Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach

Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott

Phys. Rev. Lett. **120**, 024102 — Published 12 January 2018

DOI: 10.1103/PhysRevLett.120.024102

Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Machine Learning Approach

Jaideep Pathak,[1,2] Brian Hunt,[3,4] Michelle Girvan,[1,3,2] Zhixin Lu,[1,3] and Edward Ott[1,2,5]

[1]*Institute for Research in Electronics and Applied Physics, University of Maryland, College Park, Maryland 20742, USA*
[2]*Department of Physics, University of Maryland, College Park, Maryland 20742, USA.*
[3]*Institute for Physical Science and Technology, University of Maryland, College Park, Maryland 20742, USA*
[4]*Department of Mathematics, University of Maryland, College Park, Maryland 20742, USA.*
[5]*Department of Electrical and Computer Engineering, University of Maryland, Maryland 20742, USA.*

(Dated: December 12, 2017)

We demonstrate the effectiveness of using machine learning for model-free prediction of spatiotemporally chaotic systems *of arbitrarily large spatial extent and attractor dimension* purely from observations of the system's past evolution. We present a parallel scheme with an example implementation based on the reservoir computing paradigm and demonstrate the scalability of our scheme using the Kuramoto-Sivashinsky equation as an example of a spatiotemporally chaotic system.

Recently, machine learning techniques have proven useful for a wide variety of tasks, from speech recognition [1] to playing Go [2]. In this paper we show that machine learning can be used for model-free prediction of the evolution of the state of a large spatiotemporally chaotic system. The accomplishment of this task is of great potential application, e.g., for prediction of geophysical dynamical systems. Specifically, we consider a situation where a mechanistic description of the dynamics is unavailable or insufficient for the desired purpose, but reasonably accurate and complete observational data for the evolution of the state of the system of interest can be obtained. Assuming this situation, the goal of this paper is to formulate an effective technique for predicting the future evolution of very large spatiotemporally chaotic systems from data, an especially difficult problem presently without a robust solution using existing techniques. We note that model-free techniques for prediction based on delay coordinate embedding are well established [3]. These techniques are effective for *low dimensional* chaos, and extensions have been proposed for large spatiotemporally chaotic systems [4]. Within the machine learning community, there have been a number of rapid advances in prediction using the technique known as reservoir computing [5–7]. In particular, Jaeger and Haas [8] have applied reservoir computing to predict low dimensional chaotic systems with good results. Although we focus on reservoir computing, we expect that other machine learning techniques, e.g., deep learning [9, 10], might also be useful for the task we address. On the other hand, we speculate that, because of their essential dynamical character (see below), artificial neural networks with recurrent connections [11–13], such as reservoir computers, may be inherently well-suited for tasks which are themselves dynamical in character such as prediction or inference of unmeasured state variables of a deterministic system [14]. We find that our reservoir-based spatiotemporal prediction technique yields excellent prediction results of unprecedented quality at reasonable expense.

