Journal of Nonlinear Science (2020) 30:713–735
https://doi.org/10.1007/s00332-019-09588-7

Journal of Nonlinear Science

Predicting Spatio-temporal Time Series Using Dimension Reduced Local States

Jonas Isensee[1,2] · George Datseris[1,2] · Ulrich Parlitz[1,2]

Received: 20 January 2019 / Accepted: 4 October 2019 / Published online: 26 October 2019

Abstract
We present a method for both cross-estimation and iterated time series prediction of spatio-temporal dynamics based on local modelling and dimension reduction techniques. Assuming homogeneity of the underlying dynamics, we construct delay coordinates of local states and then further reduce their dimensionality through Principle Component Analysis. The prediction uses nearest neighbour methods in the space of dimension reduced states to either cross-estimate or iteratively predict the future of a given frame. The effectiveness of this approach is shown for (noisy) data from a (cubic) Barkley model, the Bueno-Orovio–Cherry–Fenton model, and the Kuramoto–Sivashinsky model.

Keywords Data driven modelling · Nearest neighbours prediction · Spatio-temporal chaos

Mathematics Subject Classification 37N99

1 Introduction

In many experiments, some variables of the system are more easily observable than others. If the underlying dynamics is deterministic, in general, the observable of interest is nonlinearly related to other variables of the system which might be more accessible. In such cases, one may try to estimate any observable which is difficult to measure

Communicated by Dr. Paul Newton.

✉ Ulrich Parlitz
Ulrich.parlitz@ds.mpg.de

[1] Max Planck Institute for Dynamics and Self-Organization, Am Fassberg 17, 37077 Göttingen, Germany

[2] Institut für Dynamik komplexer Systeme, Georg-August-Universität Göttingen, Friedrich-Hund-Platz 1, 37077 Göttingen, Germany

from time series of those variables which are at one's disposal. Another task frequently encountered with observed time series is forecasting the dynamical evolution of the system and the time series. For both tasks, time series prediction methods have been devised using delay coordinates (Packard et al. 1980; Takens 1981; Sauer et al. 1991; Kantz and Schreiber 2004; Abarbanel 1996; Abarbanel et al. 1994; Bradley and Kantz 2015) and approximations of the flow in delay coordinate space, for example, using nearest neighbours methods (also called local modelling) (Farmer and Sidorowich 1987; Casdagli et al. 1992; Atkeson et al. 1997; Kugiumtzis et al. 1998; Mc Names et al. 1999; Engster and Parlitz 2006).

Here, we present an approach for cross-estimation and iterated time series prediction for multivariate time series from extended spatio-temporal systems which is based on (spatially) local delay coordinate maps, linear (PCA) dimension reduction, and nearest neighbour methods for local modelling.

Local delay coordinate maps (Parlitz 1998; Parlitz and Merkwirth 2000; Mandelj et al. 2001; Coca and Billings 2001; Mandelj et al. 2004; Guo and Billings 2007) are motivated by the fact that it often is impractical to predict the behaviour of systems with a large spatial extent all at once. If instead one combines a spatial and temporal neighbourhood around each measurement to find a description of the local system state, it becomes possible to make predictions for each point in space independently. For performing cross-estimation or prediction based on local states, one can either use nearest neighbours methods (Parlitz and Merkwirth 2000) or employ some other black-box modelling approach like, for example, echo state machines (Pathak et al. 2018; Zimmermann and Parlitz 2018). In the following, we shall use local modelling by selecting for each local delay coordinate vector similar vectors from a training data set whose relations to other observables and/or future temporal evolutions are known and can be exploited for cross-estimation or time series prediction.

Successful modelling of high-dimensional dynamics in extended systems, however, requires very large embedding dimensions which is a major challenge in particular for nearest neighbour methods. Therefore, a crucial point in making the conceptually simple nearest neighbours algorithm performant is dimension reduction. As a means of dimension reduction to find lower a dimensional representation of the local states, we employ Principle Component Analysis (PCA) which turns out to improve performance in particular for noisy data.

2 Predicting Spatio-temporal Time Series

In this section, we shall introduce the main concepts for predicting spatio-temporal time series, including local delay coordinate maps, linear dimension reduction, and nearest neighbours methods for local modelling of the dynamical evolution or any other relation between observed time series.

2.1 Local Modelling

Let $\mathbf{x}_t$ be a state of some dynamical system evolving in time t and let us assume that the dynamical equations generating the flow in state space are unknown, but only a set

$\mathcal{S}$ of M states $\mathbf{x}_{t_m}$ is available, for which also future values $\mathbf{x}_{t_m+T}$ are known (due to previous measurements, for example). This data set $\mathcal{S}$ can be used to predict the future value $\mathbf{x}_{t+T}$ of a given state $\mathbf{x}_t$ by selecting the k nearest neighbours $\mathbf{x}_{t_i}$ ($i \in \{1, ..., M\}$) of $\mathbf{x}_t$ in $\mathcal{S}$ and using their future values $\mathbf{x}_{t_i+T}$ for approximating $\mathbf{x}_{t+T}$, for example, by (distance weighted) averaging. In the following numerical examples, we use the average with weights

$$\omega_i = \left(1.1 - \left(\frac{d_i}{d_{max}}\right)^2\right)^4 \quad \text{and} \quad \omega_i = 1 \quad \text{if} \quad d_{max} = 0$$

where d_i are the euclidean distances of the k neighbours $\mathbf{x}_{t_i}$ to the query $\mathbf{x}_t$ and d_{max} is their maximum. The prediction is then given by

$$\hat{\mathbf{x}}_{t+T} = \frac{\sum_{i=1}^{k} \omega_i x_{t_i+T}}{\sum_{i=1}^{k} \omega_i}. \tag{1}$$

In most practical applications of this kind of local nearest neighbour modelling, the required states are reconstructed from a measured time series using the concept of delay coordinates (to be introduced in the next section). Local modelling in delay coordinate space is a powerful tool for purely data-driven time series prediction (Farmer and Sidorowich 1987; Casdagli et al. 1992; Atkeson et al. 1997; Kugiumtzis et al. 1998; Mc Names et al. 1999; Engster and Parlitz 2006). Its main ingredients are a proper state-space representation of the measured time series, fast nearest neighbour searches, and local models such as low order polynomials which can accurately interpolate and predict the (nonlinear) relation between (reconstructed) states and target values.

2.2 Delay Coordinates

The most important part of time series-based local modelling is the representation of data, i.e. proper reconstruction of states from data. Typically this representation is found utilizing delay coordinates and Takens' Embedding Theorem (Packard et al. 1980; Takens 1981; Sauer et al. 1991; Kantz and Schreiber 2004; Abarbanel 1996; Bradley and Kantz 2015) such that a scalar time series $\{s_t\}$ with $t \in \mathbb{N}$ is reconstructed to state vectors

$$\mathbf{x}_t = (s_{t-\gamma\tau}, \ldots, s_{t-\tau}, s_t) \in \mathbb{R}^{\gamma+1}$$

by including γ past measurements each separated by τ time steps.[1] These reconstructed state vectors $\mathbf{x}_t$ can then, for example, be used for predicting the (future) time series value s_{t+1} using the nearest neighbours method discussed in the previous Sect. 2.1. To do so, a training set of reconstructed states is generated whose (short term) future evolution one time step ahead is known. Then k nearest neighbours of the current (reference) state $\mathbf{x}_t$ are selected from this training set and the corresponding time series values one step ahead are used to estimate s_{t+1}.

