Predicting high-dimensional heterogeneous time series employing generalized local states

Sebastian Baur [1,2] and Christoph Räth [1]
[1] *Institut für Materialphysik im Weltraum, Deutsches Zentrum für Luft-und Raumfahrt, Münchner Straße 20, 82234 Weßling, Germany*
[2] *Fakultät für Physik, Ludwig-Maximilians-Universität München, Schellingstraße 4, 80779 Munich, Germany*

(Received 4 January 2021; accepted 12 April 2021; published 17 June 2021)

We generalize the concept of local states (LS) for the prediction of high-dimensional, potentially mixed chaotic systems. The construction of generalized local states (GLS) relies on defining distances between time series on the basis of their (non-)linear correlations. We demonstrate the prediction capabilities of our approach based on the reservoir computing (RC) paradigm using the Kuramoto-Sivashinsky (KS), the Lorenz-96 (L96), and a combination of both systems. In the mixed system a separation of the time series belonging to the two different systems is made possible with GLS. More importantly, prediction remains possible with GLS, where the LS approach must naturally fail. Applications for the prediction of very heterogeneous time series with GLSs are briefly outlined.

DOI: 10.1103/PhysRevResearch.3.023215

I. INTRODUCTION

Tremendous advances in predicting the short- and long-term behavior of complex systems have been made in recent years by applying machine learning [1–7]. Reservoir computing (RC) turned out to be a very promising machine learning approach as it combines outstanding performance with conceptual advantages, such as very fast and comparably transparent learning and possibly appealing hardware realizations of RC [8,9].

For high-dimensional systems, RC suffers like other machine learning methods from the "curse of dimensionality" meaning that the number of nodes of the network representing the reservoir has to be considerably larger than the dimensionality of the input data rendering the training unfeasible with a naive RC approach. With a parallel prediction scheme based on local states (LS) [10], however, the forecasting of high-dimensional chaotic spatiotemporal systems of arbitrarily large extent becomes possible [3,11].

The definition of LS relies on defining spatial local neighborhoods for each time series to be predicted. Thus, the knowledge of the position of the time series in space is a necessary prerequisite for defining LS.

The similarity of time series can be defined in a much more general way by deducing a distance measure and thus a local neighborhood from the correlations among the time series [12]. Those generalized similarities led to - among others - reasonable, fully data-driven taxonomy of the stock market [13] while crucial differences between the linear and the nonlinear correlation structure of the stock market, especially during crises, were reported later [14].

In this article, we employ this approach to define generalized local states (GLS) for the prediction of high-dimensional systems with which some of the shortcomings of the LS approach can be overcome.

II. SYSTEMS AND SIMULATION DETAILS

For modeling high-dimensional, spatiotemporal, chaotic systems, the Lorenz-96 (L96) and Kuramoto-Sivashinsky (KS) systems have become widely used in the RC community [3,15–17]. The L96 [18,19] system is defined as

$$\frac{dx_j}{dt} = (x_{j+1} - x_{j-2})x_{j-1} - x_j + F, \tag{1}$$

where $\boldsymbol{x}(t)$ is the D-dimensional state vector of the system and F is the forcing parameter. In this study, we set $D = 40$ and $F = 5$ resulting in a chaotic system for which we calculate a maximal Lyapunov exponent (MLE) of $\Lambda_{\max} = 0.45$ using Sprott's method of orbit separation (OS) [20]. We use the fourth-order Runge-Kutta method [21] for the simulation, with a time step of $\Delta t = 0.05$.

The second model, widely used to model a variety of weakly turbulent fluid systems, is the KS system [22,23]. Its PDE reads

$$\partial_t u + \partial_x^4 u + \partial_x^2 u + u\partial_x u = 0\,, \tag{2}$$

where the field $u(x,t)$ is defined on some domain size L. In this study, we use a domain size of $L = 22$ with periodic boundary conditions $u(x+L,t) = u(x,t)$ for all $0 \leqslant x \leqslant L$. For the numerical treatment, the equations are discretized on a grid of $D = 40$ points, the same size as the L96 system, and numerically integrated, with a time step of $\Delta t = 0.5$ using the fourth-order time-stepping method ETDRK4 [24]. Using OS we find a MLE of $\Lambda_{\max} = 0.049$.

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

III. RESERVOIR COMPUTING

At the center of RC is a network, in the following called the reservoir $\mathbf{A}$, created as a sparse random $D_r \times D_r$ network with average node degree κ. After network generation, all its random, uniformly distributed connection strengths are scaled to have a predetermined fixed spectral radius ρ. The D_{in} dimensional input $\boldsymbol{x}(t)$ interacts with the reservoir state $\boldsymbol{r}(t)$ through an input coupler, which in our case is a sparse $D_r \times D_{\text{in}}$ matrix $\mathbf{W}_{\text{in}}$. Following Ref. [1], $\mathbf{W}_{\text{in}}$ is created such that one element in each row is chosen uniformly between $[-\omega, \omega]$, where ω is the input coupler scaling parameter. All other elements of $\mathbf{W}_{\text{in}}$ are zero. The input data then connect with the reservoir state $\boldsymbol{r}(t)$ of the previous time step via activation function $\tanh(\cdot)$ to advance the reservoir state by one step in time,

$$\boldsymbol{r}(t + \Delta t) = \tanh\left[\mathbf{A}\boldsymbol{r}(t) + \mathbf{W}_{\text{in}}\boldsymbol{x}(t)\right]. \tag{3}$$

We choose a simple matrix as output coupler $\mathbf{W}_{\text{out}}$, whose elements are determined in the training phase via ridge regression,

$$\mathbf{W}_{\text{out}} = \min_{\mathbf{W}_{\text{out}}}\left[\|\mathbf{W}_{\text{out}}\tilde{\boldsymbol{r}}(t) - \boldsymbol{y}_{\text{T}}(t)\| + \beta\|\mathbf{W}_{\text{out}}\|\right], \tag{4}$$

where $\boldsymbol{y}_{\text{T}}(t)$ is the D_{out} dimensional target output and $\tilde{\boldsymbol{r}}$ is a nonlinear transformation of the reservoir state $\boldsymbol{r}$ here chosen to be $\tilde{\boldsymbol{r}} = [\boldsymbol{r}, \boldsymbol{r}^2]^T = [r_1, r_2, ..., r_{D_r}, r_1^2, r_2^2, ...r_{D_r}^2]^T$. This blowup of the reservoir states first introduced by Lu *et al.* [15] actually serves to break the symmetries in the reservoir equations [25].