We now briefly introduce the basic ideas of reservoir computing. An input vector $\mathbf{u}(t)$ of dimension D_{in} (Fig. 1(a)) is coupled through an I/R (input-to-reservoir) coupler to a high dimensional dynamical system (labeled R in Fig. 1(a)) called the "reservoir", from which an output vector $\mathbf{v}(t)$ of dimension D_{out} is coupled through an R/O (reservoir-to-output) coupler. The R/O coupler is assumed to depend on many (D_p) adjustable parameters $\mathbf{p}$, and to create outputs $\mathbf{v}(t)$ that depend linearly upon the parameters $\mathbf{p}$. Denoting the state of the D_r dimensional reservoir by the vector $\mathbf{r}(t)$, the I/R, reservoir, and R/O functions can be represented in discrete time ($t = 0, \Delta t, 2\Delta t, \dots$) by $\mathbf{r}(t+\Delta t) = \mathbf{G}[\mathbf{r}(t), \mathbf{W}_{in}(\mathbf{u}(t))], \mathbf{v}(t) = \mathbf{W}_{out}[\mathbf{r}(t), \mathbf{p}]$, where $\mathbf{W}_{in}$ (respectively $\mathbf{W}_{out}$) is a mapping from the D_{in} (D_r) dimensional input state space (reservoir state space) to the D_r (D_{out}) dimensional reservoir state space (output state space). We note that while, in this paper, we consider time to be discrete (and will subsequently take Δt to be small), the analogous continuous time reservoir is also commonly employed. The goal is to train this system to make $\mathbf{v}(t)$ closely approximate the desired outputs $\mathbf{v}_d(t)$ appropriate to the inputs $\mathbf{u}(t)$ (e.g., if the function of the system is speech recognition [1], $\mathbf{u}(t)$ might be an electronic signal derived from a person speaking, while $\mathbf{v}_d(t)$ would represent the words being spoken). To accomplish this, one uses training data consisting of pre-recorded and stored measurements of $\mathbf{u}(t)$ and the resulting $\mathbf{r}(t)$ in some time interval, $-T \le t \le 0$, and chooses the output parameters $\mathbf{p}$ so as to minimize the least squares difference between $\mathbf{v}_d(t)$ and $\mathbf{v}(t)$ over the time interval $-T \le t \le 0$. Since $\mathbf{v} = \mathbf{W}_{out}[\mathbf{r}, \mathbf{p}]$ is assumed to be linear in the parameters $\mathbf{p}$, the problem of determining $\mathbf{p}$, and hence $\mathbf{W}_{out}$, is a simple linear regression [15]. With $\mathbf{p}$ determined, if all goes well, the reservoir system can be used to fulfill its intended task for $t \ge 0$. Indeed, for large enough D_p and D_r, this framework has proven to be extremely successful for a variety of tasks [7].

Here we are interested in the task of predicting the future, $t > 0$, evolution of $\mathbf{u}(t)$ from training data in $-T \leq t \leq 0$. The prediction task via reservoir computing has been previously addressed with excellent results for a situation where $\mathbf{u}(t)$ comes from the state of a low dimensional chaotic system [8]. In that reference, the desired output condition was that $\mathbf{v}(t)$ be a good approximation to $\mathbf{u}(t)$ (i.e., $\mathbf{v}_d(t) = \mathbf{u}(t)$). After "training" $\mathbf{v}(t)$ to approximate $\mathbf{u}(t)$, the future evolution of $\mathbf{u}(t)$ for $t > 0$ is predicted by replacing the input $\mathbf{u}(t)$ in Fig. 1(a) by $\mathbf{v}(t)$, as shown in Fig. 1(b). As we will demonstrate, prediction with a single reservoir becomes computationally unfeasible as D_{in} increases. We will propose and illustrate a solution to this problem for spatiotemorally chaotic systems using parallel reservoirs assigned to different spatial regions.

In this paper, we focus on the following specific implementation choices, which are similar to those in Ref. [8]. (We emphasize here that our choices are illustrative and that many others are possible.) The I/R coupler is $\mathbf{W}_{in}(\mathbf{u}) = \mathbf{W}_{in}\mathbf{u}$ (where $\mathbf{W}_{in}$ is a $D_r \times D_{in}$ matrix whose input elements are drawn from a uniform distribution in $[-\sigma, \sigma]$). The reservoir is a large, low degree (κ), directed Erdös-Rényi network with a $D_r \times D_r$ adjacency matrix $\mathbf{A}$, appropriately scaled so that its largest eigenvalue is equal to ρ. The state of each network node j is a scalar $r_j(t)$ which, in the set-up of Fig. 1(a), evolves according to

$$\mathbf{r}(t + \Delta t) = \tanh\left[\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{in}\mathbf{u}(t)\right], \qquad (1)$$

where, for a vector $\mathbf{q} = [q_1, q_2, \dots]^T$, $\tanh(\mathbf{q})$ is the vector $[\tanh(q_1), \tanh(q_2), \dots]^T$. The R/O coupler is $\mathbf{W}_{out}(\mathbf{r}) = \mathbf{P}_1\mathbf{r} + \mathbf{P}_2\mathbf{r}^2$, where $\mathbf{P_1}$ and $\mathbf{P}_2$ are $D_{out} \times D_r$ matrices, $\mathbf{p} = (\mathbf{P}_1, \mathbf{P}_2)$, and $\mathbf{r}^2$ is the D_r dimensional vector whose j^{th} component is r_j^2. (We found that the simpler choice $\mathbf{W}_{out}(\mathbf{r}) = \mathbf{P}_1\mathbf{r}$ typically did not work for our illustrative example [16].) While, for illustration, we use the specific reservoir dynamics of Eq. (1), we emphasize that there is great versatility in the scheme of Fig. 1. E.g., for tasks other than prediction, very fast processing has been achieved by using high dimensional photonic systems as the reservoir [17–20] (see also Ref. [21]).

