The optimal reservoir computer for nonlinear dynamics

Pauliina T. Kärkkäinen and Riku P. Linna*

Department of Computer Science, Aalto University, P.O. Box 15400, FI-00076 Aalto, Finland

(*Corresponding author: riku.linna@aalto.fi)

Analysis and prediction of real-world complex systems of nonlinear dynamics relies largely on surrogate models. Reservoir computers (RC) have proven useful in replicating the climate of chaotic dynamics. The quality of surrogate models based on RCs is crucially dependent on judiciously determined optimal implementation that involves selecting optimal reservoir topology and hyperparameters. By systematically applying Bayesian hyperparameter optimization and using ensembles of reservoirs of various topology we show that the topology of linked reservoirs has no significance in forecasting dynamics of the chaotic Lorenz system. By simulations we show that simple reservoirs of unconnected nodes outperform reservoirs of linked reservoirs as surrogate models for the Lorenz system in different regimes. We give a derivation for why reservoirs of unconnected nodes have the maximum entropy and hence are optimal. We conclude that the performance of an RC is based on mere functional transformation, not in its dynamical properties as has been generally presumed. Hence, RC could be improved by including information on dynamics more strongly in the model.

I. INTRODUCTION

Extracting a model of a physical system from experimental data is a ubiquitous challenge [1]. In nonlinear science surrogate models are used both to gain understanding on the system and predict its states. The simple machine-learning method reservoir computing [2–7] has been shown to be faster and more accurate than more conventional methods [8] for nonlinear systems.

Reservoir computers (RC) have been used successfully in learning the invariant ergodic properties of chaotic systems. This success in reproducing the correct climate ensures that the right kind of dynamics will be reproduced although the detailed time series generated by the reservoir computer (RC) may deviate appreciably from the one generated by the dynamical system under study. There is great demand of computationally effective surrogate models that could predict signals in real time with measurements on a nonlinear system, the brain perhaps being the most challenging [9]. RC is a natural candidate for real-time prediction of dynamical systems, since due to its simplicity it facilitates fast hardware implementations [10].

Since reservoir computing was introduced as a variant of neural network methods where the reservoir network remains fixed, much of the RC optimization has concentrated on finding optimal recurrent network topologies. The performance of networks of different topologies have been compared to that of the random Erdős–Rényi (ER) networks for reservoirs as large as 10000 nodes. No preferential topology was found for these large reservoirs [4]. Since exact theory is lacking, optimization has been based on reasonable assumptions like "in order to learn chaotic dynamics the reservoir must be at the edge of chaos/instability", which was recently shown not to hold in general [11]. Recurrence of the network and an activation function of nonlinear form have both been considered essential. Moreover, using a large reservoir was seen important for providing a sufficient number of degrees of freedom, or capacity, for learning rich dynamics. Jaeger et al. [5] pointed out that sparse interconnectivity within the reservoir of a recurring network could be beneficial in that it lets the reservoir decompose into many loosely coupled subsystems, establishing a richly structured reservoir of excitable dynamics. By simulations also this argumentation was claimed invalid [12]. Contrariwise and supportive of Jaeger’s intuition, judiciously selected low-connectivity reservoirs were found to perform as well as high-connectivity reservoirs in forecasting chaotic systems [13].

Fundamentally, in the quest of determining the optimal RC the reservoir entropy should be maximized to minimize the entropy in the predicted signal. In search of the optimal reservoir network, topological complexity of the reservoir has often been implicitly taken as a measure of reservoir entropy. Findings by Carroll [12] are in part supportive of this. Based on measuring the mean free path within reservoirs it was argued that under certain constraints increasing reservoir connectivity increases entropy.

Since the emphasis in research on nonlinear dynamics is on understanding the physical system under study, there should be a decent understanding on how to determine the optimal RC to be used as a surrogate model and a common ground so that findings are verifiable. Although there are some promising systematic approaches, see e.g. [3, 14–17], a comprehensive theoretical understanding of reservoir computing is still missing. Consequently, optimization needs to be done by experimentation [13]. 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. [18].

In this paper we show that an ensemble of unconnected nodes is the optimal reservoir for a RC. We emphasize that we initially set out to find the optimal RC topologies for predicting nonlinear systems of different characteristics expecting the optimal reservoir to be recursive and not necessarily sparse. The outcome is thus the result of unbiased evaluations of criteria for finding optimal RCs. To our knowledge, this the first systematic study to use reasonable statistics in order to determine the optimal surrogate RC model for nonlinear systems.