[1] Here and in the following, we assume that the time series is sampled with some sampling time t_{sample} and integers t are used as time index.

For multivariate time series $\{\mathbf{s}_t\}$, one can do the same for each of the components $s_{i,t}$ resulting in $d(\gamma + 1)$ dimensional state vectors

$$\mathbf{x}_t = (s_{1,t-\gamma\tau}, \ldots, s_{1,t}, \ldots, s_{d,t-\gamma\tau}, \ldots, s_{d,t}).$$

where d is the number of observables.

2.3 Spatial Embedding

In principle, delay embedding could also be employed to reconstruct (global) states of high-dimensional spatially extended systems using multivariate time series sampled at many spatial locations. Such global state vectors are (and have to be) very high dimensional, in particular, for systems exhibiting extensive chaos where the attractor dimension increases with size of the domain of the system [see for example Lilienkamp et al. (2017) and references therein]. The runtime of nearest neighbour searches, however, and particularly the memory usage of such reconstructions grows rapidly with the dimension of the reconstructed global states. Furthermore, and even more important is the fact that (with a finite number of data) the density of points becomes very low and (Euclidean) distances between points tend to be all the same. These issues are also called "curse of dimensionality" and to avoid them it has been proposed (Parlitz 1998; Parlitz and Merkwirth 2000; Mandelj et al. 2001; Coca and Billings 2001; Mandelj et al. 2004; Guo and Billings 2007) to reconstruct (relatively) low-dimensional spatially local states and to use them to predict spatially extended systems point by point instead of the whole global state at once. This approach is motivated by the fact that most spatially extended physical systems posses a finite speed at which information travels. Therefore, the future value of any of the variables depends solely on its past and its spatial neighbours.[2] Instead of trying to describe the state of the whole system in one vector, we limit ourselves to include small neighbourhoods of all points that carry enough information to predict one point one time step into the future. As an additional benefit, the unfeasibly large embedding dimension that would result from embedding the entire space into a single state is greatly reduced. The idea of local delay coordinate spaces was first applied to spatially one-dimensional systems (Parlitz 1998; Parlitz and Merkwirth 2000; Mandelj et al. 2001) and was used, for example, to anticipate extreme events in extended excitable systems (Bialonski et al. 2015).

In the following, we will present the embedding procedure for spatio-temporal time series represented by $u_{t,\alpha}$, where t denotes time and α a point in space. For 2D space, α takes the values $\alpha = (i, j) \quad \forall 1 \leq i \leq N_x,\ 1 \leq j \leq N_y$.

In the most general case, such a local delay coordinate vector could consist of arbitrary combinations of neighbours in all directions of space and time. For practical purposes, we will limit ourselves to a certain set of parameters to describe which neighbours will be included into a local delay coordinate map. We parameterize the map with the number γ of past time steps and their respective temporal delay (or time lag) τ. All neighbouring grid points in space that are within the radius r, referring to the Euclidean distances in a unit grid, will be included as well. For each included

[2] The situation is different, if additional long-range connections exist linking remote locations.

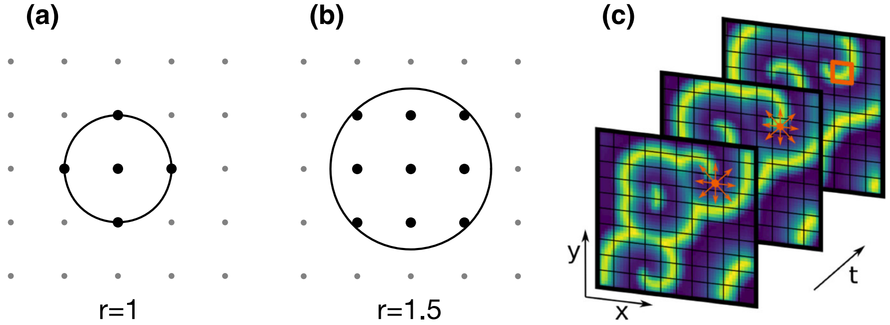

Fig. 1 Visualization of spatial regions included in a local delay coordinate vector. **a**, **b** illustrate the size of neighbourhood for radii $r = 1$ and $r = 1.5$, respectively, where all points within the circle spanned by r are included in the vector. **c** illustrates how data from spatial neighbours at different time steps are combined to predict a future value of the data array (indicated by the pixel framed in red) (Color figure online)

time step, this amounts to $d_r = |\{\alpha \in \mathbb{Z}^2 : ||\alpha||_2 \leq r\}|$ points. The resulting shape of the map is comparable to a cylinder in 2+1 dimensional space–time with dimension $D_\mathrm{E} = (\gamma + 1)d_r$. To make this clearer, a visualization of the spatially local delay coordinate vector in a two-dimensional system is displayed in Fig. 1 for different radii r.

In the following, we shall assume that the dynamics underlying the observed spatio-temporal time series is invariant with respect to translations, i.e. that the system is homogeneous. In this case, local delay coordinate vectors from different points in space can be combined to a single training set providing the database for cross-estimation or time series prediction as will be discussed in more detail in Sect. 2.4. However, even if the dynamical rules are the same for all locations, special care needs to be taken at the boundaries. This becomes obvious when trying to include non-existent neighbours from outside the grid. For periodic boundary conditions, the canonical solution is to wrap around at the edges, but for constant boundaries, the solution is not so obvious. In many cases, the effective dynamics near the boundary may also differ from dynamics far from it. It is therefore desirable to treat boundaries separately during nearest neighbour predictions. A solution proposed in Parlitz and Merkwirth (2000) is to artificially enlarge the domain of the system by a boundary region with chosen constant value. The missing spatial neighbours outside the original domain are thus replaced by the constant when generating the local delay coordinate vectors. If the chosen constant is significantly larger than typical values of the internal dynamics, the state vectors from the boundary fill regions in delay coordinate space isolated from state vectors of internal dynamics. This has the desired effect as nearest neighbour searches will always find boundary states when given a boundary state as query and similarly for internal states.

2.4 Dimension Reduction

The feasibility of any nearest neighbour search depends heavily on the memory consumption because N points of dimension $D_\mathrm{E} = (\gamma + 1)d_r$ need to be stored in memory.

A crucial part of our algorithm is therefore about creating a proper low-dimensional representation. Limiting the range of parameters γ and r to produce low-dimensional states is a severe restriction and gives poor predictions for the systems that are used in the following. Therefore, instead of choosing a small dimension for the local delay coordinate map from the start, we propose to perform some means of dimension reduction on the resulting local delay coordinate vectors. For this task, we use Principal Component Analysis (PCA) as it is a straight-forward standard technique for (linear) dimension reduction, where the vectors $\mathbf{x}_t$ are projected onto the eigenvectors of the covariance matrix corresponding to the largest eigenvalues (Gareth et al. 2015). In the field of nonlinear time series analysis, PCA has first been used by Broomhead and King (1986) who suggested to use dimension reduction applied to high-dimensional delay reconstructions with time series densely sampled in time.

