Reservoir computing systems, a class of recurrent neural networks, have recently been exploited for model-free, data-based prediction of the state evolution of a variety of chaotic dynamical systems. The prediction horizon demonstrated has been about half dozen Lyapunov time. Is it possible to significantly extend the prediction time beyond what has been achieved so far? We articulate a scheme incorporating time-dependent but sparse data inputs into reservoir computing and demonstrate that such rare “updates” of the actual state practically enable an arbitrarily long prediction horizon for a variety of chaotic systems. A physical understanding based on the theory of temporal synchronization is developed.

A recently emerged interdisciplinary field is machine-learning based, model-free prediction of the state evolution of nonlinear/chaotic dynamical systems [1–12]. A paradigm that has been exploited is reservoir computing [13–16], a class of recurrent neural networks. Starting from the same initial condition, a well-trained reservoir system can generate a trajectory that stays close to that of the target system for a finite amount of time, realizing short-term prediction. Because of the hallmark of chaos - sensitive dependence on initial conditions, the solution of the reservoir system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.

Is it possible to extend significantly the prediction horizon of reservoir computing? We provide an affirmative answer in this paper. The key observation is that, after training, prediction is enabled because the neural network system can replicate the dynamical evolution of the target system (or synchronize with it) but only for a transient period of time. In the conventional scheme, data from the target system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.

Is it possible to extend significantly the prediction horizon of reservoir computing? We provide an affirmative answer in this paper. The key observation is that, after training, prediction is enabled because the neural network system can replicate the dynamical evolution of the target system (or synchronize with it) but only for a transient period of time. In the conventional scheme, data from the target system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.

Is it possible to extend significantly the prediction horizon of reservoir computing? We provide an affirmative answer in this paper. The key observation is that, after training, prediction is enabled because the neural network system can replicate the dynamical evolution of the target system (or synchronize with it) but only for a transient period of time. In the conventional scheme, data from the target system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.

Is it possible to extend significantly the prediction horizon of reservoir computing? We provide an affirmative answer in this paper. The key observation is that, after training, prediction is enabled because the neural network system can replicate the dynamical evolution of the target system (or synchronize with it) but only for a transient period of time. In the conventional scheme, data from the target system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.

Is it possible to extend significantly the prediction horizon of reservoir computing? We provide an affirmative answer in this paper. The key observation is that, after training, prediction is enabled because the neural network system can replicate the dynamical evolution of the target system (or synchronize with it) but only for a transient period of time. In the conventional scheme, data from the target system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.

Is it possible to extend significantly the prediction horizon of reservoir computing? We provide an affirmative answer in this paper. The key observation is that, after training, prediction is enabled because the neural network system can replicate the dynamical evolution of the target system (or synchronize with it) but only for a transient period of time. In the conventional scheme, data from the target system will diverge from that of the original system exponentially. Nonetheless, if training is done properly so that the single-step error is orders-of-magnitude smaller than the oscillation range of the chaotic signal [12], accurate prediction can be achieved in short time. So far, the prediction horizon achieved is about five or six Lyapunov time [7], where one Lyapunov time is the inverse of the maximum Lyapunov exponent.
data from a component is fed into the corresponding node. The state of the reservoir network is updated according to some nonlinear function, and the resulting state vector is sent to a reservoir-to-output (R/O) module whose function is opposite to that of the I/R module, i.e., to convert the high-dimensional vector of the reservoir back into a vector of the same low-dimension as that of the original data from the target system. The reservoir network, once chosen, is fixed, so all parameters associated with it are hyperparameters. Training is done through a relatively small set of adjustable parameters associated with the R/O module, which can be tuned (or “learned”) based on the available input data. During the training phase, the system is open as it requires input from the target system. After training, one feeds the output from R/O directly into the I/R module, closing the system. The system then evolves by itself. Some key numbers involved in reservoir computing for model-free prediction are as follows. For example, for the Kuramoto-Sivashinsky equation (KSE) with spatiotemporally chaotic solutions [7], the typical number of the spatial measurement sites (the dimension of the input vector) is 64 and the size of the reservoir network is about 5000. The number of parameters in the R/O module to be trained is 5000 × 64.