In the prediction phase, $t > 0$, $\mathbf{u}(t)$ in Eq. (1) is replaced by $\mathbf{v}(t) = \mathbf{W}_{out}(\mathbf{r}(t))$. Regardless of the short-term quality of the predictions $\mathbf{v}(t)$, they will eventually diverge from the true state $\mathbf{u}(t)$ due to the exponential separation of trajectories that is a characteristic of chaotic systems. Consider now the situation where at some future time $\theta > 0$, one wants to predict $\mathbf{u}(t)$ for $t > \theta$ based on measurements of $\mathbf{u}$ up to time θ. The reservoir can then be reinitialized using Eq.(1) for a short period of time preceding θ, i.e., $(\theta - \epsilon \leq t \leq \theta)$, to determine $\mathbf{r}(\theta)$, and then used to predict for $t > \theta$. (Once the training is done, it need not be repeated for predictions of subsequent time intervals.)

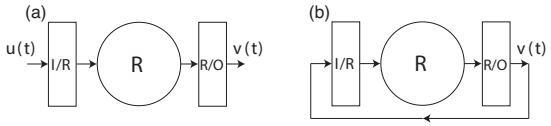

FIG. 1. I/R : (Input-Reservoir Coupler); R : (Reservoir); R/O : (Reservoir-Output Coupler). (a) Training data gathering phase. (b) Predicting phase. It is assumed that the parameters of the reservoir are chosen such that the "echo state property" is satisfied [7]: all of the conditional Lyapunov exponents of the training reservoir dynamics conditioned on $\mathbf{u}(t)$ are negative so that, for large t, the reservoir state $\mathbf{r}(t)$ does not depend on initial conditions.

As an illustrative model for a spatiotemporally chaotic system, we consider the Kuramoto-Sivashinsky (KS) equation modified by the addition (last term in Eq. (2)) of a spatial inhomogeneity term,

$$y_t = -yy_x - y_{xx} - y_{xxxx} + \mu\cos\left(\frac{2\pi x}{\lambda}\right). \qquad (2)$$

The scalar field $y(x,t)$ is periodic in the interval $[0, L)$ and L is an integer multiple of λ. Note that the attractor dimension depends directly on the dimensionless parameter L and scales linearly with L for large L [22]. For later comparison, we note that for $L \geq 100$, the RMS value of y_t is about 0.34, which can be compared to the value of μ to roughly assess the strength of the inhomogeneity on the dynamics. This equation reduces to the standard KS equation when $\mu = 0$. The cosine perturbation breaks the translation symmetry when $\mu \neq 0$. In this paper, we will consider both $\mu = 0$ and $\mu \neq 0$ in order to probe the effect of spatial homogeneity on our predictions. Equation (2) is integrated on a grid of Q equally spaced points with $\Delta t = 0.25$, giving a simulated data set with Q time series, which we denote by the vector $\mathbf{u}(t)$ and use as the reservoir input. Figure 2(a) shows our numerical solution of Eq. (2) for a KS system with $L = 22, Q = 64$, and $\mu = 0$, while figure 2(b) shows a reservoir performed prediction using the scheme described above (Fig. 1). Figure 2(c) shows the difference between the prediction and the actual solution (we remark that this error metric may overemphasize errors due to spatial shifting of the patterns).