Let $\{\mathbf{x}^n\}$ be the set of all N local delay coordinate vectors $\mathbf{x}^n = (x_1^n, \dots, x_{D_\mathrm{E}}^n) \in \mathbb{R}^{D_\mathrm{E}}$ (at different times t and locations α, assuming stationary and spatially homogeneous dynamical rules). To perform PCA first mean values, $\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n = (\bar{x}_1, \dots, \bar{x}_{D_\mathrm{E}})$ with $\bar{x}_i = \frac{1}{N}\sum_{n=1}^{N} x_i^n$ are subtracted resulting in shifted states $\tilde{\mathbf{x}}^n = \mathbf{x}^n - \bar{\mathbf{x}} = (\tilde{x}_1^n, \dots, \tilde{x}_{D_\mathrm{E}}^n)$. The covariance matrix

$$
\begin{aligned}
C_X &= \frac{1}{N}\sum_{n=1}^{N} (\tilde{\mathbf{x}}^n)^{\mathrm{tr}} \cdot \tilde{\mathbf{x}}^n \\
&= \frac{1}{N}\begin{bmatrix}
\sum_{n=1}^{N} \tilde{x}_1^n \tilde{x}_1^n & \sum_{n=1}^{N} \tilde{x}_1^n \tilde{x}_2^n & \dots & \sum_{n=1}^{N} \tilde{x}_1^n \tilde{x}_{D_\mathrm{E}}^n \\
\sum_{n=1}^{N} \tilde{x}_2^n \tilde{x}_1^n & \sum_{n=1}^{N} \tilde{x}_2^n \tilde{x}_2^n & \dots & \sum_{n=1}^{N} \tilde{x}_2^n \tilde{x}_{D_\mathrm{E}}^n \\
\vdots & \vdots & \ddots & \vdots \\
\sum_{n=1}^{N} \tilde{x}_{D_\mathrm{E}}^n \tilde{x}_1^n & \sum_{n=1}^{N} \tilde{x}_{D_\mathrm{E}}^n \tilde{x}_2^n & \dots & \sum_{n=1}^{N} \tilde{x}_{D_\mathrm{E}}^n \tilde{x}_{D_\mathrm{E}}^n
\end{bmatrix}
\end{aligned}
$$

is computed by iteratively producing individual local delay coordinate vectors $\tilde{\mathbf{x}}^n$ from the dataset and summing the terms $(\tilde{\mathbf{x}}^n)^{\mathrm{tr}} \cdot \tilde{\mathbf{x}}^n$ into the preallocated matrix C_X (here x^{tr} stands for the transpose operation).

Local states $\mathbf{y}^n$ of lower dimension $D_\mathrm{R} \leq D_\mathrm{E}$ are obtained by projecting the shifted states $\tilde{\mathbf{x}}$

$$
\mathbf{y}^n = P\tilde{\mathbf{x}}^n
$$

using a (globally valid) $D_\mathrm{R} \times D_\mathrm{E}$ projection matrix P whose rows are given by the D_R eigenvectors of the matrix C_X corresponding to the largest D_R eigenvalues. The dimensionality D_R of the subspace spanned by eigenvectors to be taken into account can either be set explicitly or determined such that some percentage such as 99% of the original variance of the local delay coordinate vectors is preserved.

The whole data set can thus be mapped into the space with reduced dimension D_R by mapping each point of the data set into the high-dimensional space $\mathbb{R}^{D_\mathrm{E}}$ and projecting it into the lower dimensional space $\mathbb{R}^{D_\mathrm{R}}$ using the PCA projection matrix P computed beforehand. For the subsequent prediction process, the projected local delay coordinate vectors $\mathbf{y}^n$ are then fed into a tree structure such as a *k*–*d* tree (Bentley 1975; Carlsson 2018) for fast nearest neighbour searching.

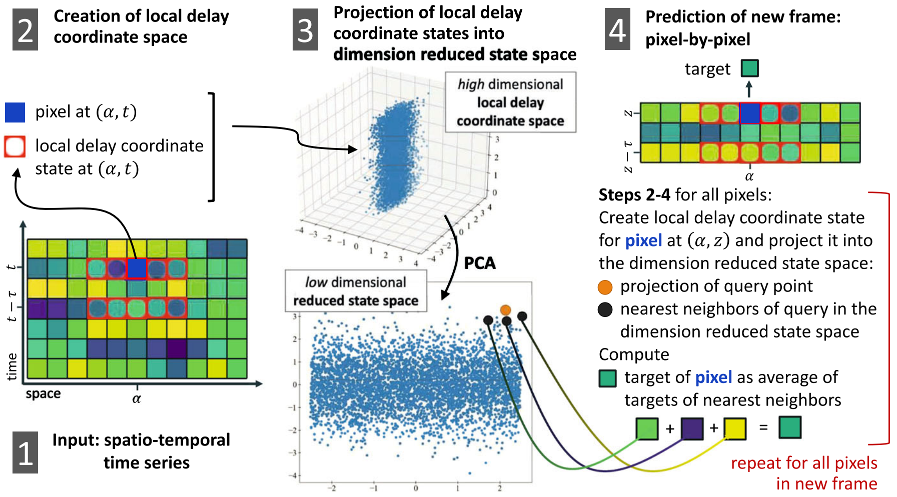
2 Creation of local delay coordinate space
3 Projection of local delay coordinate states into dimension reduced state space
4 Prediction of new frame: pixel-by-pixel
pixel at (α, t)
local delay coordinate state at (α, t)
high dimensional local delay coordinate space
PCA
low dimensional reduced state space
target
z
$z-\tau$
α
t
$t-\tau$
time
space
α
Steps 2-4 for all pixels:
Create local delay coordinate state for pixel at (α, z) and project it into the dimension reduced state space:
projection of query point
nearest neighbors of query in the dimension reduced state space
Compute
target of pixel as average of targets of nearest neighbors
+ + =
repeat for all pixels in new frame
1 Input: spatio-temporal time series

Fig. 2 Overview of the prediction algorithm. After sampling the input data in step 1 local delay coordinate vectors are created in step 2 for each pixel at location α and time t. Then, in step 3, using PCA the local delay coordinate vectors are projected into a lower dimensional reduced state space where in step 4 neighbours of (projections of) given query points provide target values that are used to approximate the target of the query point [here using weighted averaging over target values, like in Eq. (1)]. Targets can be future pixels of the same field or pixels of some other fields related to the input data. The projection matrix of the PCA and the k–d tree for searching nearest neighbours in the reduced state space are computed before based on a training data set

One issue arises with points near boundaries. Since the dynamics close to the boundaries may differ from the rest of the system, they were separated from other local delay coordinate vectors in phase space. This was achieved by setting the non-existent neighbours of boundary points to a large constant value (Parlitz and Merkwirth 2000). The power of PCA however relies on its assumption of a single cloud of points in (state) space within or close to a low-dimensional linear subspace. This is no longer the case when constant boundaries come into play. To sidestep this issue, we suggest changing the second step of the procedure described above. Simply exclude all boundary states from the computation of the projection matrix P but project them with the resulting matrix P nonetheless. In principle, this could eliminate the offset meant to separate internal and boundary dynamics but in practice the projection matrices rarely posses zero-valued entries. Therefore, it is highly unlikely that this would become a problem as long as boundary offset values are chosen large enough.

2.5 Prediction Algorithm

An overview of the prediction algorithm is provided in Fig. 2. While the dimension of the local delay coordinate space has changed in the dimension reduction process, the ordering of the vectors $(t, \alpha) \leftrightarrow n$ within the data set of dimension $\mathbb{R}^{D_\mathrm{R}}$ and the search tree remained unaffected and is thus known. It is therefore sufficient to find the indices of nearest neighbours for a given query. To make predictions, we assign each local delay coordinate vector $\mathbf{x}_{t,\alpha}$ a target value from the original training data and the only difference between temporal prediction and cross-estimation lies in the choice of these target values.