Our proposed reservoir computing system functioning in the prediction phase leading to arbitrarily long prediction horizon is shown schematically in Fig. 1 where the three modules inside the blue dashed box represent the conventional system and the one inside the red dashed box is a new module incorporating rare state updates. The I/R module is described by a matrix $W_{in}$, a $D_r \times D_{in}$ random matrix that maps a $D_{in}$-dimensional input vector $v$ into a $D_r$-dimensional vector $r(t)$, where $D_{in} \ll D_r$. The elements of $W_{in}$ are generated from a uniform distribution in $[-\sigma, \sigma]$. The reservoir is a complex network of $D_r$ nodes with average degree $d$, whose connecting topology is described by the $D_r \times D_r$ matrix $A$. (For simplicity, we choose it to be a directed random network.) A recent work [12] has revealed that successful training and finite-time prediction can be achieved if the spectral radius of the network is in a finite range, which can be adjusted by properly normalizing the link weights in the network. The R/I module is represented by a matrix $D_{out} \times D_r$, whose elements are parameters determined through training [3, 7]. Typically, we have $D_{out} = D_{in}$. Without any state update, in the prediction phase, the reservoir computing system is a self-evolving dynamical system described by $r(t + \Delta t) = tanh \left[ A \cdot r(t) + W_{in} \cdot v(t) \right] + v(t + \Delta t) = W_{out} \cdot f[r(t + \Delta t)]$, where $f(r)$ is the output function [3, 7]: $f_i(r) = r_i$ for odd $i$, $f_i(r) = r_i^2$ for even $i$, $i = 1, 2, \ldots, D_r$, respectively. Associated with the dynamical evolution of the reservoir system are two sets of dynamical variables: the high-dimensional reservoir state vector $r$ and the (typically) low-dimensional output vector $v$. The matrices $W_{in}$ and $A$ are pre-defined while $W_{out}$ is determined by training during which $v$ is replaced by the state vector $u$ of the target system. After training is completed, to set the initial condition for the reservoir network, one approach is to continue to use $u$ in place of $v$ but only for a few time steps, after which the reservoir system executes natural dynamical evolution by itself.

In the conventional scheme [1, 12], after withdrawing the true state vector $u$ so that the system is closed, real measurements are no longer used, resulting in a relatively short prediction horizon for chaotic systems because of the exponential divergence between the trajectories of the reservoir and true systems. Our idea, as shown in Fig. 1, is to update the reservoir state with sparsely sampled real state vector $u'$ before the divergence exceeds a pre-defined tolerance limit, i.e., the updates are needed only rarely. In particular, during the update, the input to the I/R module can be written as $v'(t) = v(t) + c(u'(t) - u(t))$, where $u'$ contains data at $t$ and $c$ is the coupling parameter. Most of the time during the evolution, we still have $v'(t) = v(t)$. Updating is effectively an on-off coupling process between the reservoir and the true systems, where the “on” phase is significantly more sparse than the “off” phase. Prediction can then be viewed as a synchronization process [11, 17] between the two systems that are coupled but only intermittently [18, 19].

We test the predictive power of our proposed reservoir computing scheme with a large number of chaotic systems. Here we present two examples of high-dimensional chaotic systems: KSE and the complex Ginzburg-Landau equation (cGLE). (Examples of a number of low-dimensional chaotic systems are presented in Appendices.) The KSE is $y_{\theta} + y_{xx} + y_{xxxx} = 0$, where $y(x,t)$ is a scalar field in the interval $x \in (0, L]$ with periodic boundaries. We divide the spatial domain into $M$ uniform subintervals. Figure 2(a) shows a typical spatiotemporal chaotic solution for $L = 22$, where the numerical integration parameters are $M = 64$ and $\Delta t = 0.25$, and the maximum Lyapunov exponent is $\Lambda_{max} \approx 0.05$. Thus, approximately 80 time steps correspond to one Lyapunov time. The $M$-dimensional data vector is fed into the reservoir computing system with parameters $D_{in} = D_{out} = M$, $D_r = 4992$, $\sigma = 1$, $d = 3$, and $\rho = 0.1$. In addition, to avoid overfitting of $W_{out}$ during the training process, we set the relevant bias parameter [17] to be $\eta = 1 \times 10^{-4}$. Figure 2(b) shows the difference between the evolution of the reservoir and true system, i.e., the prediction error (color coded). The prediction horizon is about five Lyapunov time.

