ | $T_{lv}^l$ |
|--------|------|--------------|-------------|--------------|------------|
| Lorenz: |      |              |             |              |            |
| $r = 28$ | ER   | 0.020        | 5.29        | 0.017        | 6.03       |
| $r = 100$ | ER   | 0.031        | 12.08       | 0.024        | 14.55      |
| $r = 147$ | ER   | 0.060        | 57.47       | 0.020        | 82.26      |
| $r = 28$ | RUN  | 0.020        | 5.97        | 0.016        | 6.77       |
| $r = 100$ | RUN  | 0.032        | 12.18       | 0.023        | 15.71      |
| $r = 147$ | RUN  | 0.058        | 66.56       | 0.022        | 88.20      |
| Wilson-Cowans: |      |              |             |              |            |
| $\alpha = 1.3$ | ER   | 0.015        | 11.55       | 0.014        | 14.75      |
| $\alpha = 1.9$ | ER   | 0.007        | 45.36       | 0.006        | 36.56      |
| $\alpha = 4$ | ER   | 0.007        | 27.42       | 0.007        | 40.47      |
| $\alpha = 1.3$ | RUN  | 0.010        | 22.97       | 0.008        | 25.33      |
| $\alpha = 1.9$ | RUN  | 0.006        | 71.38       | 0.005        | 75.04      |
| $\alpha = 4$ | RUN  | 0.005        | 34.74       | 0.005        | 51.23      |

TABLE IV. Median values of figures of merit. Short training: $\epsilon^s$ and $T_{sv}^s$. Long training: $\epsilon^l$ and $T_{lv}^l$. Optimized RCs with reservoirs of Erdős-Rényi (ER) topology and unconnected nodes (RUN) were used for Lorenz and coupled Wilson-Cowan systems.

$T_{train} = 10000$ again decreases the error and is long enough to bring it to its minimum value of $\epsilon \approx 0.005$ for these two regimes. However, $\epsilon$ for the aperiodic regime does not reach this minimum even for longer training, reflecting the spiky dynamics of the coupled Wilson-Cowan system.

Similarly to the chaotic Lorenz system, RUNs perform forecasting slightly better than ERs. Hence, we conclude that this surprising result generalizes well to different noiseless systems.

Why does forecasting the cW-C system in the periodic regime seem to be so inconsistent? To a human, predicting values for a system in a periodic regime is far easier than in any of the other regimes studied here. Fig. 8 shows sample signals from the dynamical systems and the forecasted signals from RUNs. In Fig. 8 (b) we see that after first following the cW-C signal quite accurately the RC output flies off and follows a different oscillatory pattern at different output values. It is evident that RC effectively constructs a system of nonlinear dynamics where there is an unstable limit cycle by which the correct output is generated but after a while the trajectory veers off from this limit cycle and ends up following the stable limit cycle in its vicinity. Not only stable limit cycles were found for the RCs forecasting periodic cW-C system. RC dynamics also exhibited stable spirals and stable fixed points in the vicinity of the unstable limit cycle.

In Fig. 8 are shown the trajectories corresponding to the signals in Fig. 8 excepting Fig. 8 (b) that for clarity shows a RC forecast more successful than the one shown in Fig. 8 (b). As seen in Fig. 8 (b), trajectories of the periodic W-C system follow a very fine path. Due to this the RC is trained to just follow this fine trajectory. In training for this system less information is received on the phase space than in training for the system where trajectories cover a broader region...
FIG. 9. Sample trajectories of the Lorenz and coupled Wilson-Cowan systems (blue) and their RC (RUN) forecast (orange). (a) Lorenz system in the periodic regime. (b) cW-C in the periodic regime. (c) Lorenz in the chaotic regime. (d) cW-C in the aperiodic regime.

in the phase space. For an original and a learned trajectory in the aperiodic regime, see Fig. 9(d). Accordingly, RCs forecasting the cW-C systems in quasiperiodic and aperiodic regimes learn to reproduce more general features of the trajectories, that is, the climate. For a periodic system to work properly after one period the system has to return to the point where it started. Even a slight deviation from this can shift the trained reservoir to an "uncharted" territory, where it performs dynamics that is not inherent to the periodic system it was trained with.