In the following section we describe the implementation and evaluation criteria of RC. Sec. [11] states the Lorenz systems. In Sec. [14] we present the results on the relevant aspects of RCs as surrogate models for nonlinear systems,
such that the given spectral radii

\[ \tilde{r} \]

is a transformation that removes unwanted symmetries in \( \tilde{r} \) in the combined system of the reservoir and Lorenz dynamics. The input matrix \( W_{\text{in}} \) and the topology of the reservoir via \( A \) are determined before training and remain fixed after this. Input connections are first established with a given probability, after which the strength for each existing connection is taken as a number chosen randomly from a standard normal distribution. The elements of \( A \) and \( W_{\text{in}} \) are scaled such that the given spectral radii \( \rho_i \) and \( \rho_{in} \) are obtained.

To exclude the initial transient, half of \( r(t) \) from the beginning is discarded. The output layer then transforms the remaining reservoir’s output \( r' \) to \( W_{\text{out}} \tilde{r}(t) \). The symmetry in the combined system of the reservoir and Lorenz dynamics may deteriorate prediction \( f \). Hence, \( \tilde{r}(t) = f_{\text{out}} r'(t) \) is a transformation that removes unwanted symmetries in the reservoir. We use the symmetry-breaking form of \( f_{\text{out}} \) that transforms the reservoir node values as \( \tilde{r}_i(t) = r_i(t) \) for \( i \leq N/2 \) and \( r_i(t)^2 \) for \( i > N/2 \).

In the next stage of training \( W_{\text{out}} \) is adjusted such that \( W_{\text{out}} \tilde{r}(t) \) 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

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

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

Forecasting is done by using the trained RC. At this stage \( B(t) = W_{\text{out}} \tilde{r}(t) \) in Eq. (1) and the output \( W_{\text{out}} \tilde{r}(t) \) is continuously fed back to the input so RC runs autonomously.

In the numerical form of Eq. (1) we use time step \( \Delta t = 10^{-2} \) and leakage parameter \( \beta = \gamma \Delta t \in [0, 1] \). It 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.

Hyperparameter values are found by Bayesian optimization. Optimal values are found for six hyperparameters: spectral radius \( \rho_r \), probability of connecting an element of \( W_{\text{in}} \) to RC \( \sigma \), spectral radius of \( W_{\text{in}} \rho_{in} \), leakage parameter \( \beta \), regression parameter \( \mu \), and in-degree \( k \).

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

\[
\varepsilon = \left\{ \frac{1}{P} \sum_{t=1}^{P} \varepsilon_t^2 \right\}^{1/2}, \tag{3}
\]

where \( \varepsilon_t^2 = \frac{\Delta t}{T_{\text{eval}}} \sum_{t=t_i}^{t_i+T_{\text{eval}}} |u(t) - W_{\text{out}} \tilde{r}(t)|^2 \). In the Lorenz system \( T_{\text{eval}} = 1/\lambda \), that is, one Lyapunov period, see Section III. Before each start time \( t_i \), the reservoir is run with input for the time from \( t_i - \xi \) to \( t_i \) in order to synchronize the state of the reservoir to the state of the system. We use \( \xi = 10 \) and \( P = 50 \). In a chaotic system the \( P \) start times define 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 evaluating RC performance.

A figure of merit more directly related to signal prediction is the valid time \( T_v \), defined as the elapsed time before the normalized error \( E(t) = ||u(t) - W_{\text{out}} \tilde{r}(t)||/||u(t)||^2 \) exceeds some value \( f \in [0, 1] \). We use \( f = 0.4 \). \( T_v \) is averaged over 20 start times. Here \( || \cdot || \) denotes the \( L_2 \)-norm. The time \( T_{\text{eval}} \) over which \( \varepsilon \) in Equation (3) is computed is much shorter than the measured \( T_v \), so despite the likeness of the involved error term, \( T_v \) is more strongly related to the characteristics of the dynamical system under study, whereas \( \varepsilon \) is a direct measure of the precision of the method used for forecasting, as seen in Section III. Reservoir computers’ capacity of reproducing the ergodic properties of the Lorenz system in the chaotic regime is evaluated by measurements of Lyapunov exponents and the fractal dimension of the attractor, Sec. IV C.

FIG. 1. Schematic depiction of RC in the training phase. In the prediction phase the output of the reservoir is fed back to the input and the reservoir runs autonomously.
Unless noted otherwise, this system is used in the chaotic regime for the Lorenz system in intermittently chaotic (periodic regimes) and to real-world noisy dynamics. The inverse of the largest positive Lyapunov exponent of the standard chaotic Lorenz system, \( \lambda = 0.9056 \), is used as time reference. By adjusting \( r \), we also briefly evaluate RCs as surrogate models for the Lorenz system in intermittently chaotic (\( r = 100 \)) and periodic regimes (\( r = 150 \)).