We now instigate rare updates with the true data $y_{act}(x,t)$, where the coupling function is $y_{rc}(x,t) = y_{rc}(x,t) + c[y_{act}(x,t) - y_{rc}(x,t)]$ if there is data update at the space-time point $(x,t)$, otherwise we have $y_{rc}(x,t) = y_{rc}(x,t)$, where $y_{rc}$ is the output of the reservoir system. We divide the output time series into equal time intervals, each of $T$ time steps. In one interval, the actual data points are available for consecutive $T_0$ steps, and there is no update for the remaining $T - T_0$ steps. In the space, we select $M_t$ uniform spatial points from the total $M$ measurement points. For illustration, we let $T_0 = 1$ and $M_t = 64$, i.e., each component of the input vector to the reservoir system (corresponding to a distinct measurement point in space) receives one true data point every $T$ time steps. Figure 2(c-e) show the prediction error for $T = 240, 160$, and $80$ time steps, respectively, corresponding to three, two, and one Lyapunov time. For $T = 240$ (Fig. 2(c)), the error is reduced as compared the case of no data update (Fig. 2(b)) and exhibits intermittency.
implying an intermittently synchronous behavior between the reservoir and the true system. In this case, the time interval in which the error is below the tolerance emerges intermittently in time. As the updating becomes twice as frequent, the small-error intervals are generally enlarged, as shown in Fig. 2(d) for $T = 160$. Remarkable, for $T = 80$, the error is essentially zero in the whole time interval considered. This means that, insofar as a single true data point is used to update the reservoir system in every Lyapunov time, the prediction horizon becomes arbitrarily long!

To obtain a systematic picture of the prediction horizon, we calculate the average error $\delta e$ over a long time interval, e.g., about 100 Lyapunov time, in the parameter plane $(T_0, T)$ (for $T_0 \leq T$). Figure 3(a) shows, for $M_c = M/2 = 32$ (the number of spatial coupling channels) and $c = 2.0$, the error behavior. In most of the parameter region, the error is small: $\delta e < 0.01$. Long term prediction fails only for extremely rare updates, i.e., for small values of $T_0$ and large values of $T$. Fixing the value of $T_0$, we obtain the curves of $\delta e$ versus $T$. For $T_0 = 1$ (black line), $\delta e$ increases rapidly with $T$. As more data points are included in each update, e.g., as the value of $T_0$ is increased from two to ten, $\delta e$ decreases gradually. Figure 3(c) shows that, for $M_c = M = 64$, the error remains small in all cases, giving rise to a significantly augmented prediction horizon. Figure 3(d) shows the prediction error for $T$ up to 500 time steps. For a change from $T_0 = 1$ to $T_0 = 2$, error $\delta e$ decreases sharply. However, for $T_0 \geq 2$, the predictions error becomes saturated. Comparing Figs. 3(b) with 3(d), we observe a dramatic reduction in $\delta e$ for $M_c = 64$.