To avoid that the arbitrary initial state of the reservoir influences the regression results, training only starts after a washout phase of 20,000 time steps.

Once trained, the output $\boldsymbol{y}(t)$ can be calculated from the reservoir states $\boldsymbol{r}(t)$ as $\mathbf{y}(t) = \mathbf{W}_{\text{out}}\tilde{\boldsymbol{r}}(t)$. When using RC for prediction, it can then be run autonomously by using the prediction of the previous time step $\boldsymbol{y}(t)$ as the input $\boldsymbol{x}_{\text{pred}}(t)$ to calculate the next predicted time step $\boldsymbol{y}(t + \Delta t)$. In this case one finds

$$\boldsymbol{r}(t + \Delta t) = \tanh\left[\mathbf{A}\boldsymbol{r}(t) + \mathbf{W}_{\text{in}}\boldsymbol{x}_{\text{pred}}(t)\right]. \tag{5}$$

IV. GLS

A. Generalizing LS

GLS is based on the LS approach proposed by Parlitz *et al.* [10] and used for RC prediction by Pathak *et al.* [3]. While nonlocal implementations of RC algorithms use just one network to process all input data, LS and GLS partition the input data into multiple subsets of smaller dimension, each with their own reservoir. The reservoirs are then trained on and predict these subsets only. This provides an effective workaround for the curse of dimensionality by essentially parallelizing the prediction of one high-dimensional dataset using many lower-dimensional subsets. These subsets are in the following called *neighborhoods*. Each neighborhood itself consists of a number of *core* variables and *neighbor* variables and is assigned its own reservoir. Each reservoir in question then uses only the variables of the input making up its neighborhood to predict its core variables as accurately as possible.

As such, the input time series for the reservoir assigned to the i-th neighborhood is not the full D_{in} dimensional input time series $\boldsymbol{x}$ anymore, but instead a slice unique to this neighborhood $\boldsymbol{x}^i$ of dimension $D_{\text{in}}^i \leqslant D_{\text{in}}$. Similarly, the trained output of the i-th neighborhood $\boldsymbol{y}^i$ is given by just its core variables and hence is an even smaller subset of the neighborhood of dimension $D_{\text{out}}^i \leqslant D_{\text{in}}^i$. These different neighborhoods are depicted in Fig. 1.

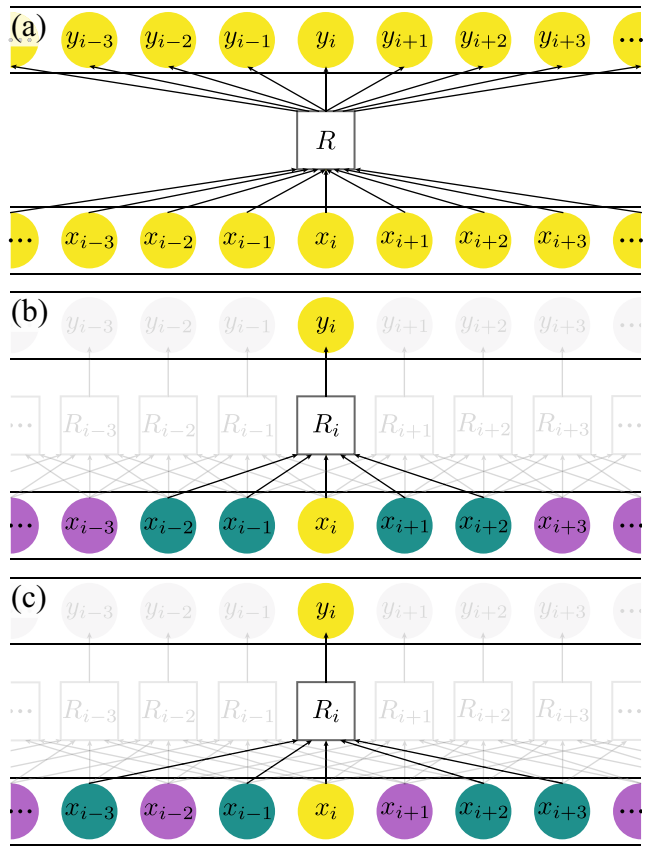

FIG. 1. Schematic depicting different neighborhoods and associated data flow. Yellow circles mark the core variables, green circles are the neighbors. and purple circles are all other variables of the full input. (a) The neighborhood of simple, nonlocal RC. All dimensions of the input vector $\boldsymbol{x}(t)$ are core dimensions of the reservoir R. No neighbors or other reservoirs exist. (b) LS RC. The highlighted neighborhood of the reservoir has one core variable x_i and four locally adjacent neighbors. These five variables are used as input for the reservoir R_i to predict the core variables future state y_i. Many more reservoirs exist, each with its own neighborhood. (c) GLS RC. As in (b), the neighborhood has one core variable and four neighbors, with the crucial difference being that the neighbors do not have to be locally adjacent to the core.

Between each prediction step, the neighborhoods need a new input which, as typically $D_{\text{out}}^i < D_{\text{in}}^i$, can only come from the other neighborhoods' prediction. Hence between each prediction step, a new input vector $\boldsymbol{x}_{\text{pred}}$ is formed by the combined core variable prediction of all neighborhoods. As a consequence, each variable of the original input must be in one and only one neighborhood as a core variable.