### III. THE LORENZ SYSTEM

We use the Lorenz system [19] for optimization and evaluation of RCs. This dynamical system is used to generate data \( \mathbf{u}(t) \) extending over time \( t \in [0, T] \). The three-dimensional dynamical system is defined as

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

(4)

Unless noted otherwise, this system is used in the chaotic regime with the parameter values \( \sigma = 10 \), \( b = 8/3 \), and \( r = 28 \), originally used by Lorenz. This is the most generally used test bench for RCs. For deterministic systems observed over a finite time interval, chaotic dynamics has the closest resemblance to real-world noisy dynamics. The inverse of the largest positive Lyapunov exponent of the standard chaotic Lorenz system, \( \lambda = 0.9056 \), is used as time reference. By adjusting \( r \), we also briefly evaluate RCs as surrogate models for the Lorenz system in intermittently chaotic (\( r = 100 \)) and periodic regimes (\( r = 150 \)).

### IV. RESULTS

#### A. Reservoir topology and data representation

The performance of an RC can be improved by judiciously normalizing the data \( \mathbf{u}(t) \), as shown later. We first normalized by the variance, coined here error normalization (EN) as it results in normalizing the error of the predicted signal. The normalized quantity for each component \( i \in [1, D] \), where \( D \) is the dimension of the system, is obtained as \( \hat{u}_i(t) = [u_i(t) - \bar{u}_i]/\text{Var}(u_i) \), where \( \bar{u}_i \) is the mean and \( \text{Var}(u_i) \) is the variance of component \( u_i \) computed over the training time.

First, we used the same ranges for hyperparameter optimizations as Griffith et al. [13], see the first row, labeled R, in Table I. Training time \( T_{\text{train}} = 100 \). To evaluate the optimization procedure consisting of 100 iterations we ran it for 20 times for each topology. 200 reservoirs of size \( D_i = 100 \) were generated using median parameter values from the optimization. Topologies cover the available types at low connectivity: a random Erdős-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 \( (k = 1) \).

Distributions of optimal hyperparameter values in Fig. 2 are seen to be wide. Hence, the requirement of sufficient statistics for determining optimal values is quite severe, which alone would explain why efforts based on simulations of a single or at most a couple of reservoirs for finding optimal topologies were not successful.

Fig. 3 shows distributions of averaged short time prediction errors \( \epsilon \), Eq. 3, for the different reservoir topologies. The vertical lines show the median values \( \text{Med}(\epsilon) \). For the first column hyperparameter values that in [13] were found optimal after one optimization were used for all 200 RC realizations of each topology. Our RC implementations give very similar results as in [13]. These results having different \( \text{Med}(\epsilon) \) for different topologies would suggest that reservoir topology matters. Most notably, the single-cycle topology gives the largest error. For the second column hyperparameter values out of 20 optimizations were used for 200 RCs of each topology. There are no discernible differences between different reservoir topologies. \( \text{Med}(\epsilon) \) are equal for all topologies. We conclude that low-connectivity reservoir topology does not affect the accuracy of RC.

Since optimal values in Fig. 2 are seen to lie close to the limits of the allowed parameter ranges, we applied larger ranges and concurrently investigated if data representation affects RC quality. EN, commonly used in gradient-descent based learning, does not preserve the relative magnitudes of \( u_i(t) \). However, preserving them would seem preferable. To test this we used data in different forms in optimizations of ER reservoirs: raw data (RD), \( \mathbf{u}(t) \), variance-normalized data (EN) \( \tilde{u}(t) \), and data normalized (DN) as \( \hat{u}_i(t) \in [0, 1]: \tilde{u}(t) = [u(t) - \min_t[u(t)]]/\max_t[u(t) - \min_t[u(t)]] \), where \( u(t)_{\min} \) and \( u(t)_{\max} \) are the minimum and maximum of all components \( u_i(t) \) and \( t \in [0, T_{\text{train}}] \).

In optimization of ER reservoirs, additional wider sets A and B of parameter ranges were used, see Table I. Set B includes \( \rho = 0 \) and, consequently, RUNs are allowed. Since

| \( R \) | \( A \) | \( B \) |
|---|---|---|
| \( \rho \) | \( \rho_{\text{m}} \) | \( \rho_{\text{m}} \) |
| \( \beta \) | \( \beta_{\text{m}} \) | \( \beta_{\text{m}} \) |
| \( \log \mu \) | \( \mu_{\text{m}} \) | \( \mu_{\text{m}} \) |
| \( k \) | \( k_{\text{m}} \) | \( k_{\text{m}} \) |

TABLE I. Hyperparameter optimization ranges for RCs with ER networks.

FIG. 2. Histograms of optimized parameter values 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 to be quite similar when optimization was run for different sets of 20 reservoirs.

FIG. 3 shows distributions of averaged short time prediction errors \( \epsilon \), Eq. 3, for the different reservoir topologies. The vertical lines show the median values \( \text{Med}(\epsilon) \). For the first column hyperparameter values that in [13] were found optimal after one optimization were used for all 200 RC realizations of each topology. Our RC implementations give very similar results as in [13]. These results having different \( \text{Med}(\epsilon) \) for different topologies would suggest that reservoir topology matters. Most notably, the single-cycle topology gives the largest error. For the second column hyperparameter values out of 20 optimizations were used for 200 RCs of each topology. There are no discernible differences between different reservoir topologies. \( \text{Med}(\epsilon) \) are equal for all topologies. We conclude that low-connectivity reservoir topology does not affect the accuracy of RC.