The cases considered so far are for rare but regular updates. What about random updates? We examine the parameter plane $(p_t, p_s)$, where $p_t$ is the probability of update in each time step and $p_s$ is the probability for a spatial point to receive updates. Figure 3(e) shows the average prediction error, which exhibits an approximately symmetric behavior with respect to $p_t$ and $p_s$. Figure 3(f) shows, for fixed $p_s = 0.5$ and...
FIG. 3. Behaviors of prediction error for KSE. (a) Prediction error in the parameter plane $(T_0, T)$ for $M_c = 32$ and $c = 2.0$. (b) For $M_c = 32$, prediction error $\delta e$ versus $T$ for different values of $T_0$. (c) Prediction error for $M_c = 64$ and $c = 1.0$. (d) For $M_c = 64$, $\delta e$ versus $T$ for different values of $T_0$. (e) For random updating, prediction error in the parameter plane $(p_t, p_s)$ for $c = 1.2$ and (f) $\delta e$ versus $p_t$ (or $T_0/T$).

$p_s = 1$ (corresponding to the cases of $M_c = 32$ and $M_c = 64$ with regular coupling, respectively), random updating yields somewhat larger errors than those with regular coupling.

We now demonstrate the predictive power of our reservoir computing scheme for the 1D cGLE in the regime of spatiotemporal chaos: $A_t = (1 + i\alpha)A_{xx} + A - (1 + i\beta)|A|^2 A$, where $A(x, t)$ is a complex field in the interval $x \in (-L/2, L/2)$ with the periodic boundary condition, $\alpha$ and $\beta$ are parameters. The cGLE is a general model for a variety of physical phenomena [20–22]. For $L = 18$, $\alpha = 2$ and $\beta = -2$, the 1D cGLE exhibits spatiotemporal chaos with the maximum Lyapunov exponent $\Lambda \approx 0.23$. 

FIG. 4. Long time prediction of the state evolution of one-dimensional cGLE and . The system size is $L = 18$ and the number of measurement sites is $M = 32$. (a) Spatiotemporal pattern of true state evolution. (b-e) The difference between the predicted and actual state evolution for $T = \infty$, $T = 195$, $T = 130$, and $T = 65$, respectively. (f) For regular updating, prediction error in the parameter plane $(T_0, T)$ for $M_c = 32$ and $c = 0.4$. (g) For random updating, prediction error in the parameter plane $(p_t, p_s)$ for $c = 0.4$.

We divide the whole interval into $M = 32$ equally spaced points - the measurement sites. For integration step $\Delta t =$
0.07, one Lyapunov time corresponds to about 65 time steps. Figure 4(a) shows the true spatiotemporal evolution pattern $A(x,t)$. Because the field $A(x,t)$ is complex, it is necessary to use both the real and imaginary parts of the field $A'(x,t)$ and imaginary $A''(x,t)$ parts for training the reservoir system. The parameters of the reservoir system are $D_{in} = D_{out} = 2M$, $D_r = 9984$, $\sigma = 1$, $d = 3$, $\rho = 0.1$, and $\eta = 2 \times 10^{-5}$. Without any update, the error between the predicted and true state evolution is shown in Fig. 4(b), where the prediction horizon is about five Lyapunov time. To introduce rare updates of the true state, we use the coupling scheme $A_{rc}^p(i)(x,t) = A_{rc}^p(i)(x,t) + 0.4[A_{act}^p(i)(x,t) - A_{rc}^p(i)(x,t)]$ if there is actual data or $A_{rc}^p(i)(x,t) = A_{rc}^p(i)(x,t)$ if there is no update. State updates, even being rare, reset the synchronization error from time to time. Of particular interest is the finding that, insofar as there is no update of a single data point within one Lyapunov time, the prediction horizon can be made arbitrarily long. The machine learning scheme proposed and studied here has the potential to extend significantly the application scope of reservoir computing in predicting complex dynamical systems.

The novelty of our work is threefold: (1) we have introduced intermittent data updates into machine learning, (2) we have demonstrated that long prediction time can be achieved, and (3) we have introduced the concept of temporal synchronization with on-off coupling to understand the working of the reservoir computing machine with state updating. Our work sheds new light in the working of reservoir computing in predicting the state evolution of chaotic systems. In particular, in the synchronization-based scenario, the predictability of reservoir computing is the result of its ability to synchronize with the target chaotic system for a finite amount of time. At the start of the prediction phase, the reservoir system has the same initial condition as the target system. Because of temporal synchronizability, the reservoir system is able to follow the target system for sometime before the synchronization error becomes significant. Without state updates, the time it takes for the error to grow to a predefined threshold value determines the prediction time. State updates, even being rare, reset the synchronization error from time to time, insofar as a new update is provided before the error exceeds the threshold.