Although the results of Fig. 2 indicate the potential for reservoir-computer-based prediction of spatiotemporal chaos, we note that, as L increases, the size D_r of the reservoir network required to predict the system using a single reservoir (as described by Fig. 1) increases. We find that this makes prediction with a single reservoir intractable for much larger values of L. In order to treat large systems, we take advantage of the local nature of interactions in typical spatiotemporally chaotic systems, as was done in Ref. [4] in the context of delay co-ordinates. We propose a parallelized scheme consisting of a large set of reservoirs of moderate size, each of which predicts

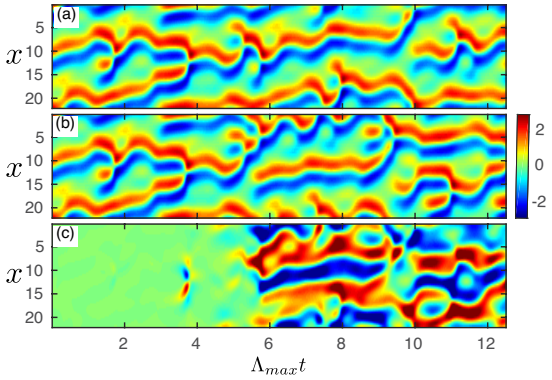

FIG. 2. Prediction of a KS equation with $L = 22$, $\mu = 0$ using a single reservoir of size $D_r = 5000$. (a) Actual data from the KS model. (b) Reservoir prediction. (c) Error (panel (b) minus panel (a)) in the reservoir prediction. We multiply t by the largest Lyapunov exponent (Λ_{max}) of the model, so that each unit on the horizontal axis represents one Lyapunov time, i.e., the average amount of time for errors to grow by a factor of e.

a local region of the spatio-temporal system. We comment that a somewhat similar structure is employed by convolutional neural networks (CNN's), e.g., see chapter 9 of Ref. [10]. CNN's are widely used in deep learning for image processing tasks, and employ a translationally invariant structure (as we will later discuss for our KS example with $\mu = 0$).

Consider a spatiotemporal system on a one dimensional grid of size Q with periodic boundary conditions, giving us a multivariate data set with Q time series which we denote by the vector $\mathbf{u}(t)$. The Q variables $u_j(t)$ are split into g groups, each group consisting of q spatially contiguous variables such that $gq = Q$. We denote the states of the spatial points in each of the g groups by the vectors $\mathbf{g}_i(t)$: $\mathbf{g}_1(t) = (u_1(t), u_2(t), \cdots u_q(t))^T$, $\mathbf{g}_2(t) = (u_{q+1}(t), u_{q+2}(t), \cdots u_{2q}(t))^T$, and so on. Each group of time series, $\mathbf{g}_i$, is predicted by a reservoir R_i with adjacency matrix $\mathbf{A}_i$, internal state $\mathbf{r}_i(t)$ and input weights $\mathbf{W}_{in,i}$. We denote the input to the i^{th} network by $\mathbf{h}_i(t)$, where $\mathbf{h}_i(t)$ is such that each reservoir accepts inputs from all of the spatial points in the i^{th} group as well as from two contiguous buffer regions of l spatial points on its left and right hand sides, $\mathbf{h}_i(t) = (u_{(i-1)q-l+1}(t), u_{(i-1)q-l+2}(t), \cdots, u_{iq+l}(t))^T$ (the subscript j in u_j is taken modulo Q). Thus, adjacent reservoir networks have overlapping inputs with the size of the overlap set by the locality parameter l (see Fig. 3).

The data from $t = -T$ to $t = 0$ is used to train the reservoir network, while the data from $t > 0$ is used to evaluate the quality of the reservoir predictions. Similar to Eq. (1), in the training phase, each of the g reservoirs evolves in parallel according to $\mathbf{r}_i(t + \Delta t) = \tanh(\mathbf{A}_i\mathbf{r}_i(t) + \mathbf{W}_{in,i}\mathbf{h}_i(t))$, $1 \le i \le g$, from $t = -T$ to $t = 0$. The g reservoirs are then trained by finding a set of output weights $\mathbf{p}_i = (\mathbf{P}_{1,i}, \mathbf{P}_{2,i})$ for each reservoir such that $\mathbf{P}_{1,i}\mathbf{r}_i(t) + \mathbf{P}_{2,i}\mathbf{r}_i^2(t) \simeq \mathbf{g}_i(t)$. The trained reservoirs with their output weights are now used to predict the time series, $\tilde{\mathbf{g}}_i(t) = \mathbf{P}_{1,i}\mathbf{r}_i(t) + \mathbf{P}_{2,i}\mathbf{r}_i^2(t)$, $\mathbf{r}_i(t + \Delta t) = \tanh(\mathbf{A}_i\mathbf{r}_i(t) + \mathbf{W}_{in}\tilde{\mathbf{h}}_i(t))$, where $\tilde{\mathbf{h}}_i(t)$ is determined from $\tilde{\mathbf{g}}_i(t)$ and the output of the neighboring reservoirs, and we use a superscribed tilde to denote a predicted quantity.