For time series prediction, we choose $\mathbf{x}_{t,\alpha} \to u_{t+1,\alpha}$ where $\mathbf{x}_{t,\alpha}$ are the local delay coordinate vectors from the spatio-temporal time series $\{u_{t,\alpha}\}$ and $u_{t+1,\alpha}$ target values. The prediction process then consists of producing vectors $\mathbf{x}_{T,\alpha}$ from the end of the time series by applying the same local delay coordinate map, subsequent dimension reduction using the projection matrix P that was computed for the training set, and local nearest neighbour modelling providing the target values $u_{T+1,\alpha}$. Once a prediction for each point (denoted by α) has been made, all future values $u_{T+1,\alpha}$ of the (input) field u are known and the procedure can be repeated for predicting $u_{T+2,\alpha}$. Using this kind of iterated prediction, spatio-temporal time series can, in principle, be forecasted for any period of time (with the well known limits of predictability of chaotic dynamics).

The case of cross-estimation is even simpler than time series prediction. Here, we are given a training set of two fields: an input variable $u_{t,\alpha}$ and a target variable $v_{t,\alpha}$. The values of the input field $u_{t,\alpha}$ are mapped into local delay coordinate vectors $\mathbf{x}_{t,\alpha}$. Using PCA and nearest neighbours search, we find similar instances in the training set for which the corresponding values of the target variables are known and can be used for estimating the current target $v_{t,\alpha}$.

2.6 Error Measures

In Sect. 4, we will test the presented prediction methods on the model systems described in Sect. 3. For evaluation, we compare any predicted field $\hat{v}$ with the corresponding correct values (i.e. test values) $\check{v}$ by considering spatial averages of the quadratic error over all sites α. This so-called Mean Squared Error (MSE) is then normalized by the MSE obtained when using the (spatial) mean value $\bar{v}$ for prediction. The resulting Normalized Mean Squared Error (NRMSE) is defined as

$$\mathrm{NRMSE}(\check{v}, \hat{v}) = \sqrt{\frac{\mathrm{MSE}(\check{v}, \hat{v})}{\mathrm{MSE}(\check{v}, \bar{v})}}, \qquad \text{where} \qquad \mathrm{MSE}(\check{v}, \hat{v}) = \frac{1}{A} \sum_{\alpha} \left(\check{v}_\alpha - \hat{v}_\alpha\right)^2 \tag{2}$$

where A is the number of spatial sites α taken into account. Any good estimate or forecast should be (much) better than the trivial prediction using mean values and result in NRMSE values (much) smaller than one.

2.7 Software

All software used in this paper has been published in the form of an open source software library under the name of *TimeseriesPrediction.jl* (https://github.com/JuliaDynamics/TimeseriesPrediction.jl) along with extensive documentation and various examples. It is written using the programming language Julia (Bezanson et al. 2017) with extensibility in mind, such that it is compatible with different spatial dimensions as well as arbitrary spatio-temporal delay coordinate maps. This is made possible through a modular design and Julia's multiple dispatch.

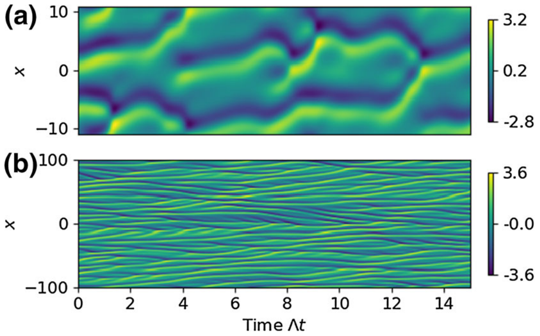

Fig. 3 Temporal evolution of the KS model (3) for two different system sizes. Pane **a** has parameters $L = 22$ and $Q = 64$, while the larger system **b** has $L = 200$ and $Q = 512$

3 Model Systems

The Kuramoto–Sivashinsky (KS) model (Kuramoto 1978; Sivashinsky 1980, 1988) has been devised for modelling flame fronts and will in our case be used as a benchmark system for iterated time series prediction. The Barkley model (Barkley 1991) describes an excitable medium that shows chaotic interplay of travelling waves. The third and most complex model is the Bueno-Orovio–Cherry–Fenton (BOCF) model (Bueno-Orovio et al. 2008), which is composed of four coupled fields describing electrical excitation waves in the heart muscle.

3.1 Kuramoto–Sivashinsky System

The Kuramoto–Sivashinsky (KS) system (Kuramoto 1978; Sivashinsky 1980, 1988) is defined by the following partial differential equation:

$$\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^4 u}{\partial x^4} + \left| \frac{\partial u}{\partial x} \right|^2 = 0 \tag{3}$$

typically integrated with periodic boundary conditions. It is widely used in literature (Parlitz and Merkwirth 2000; Pathak et al. 2018) because it is a simple system consisting of just one field while still showing high-dimensional chaotic dynamics.

The dynamics were simulated with an EDTRK4 algorithm (Rackauckas and Nie 2017) and the parameters for integration are the time step $\Delta t = 0.25$ and the system size L with spatial sampling Q. Two example evolutions with $L = 22$, $Q = 64$ and $L = 200$, $Q = 512$ are shown in Fig. 3.

3.2 Barkley Model

The Barkley model (Barkley 1991) is a simple system that exhibits excitable dynamics. We will use a modification with a cubic term u^3 in the differential equation of the v

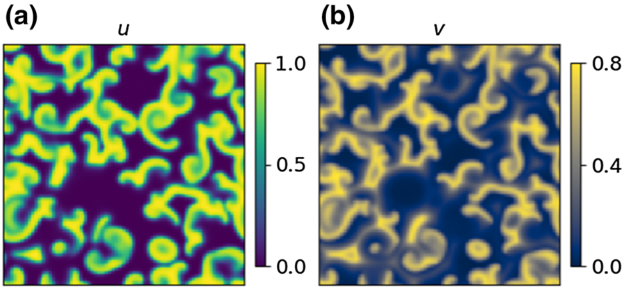

Fig. 4 Snapshot of the chaotic Barkley model (4) on a grid of size 150 × 150 with constant boundary conditions and after transients decayed. The u variable is displayed in (**a**) and v in (**b**)

variable that can be used to generate spatio-temporal chaos such that:

$$\begin{aligned}\frac{\partial u}{\partial t} &= \frac{1}{\varepsilon}u(1-u)\left(u-\frac{v+b}{a}\right)+D\nabla^2 u\\ \frac{\partial v}{\partial t} &= u^3 - v,\end{aligned} \tag{4}$$

where the parameter set $a = 0.75$, $b = 0.06$, $\varepsilon = 0.08$ and $D = 0.02$ leads to chaotic behaviour. For integration we used $\Delta t = 0.01$ and $\Delta x = 0.1$ in combination with an optimized FTCS scheme like the one described in Barkley (1991) (Fig. 4).