In essence, the basic idea of our method is similar to that of data assimilation, where models and measurements (true state updates) are combined to generate accurate predictions with applications in, e.g., weather forecasting [29][31]. In a recent study, data assimilation and machine learning have been combined to emulate the Lorenz 96 model from sparse and noisy observations [32]. It is also noteworthy that, in predicting chaotic dynamical systems, an alternative machine-learning based framework is radial basis function networks - artificial neural networks employing radial basis functions as activation functions [33][34]. Given a set of inputs, such a network outputs a signal that is a linear combination of radial basis functions of the inputs, and the parameters of the artificial neurons are determined through training based on, e.g., the standard backpropagation scheme. The method has been demonstrated to be effective for low-dimensional chaotic systems such as the logistic map and the classic Lorenz chaotic oscillator [35][38]. Whether the method can be superior to reservoir computing in predicting spatiotemporal chaotic systems is an open question.
ACKNOWLEDGMENT

The first two authors contributed equally to this work. This work is supported by the Office of Naval Research through Grant No. N00014-16-1-2828.

Appendix A: Mechanism for reservoir computing to achieve arbitrarily long prediction horizon - a heuristic analysis based on synchronization

To understand the mechanism behind the realization of long prediction horizon in our proposed reservoir computing scheme with state updating, we resort to synchronization analysis. To facilitate the analysis, we apply our scheme to the chaotic Rössler oscillator given by [25]:

\[
\begin{align*}
\dot{x} &= -y - z, \\
\dot{y} &= x + 0.2y, \\
\dot{z} &= 0.2 + (x - 9.0)z.
\end{align*}
\]

We obtain the three time series \([x(t), y(t), z(t)]\) at the step size \(\Delta t = 0.05\). The values of the parameters of the reservoir computing system are \(D_{in} = D_{out} = 3, D_r = 600, \sigma = 0.15, d' = 0.2, \rho = 0.2, \) and \(\eta = 1 \times 10^{-7}\), where \(d'\) is the link density of the complex neural network. The elements of the \(D_r \times D_{in}\) input matrix \(W_{in}\) are generated from the uniform distribution \([-\sigma, \sigma]\). The non-zero elements of the \(D_r \times D_r\) matrix \(A\) are generated from the uniform distribution \([-1, 1]\). Figure 5(a) shows the prediction result with the conventional scheme, where the prediction horizon is about \(t \approx 75\) (corresponding to approximately 15 average cycles of oscillation). Figure 5(b) shows that, with our proposed scheme, the prediction horizon is practically infinite, where an update of the actual data of \(y_{act}\) is coupled into the system \((T_0 = 1, T = 50)\) once every 50 time steps. More specifically, in the iterative process of reservoir computing system, the input data \(y_{rc}\) is replaced by \(y'_{rc} = y_{rc} + c(y_{act} - y_{rc})\) every 50 time steps for \(c = 0.8\). Note that, in Fig. 5(b), only the true and predicted \(x\) time series are shown, but time series from the other two dynamical variables give essentially the same prediction result. Figure 5(c) shows the color-coded average prediction error \(\delta e\) in the parameter plane \((T_0, T)\), where there are multiple parameter regions in which the error is small. The patterns in Fig. 5(c) are reminiscent of the phenomenon of ragged synchronization in coupled chaotic oscillator systems [39, 40], suggesting the use of synchronization theory to understand the working mechanism of our articulated reservoir computing machine.