Pathak *et al.* [3] originally introduced their LS approach in the context of the KS system, a purely locally interacting system. As such, their neighborhoods were chosen to have

only spatially adjacent cores surrounded by contiguous buffer regions of neighbors [see Fig. 1(b)].

For systems where such a local interaction is not present, this scheme cannot be used. Nonetheless, the idea of locality in the sense of importance to a prediction is generalizable to something which we will call *similarity*. As the choice of neighborhoods is essentially arbitrary, this allows the creation of neighborhoods not by which variables are locally closest to the core variables, but by instead including the variables most *similar* to the cores as neighbors. Such a nonlocal GLS neighborhood is shown in Fig. 1(c).

The effectiveness of this procedure of course depends on an appropriate choice of neighborhoods, such that the information necessary to predict their core variables is present in their (nonlocal) neighbors.

B. Similarity Measures

Our first measure to estimate the similarity of the time series is the (linear) cross-correlation (CC) coefficient

$$C_{X_i,X_j} = \frac{\sum_{t=1}^{n}(x_{i,t}-\bar{x}_i)(x_{j,t}-\bar{x}_j)}{\sqrt{\sum_{t=1}^{n}(x_{i,t}-\bar{x}_i)^2}\sqrt{\sum_{t=1}^{n}(x_{j,t}-\bar{x}_j)^2}}, \quad (6)$$

where x_i is the i-th variable of the time series $\boldsymbol{x}$. To transform the CC into a similarity measure (SM) we take its absolute value

$$\mathrm{SM}_{\mathrm{CC}}(X_i, X_j) = \left|C_{X_i,X_j}\right| \in [0, 1], \quad (7)$$

and associate a larger value as more similar. As such, both a high correlation as well as a high anticorrelation correspond to a high similarity.

Considering that chaotic time series are by their very nature nonlinear, we use one more measure that captures these nonlinear relationships, the mutual information (MI). As we are working with long (10^5 time steps), well-behaved time series, we will implement the widely used binning method [14,26,27] as the estimator for the MI. Akin to Ref. [14], we find empirically that for our time series of length $T = 10^5$ a choice of $\lfloor\sqrt{T/4}\rfloor = 158$ bins of equal size works well. We normalize the MI as described by Strehl *et al.* [28] leading to our MI SM

$$\mathrm{SM}_{\mathrm{MI}}(X_i, X_j) = \frac{I(X_i, X_j)}{\sqrt{H(X_i)H(X_j)}} \in [0, 1], \quad (8)$$

where $I(X_i, X_j)$ is the MI while $H(X_i)$ and $H(X_j)$ are the Shannon entropies of X_i and X_j, respectively.

C. Neighborhood definitions

Once a SM has been chosen and calculated for all variables of the full input data, one needs to use it to create the neighborhoods. As the prediction output of each neighborhood is only given by its core variables, predicting these core variables as accurately as possible is most important. While a variety of choices is possible, we restrict ourselves to associate each neighborhood with exactly one core variable. In the LS case, the neighbors of the core are simply the variables spatially closest to that core. We will call this a spatial neighborhood (SN). In the GLS case, the exact analog is possible, where the neighbors of the core are the variables most similar to it as defined be the SM. We call the corresponding neighborhoods CC or MI neighborhoods, depending on the SM used.

Lastly, it should be emphasized that the GLS method is in principle independent of not only the specific SM used but also the chosen prediction method. Many other choices for the SM, e.g., the transfer entropy, or the network, e.g., LSTMs, are conceivable.

TABLE I. RC hyperparameters used throughout this article.

reservoir dimension	D_r	5000
average node degree	κ	3
spectral radius	ρ	0.5
input coupler scaling	ω	0.5
ridge regression parameter	β	10^{-6}
noise level	α	1%

V. EXPERIMENTS

A. Methods and parameters

To enable a fair comparison of the different neighborhood generation methods, the RC hyperparameters are optimized for the LS neighborhoods and then copied for the GLS neighborhoods without further adjustments. Furthermore, we added noise to all training data following Vlachas *et al.* [2]. Without this noise, we found the short-term prediction accuracy to often be higher, but at the cost of an increased rate of failed realizations and lower quality long-term predictions in general. The added noise proved decisive in minimizing variance between network realizations and reducing the number of failed realizations, especially for the L96 system. Heuristically, we found normally distributed noise with standard deviation (SD) $\sigma_{\mathrm{noise}} = 1\%\,\sigma_{\mathrm{data}}$, where σ_{data} is the SD of the training data, to be a sweet spot. The hyperparameters used for all RC are given in Table I.

Transient effects of the simulated data were discarded before any synchronization, training, or prediction took place. Similarly, each reservoir training and prediction is preceded by 2000 synchronization steps. All reservoirs were trained for a total of 10^5 time steps using the noisy training data.

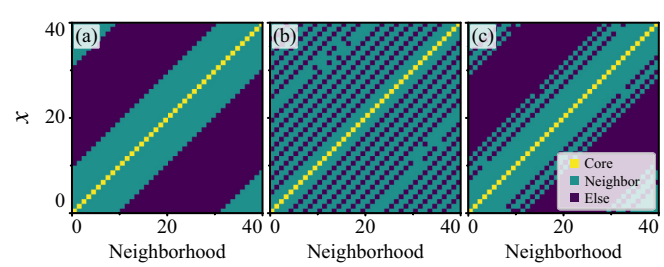

FIG. 2. Example neighborhoods for the L96 system. Each row depicts one variable of the time series, while each column represents a neighborhood. Each neighborhood is defined by the core variable (yellow) and its neighbors (green). (a) SN neighborhoods. The SN neighborhood consists only of a core and its 18 nearest neighbors. (b) CC neighborhoods. Each CC neighborhood includes 28 neighbors chosen with the CC SM. (c) MI neighborhoods with 18 neighbors.