Since optimal values in Fig. 2 are seen to lie close to the limits of the allowed parameter ranges, we applied larger ranges and concurrently investigated if data representation affects RC quality. EN, commonly used in gradient-descent based learning, does not preserve the relative magnitudes of \( u_i(t) \). However, preserving them would seem preferable. To test this we used data in different forms in optimizations of ER reservoirs: raw data (RD), \( \mathbf{u}(t) \), variance-normalized data (EN) \( \tilde{u}(t) \), and data normalized (DN) as \( \hat{u}_i(t) \in [0, 1]: \tilde{u}(t) = [u(t) - \min_t[u(t)]]/\max_t[u(t) - \min_t[u(t)]] \), where \( u(t)_{\min} \) and \( u(t)_{\max} \) are the minimum and maximum of all components \( u_i(t) \) and \( t \in [0, T_{\text{train}}] \).

In optimization of ER reservoirs, additional wider sets A and B of parameter ranges were used, see Table I. Set B includes \( \rho = 0 \) and, consequently, RUNs are allowed. Since
FIG. 3. Distributions of $\varepsilon$ from 200 simulations, visualized as Gaussian kernel density estimations in log$_{10}$ $\varepsilon$ with bandwidth of 0.1 [20]. The first column: hyperparameters the same as in [13] and obtained from one optimization. The second column: hyperparameters from 20 runs of Bayesian optimization. Vertical lines show median values.

| Range | Data | $\rho$ | $k$ | $\rho_{in}$ | $\rho_{in}$ | $v$ | $T$ | $G_{T}$ |
|-------|------|-------|-----|-------------|-------------|-----|-----|--------|
| A     | RD   | 0.10  | 2.00| 0.10        | -4          | 0.02| 4.67|        |
| A     | EN   | 0.10  | 3.00| 0.30        | 0.09        | -5  | 0.05| 4.32   |
| A     | DVE  | 0.74  | 2.00| 0.77        | 1.28        | -5  | 0.02| 5.20   |
| B     | RD   | 0.36  | 2.00| 0.45        | 0.06        | -4  | 0.03| 4.45   |
| B     | EN   | 0.44  | 3.00| 0.77        | 0.29        | -4  | 0.08| 3.50   |
| B     | DVE  | 1.01  | 1.64| 0.64        | 1.31        | -5  | 0.02| 5.15   |

TABLE II. ER reservoirs. Optimal hyperparameter values and figures of merit, $\varepsilon$ and $T_r$, for PNN with different data transformations in 200 RCs (see text). Sets A and B of hyperparameter ranges (see Table I). The narrower ranges of A results in slightly better performance due to faster optimization.

wider ranges require longer optimization, optimization was run for 200 and 500 iterations for A and B, respectively. Optimal hyperparameter values, errors $\varepsilon$, and valid times $T_r$ obtained for 200 RCs using the data in the described forms are shown in Table II. EN is seen to result in the poorest performing RCs: $\varepsilon$ and $T_r$ are the largest and $T_r$ the shortest. DN improves the performance of RCs compared to using RD. 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. Since EN also scales the data but results in poorer RC performance than RD, preserving the relative magnitudes within data seems to be more important than scaling. Consequently, from this point on we normalize the data.

Within both ranges optimal reservoirs are sparse as seen from the combined values of $\rho$ and $k$. It is noteworthy that for the set of ranges B where RUN is allowed, optimization resulted in RUN ($\rho = 0$) in more than 20% of the cases for RD and in almost 40% of the cases for DN. We also made sure that the found optimal reservoirs of very low connectivity are not just optima within the chosen hyperparameter ranges by using the wider range of $k$ in $[1,10]$ and several ranges for $\rho$. These optimizations resulted in minimal $k$ values. Summarizing, optimization that allows high connectivity leads to very sparsely connected reservoirs, the optima of which are reservoirs of no connections.

B. Reservoir size

Here, we investigate the effect of reservoir size to RC performance. First, $T_r$ were measured for 100 RCs of the five different topologies of $D_r \in \{50, 100, 150, \ldots, 450\}$ using hyperparameters from optimizations of RCs of sizes $D_r^O = 100$ or 300. For $D_r^O = 100$, $T_r$ increased with $D_r$ to a clear maximum at $D_r = 100$ deteriorating with increasing $D_r > 100$ (not shown). For $D_r^O = 300$, $T_r$ increased with $D_r$ saturating for $D_r > 300$, see Fig. 4(a). We also tested two training times $T_{train} = 100$ and 1000. For the shorter $T_{train}$ RUN was seen to perform as well as or better than other RCs. Increasing $T_{train}$ improved the performance of RUNs more than other RCs. For $T_{train} = 1000$, RUN clearly outperformed other RCs, see Fig. 4(a).