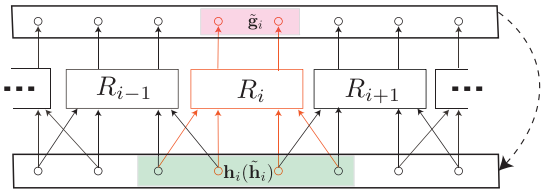

FIG. 3. Illustration of the parallellized reservoir scheme ($q = 2$, $l = 1$). The pink shaded vector above R_i represents its output $\tilde{\mathbf{g}}_i$. The green shaded vector below R_i represents its input $\mathbf{h}_i$ (during training) and $\tilde{\mathbf{h}}_i$ (during prediction). The dashed arrow shows the feedback connection applied during the autonomous prediction phase ($t \ge 0$).

We now present numerical results; unless otherwise specified, the reservoir parameters used are $D_r = 5000$, $T = 70000$ steps, $\rho = 0.6$, $\sigma = 1.0$, $l = 6$ and $\kappa = 3$. Once the reservoir is trained and the output weights are determined, the resulting autonomous reservoir is used to make a series of predictions, which are then compared with the evaluation data set. We perform predictions on $K = 30$ non-overlapping time intervals, $\theta_k \le t < \theta_k + \tau$, each of length $\tau = 1000$ in the evaluation data set. Here $\theta_k = (k - 1)\tau$ denotes the start of each prediction interval. Before the start of each prediction interval, all reservoir states are reset to $\mathbf{r}_i = \mathbf{0}$ and the reservoirs are then evolved with the true measurements $\mathbf{u}(t)$ for $\epsilon = 10$ time steps, i.e., from $t = \theta_k - \epsilon$ to θ_k, according to $\mathbf{r}_i(t + \Delta t) = \tanh(\mathbf{A}_i\mathbf{r}_i(t) + \mathbf{W}_{in,i}\mathbf{h}_i(t))$, $1 \le i \le g$. This gives the reservoir appropriate initial conditions to begin predicting autonomously for the next τ steps. The RMS error between $\mathbf{u}(t)$ and $\tilde{\mathbf{u}}(t) = (\tilde{\mathbf{g}}_1(t), \ldots, \tilde{\mathbf{g}}_g(t))$ is averaged over the K independent predictions to give an estimate of the typical quality of prediction. We perform the same prediction 10 times, for different random reservoir realizations, and calculate the average RMSE over all the trials. Figure 4 shows the results for a KS equation ($L = 200$, $\mu = 0.01$, $Q = 512$) where panel (a) is the numerical solution of Eq. (2), panel (b) is the reservoir prediction using $g = 64$ reservoirs of size $D_r = 5000$ each, and panel (c) is the prediction error (panel (a) minus panel (b)). We see that low prediction error is obtained for about 8 Lyapunov times. As a performance benchmark, panel (d) shows the error of the prediction made by integrating the KS equation (with the same solution method as panel (a)) using the output of the reservoir at $t = 0$ as its initial condition. Thus, panels (c) and (d) have the exact same error at $t = 0$. We see that the

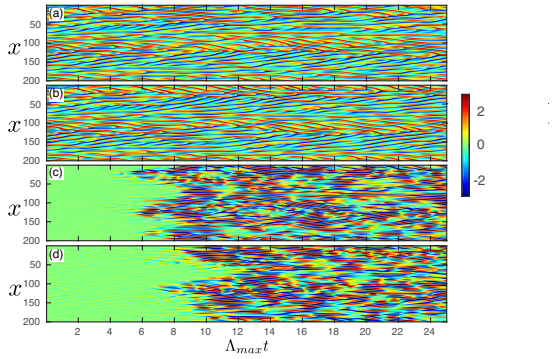

FIG. 4. Prediction of KS equation ($L = 200$, $Q = 512$, $\mu = 0.01$, $\lambda = 100$) with the parallelized reservoir prediction scheme using $g = 64$ reservoirs. (a) Actual KS equation data. (b) Reservoir prediction ($\tilde{\mathbf{u}}(t)$). (c) Error in the reservoir prediction. (d) Error in a prediction made by integrating the KS equation when it uses the reservoir output at $t = 0$, $\tilde{\mathbf{u}}(0)$, as its initial condition.