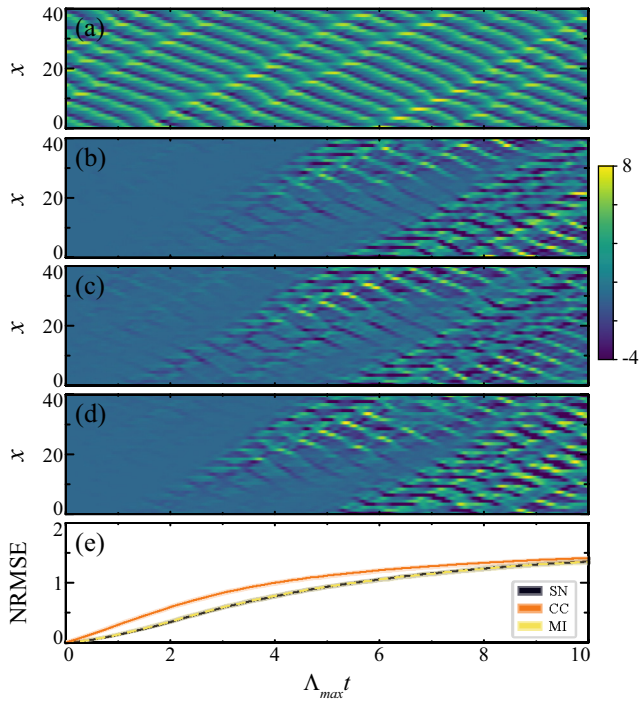

FIG. 3. L96 system short-term prediction comparisons. (a) Simulated data of the L96 system. (b)–(d) Exemplary difference of the RC prediction to the simulated data when using the (b) SN, (c) CC, and (d) MI neighborhoods. (e) NRMSE of SN, CC, and MI prediction data averaged over first the 300 predicted sections and then the 100 network realizations. The error bands correspond to the 3σ SD of the random network realizations. We multiply t by the MLE $\Lambda_{\max}$ of the model, so that each unit on the horizontal axis represents one Lyapunov time.

To calculate the SMs, we use the same noisy data used to train the reservoirs. As a result, all neighborhoods shown here are representative examples, but not uniform for all realizations. The neighborhoods are calculated for a single core and 18 neighbors in the case of SN and MI neighborhoods and 28 neighbors in the case of a CC neighborhood.

These fixed neighborhood sizes for SN and MI were chosen to be similar to Pathak's original paper [3], which we found to be a good compromise between computation speed (small neighborhoods) and prediction accuracy (large neighborhoods). Although this results in good predictions for them, this leads to essentially all predictions diverging for the CC SM. This is likely the result of the linear CC SM not recognizing the importance of the core's nearest neighbors in these systems. As such, the CC neighborhood size was increased to a total size of 29, the minimum where no predictions diverged.

TABLE II. MLEs calculated via the OS method for the simulated (SIM) and predicted trajectories using the SN, CC, and MI neighborhoods for the KS and L96 data. The errors represent the 1σ SD between network realizations.

System	SIM	SN	CC	MI
L96	0.45	0.43 ± 0.01	0.47 ± 0.02	0.44 ± 0.02

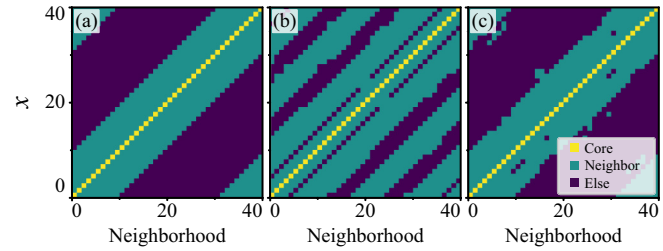

FIG. 4. Example neighborhoods for the KS system. Each row depicts one variable of the time series, while each column represents a neighborhood. Each neighborhood is defined by the core variable (yellow) and its neighbors (green). (a) SN, (b) CC, and (c) MI neighborhoods.

CC neighborhoods of different sizes for the L96 system are discussed in Appendix A.

B. Predicting the L96 system

Example neighborhoods for the L96 systems are shown in Fig 2.

While the SN neighborhoods are simply defined as a single core and its nearest neighbors, the CC and MI neighborhoods warrant a closer look. First and foremost, even though the MI neighborhoods were calculated dynamically, without directly using the knowledge of L96 being a locally interacting system, the resulting MI neighborhoods closely resemble the SN

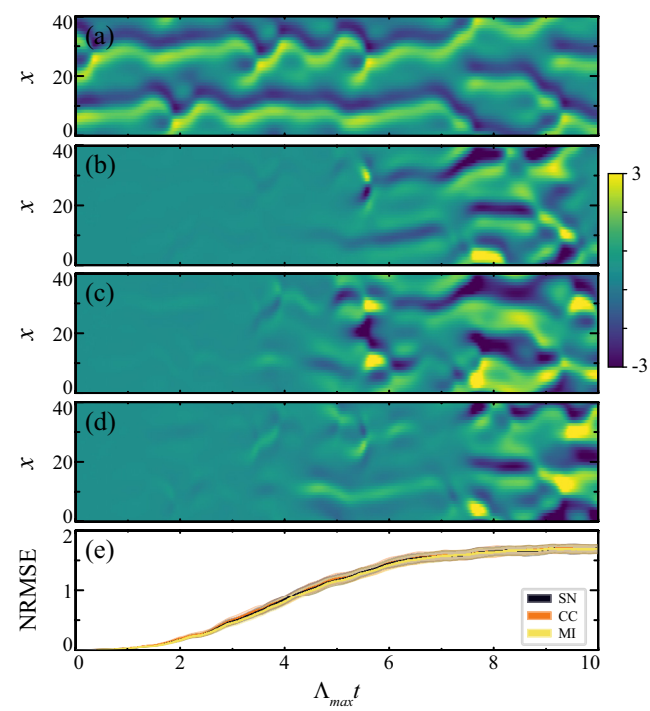

FIG. 5. KS system short-term prediction comparisons. (a) Simulated KS data. (b)–(d) Exemplary error in the RC prediction when using the (b) SN, (c) CC, and (d) MI neighborhoods. (e) NRMSE of SN, CC, and MI prediction data averaged over first the 300 predicted sections and then the 100 network realizations. The error bands correspond to the 3σ SD of the random network realizations.