The periodic regime is the most sensitive to training time also for Lorenz systems, as seen by the large $\epsilon = 0.06$ when shorter training is applied. The same characteristics as for the periodic cW-C system also apply for the periodic Lorenz system, see Figs. 8(a) and 9(a). The reason why RCs trained to follow this system do not lose track of it as easily as RCs following a periodic cW-C system lies in the fact that trajectories even in the periodic Lorenz system cover a reasonably wide region, see Fig. 9(a). Hence, RC receives information about the dynamics in a wider region in the phase space than just about a narrow trajectory as in the periodic cW-C system. Consequently, a RC is able to apply close to correct dynamics even after a slight deviation from the exact trajectory at a given time and so remain in its vicinity.

VII. SUMMARY

We studied forecasting by reservoir computers (RC) of two dynamical systems of highly different characteristics, the Lorenz and a coupled Cowan-Wilson system in different dynamical regimes. Our motivation was, on one hand, to gain understanding on the most prominent factors governing the RC performance in forecasting systems of nonlinear dynamics and, on the other hand, to determine judicious guidelines for finding optimal RCs for this purpose. Some novel understanding was gained. Optimization consistently led to reservoirs of low connectivity. This surprising outcome of the Bayesian hyperparameter optimization was verified by forecasting simulated dynamical systems. By applying a consistent method and sufficient statistics we also showed that the topology of reservoirs of low connectivity has no significance in the forecasting performance.

The most surprising outcome of the present study is the performance of a reservoir consisting of unconnected nodes (RUN). These reservoirs consistently performed best in forecasting noiseless systems of nonlinear dynamics. An unconnected ensemble of nodes with their individual values can be regarded as analogous to base functions by which a signal is represented. With finite training time the solution of this ensemble was seen to converge faster toward the correct representation than an ensemble of linked nodes. In other words, the extra degrees of freedom brought in by the links slows down convergence so within realistic training times the forecast remained farther from the correct representation when links were present in the reservoir.

Measurement noise was seen to change the setting in RC forecasting. In the presence of even a very weak noise RUN no longer performed better than reservoirs of low connectivity. Its performance turned out to be somewhat weaker for the noise levels used in the present study. At very large noise levels no differences in the performance of these RCs are seen. At sufficiently high noise levels the information content of the data diminishes such that differences in learning
Based on our findings, a workable optimization procedure for RCs to be used in forecasting can be outlined as follows:

Hyperparameters are determined by Bayesian optimization. If forecasting is performed for a theoretical simulated system or, more generally, for a virtually noiseless system, the best choice as the reservoir is an ensemble of unconnected nodes. In the presence of noise a reservoir of any topology having low connectivity can be used. If connectivity is included in this Bayesian optimization, then one chooses the resulting optimal value for the average input connectivity $k$. Sufficient statistics is needed. In the present study these parameters were determined as medians of optimal values obtained for 20 realizations of the selected reservoir topology. For $k$ it is preferable to use the most probable value, instead of the median value. At sufficiently large noise levels the computationally most effective reservoirs of unconnected nodes ($k = 0$) become again the best choice and should be used. It is fair to state that in noisy systems the performance of RC forecasting in its basic form is moderate. For such real-world systems improvements in the method are called for.

If RCs will be optimized and trained for each case, for example at the beginning of a measurement over a time interval, based on which forecasts are made, then these procedures should be done for one RC and use that in the forecasting. However, a very common usage is that a reasonable RC should be found for varying situations (data) that hopefully have enough common characteristics for forecasting to succeed. In this case one should train and use a sufficiently large reservoir. A few test runs with reservoirs of different sizes should be made and for computational efficiency choose the minimum size that gives sufficiently good figures of merit, in the present context the error and the valid time. In this latter case also overfitting should be avoided. If indications of this are seen in the test runs, regularization should be strengthened.

Regarding optimization and evaluation of RCs, the very general main finding is that they should be made using sufficient statistics (number of RC realizations). Most of the earlier findings in RC forecasting were based on very few, or even single, selected RCs, which gave misleading outcomes. The fundamental reason for this is that distributions of optimal hyper parameter values are very broad, as shown in the present study.