We compared performances of ER and RUN in more detail. Optimal hyperparameter values were found out of 50 reservoirs for each $D_r \in \{50, 100, 200, 300, 500, 1000\}$. $T_r$ vs $D_r$ out of 1000 reservoirs of each $D_r$ are shown in Fig. 4(b). Increasing $T_{train}$ is seen to improve RUN more than ER. The difference between the performance of RUN and connected reservoirs is seen to diminish with increasing $D_r$. This is in keeping with our maximum entropy argument as explained in Section [IV-E].

C. Climate replication

To evaluate replication of ergodic properties of the chaotic Lorenz system by the different reservoirs we measured their correlation dimensions $d_i$ and Lyapunov exponents $\lambda_i$, $i \in [1,2,\ldots,D_r]$. We report results for the three best-performing
RCs: ER, RUN and the reservoir including a single cycle \((k = 1)\) (RIS) of sizes \(D_r = 100\) and 300.

We computed correlation dimension \(d_c\) that gives an estimate of the fractal dimension \(d_f\) of the signal using Grassberger-Procaccia algorithm [21][22]. The true value for the standard chaotic Lorenz system is \(d_f \approx 2.06\). The measured \(d_c\) for \(D_r = 100\) and 300, respectively, are ER: 2.02 and 2.04; RUN: 2.01 and 2.03; RIS: 2.00 and 1.98. All reservoirs give \(d_c\) with reasonable precision. Except for RIS, increasing \(D_r\) from 100 to 300 slightly improves the estimate.

Lyapunov exponents of RCs were computed as described in [6]. The whole spectrum of these exponents \(\lambda_i\), where \(i \in [1,2,\ldots,D_r]\), was computed for 20 optimized RCs of \(D_r = 100\) and 300. In Fig. [3] are shown distributions of the values of \(\lambda_i\), where \(i \in [1,2,3]\) for \(D_r = 300\) together with the true exponent values, \(\lambda_1 = 0.91, \lambda_2 = 0.00\), and \(\lambda_3 = -14.6\). Distributions for \(D_r = 100\) are qualitatively similar, differences being largest for ER, which is to be expected due to the highest level of randomness involved.

Median values with standard deviations are:

| RC | \(D_r\) | \(\lambda_1\) | \(\lambda_2\) | \(\lambda_3\) |
|----|--------|--------------|--------------|--------------|
| ER | 100    | 0.964 ± 0.035| 0.003 ± 0.025| -7.321 ± 5.129|
| ER | 300    | 0.965 ± 0.038| 0.013 ± 0.019| -2.618 ± 5.001|
| RUN| 100    | 0.933 ± 0.035| -0.021 ± 0.016| 14.480 ± 0.490|
| RUN| 300    | 0.919 ± 0.027| 0.012 ± 0.018| -14.270 ± 0.603|
| RIS| 100    | 0.947 ± 0.034| 0.009 ± 0.022| -9.713 ± 5.218|
| RIS| 300    | 0.978 ± 0.053| 0.013 ± 0.032| -6.498 ± 5.662|

RUN is seen to give the most accurate estimates. Standard deviations are far largest for \(\lambda_3\). For RUN these are almost an order of magnitude smaller than for ER and RIS.

An additional advantage of RUNs is that due to there being no randomly inserted connections, distributions of \(\lambda_i\) are very narrow around the correct values. Hence, in contrast to ERs, any realization of RUN gives a reasonably good estimate of the ergodic properties of the dynamical system. This also reflects in \(\lambda_i\), where \(i \geq 4\). The true dynamical system has dimension 3, so ideally \(\lambda_i = -\infty\) for \(i \geq 4\). 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 may have an even larger value than \(\lambda_1\) for the other topologies. So, unlike other reservoirs, RUN has very precisely the dimension of the true dynamical system.

This reflects directly in Kaplan-Yorke dimension \(d_{KY}\) that, like \(d_c\), is an estimate of the signal fractal dimension. \(d_{KY}\) is computationally more precise than \(d_c\) [23][24]. It is not an independent estimate as it is computed from the measured Lyapunov exponents as \(d_{KY} = j + \sum_{i=1}^{j} \frac{\lambda_i}{|\lambda_i|}\), where \(j\) is the largest integer for which the cumulative sum of the Lyapunov exponents is positive. The computed \(d_{KY}\) for \(D_r = 100\) and 300, respectively, are ER: 2.225 ± 0.242 and 2.412 ± 0.321, RUN: 2.063 ± 0.002 and 2.064 ± 0.003, and RIS: 2.169 ± 0.145 and 2.273 ± 0.301. Not only is \(d_{KY}\) for RUN exactly \(d_f\) of the Lorenz system, also the error of the estimate is almost two orders of magnitude smaller than for the other reservoirs.