TABLE III. Same as Table II but for the KS instead of the L96 system.

System	SIM	SN	CC	MI
KS	0.049	0.046 ± 0.004	0.048 ± 0.001	0.048 ± 0.001

neighborhoods. The CC neighborhoods, in contrast, include many variables spatially far away from the core.

One hundred distinct random network realizations are generated for each system-SM combination. They are then trained and used to predict the same 300 sections of 10 Lyapunov times length on the chaotic attractor of the L96 and KS systems.

To quantify the short-term prediction accuracy, we define the normalized root mean square error (NRMSE) as

$$\mathrm{NRMSE}(\hat{\boldsymbol{y}}) = \frac{\sqrt{(\hat{\boldsymbol{y}} - \boldsymbol{y})^2}}{y_{\max} - y_{\min}}, \tag{9}$$

where $\hat{\boldsymbol{y}} \in \mathbb{R}^D$ is the prediction at a single time step, $\boldsymbol{y} \in \mathbb{R}^D$ is the true signal, and $y_{\max}$ ($y_{\min}$) is the largest (smallest) value taken of any variable in the simulated data set. short-term prediction results are shown in Fig. 3.

Looking at the short-term predictions of the L96 system, it is very striking that the averaged NRMSE of the SN and MI neighborhoods coincide more or less exactly, while the CC prediction is significantly worse. This performance drop is likely the result of the CC neighborhood's inclusion of many variables which are far from the ostensibly most important close region around the core (see Fig. 2(b).

It should be noted that the statistical stability that has been assessed here for the first time is remarkable. Often RC algorithms exhibit a much larger spread of prediction qualities between different random networks [29]. This stability is partly attributable to finding the correct hyperparameters for the systems at hand, as suboptimally chosen hyperparameters often leave the best predictions intact while increasing the variance toward completely failed predictions drastically [16,29]. However, empirically we found that the role of the noise added to the training data in achieving this stability, especially for the L96 system, cannot be understated either.

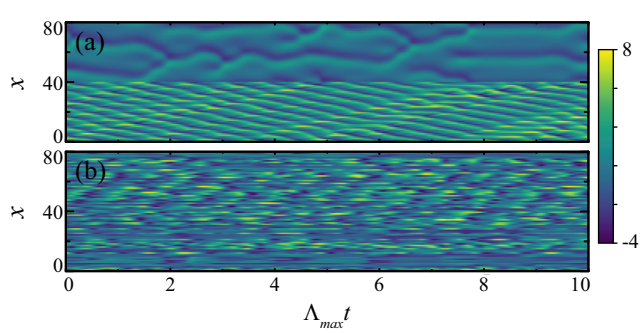

FIG. 6. The concatenated system. (a) Simulated data combined from the L96 (variables 1–40) and KS (variables 41–80) systems. (b) Simulated data as in (a) with the variables shuffled. The time axis is scaled by the MLE $\Lambda_{\max}$ of the KS model.

TABLE IV. MLEs $\Lambda_{\max}$, time step size Δt, Lyapunov time T_L, and the number of time steps per T_L for the KS and L96 systems.

System	$\Lambda_{\max}$	Δt	T_L	Time steps per T_L
KS	0.049	0.5	20	41
L96	0.45	0.05	2.2	45

Using the OS method, we can calculate the MLEs of the predictions as another characteristic to quantify long-term prediction accuracy. For this purpose, we let each of our reservoir realizations predict three distinct sections of the system attractor each 1000 Lyapunov times in length which are then used to calculate the MLEs shown in Table II.

The MLEs of all three neighborhoods agree well with the MLEs calculated directly from the simulations. The details of the OS procedure used are described in Appendix B.

C. Predicting the KS system

In preparation for the nonlocal system discussed in the following section, we predict and analyze the KS system as we have just done for the L96 system.

Example neighborhoods and short-term prediction results for the KS system are shown in Figs. 4 and 5, respectively.

Looking at the short-term predictions of the KS system shown in Fig. 5, it is striking that the averaged NRMSE of all three neighborhoods coincide more or less exactly. This is remarkable, especially when compared to the L96 system in which the CC neighborhoods performed significantly worse.

Using OS to calculate the MLE, we find them again to agree excellently with the simulated MLE as depicted in Table III.

D. Predicting a nonlocal system

To test the usefulness of GLS for nonlocally interacting systems, we use the KS and L96 systems to artificially create such a nonlocally interacting test system. As depicted in Fig. 6, we do this by concatenating both systems and then randomly shuffling the 80 variables of the combined system.

As this combined system now is a composite of two systems with different time steps, it does not have a well-defined Lyapunov exponent any more. For the sake of consistency, we nonetheless continue the time axis rescaling in terms of Lyapunov exponents. To do so we use the larger of the two system's time steps per Lyapunov time as calculated in Table IV, hence at worst slightly underestimating our short-term prediction results.

As before, we can also calculate the SN, CC, and MI neighborhoods for this new concatenated system. The CC and MI neighborhoods are shown in Fig. 7.