3.3 Bueno-Orovio–Cherry–Fenton Model

The Bueno-Orovio–Cherry–Fenton (BOCF) model (Bueno-Orovio et al. 2008) is a more advanced set of equations that serves as a realistic but relatively simple model of (chaotic) cardiac dynamics. It consists of four coupled fields that can be integrated as PDEs on various geometries. For the sake of simplicity, we consider a two-dimensional square. The four variables u, v, w, s are given by the following equations:

$$\begin{aligned}\frac{\partial u}{\partial t} &= D\cdot\nabla^2 u-(J_{\mathrm{si}}+J_{\mathrm{fi}}+J_{\mathrm{so}})\\ \frac{\partial v}{\partial t} &= \frac{1}{\tau_{\mathrm{v}}^{-}}(1-H(u-\theta_{\mathrm{v}}))(v_{\infty}-v)-\frac{1}{\tau_{\mathrm{v}}^{+}}H(u-\theta_{\mathrm{v}})v\\ \frac{\partial w}{\partial t} &= \frac{1}{\tau_{\mathrm{w}}^{-}}(1-H(u-\theta_{\mathrm{w}}))(w_{\infty}-w)-\frac{1}{\tau_{\mathrm{w}}^{+}}H(u-\theta_{\mathrm{w}})w\\ \frac{\partial s}{\partial t} &= \frac{1}{2\tau_{\mathrm{s}}}((1+\tanh(k_{\mathrm{s}}(u-u_{\mathrm{s}})))-2s)\end{aligned} \tag{5}$$

where the currents J_{si}, J_{fi} and J_{so} and all parameters are defined in the appendix. Variable u represents the voltage across the cell membrane and provides spatial coupling due the diffusion term, whereas v, w, and s are governed by local ODEs without any spatial coupling. Figure 5 shows a snapshot of all four fields. To make it easier to

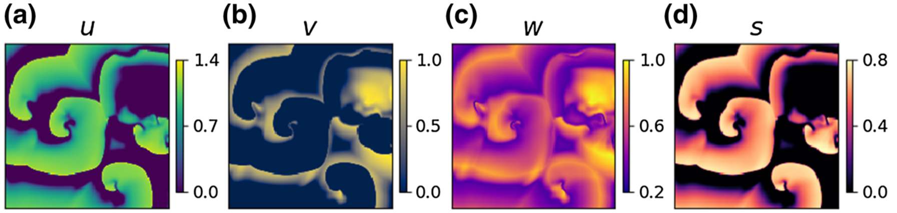

Fig. 5 Snapshot of the four variables of the BOCF model simulated on a 500 × 500 grid and coarse grained to a 150 × 150 grid using the software by Zimmermann and Parlitz (2018)

tell the different fields apart each one has been assigned its own colour map that will be used consistently. For simulation we used an implementation by Zimmermann and Parlitz (2018), that simulates the dynamics of the BOCF model using an FTCS scheme on a 500×500 grid with integration parameters $\Delta x = 1$, $\Delta t = 0.1$, diffusion constant $D = 0.2$, no-flux boundary conditions and a temporal sampling of $t_{\text{sample}} = 2.0$. The dense spatial sampling is needed for integration but impractical for our use. Therefore the software by Zimmermann coarse-grains the data to a grid of size 150×150.

4 Cross-Estimation

For cross-estimation, we analyze the Barkley model and the BOCF model. In the beginning, both systems are simulated for more than 10,000 time steps so that different subsets can be chosen for model training and testing. All training sets consisted of 5000 consecutive time steps. Due to the dense temporal sampling, the first few frames after the end of the training set are potentially predicted much better than the following ones, because data very similar to the desired estimation output are already included in the training set. To avoid this issue, predictions were offset by 1000 frames after the end of the training sequence and averaged over 20 predicted frames, where each frame was again offset by 100 time steps from the next, to reduce fluctuations and compute a standard deviation for the error measures.

To simulate uncertainty in measurements, normally distributed random numbers were added to the observed variable in the test set. Adding such noise with mean $\mu = 0$ and standard deviation $\sigma_{\mathcal{N}} = 0.075$ resulted in signal-to-noise ratios

$$\mathrm{SNR}_{\mathrm{db}} = 10 \log_{10} \frac{\langle u_{t,\alpha}^2 \rangle}{\sigma_{\mathcal{N}}^2}$$

of 18.5 dB and 14.6 dB for u and v in the Barkley model, respectively. For the fields, u, v, w, s of the BOCF model SNRs were 20.1 dB, 13.2 dB, 18.1 dB, and 15.4 dB, respectively. For an intuition of the noisyness Fig. 6 shows the variable u and v of the Barkley model and the variables u and w of the BOCF model with added noise.

To optimize the choice of the various algorithm parameters we employ the approach described in “Appendix B”.

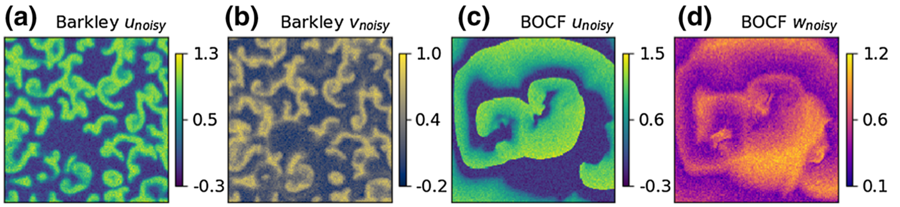

Fig. 6 Snapshots of the variables u and v of the Barkley model and the variable u and w of the BOCF model after addition of normally distributed noise

Table 1 Identified optimal parameters and average cross-estimation errors for noisy data from the Barkley model (4) with temporal sampling $t_{\text{sample}} = 0.01$. $D_{\text{E}} = (\gamma + 1)d_r$ is the initial local delay coordinate space dimension and D_{R} is the reduced dimension used to make the prediction. For both predictions, we used the constant value of 200 for the beyond the boundary pixels

	$u \to v$	$v \to u$
γ	500	30
τ	1	5
r	0	3
d_r	1	29
D_{E}	501	899
D_{R}	15	15
NRMSE	0.0354 ± 0.0013	0.096 ± 0.006

4.1 Barkley Model

For the Barkley model (4), only the u variable has a diffusion term. Therefore, the dynamics of v solely depends on u and its past. This significantly reduces the parameter space as spatial neighbourhoods may only be needed for noise reduction during PCA and can likely be small. For the prediction direction, $u \to v$ the local delay coordinate map with least prediction error was $\gamma = 500$, $\tau = 1$ and $r = 0$. These parameters produce a highly redundant map which allows PCA to efficiently filter out noise. The other direction $v \to u$ needs spatial neighbourhoods for effective cross-estimation and the parameters were $\gamma = 30$, $\tau = 5$ and $r = 3$.

The results evaluated according to the error measure (2) are listed in Table 1. A visualization of the predictions is shown in Fig. 7 along with additional predictions performed with identical parameters but for noiseless input.

4.2 BOCF Model

Similar to the Barkley model, only the u variable of the BOCF model (5) has a diffusion term which simplifies the predictions of $u \to \{v, w, s\}$. All local delay coordinate map parameters are listed along with the prediction errors in Table 2. In most of these cases, we observed that local delay coordinate maps covering a large time window $\gamma\tau$ along with a small spatial neighbourhood performed best.