In summary, RUN replicates the climate of the dynamical system with much higher precision than other RCs. In reservoirs with connections the Lyapunov exponents \(\lambda_i\), where \(i \geq 4\), can have large values, even larger than \(\lambda_3\). These large values increase Kaplan-Yorke dimension. In [13] by varying reservoir parameters of densely connected reservoirs \(\lambda_4\) was kept small so that it does not overlap with the Lyapunov exponent spectrum of the driving system. Here we find that this criterion is fulfilled completely by RUNs, which in keeping with RUNs being the maximum entropy reservoirs, see Section [IV.E].

D. The Lorenz system in different regimes

The figures of merit measured for 100 RCs of \(D_r = 100\) and 300 for Lorenz systems in chaotic \((r = 28)\), intermittently chaotic \((r = 100)\), and periodic \((r = 150)\) regimes are given in Table [III]. \(r = 150\) is within the period-doubling regime \(r \in (145,166)\). Intermittently chaotic dynamics turned out to require longer training, hence \(T_{train} = 1000\) was used. Due to distinctly different intermittent periodic and chaotic modes, \(\epsilon\) are seen to be large for \(r = 100\). RUN outperforms ER in all regimes, although differences are very small for \(r = 100\). For \(r = 150\) the observation time \(T_o = 109.99\). Unlike ER, RUN is able to predict the periodic signal throughout this time already for the smaller reservoir. For \(D_r = 300\), \(T_o = T_o\) also for ER. This suggests that RUN has larger entropy than ER, as shown in Sec. [IV.E] Fig. [6] shows samples of driving signals together with signals predicted by RUN and the corresponding trajectories in the phase space for the Lorenz systems in the three regimes. In keeping with findings in [23], RCs are seen to predict extremely well also in the periodic regime where dynamics is Hamiltonian.
TABLE III. Errors $\varepsilon$ and valid times $T_v$ measured for 100 ER and RUN RCs trained for $T_{run} = 1000$. The Lorenz system in chaotic ($r = 28$), intermittently chaotic ($r = 100$), and periodic ($r = 150$) regimes.

| Lorenz | $D_r = 100$ | $D_r = 300$ |
|--------|--------------|--------------|
| $r = 28$ | ER | 0.25 | 5.71 | 0.018 | 6.61 |
| $r = 100$ | ER | 0.123 | 7.31 | 0.089 | 7.47 |
| $r = 150$ | ER | 0.016 | 77.60 | 0.012 | 109.99 |
| $r = 28$ | RUN | 0.015 | 6.73 | 0.017 | 6.82 |
| $r = 100$ | RUN | 0.120 | 7.37 | 0.125 | 7.53 |
| $r = 150$ | RUN | 0.016 | 109.99 | 0.017 | 109.99 |

FIG. 6. Samples of the output $x$ of the Lorenz systems (blue), its RC (RUN) prediction (orange), and the corresponding trajectories. The standard chaotic regime, $r = 28$: (a) and (b). A regime of intermittent chaos, $r = 100$: (c) and (d). A periodic regime, $r = 150$: (e) and (f).

E. RUN is the maximum-entropy reservoir

According to the maximum-entropy principle the information should be obtained from the probability distribution that maximizes the entropy, subject to the constraints [26]. In the context of RCs, the entropy of the system consisting of the reservoir and the output matrix $W_{out}$ should then be maximized subject to the information, which constitutes samples $u(t_k)$ at times $t_k$, $k \in [1, n-1]$, before the present time $t_0$. This way the entropy of $u(t_k)$ will be minimized. As each node is connected to the output via $W_{out}$, the probability distribution whose entropy is to be maximized is determined by the reservoir. We form the predicted signal value as $u(t+\Delta t) = \sum c_k f(r_k(t))$, where $c_k$ are the elements of $W_{out}$. In RUN $r_k(t)$ are determined independently of $r_j(t-\Delta t)$, where $j \neq i$, by regressive fitting of $c_k$, whereas in networks with connections node $i$ may be connected to node $j$ and accordingly $r_j(t)$ depends directly on $r_i(t-\Delta t)$. The probability distribution of the reservoir can be written as $\sum p(r_k(t))$. The entropy of this distribution is at maximum when there are no connections between nodes, since connecting node $i$ to $j$ reduces direct dependence of $p(r_j(t))$ on $p(r_i(t-\Delta t))$. Viewing the reservoir entropy as formed by constituent entropies of each node, then connecting node $i$ to $j$ means that $p(r_j(t)) p(r_i(t-\Delta t)) > 0$ and the corresponding conditional entropy is smaller than the entropy $p(r_i(t))$ of the independent node $i$.