The SN neighborhoods have been omitted from Fig. 7 as they are, by definition, always the same. Fascinatingly, both the CC and the MI neighborhoods in the combined but not shuffled systems look like they are composed of the individual system's neighborhoods. In fact, this is exactly the case, as both the CC and the MI SMs are able to completely separate the KS and L96 systems. In the case of experimental data, such an analysis of the similarity structure can be highly

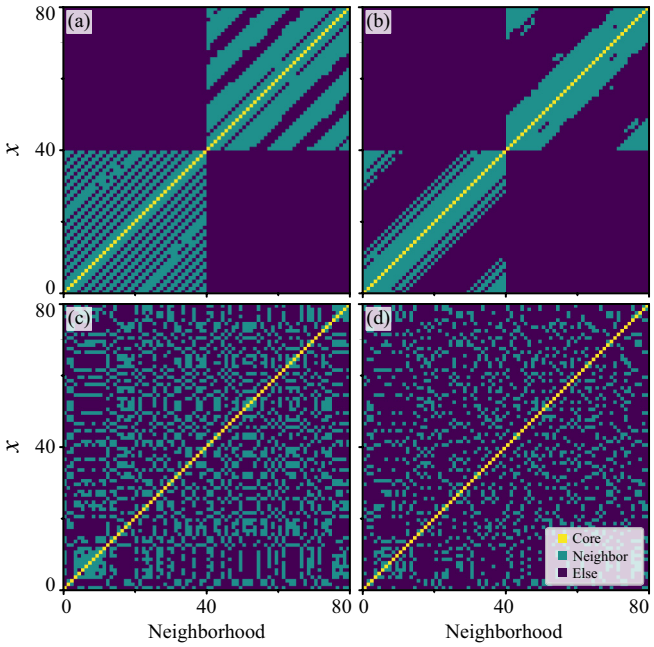

FIG. 7. Neighborhoods of the concatenated L96 and KS systems. (a) The CC and (b) the MI neighborhoods for the concatenated, unshuffled system. Variables 1–40 of the data used to compute these neighborhoods come from the L96 system with variables 41–80 originating from the KS simulation; (c) the CC and (d) MI neighborhoods for the shuffled system are shown.

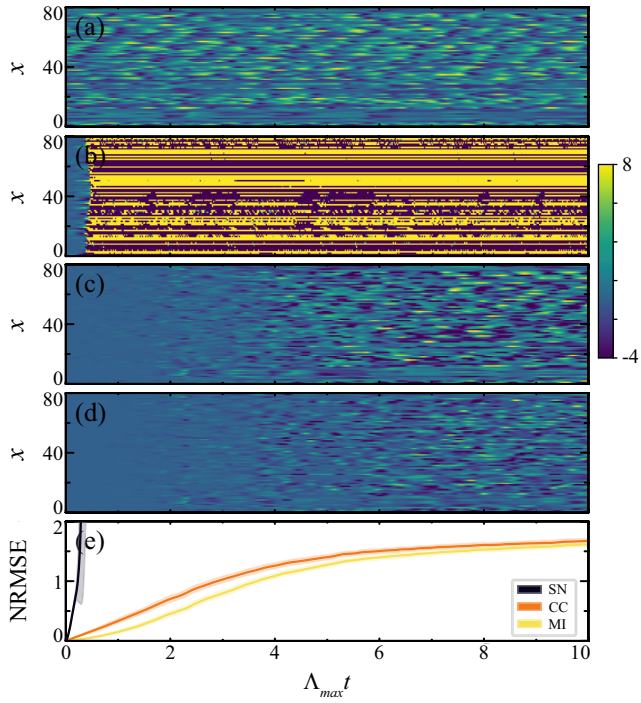

FIG. 8. Shuffled system short-term prediction comparison. (a) Simulated data of the combined shuffled KS-L96 system. (b)–(d) Exemplary difference of the RC prediction to the simulated data when using the (b) SN, (c) CC, and (d) MI neighborhoods. The color scale of the diverging prediction was cut to the ensure legibility of the other plots (e) NRMSE of SN, CC, and MI prediction data averaged over first the 300 predicted sections and then the 100 network realizations. The error bands correspond to the 3σ SD of the random network realizations.

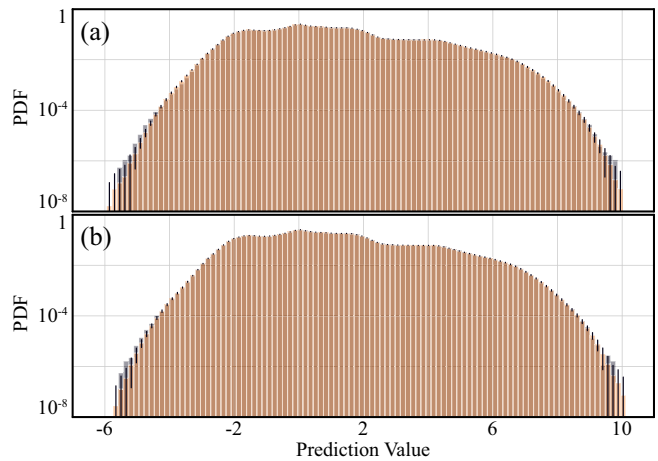

FIG. 9. PDFs for the long-term predictions of the combined and shuffled KS-L96 system. The PDFs are estimated via two superimposed 100 bin histograms of the simulated (gray) and the predicted (orange) data of the (a) CC or (b) MI neighborhood. Each entry in the histogram is generated by the prediction of a single variable value, with the data set coming from the prediction of three 1000 Lyapunov time-long sequences. The error bars represent the 1σ SD of the predicted histogram bins.

informative as it exposes the underlying relationships between dimensions.

As before, we quantify the short-term prediction accuracy using the NRMSE. The results are shown in Fig 8. Immediately noticeable is the almost instant divergence of all SN predictions. This is of course expected, considering the SN neighborhoods assume a locally interacting system, which the shuffled system is not. Furthermore, the NRMSE of the CC and MI neighborhoods is the combination of the NRMSE of the individual systems. This, again, makes sense due to the perfect separation between the L96 and the KS systems shown in Fig. 7. As for the KS system, this results in the MI SM delivering significantly better results than the CC SM.