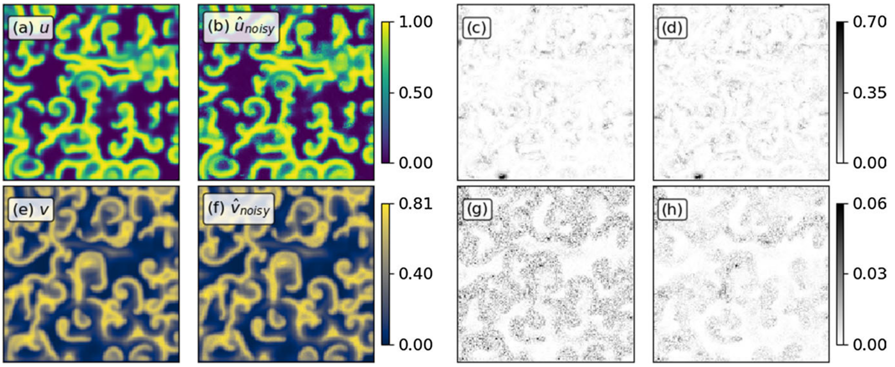

Fig. 7 Cross-estimation of data generated by the Barkley model (4) from a noisy v field to u (first row) and vice versa (second row). **a**–**d** show estimates of the u field where **a** is the actual u field, **b** the predicted field $\hat{u}$ (from noisy input), **c** the absolute difference between the two, and **d** a reference estimation error for noiseless input with identical parameters and training set. Panes **e**–**h** show the same for the field v. The parameters are listed in Table 1

Table 2 Parameters and average cross-estimation errors for noisy data from the BOCF model (5) with temporal sampling of $t_{\text{sample}} = 2.0$. A value of 200 was used for the pixels beyond the boundary. $D_{\text{E}} = (\gamma + 1)d_r$ is the initial dimension of local delay coordinate space and D_{R} is reduced dimension used for nearest neighbour searches

	γ	τ	r	D_{E}	D_{R}	d_r	NRMSE
$u \to v$	100	1	1	505	15	5	0.041 ± 0.006
$u \to w$	100	1	1	505	15	5	0.048 ± 0.007
$u \to s$	100	1	1	505	15	5	0.037 ± 0.004
$v \to u$	100	2	2	1313	15	13	0.12 ± 0.02
$v \to w$	200	1	1	1005	15	5	0.054 ± 0.011
$v \to s$	100	2	2	1313	15	13	0.093 ± 0.014
$w \to u$	10	2	4	539	9	49	0.26 ± 0.03
$w \to v$	10	2	4	539	9	49	0.26 ± 0.04
$w \to s$	10	2	4	539	9	49	0.22 ± 0.03
$s \to u$	100	1	1.5	909	15	9	0.071 ± 0.005
$s \to v$	100	1	1.5	909	15	9	0.057 ± 0.008
$s \to w$	100	1	1.5	505	15	9	0.059 ± 0.008

This is likely due to the dense temporal sampling relative to the propagation speed of wavefronts within the simulated medium. In this way, the highly redundant map and PCA for dimension reduction provide an effective method of noise reduction. The w field however presents itself as a somewhat smeared out version of the other variables thus requiring a larger spatial neighbourhood to recover the positions of wavefronts.

To visualize a few results, we chose the best and worst performing estimations. Figure 8 contains results for $w_{\text{noisy}} \to \{u, v, s\}$ and Fig. 9 shows estimations from a noisy u field to all other variables. The NRMSE values in Table 2 indicate that the estimations from field w perform about one order of magnitude worse than the estimations from field u. Figures 8 and 9 on the other hand reveal that, even in the latter estimations, the erroneous pixels are concentrated around the wavefronts. Thus, the overall prediction for most of the area is very accurate in both cases.

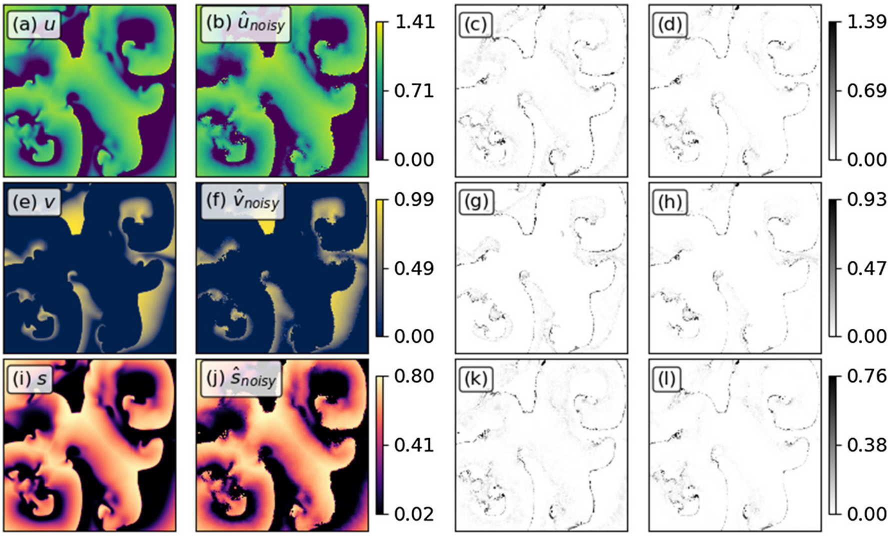

Fig. 8 Cross-estimation of data generated by the BOCF model (5) from w_{noisy} to all three other variables. **a**–**d** show estimates of the u field where **a** is the actual u field, **b** the predicted field $\hat{u}$ (from noisy input), **c** the absolute difference between the two, and **d** a reference estimation error for noiseless input with identical parameters and training set. Panes **e**–**h** and **i**–**l** show the same for their fields v and s, respectively. The parameters are listed in Table 2

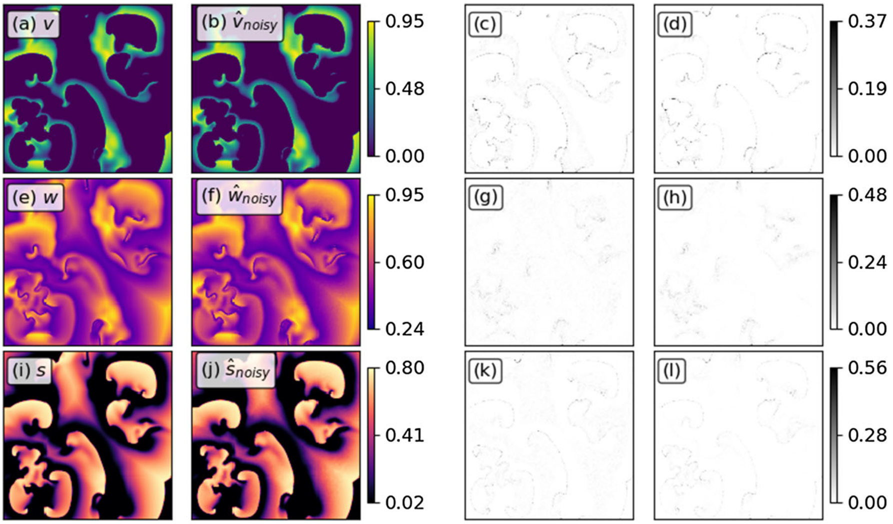

Fig. 9 Cross-estimation of data generated by the BOCF model (5) from u_{noisy} to all three other variables. **a**–**d** show estimations for the v field where **a** is the actual v field, **b** the predicted field (from noisy input), **c** the absolute difference between the two, and **d** a reference estimation error for noiseless input with identical parameters and training set. Panes **e**–**h** and **i**–**l** show the same for their fields w and s, respectively. The parameters are listed in Table 2

5 Iterated Time Series Prediction

In the following, we will analyze the performance of local modelling for spatially extended systems in the context of iterated time series prediction. For this, we use the Kuramoto–Sivashinsky model (3) and the Barkley model (4).