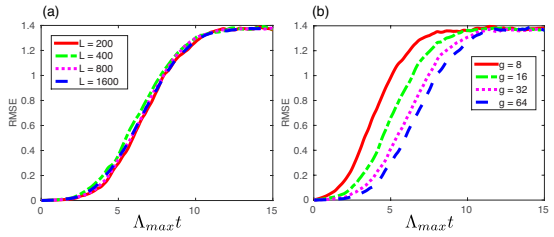

FIG. 5. (a) RMS error in the predictions of the KS system as function of time for different system sizes $L = 200, 400, 800, 1600$ with L/g held fixed at $200/64$ for all four curves. (b) Improvement of the prediction performance as the number (g) of reservoirs employed is increased; $L = 200$, $Q = 512$, $\mu = 0.01$, $\lambda = 100$.

prediction time in panel (d) is only slightly longer than that for panel (c), indicating good reproduction of the true dynamics by the reservoir system.

Figure 5(a) shows that we can obtain predictions comparable to Fig. 4 independent of the system size L. Table I indicates the largest Lyapunov exponent Λ_{max} and estimated Kaplan-Yorke dimension [23] of the KS system along with the number of reservoirs (g) and the total number of nodes N_T in the g reservoirs used for Fig. 5(a).

When the strength of the cosine perturbation term is set to $\mu = 0$, the KS equation (Eq. (2)) has translation symmetry which can be exploited to drastically reduce the computational cost of training the output weights. We find that it is then sufficient to train a single reservoir (say R_1) by evolving it according to $\mathbf{r}_1(t + \Delta t) = \tanh(\mathbf{A}_1\mathbf{r}_1(t) + \mathbf{W}_{in,1}\mathbf{h}_1(t))$ and then calculating $(\mathbf{P}_{1,1}, \mathbf{P}_{2,1})$. We then use g identical reservoir systems with $\mathbf{W}_{in,i} = \mathbf{W}_{in,1}$, $\mathbf{A}_i = \mathbf{A}_1$, and $(\mathbf{P}_{1,i}, \mathbf{P}_{2,i}) = (\mathbf{P}_{1,1}, \mathbf{P}_{2,1})$ for $1 \le i \le g$ in the prediction phase equations. As shown by the agreement between

L	Λ_{max}	D_{KY}	g	N_T ($\times 10^5$)
100	0.09	23	32	1.6
200	0.09	43	64	3.2
400	0.09	85	128	6.4
800	0.1	167	256	12.8
1600	0.1	338	512	25.6

TABLE I. Largest Lyapunov Exponent (Λ_{max}) and Kaplan-Yorke Dimension (D_{KY}) of the attractor ($\lambda = 100, \mu = 0.01$) along with the number of parallel reservoirs (g) and the total number (N_T) of all nodes in the g reservoirs of the parallelized reservoir scheme used.

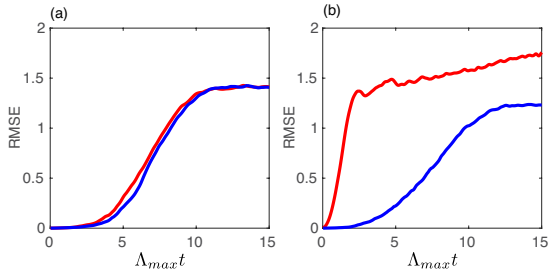

FIG. 6. Reservoir prediction performance for the KS equation with $L = 200, \lambda = 100$ (a): $\mu = 0$ and (b) $\mu = 0.01$. The red curve shows the RMSE curve when all $g = 64$ reservoirs are identical and have the same output weights. The blue curve shows the RMSE when the g parallel reservoirs are independently trained.

the red (identical weights) and blue (individually trained weights) curves in Fig 6(a), this works well. However, when $\mu = 0.01$, the method of identical weights fails as expected (Fig. 6(b)). Note that the Lyapunov spectrum for $\mu = 0.01$ is very close to the spectrum for $\mu = 0$ (see supplement).