In RUN all nodes need to be connected to input, that is, each node receives one or more of the components of the driving signal. In connected reservoirs the fraction of nodes receiving input $i$ varies through optimization. We tested the applicability of our argumentation by removing a fraction of input connections and simultaneously replacing each input connection by one from a randomly chosen node. We then added $l^+ = 0, 25, 50, 100, 200$ random inter-node connections and measured $T_v$ averaged over 100 reservoirs. For $f_i = 80\%$, a clear maximum of $T_v \approx 5.15$ was obtained for $l^+ = 100$. For $f_i = 95\%$, the maximum $T_v \approx 5.3$ was obtained for $l^+ = 0$. For $l^+ \geq 50$, $T_v$ increased with $l^+$.

Our argumentation thus strictly holds only for $f_i = 100\%$. With decreasing $f_i$ and increasing inter-node connections the entropy of the reservoir becomes intractable due to recurrence. With 5\% replacement the reservoirs with $l^+ = 0$ perform best. So for $f_i = 95\%$ the least connected reservoir still has the largest entropy. For $f_i = 80\%$ that in the three-dimensional system, where probability of no connection per node is $(1 - p_{in})^3$ corresponds to input connectivity $p_{in} = 0.42$, this no longer holds. As seen in Table [I] $p_{in}$ of the optimized ERs are well above this value, so according to our estimate, of the optimized reservoirs RUN has the maximum entropy.

The decrease of the difference between the performance of RUN and connected reservoirs with increasing $D_r$, see Sec. [IV B], is in keeping with our maximum entropy argument. This difference decreases for the same reason as the performance of all reservoirs saturate for increasing $D_r$: the reservoir entropy becomes increasingly sufficient for signal inference.

F. RUN and linear regression models

Recently, next generation reservoir computing (NG-RC) [27] and domain-driven regularized regression (D2R2) [28] have been shown to outperform RCs with connected reservoirs. Neither of these methods has a reservoir. They are essentially linear regression models with nonlinear input features and use polynomial features [29]. NG-RC is a generalized version of D2R2. The two are equivalent when only the present state is used for polynomial features.

RUN with $\beta = 1$ can be viewed as D2R2 with $f_{out} \tanh(W_{in} u_i)$ replacing the polynomial features as the extraction function. The features are (squared) hyperbolic tan-
gents of randomly weighted sums of inputs. Qualitatively the main difference between D2R2 and RUN is that features in RUN include randomness. The optimal reservoir computer being analogical and under definite constraints similar with a linear regression model further corroborates that reservoir computing is based on functional transformation, not on the topology of the reservoir network.

V. SUMMARY AND DISCUSSION

Here, we set out to determine the optimal reservoir computer (RC) for forecasting nonlinear dynamical systems. One motivation was to set a common framework for studying nonlinear systems using RCs such that inconsistencies in results would not be due to RC implementations. We mostly used the standard chaotic Lorenz system as the test dynamical system. After systematic Bayesian optimization of hyperparameters and measurements of RC performance we showed that sparsely connected reservoirs outperform more densely connected ones. Among the sparsely connected reservoirs singly connected performed best. We then determined that all selected topologies of singly connected RCs performed exactly equally well. Hence, there exists no optimal reservoir topology. Further expanding the range of possible hyperparameter values, optimizations tended to end up in reservoirs with no connections.

These reservoirs of unconnected nodes (RUN) consistently performed best both in replicating the climate of the dynamical system and predicting the actual signals, which indicated that RUN would be the reservoir of maximum entropy. In keeping with this is also that the difference in the capacity of RUN and other reservoirs were found to grow smaller with increasing reservoir size, since the relative difference due to topology diminishes as the available entropy increases with the reservoir size for all reservoirs. We derived RUN to be the maximum-entropy reservoir under the condition that all reservoir nodes are connected to input and estimated this to hold for connectivities obtained for optimized Erdös-Rényi reservoirs.

All RCs proved to be valid surrogate models also for the Lorenz system in the intermittently chaotic and periodic regimes. This corroborates the study where it was shown that, contrary to the common expectation, reservoir computing can well replicate the dynamics of Hamiltonian systems. Our findings were found to hold also for the Wilson-Cowan system that is very different from the Lorenz system. As further support to RUN being the maximum-entropy reservoir we pointed to the close analogy, and even equality under certain conditions, of RUN and recently introduced linear regression models that were found to outperform connected RCs.

In conclusion, RUNs perform clearly best in reservoir computers used as surrogate models for nonlinear systems. RUN has no recurrence and can be viewed as merely a functional transformation, instead of a dynamical system with some topology. The fact that such a reservoir outperforms ones with recurrence sets reservoir computers in new perspective. It suggests that judiciously incorporating knowledge about the dynamics of the system in the reservoir computer is a logical step for improving its capacity. A first attempt at this was made in [29]. Judicious evaluation of such a procedure must be made using optimal reservoirs, since according to our studies including dynamical information may improve the performance of a reasonable RC but not an optimal RC.