The obvious performance measure in this case is the time it takes before the prediction errors exceed a certain threshold. Time however is not an absolute concept in dimensionless systems. Therefore we will also define characteristic timescales of each system which will give a context to the prediction times.

5.1 Predicting Barkley Dynamics

The data sets used during cross-estimation were sampled with $t_{\text{sample}} = 0.01$ which could be considered nearly continuous relative to the timescale of the dynamics. To provide a useful example for temporal prediction with a reasonable amount of predicted frames, we use a larger time step $t_{\text{sample}} = 0.2$, while the simulation time step was kept constant at $\Delta t = 0.01$ for accurate numerical integration.

Figure 10 shows one such prediction of the u variable in the Barkley model. The figure consists of seven subplots where the top two rows show the system state at the prediction time steps $n = 25, 50$ as well as the corresponding iterated predictions. The very right column displays the absolute errors of the prediction defined by $|u_{t,\alpha} - \hat{u}_{t,\alpha}|$. At the bottom is the time evolution of the NRMSE for the prediction. Looking closely at the snapshots in the figure reveals that indeed the maximum prediction error increases quickly, as can be seen by the dark spots of the error plots (c) and (f). The overall error however increases much more slowly which is confirmed by comparing the original state with the prediction.

To set the above results into perspective, we calculate a characteristic timescale for the Barkley model. Here, we will use the average time between two consecutive local maxima for each pixel, which in good approximation gives the average period of the rotating spiral waves. Averaging over 100 × 100 pixels and 4000 time steps gave this time as $t_c \approx 5$. This means that the error of the u field prediction increased to $\text{NRMSE}(u, 2t_c) \approx 0.5$ within two characteristic times.

5.2 Predicting Kuramoto–Sivashinsky Dynamics

The Kuramoto–Sivashinsky (KS) model (3) is a one-dimensional system that has just a single field. As in the iterated time series prediction of the Barkley model we will need a characteristic timescale for the dynamics of the KS model to assess the quality of the forecast. Here, we choose the Lyapunov time which was defined and calculated for the KS model by Pathak et al. (2018). The following figures are scaled according to the Lyapunov time Λt with $\Lambda \approx 0.1$.

It is possible to integrate the KS model with different sizes L and spatial samplings Q. We will attempt to predict the time evolution for $L = 22$, $Q = 64$ and a larger system with $L = 200$ and $Q = 512$. The smaller one of the two has just 64 points and thereby could be predicted by using either local or global states, where the latter are

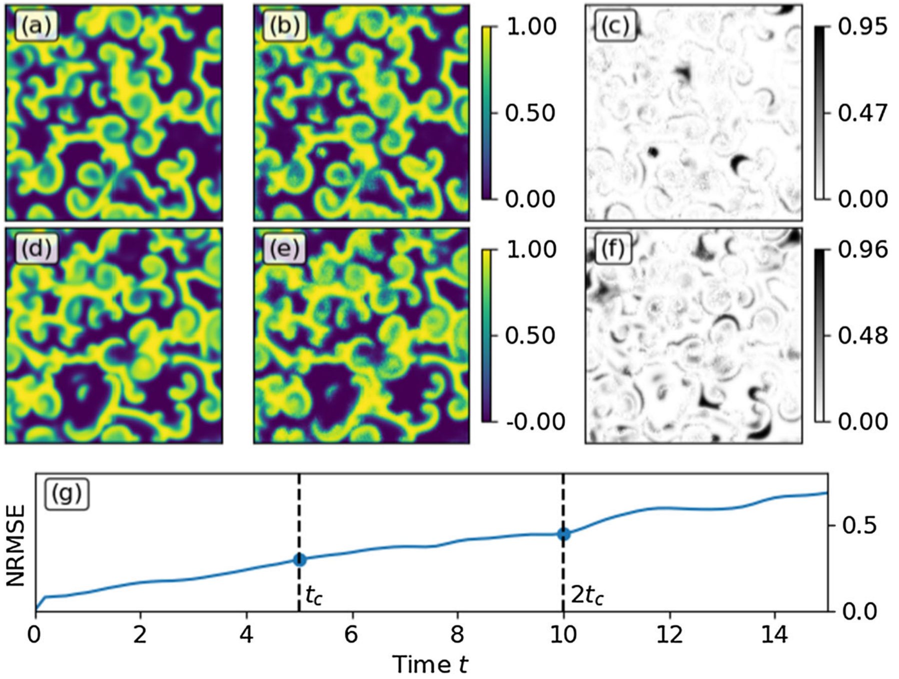

Fig. 10 Predicting field u of the Barkley model with system size 150×150 and training of 5000 states. The parameters are $\gamma = 12$, $\tau = 2$, $r = 4$, and boundary constant 200. PCA reduced the dimension from $D_{\mathrm{E}} = 637$ to $D_{\mathrm{R}} = 15$. Panes **a** and **d** show the true evolution at time $t = 5$ and $t = 10$. Panes **b** and **e** contain the iterated prediction at that time and **c** and **f** the corresponding absolute error. **g** shows the accumulation of the NRMSE in the prediction. The dashed lines note t_c, the bullets note the times 5 and 10

given by combining samples from all Q sites in a state vector. The global states have a higher dimension and may require larger training sets to densely fill the reconstructed space but in return each vector represents the state of the whole system. Sample predictions for both approaches are shown in Fig. 11 using the same training set of 10^5 states.

A notable observation with the ($L = 22$, $Q = 64$) KS model is its variable predictability as it strongly depends on the initial conditions, i.e. the current position on the chaotic attractor.

Figure 12 supports this claim by showing box-plots of the predictability for 500 different initial conditions for three different training sets of length 10^5, 10^6, and 10^7. The prediction horizon is computed as the time it takes for the NRMSE error to grow to a value of 0.1. For an intuition, these time steps are highlighted in red in Fig. 11. For $L = 22$, both the length of the predictions and its variability increase for larger training sets. In some rare cases, the errors exceeded 0.1 only after $17\Lambda t$ (not shown in Fig. 12).