A well-trained reservoir computing system can be viewed as a high-dimensional replica of the target system. A previous calculation of the Lyapunov exponents of the reservoir dynamical network revealed that the first few exponents are indeed approximate values of the exponents of the true system, and the vast set of remaining exponents have large negative values [3]. This is anticipated as the large negative exponents are necessary to reduce the exceedingly high dimension of the reservoir network to the low-dimensional target system through a strong compression of the dynamics along vast majority of orthogonal directions in the phase space. For the chaotic Rössler system, the considerations suggest that the dynamics of the reservoir computing system be approximately described by

\[
\begin{align*}
\dot{x}_{rc} &= -y_{rc} - z_{rc}, \\
\dot{y}_{rc} &= x_{rc} + 0.2y_{rc} + \varepsilon(t)(y_{act} - y_{rc}), \\
\dot{z}_{rc} &= 0.2 + (x_{rc} - 9)z_{rc},
\end{align*}
\]

where, in the coupling term \(\varepsilon(t)(y_{act} - y_{rc})\), \(\varepsilon(t)\) specifies the on-off nature of the coupling: \(\varepsilon(t) = \varepsilon\) if \(nT\Delta t < t < T_0\Delta t + nT\Delta t\) \((n = 0, 1, 2, \ldots)\) and \(\varepsilon(t) = 0\) otherwise.

There is a linear relation between \(\varepsilon\) and \(c\): \(c = \varepsilon\Delta t\). The data of \(y_{act}\) are generated from the target system Eq. (A1). In our scheme, predictability implies synchronization between the reduced reservoir computing system Eq. (A2) and the true system Eq. (A1) with only on-off coupling, where the “on” phase is typically significantly shorter than the “off” phase.

We use stability analysis to quantify synchronization. Let \(\delta x = x_{rc} - x_{act}\), \(\delta y = y_{rc} - y_{act}\), and \(\delta z = z_{rc} - z_{act}\) be the infinitesimal perturbations transverse to the synchronization manifold. The variational equations can be obtained by linearizing Eq. (A2) with respect to the true state of the target
We obtain the time series of $\Lambda$ step size prediction horizon with rare data updating. Combining Eqs. (A1) and (A3), we calculate the maximum transverse Lyapunov exponent $\Lambda$. Stable synchronization requires $\Lambda < 0$. Figure 5(d) shows the color-coded value of $\Lambda$ in the parameter plane ($T_0, T$), which exhibits typical features of ragged synchronization [39, 40]. Comparing Fig. 5(c) with Fig. 5(d), we observe a striking degree of similarity, indicating synchronization between the reservoir computing and the target systems subject to on-off coupling as the dynamical mechanism responsible for realizing the long prediction horizon with rare data updating.

**FIG. 6.** Predicting the chaotic Lorenz system with rare state updating. (a) Prediction with the conventional reservoir computing scheme without state updating, where the prediction horizon is about ten oscillations. (b) Practically infinite prediction horizon achieved with sparse input of actual data for $T_0 = 1$, $T = 40$, and $c = 0.8$. (c,d) The difference between the predicted and true time series corresponding to the results in (a,b), respectively.

**Appendix B: Predicting chaotic Lorenz system**

We demonstrate that a practically infinite prediction horizon can be achieved for the classic chaotic Lorenz system [26] with rare state updating. The equations of the system are

$$
\begin{align*}
\dot{x} &= 10(y - x), \\
\dot{y} &= x(28 - z) - y, \\
\dot{z} &= xy - 8/3z.
\end{align*}
$$

We obtain the time series $x(t)$, $y(t)$, and $z(t)$ with integration step size $\Delta t = 0.01$. The parameter setting of the reservoir computing system is $D_{in} = D_{out} = 3$, $D_r = 600$, $\sigma = 0.1$, $d' = 0.3$, $\rho = 1.2$, and $\eta = 1 \times 10^{-5}$. Figure 6(a) shows the prediction result with the conventional scheme without any state updating, where the prediction horizon is $t \approx 5$ (corresponding to approximately ten average oscillations). Figure 6(b) shows that, with rare state updating ($T_0 = 1$ and $T = 40$) of one of the dynamical variables, mathematically represented as replacement of $y_{rc}$ by $y_{rc} + c(y_{act} - y_{rc})$ once every 40 time steps, practically an arbitrarily long prediction horizon can be achieved. Figures 6(c) and 6(d) show the difference between the predicted and true time series from the conventional and our proposed reservoir computing schemes, respectively. It can be seen that, with rare state updating, the prediction error is essentially zero for the time interval displayed, with relatively large errors occurring only at about a few dozen time steps (out of $2 \times 10^4$ time steps).