For the long-term statistical analysis we cannot use the MLE as it is not well defined due to the different time step sizes of the subsystems. Therefore we look at the probability distribution functions (PDFs) to roughly estimate the system climate, as also used in Ref. [4]. The details of the histogram generation are described in Appendix C. CC and MI PDFs are depicted in Fig. 9.

The predicted PDFs show excellent agreement with the simulated data. Similarly to the MLE results, the worst short-term predictive performance of the CC SM compared to the MI SM is not reflected in its climate reproduction quality.

VI. CONCLUSIONS

We have proposed and discussed a generalization of the concept of LS in the sense of using SM derived from correlations among (instead of spatial distances between) time series. This offers a much more versatile approach for the prediction of high-dimensional complex systems.

First, GLS can still make excellent predictions in the case of mixed systems, where LS are doomed to fail. Here it is worth noticing that the perfect neighborhood separation

we found in the combined KS-L96 system (see Fig. 7) suggests that GLS can be used to separate mixed data sets and to thus infer different origins of a set of heterogeneous time series, for which the generating processes are unknown.

Second, prediction of high-dimensional systems remains feasible when for some or all time series no spatial information is available. This is more and more the typical case in real world applications, when analyzing such heterogeneous data sets ranging from remote sensing data, financial data, social media data (e.g., Instagram and Twitter) to nowadays infection rates during the COVID-19 pandemic, etc., and a combination of the aforementioned. Current research explores applications of GLS -based predictions in those cases.

ACKNOWLEDGMENTS

We acknowledge valuable discussions with D. Mohr, P. Huber, J. Aumeier, Y. Mabrouk, and J. Herteux.

APPENDIX A: CC NEIGHBORHOOD SIZE

As discussed in the text, the CC neighborhood sizes were necessarily chosen to be significantly larger than the SN and MI neighborhood sizes.

This necessity is likely the result of the CC SM not recognizing the importance of the core's nearest neighbors in the L96 and KS systems. As such, the CC neighborhood size was increased to a total size of 29, the minimum where no predictions diverged and where the core's nearest neighbors were consistently included in its neighborhood. CC neighborhoods of sizes 19, 27, and 29 for the L96 system are depicted in Fig. 10.

While the true importance of each variable regarding the prediction of another is of course unknown, at least for the locally interacting L96 and KS systems studied here, the spatial nearest neighbors of the cores are critical for achieving a sensible prediction.

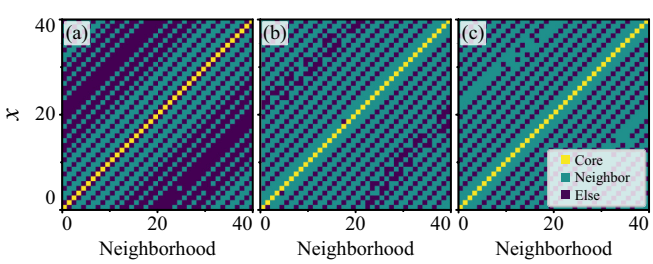

FIG. 10. CC neighborhoods of the L96 system. (a) Total neighborhood size 19; the nearest neighbors of the core variables are not included in the neighborhood. (b) Neighborhood size 27. Only a couple of nearest neighbors of the cores are missing. Nevertheless, predictions using these neighborhoods diverge regularly. (c) Neighborhood size 29. All nearest neighbors of the cores are included in the neighborhood. This is the smallest neighborhood size for which predictions consistently succeed.

APPENDIX B: ORBIT SEPARATION

The most important Lyapunov exponent is typically considered to be the MLE, defined as the largest Lyapunov exponent of any given chaotic system. Its importance stems from the fact that it is intimately tied to the predictability of the system as, given a MLE $\Lambda_{\max}$ two infinitesimally close trajectories in phase space, initially separated by the vector $\delta\boldsymbol{x}(t=0)$ diverge as

$$|\delta\boldsymbol{x}(t)| \approx e^{\Lambda_{\max} t} |\delta\boldsymbol{x}(t=0)|\,, \tag{B1}$$

where t is the time since separation [30].

To calculate the MLE, we use Sprott's method of OS [20]. By taking the logarithm and the average $\langle\cdot\rangle$ over many trajectory divergences in different parts of the chaotic attractor, we find

$$\Lambda_{\max} = \frac{1}{t_2 - t_1}\left\langle \frac{\log|\delta\boldsymbol{x}(t_2)|}{\log|\delta\boldsymbol{x}(t_1)|}\right\rangle. \tag{B2}$$

Note that for this equation to hold, we are only using the divergence data after transient effects have subsided but before the divergence size saturates due to it reaching the size of the chaotic attractor. From this, we can use a simple linear least-squares fit to calculate the MLE.

As described in the main text, we base the OS calculation on three 1000 Lyapunov time long-term prediction data sets. For each of the 100 realizations, we choose 50 trajectory positions uniformly distributed in the first of the three long-term data sets as a starting point for the OS at which we add normally distributed noise with SD $\sigma_{\text{noise}} = 10^{-10}\sigma_{\text{r}}$ to the internal reservoir states. From this perturbed internal state, we let the reservoir predict 1500 time steps and compute the separation magnitude to the unperturbed predicted time series using the least-squared fit of Eq. (B2) as described above.

The MLE of the L96 and KS simulations are computed analogously, using one 10^5 time step-long data set from which 10^4 uniformly distributed starting positions for the trajectory divergence are chosen. The resulting MLEs are given in Table IV.