Further details on the specific reservoir computer parameters, implementation and methods are given in the supplemental material at [*URL will be inserted by the publisher*] which includes Refs. [24, 25]. The additional material illustrates that the performance shown above is very robust, in that it changes little over wide ranges in the various parameters.

In conclusion, our results suggests that machine learning, and in particular reservoir computing, offers an effective potential means for model-free prediction of large spatiotemporally chaotic systems.

This work was supported by the U.S. Army Research Office under grant W911NF1210101. We thank D. Gauthier, A. Hartemink, and R. Brockett for useful comments.

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N.

Sainath, *et al.*, IEEE Signal Processing Magazine **29**, 82 (2012).
[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, Nature **529**, 484 (2016).
[3] H. Kantz and T. Schreiber, *Nonlinear time series analysis*, Vol. 7 (Cambridge university press, 2004).
[4] U. Parlitz and C. Merkwirth, Physical review letters **84**, 1890 (2000).
[5] H. Jaeger, Bonn, Germany: German National Research Center for Information Technology GMD Technical Report **148**, 13 (2001).
[6] W. Maass, T. Natschläger, and H. Markram, Neural computation **14**, 2531 (2002).
[7] M. Lukoševičius and H. Jaeger, Computer Science Review **3**, 127 (2009).
[8] H. Jaeger and H. Haas, Science **304**, 78 (2004).
[9] Y. LeCun, Y. Bengio, and G. Hinton, Nature **521**, 436 (2015).
[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning* (MIT press, 2016).
[11] S. Marzen, Physical Review E **96**, 032308 (2017).
[12] M. Inubushi and K. Yoshimura, Scientific Reports **7**, 10199 (2017).
[13] S. Hochreiter and J. Schmidhuber, Neural computation **9**, 1735 (1997).
[14] Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, Chaos: An Interdisciplinary Journal of Nonlinear Science **27**, 041102 (2017).
[15] X. Yan and X. Su, *Linear regression analysis: theory and computing* (World Scientific, 2009).
[16] We believe that this may be due to the odd symmetry of the tanh function: With $\mathbf{P}_2 = 0$, the setup in Fig. 1(b) is such that, if $\mathbf{r}(t)$ is an attracting reservoir orbit with output $\mathbf{v}(t)$ for $y(x,t)$, then $-\mathbf{r}(t)$ is also an attracting reservoir orbit and corresponds to an output $-\mathbf{v}(t)$. Thus with $\mathbf{P}_2 = 0$ the reservoir dynamics has a symmetry in conflict with the KS equation which is *not* invariant to the change $y \to -y$. Having $\mathbf{P}_2 \neq 0$ breaks this unwanted reservoir symmetry.
[17] K. Vandoorne, J. Dambre, D. Verstraeten, B. Schrauwen, and P. Bienstman, IEEE transactions on neural networks **22**, 1469 (2011).
[18] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, Nature communications **4**, 1364 (2013).
[19] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V. S. Udaltsov, Y. K. Chembo, and M. Jacquot, Phys. Rev. X **7**, 011015 (2017).
[20] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, Nature communications **2**, 468 (2011).
[21] N. D. Haynes, M. C. Soriano, D. P. Rosin, I. Fischer, and D. J. Gauthier, Physical Review E **91**, 020801 (2015).
[22] P. Manneville, Macroscopic Modelling of Turbulent Flows , 319 (1985).
[23] J. L. Kaplan and J. A. Yorke, in *Functional Differential equations and approximation of fixed points* (Springer, 1979) pp. 204–227.
[24] N. Tikhonov, Andreĭ, V. I. Arsenin, and F. John, *Solutions of ill-posed problems*, Vol. 14 (Winston Washington, DC, 1977).
[25] A.-K. Kassam and L. N. Trefethen, SIAM Journal on Scientific Computing **26**, 1214 (2005).