**Appendix C: Predicting chaotic Hindmarsh-Rose neuron**

We test our reservoir computing scheme with rare state updating for the chaotic Hindmarsh-Rose neuron model [27]:

$$
\dot{x} = y + 3x^2 - x^3 - z + 3.2, \\
\dot{y} = 1 - 5x^2 - y, \\
\dot{z} = -0.006z + 0.024(x + 1.6),
$$

where $x$ is the membrane potential, $y$ and $z$ are the transport rates of the fast and slow channels, respectively. The integration time step is $\Delta t = 0.1$. The three time series $x(t)$, $y(t)$, and $z(t)$ are used to train the reservoir computing system with parameters $D_{in} = D_{out} = 3$, $D_r = 600$, $\sigma = 0.6$, $d' = 0.2$, $\rho = 0.3$, and $\eta = 1 \times 10^{-7}$. Figure 7(a) shows the result with the conventional reservoir computing scheme without
The prediction errors corresponding to Figs. 7(a) and 7(b) are shown in Figs. 7(c) and 7(d), respectively.

Appendix D: Predicting population evolution in a chaotic food web

We demonstrate an arbitrarily long prediction horizon for the following chaotic food web system [28]:
\[
\begin{align*}
\dot{x} &= x - 0.2 \frac{xy}{1 + 0.05x}, \\
\dot{y} &= -y + 0.2 \frac{xy}{1 + 0.05x} - yz, \\
\dot{z} &= -10(z - 0.006) + yz,
\end{align*}
\]
where \(x, y\) and \(z\) represent vegetation, herbivores and predators, and the evolution displays uniform phase evolution but with chaotic amplitude modulation. The integration step size is \(\Delta t = 0.1\). The parameter values of the reservoir computing system are \(D_{in} = D_{out} = 3, D_c = 600, \sigma = 0.15, d' = 0.2, \rho = 0.2, \) and \(\eta = 1 \times 10^{-7}\). Figure 8(a) shows the prediction result from the conventional scheme without any state updating, where the prediction horizon is \(t \approx 50\) (containing seven or eight bursts in the predator population). In the food web system, vegetation data are relatively easy to be collected, so we use \(x_{act}\) to perform rare state updating. Figure 8(b) shows the predicted and actual predator time series for \(T_0 = 1\) and \(T = 50\), i.e., we replace \(x_{rc}\) by \(x'_{rc} = x_{rc} + c(x_{act} - x_{rc})\) once every 50 time steps. Visually the two types of time series cannot be distinguished. Figures 8(c) and 8(d) show the corresponding evolution of the prediction error for Figs. 8(a) and 8(b), respectively. We see that, with rare state updating, the prediction error is exceedingly small in the long time interval (10⁴ time steps) tested, indicating that a practically infinite prediction horizon has been achieved.
of network spectral radius,” Phys. Rev. Research 1, 033056 (2019).

[13] H. Jaeger, “The echo state approach to analysing and training recurrent neural networks-with an erratum note,” Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148, 13 (2001).

[14] W. Mass, T. Nachtschlaeger, and H. Markram, “Real-time computing without stable states: A new framework for neural computation based on perturbations,” Neur. Comp. 14, 2531 (2002).

[15] H. Jaeger and H. Haas, “Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication,” Science 304, 78 (2004).

[16] G. Manjunath and H. Jaeger, “Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks,” Neur. Comp. 25, 671 (2013).