The optimum reservoir resulting in an unconnected reservoir seems logical in the sense that the reservoir is a fixed random configuration. Since RC learns features of the driving dynamical system only during training and after this the fixed system is merely fitted to the dynamical system, there is little reason to assume that there was some topology that would make this fixed reservoir optimal for any dynamical system driving it. This understanding is also supported by the fact that mere linear regression models with nonlinear input features, shown to be analogous to RUNs, constitute as good surrogate models for nonlinear dynamics as RCs.

Finding the simplest possible reservoir to be optimal does not change the fact that reservoir computing outperforms conventional methods used in nonlinear science in both accuracy and efficiency. The real potential for RUNs lies in applications where complex nonlinear systems must be analyzed in real time. Especially valuable would be fast structural analysis of coupled elements of nonlinear dynamics. For example, in neuronal dynamics being able to fast determine connections between groups of neurons, or cortical areas in brain measurements, would have a huge impact on understanding such systems. It is in principal possible to determine connections in such coupled nonlinear systems using RCs. Developing reservoir computing toward usage in determining the structure of a network of coupled nonlinear elements would have a significant impact on real-time analysis and prediction of real-world nonlinear systems. Having convincingly determined that the optimal reservoir is the simplest and so computationally the most effective one, it seems timely to direct more attention to such challenging problems.

ACKNOWLEDGEMENT

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

---

[1] S. L. Brunton, J. L. Proctor, and J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proc. Natl Acad. Sci. USA 113, 3932 (2016).

[2] 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).
[3] 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 (2002).

[4] M. Lukoševičius and H. Jaeger, Reservoir computing approaches to recurrent neural network training, Comput. Sci. Rev. 3, 127 (2009).

[5] H. Jaeger and H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science 304, 78 (2012).

[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] H. Kantz and T. Schreiber, Nonlinear Time Series Analysis (The University Press, Cambridge, UK, 2004).

[9] R. Ilmoniemi and J. Sarvas, Brain Signals (The MIT Press, Cambridge, Massachusetts, 2019).

[10] 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 115, 100 (2019).

[11] T. L. Carroll, Do reservoir computers work best at the edge of chaos?, Chaos 30, 121109 (2020).

[12] T. L. Carroll, Low dimensional manifolds in reservoir computers, Chaos 31, 043113 (2021).

[13] A. Griffith, A. Pomerance, and D. J. Gauthier, Forecasting chaotic systems with very low connectivity reservoir computers, Chaos 29, 123108 (2019).

[14] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, Information processing capacity of dynamical systems, Sci. Rep. 2, 514 (2012).

[15] Z. Lu, B. R. Hunt, and E. Ott, Attractor reconstruction by machine learning, Chaos 28, 061104 (2018).

[16] L. M. Smith, J. Z. Kim, Z. Lu, and D. S. Bassett, Learning continuous chaotic attractors with a reservoir computer, Chaos 32, 011101 (2022).

[17] E. Bollt, On explaining the surprising success of reservoir computing forecasters of chaos? the universal machine learning dynamical system with contrast to var and dmd, Chaos 31, 013108 (2021).

[18] 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).

[19] E. N. Lorenz, Deterministic nonperiodic flow, J. Atmos. Sci. 20, 130 (1963).

[20] D. W. Scott, Multivariate density estimation : theory, practice, and visualization, second edition. ed. (Wiley, Hoboken, New Jersey, 1992).

[21] P. Grassberger and I. Procaccia, Measuring the strangeness of strange attractors, Physica D 9, 189 (1983).

[22] P. Grassberger and I. Procaccia, Characterization of strange attractors, Physical Review Letters 50, 346 (1983).

[23] P. Frederickson, J. L. Kaplan, E. D. Yorke, and J. A. Yorke, The liapunov dimension of strange attractors, Journal of Differential Equations 49, 185 (1983).

[24] T. L. Carroll, Dimension of reservoir computers, Chaos 30, 013102 (2020).

[25] H. Zhang, H. Fan, L. Wang, and X. Wang, Learning hamiltonian dynamics with reservoir computing, Physical Review E 104, 024205 (2021).

[26] E. Jaynes, Information theory and statistical mechanics, Physical Review 60, 620 (1957).

[27] D. J. Gauthier, E. Bollt, A. Griffith, and W. A. S. Barbosa, Next generation reservoir computing, Nature Communications 12, 5564 (2021).

[28] R. Pyle, N. Jovanovic, D. Subramanian, K. V. Palem, and A. B. Patel, Domain-driven models yield better predictions at lower cost than reservoir computers in lorenz systems, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 379, 20200246 (2021).

[29] M. O. Franz and B. Schölkopf, A Unifying View of Wiener and Volterra Theory and Polynomial Kernel Regression, Neural Computation 18, 3097 (2006).

[30] 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).

[31] A. Banerjee, J. Pathak, R. Roy, J. G. Restrepo, and E. Ott, Using machine learning to assess short term causal dependence and infer network links, Chaos 29, 121104 (2019).