APPENDIX C: PDF ESTIMATION

PDF estimation was done with the same data sets used to calculate the MLE. The three resulting predictions and reference data sets from the simulation are treated as one single data set of 3000 Lyapunov times length, respectively. The resulting data sets are flattened and then used to create the histogram. The histograms themselves consist of 100 equally sized bins. To make comparison easy, they are chosen such that bin position and size for the simulated and predicted time series histograms are the same.

[1] Z. Lu, B. R. Hunt, and E. Ott, Attractor reconstruction by machine learning, Chaos: An Interdiscip. J. Nonlinear Sci. **28**, 061104 (2018).

[2] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, Data-driven forecasting of high-dimensional chaotic systems with long short-Term memory networks, Proc. R. Soc. A **474**, 20170844 (2018).

[3] 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).

[4] A. Chattopadhyay, A. Subel, and P. Hassanzadeh, Data-driven super-parameterization using deep learning: Experimentation with multiscale Lorenz 96 systems and transfer learning, J. Adv. Model. Earth Syst. **12**, 16 (2020).

[5] Y. A. Mabrouk and C. Räth, Calibrated reservoir computers, Chaos: An Interdiscip. J. Nonlinear Sci. **30**, 113134 (2020).

[6] A. Haluszczynski, J. Aumeier, J. Herteux, and C. Räth, Reducing network size and improving prediction stability of reservoir computing, Chaos: An Interdiscip. J. Nonlinear Sci. **30**, 063136 (2020).

[7] T. L. Carroll, Do reservoir computers work best at the edge of chaos? Chaos: An Interdiscip. J. Nonlinear Sci. **30**, 121109 (2020).

[8] S. Bompas, B. Georgeot, and D. Guéry-Odelin, Accuracy of neural networks for the simulation of chaotic dynamics: Precision of training data vs precision of the algorithm, Chaos: An Interdiscip. J. Nonlinear Sci. **30**, 113118 (2020).

[9] G. Marcucci, D. Pierangeli, and C. Conti, Theory of Neuromorphic Computing by Waves: Machine Learning by Rogue Waves, Dispersive Shocks, and Solitons, Phys. Rev. Lett. **125**, 093901 (2020).

[10] U. Parlitz and C. Merkwirth, Prediction of Spatiotemporal Time Series Based on Reconstructed Local States, Phys. Rev. Lett. **84**, 1890 (2000).

[11] R. S. Zimmermann and U. Parlitz, Observing spatio-temporal dynamics of excitable media using reservoir computing, Chaos: An Interdiscip. J. Nonlinear Sci. **28**, 043118 (2018).

[12] R. N. Mantegna, Hierarchical structure in financial markets, Eur. Phys. J. B **11**, 193 (1999).

[13] J. P. Onnela, A. Chakraborti, K. Kaski, J. Kertész, and A. Kanto, Dynamics of market correlations: Taxonomy and portfolio analysis, Phys. Rev. E **68**, 056110 (2003).

[14] A. Haluszczynski, I. Laut, H. Modest, and C. Räth, Linear and nonlinear market correlations: Characterizing financial crises and portfolio optimization, Phys. Rev. E **96**, 062315 (2017).

[15] 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: An Interdiscip. J. Nonlinear Sci. **27**, 041102 (2017).

[16] P. R. Vlachas, J. Pathak, B. R. Hunt, T. P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics, Neural Networks **126**, 191 (2020).

[17] Y. Huang, Z. Fu, and C. L. E. Franzke, Detecting causality from time series in a machine learning framework, Chaos: An Interdiscip. J. Nonlinear Sci. **30**, 063116 (2020).

[18] E. N. Lorenz, Predictability: A problem partly solved, in *Proceedings of the Seminar on Predictability, 4–8 Sept. 1995* (ECMWF, Shinfield Park, Reading, 1995), Vol. 1, pp. 1–18.

[19] D. S. Wilks, Effects of stochastic parametrizations in the Lorenz '96 system, Q. J. R. Meteorol. Soc. **131**, 389 (2005).

[20] J. C. Sprott and J. C. Sprott, *Chaos and Time-Series Analysis* (Oxford University Press, Oxford, 2003), Vol. 69.

[21] W. H. Press, S. A. Teukolsky, B. P. Flannery, and W. T. Vetterling, *Numerical Recipes in Fortran 77: volume 1, Fortran Numerical Recipes: the Art of Scientific Computing* (Cambridge University Press, Cambridge, 1992).

[22] Y. Kuramoto and T. Tsuzuki, Persistent propagation of concentration waves in dissipative media far from thermal equilibrium, Prog. Theor. Phys. **55**, 356 (1976).

[23] Y. Pomeau and S. Zaleski, *The Kuramoto-Sivashinsky Equation: A Caricature of Hydrodynamic Turbulence?* (Springer, Berlin, 1985), pp. 296–303.

[24] A. K. Kassam and L. N. Trefethen, Fourth-order time-stepping for stiff PDEs, SIAM J. Sci. Comput. **26**, 1214 (2005).

[25] J. Herteux and C. Räth, Breaking symmetries of the reservoir equations in echo state networks, Chaos: An Interdiscip. J. Nonlinear Sci. **30**, 123142 (2020).

[26] C. E. Shannon, A mathematical theory of communication, Bell Syst. Tech. J. **27**, 623 (1948).

[27] A. Kraskov, H. Stögbauer, and P. Grassberger, Estimating mutual information, Phys. Rev. E **69**, 066138 (2004).

[28] A. Strehl and J. Ghosh, Cluster ensembles - A knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. **3**, 583 (2002).

[29] A. Haluszczynski and C. Räth, Good and bad predictions: Assessing and improving the replication of chaotic attractors by means of reservoir computing, Chaos: An Interdiscip. J. Nonlinear Sci. **29**, 103143 (2019).

[30] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, A practical method for calculating largest Lyapunov exponents from small data sets, Phys. D: Nonlinear Phenom. **65**, 117 (1993).