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

[18] L. Chen, C. Qu, and H. B. Huang, “Synchronization with on-off coupling: Role of time scales in network dynamics,” Phys. Rev. E 79, 045101 (2009).

[19] S. Li, N. Sun, L. Chen, and X. Wang, “Network synchronization with periodic coupling,” Phys. Rev. E 98, 012304 (2018).

[20] I. S. Aranson and L. Kramer, “The world of the complex Ginzburg-Landau equation,” Rev. Mod. Phys. 74, 99 (2002).

[21] M. C. Cross and P. C. Hohenberg, “Pattern formation outside of equilibrium,” Rev. Mod. Phys. 65, 851 (1993).

[22] Y. Kuramoto, Chemical Oscillations, Waves and Turbulence (Springer, Berlin, 1984).

[23] J. F. Heagy, L. M. Pecora, and T. L. Carroll, “Short wavelength bifurcations and size instabilities in coupled oscillator systems,” Phys. Rev. Lett. 74, 4185 (1995).

[24] L. M. Pecora and T. L. Carroll, “Master stability functions for synchronized coupled systems,” Phys. Rev. Lett. 80, 2109 (1998).

[25] O. E. Rössler, “Equation for continuous chaos,” Phys. Lett. A 57, 397 (1976).

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

[27] J. L. Hindmarsh and R. M. Rose, “A model of neuronal bursting using three coupled first order differential equations,” Proc. Roy. Soc. London Ser. B Biol. Sci. 221, 87 (1984).

[28] B. Blassius, A. Huppert, and L. Stone, “Complex dynamics and phase synchronization in spatially extended ecological systems,” Nature 399, 354 (1999).

[29] D. J. Patil, B. R. Hunt, E. Kalnay, J. A. Yorke, and E. Ott, “Local low dimensionality of atmospheric dynamics,” Phys. Rev. Lett. 86, 5878 (2001).

[30] E. Ott, B. Hunt, I. Szunyogh, A. V. Zimin, E. J. Kostelich, M. Corazza, E. Kalnay, D. J. Patil, and J. A. Yorke, “A local ensemble Kalman filter for atmospheric data assimilation,” Tellus 56A, 415 (2004).

[31] I. Szunyogh, E. Kostelich, G. Gyarmati, D. J. Patil, B. R. Hunt, E. Kalnay, E. Ott, and J. A. Yorke, “Assessing a local ensemble Kalman filter: Perfect model experiments with the national centers for environmental prediction global model,” Tellus 57A, 528 (2005).

[32] J. Brajard, A. Carrassi, M. Bocquet, and L. Bertino, “Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: a case study with the Lorenz 96 model,” Geosci. Model Dev. Discuss. (2019).

[33] D. S. Broomhead and D. Lowe, “Multivariable functional interpolation and adaptive networks,” Complex Syst. 2, 321 (1988).

[34] T. Poggio and F. Girosi, “Networks for approximation and learning,” Proc. IEEE 78, 1484 (1990).

[35] A. Gholipour, B. N. Araabi, and C. Lucas, “Predicting chaotic time series using neural and neurofuzzy models: A comparative study,” Neural Process. Lett. 24, 217 (2006).

[36] D. Chen and W. Han, “Prediction of multivariate chaotic time series via radial basis function neural network,” Complexity 18, 55 (2013).

[37] M. K. Rafsanjani and M. Samareh, “Chaotic time series prediction by artificial neural networks,” J. Comp. Methods Sci. Eng. 16, 599 (2016).

[38] V. T. Nguyen and T. A. Duong, “Chaotic time series prediction using radial basis function networks,” in 2018 4th International Conference on Green Technology and Sustainable Development (GTSD) (2018).

[39] A. Stefański, P. Perlikowski, and T. Kapitaniak, “Ragged synchronizability of coupled oscillators,” Phys. Rev. E 75, 016210 (2007).

[40] P. Perlikowski, B. Jagiello, A. Stefanski, and T. Kapitaniak, “Experimental observation of ragged synchronizability,” Phys. Rev. E 78, 017203 (2008).