Our findings regarding optimal reservoir computers and the pertinent workable optimization procedure were similar for the two systems of very different characteristics and time-series data. In conclusion, each system requires understanding of its fundamental characteristics in order to use forecasting judiciously, but the steps outlined here for arriving at optimal RCs to use in forecasting seem to be common for different systems. The present study serves as basis for our long-term effort to develop a reservoir computing based method for real-time prediction of neural data.

ACKNOWLEDGEMENT

We acknowledge the computational resources provided by the Aalto Science-IT project.

1. H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis* (The University Press, Cambridge, UK, 2004).
2. R. Ilmoniemi and J. Sarvas, *Brain Signals* (The MIT Press, Cambridge, Massachusetts, 2019).
3. H. Jaeger, “The “echo state” approach to analysing and training recurrent neural networks-with an erratum note,” German National Research Center for Information Technology GMD Technical Report 148, 13 (2001).
4. W. Maass, T. Natschlager, and H. Markram, “Real-time computing without stable states: A new framework for neural computation based on perturbations,” Neural Computation 14, 2531–2560 (2002).
5. M. Lukoševičius and H. Jaeger, “Reservoir computing approaches to recurrent neural network training,” Comput. Sci. Rev. 3, 127 – 149 (2009).
6. J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, “Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data,” Chaos 27, 121102 (2017).
7. 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,” Phys. Rev. Lett. 120, 024102 (2018).
8. P. Antonik, M. Gulina, J. Pauwels, and S. Massar, “Using a reservoir computer to learn chaotic attractors, with applications to chaos synchronizion and cryptography,” Phys. Rev. E 98, 012215 (2018).
9. J. Pathak, A. Wikner, R. Fussell, S. Chandra, B. R. Hunt, M. Girvan, and E. Ott, “Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model,” Chaos 28, 041101 (2018).
10. D. J. Gauthier, E. Bolli, A. Griffith, and W. A. S. Barbosa, “Next generation reservoir computing,” Nature Communications 12, 5564 (2021).
11. M. Dale, S. O’Keeffe, A. Sebald, S. Stepney, and M. A. Trefzer, “Reservoir computing quality: connectivity and topology,” Natural Computing 20, 205–216 (2021).
12. H. Jaeger and H. Haas, “Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication,” Science 304, 78–80 (2012).
13. N. D. Haynes, M. C. Soriano, D. P. Rosin, I. Fischer, and D. J. Gauthier, “Reservoir computing with a single time-delay autonomous boolean node,” Phys. Rev. E 91, 020801(R) (2015).
14. A. Haluszczynski and C. Räth, “Good and bad predictions: Assessing and improving the replication of chaotic attractors by means of reservoir computing,” Chaos 29, 103143 (2019).
15. J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, “Information processing capacity of dynamical systems,” Sci. Rep. 2, 514 (2012).
16 Z. Lu, B. R. Hunt, and E. Ott, “Attractor reconstruction by machine learning,” Chaos 28, 061104 (2018).
17 L. M. Smith, J. Z. Kim, Z. Lu, and D. S. Bassett, “Learning continuous chaotic attractors with a reservoir computer,” Chaos 32, 011101 (2022).
18 E. Bollt, “On explaining the surprising success of reservoir computing forecaster of chaos? the universal machine learning dynamical system with contrast to var and dmd,” Chaos 31, 013108 (2021).
19 A. Griffith, A. Pomerance, and D. J. Gauthier, “Forecasting chaotic systems with very low connectivity reservoir computers,” Chaos 29, 123108 (2019).
20 Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, “Reservoir observers: Model-free inference of unmeasured variables in chaotic systems,” Chaos 27, 041102 (2017).
21 E. N. Lorenz, “Deterministic nonperiodic flow,” J. Atmos. Sci. 20, 130–141 (1963).
22 Y. Maruyama, Y. Kakimoto, and O. Araki, “Analysis of chaotic oscillations induced in two coupled wilson-cowan models,” Biological Cybernetics 108, 355–363 (2014).