Figure 11 also shows a prediction of the KS system on a larger domain of $L = 200$, $Q = 512$ with corresponding predictability box-plots in Fig. 12. Here, the prediction horizons are shorter and the variability in predictability is much smaller

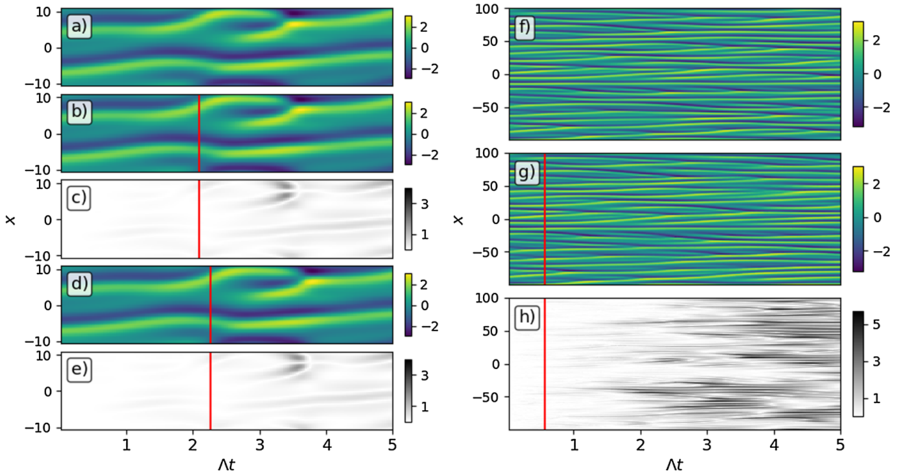

Fig. 11 Predictions of the KS dynamics with $L = 22,\ Q = 64$ **a**–**e** and $L = 200,\ Q = 512$ **f**–**h** using PCA and 1 nearest neighbour. Shown are: in **a** and **f** actual evolutions, below it in **b** and **g** predictions from local states with parameters $\gamma = 7,\ \tau = 1,\ r = 10$, and in **d** a prediction using global states ($\gamma = 0,\ r = 32$), each along with its errors (**c**, **e**, **h**). All predictions used 10^5 time steps for training. The timestep at which the NRMSE error first hit a value > 0.1 is marked with a red line (Color figure online)

compared to the smaller KS system. Figure 11 (h) shows regions in space that quickly accumulate error thereby limiting the overall predictability as well as regions that are (still) predicted accurately until $\approx 2\Lambda t$.

The KS model has previously been used by Pathak et al. (2018) for evaluating the prediction performance of some reservoir computing methods. These authors reported for $L = 22$ and $L = 200$ prediction horizons of $\approx 3\Lambda t$ (Fig.2 in Pathak et al. (2018)) when using a reservoir network and $\approx 4\Lambda t$ (Fig.6a in Pathak et al. (2018) for RMSE threshold values between 0.08 and 0.09, which corresponds to our criterium of NRMSE = 0.1) for 64 reservoirs running in parallel.

The issue of variations in predictability of the KS model hinders direct comparisons to the work of Pathak et al. (2018) who did not address this problem. In the small system, we saw initial conditions where predictions outperformed the ones by Pathak et al. but also others that were much worse. The larger system however has so far been harder to predict and we did not match the prediction accuracy of the approach of Pathak et al.

6 Benchmark of PCA

In this paper, we use principal component analysis for two reasons. The obvious purpose is to find a low-dimensional representation of the high-dimensional local delay coordinate space. One very much wanted side effect is noise reduction. All of the above presented examples used highly redundant local delay coordinate maps to allow for noise tolerance.

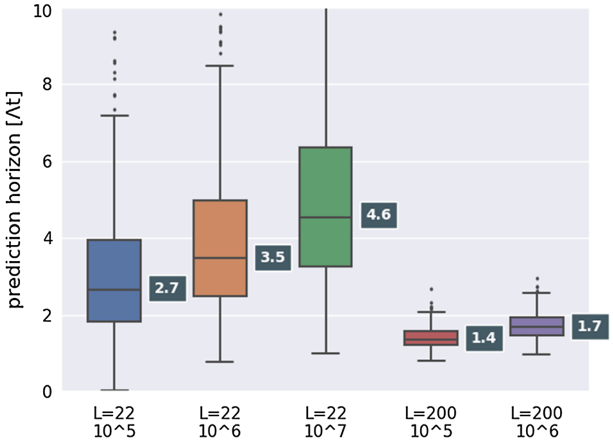

Fig. 12 Variations in predictability of the KS model where the prediction horizon for an initial condition is defined as the time predicted until the NRMSE first exceeds 0.1. Shown are results for $L = 22$, $Q = 64$ and $L = 200$, $Q = 512$ obtained with different lengths $\{10^5, 10^6, 10^7\}$ of the training set. Predictions were done on 500 different initial conditions (each offset by 100 time steps) using parameters $\gamma = 7$, $\tau = 1$, $r = 10$ and 1 nearest neighbour for modelling. The reduced dimension was automatically determined to $D_{\mathrm{R}} = 7$ for $L = 22$ and $D_{\mathrm{R}} = 8$ for $L = 200$. The numbers in the box-plots give the median of the underlying distribution and the (coloured) boxed indicate the first and the third quantile (Color figure online)

To evaluate how well PCA is suited for this purpose, we test two things: We firstly test whether a low-dimensional representation is found via PCA. The result is shown in Fig. 13. We see the dependence of the prediction error on the output dimension D_{R} of PCA in a cross-estimation of $u \rightarrow v$ in the Barkley model. It is evident that in this case no more than about 5–7 dimensions are needed to encode all information relevant to the prediction as both the prediction error as well as the ratio of retained variance saturate for larger D_{R}.

To test whether PCA also successfully eliminated the noise in the test set, we compare the two panes in Fig. 13 where the results in (b) were computed using a 20 times less redundant local delay coordinate map than in (a). The parameters in (a) were chosen identically to the identified optimal parameters listed in Table 1, whereas the less redundant parameter set for (b) was chosen to keep the covered time window of each local delay coordinate vector constant at $\gamma\tau\Delta t \approx 500\Delta t \approx 5$. The noiseless predictions perform similarly well in both cases, indicating that the additional values are indeed redundant and do not add much information to the local delay coordinate vectors. Comparing the noisy predictions highlights the effectiveness of PCA in this case as predictions from the redundant map (Fig. 13a) are consistently better by one order of magnitude (comp. Fig. 13b).

7 Conclusions

The combination of local modelling and principal component analysis for dimension reduction provides a conceptually simple yet effective approach to both cross-

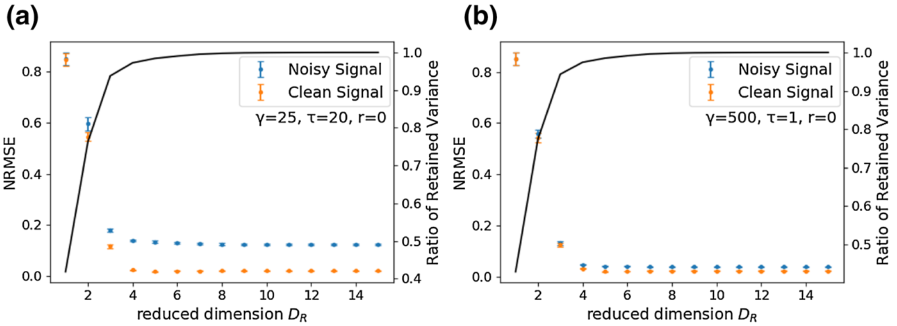

Fig. 13 NRMS Errors of cross-estimation $u \rightarrow v$ of Barkley variables vs. reduced dimension D_{R} for clean and noisy (test) input signals u with parameters (**a** $\gamma = 25$, $\tau = 20$; **b** $\gamma = 500$, $\tau = 1$) such that the covered time window $\gamma\tau$ remains constant. The estimation error is large for very small values of the reduced dimension D_{R}, but becomes almost constant for $D_{\text{R}} > 5$. The ratio of preserved variance is shown in black. PCA-based dimension reduction starting from a higher dimensional local delay coordinate map with $D_{\text{E}} = \gamma + 1 = 501$ in (**b**) proves to be more resilient to noise than the map with $D_{\text{E}} = \gamma + 1 = 26$ in (**a**)

estimation and temporal prediction of complex spatially extended dynamics. The equations for all three model systems (Barkley model, BOCF model, KS model) were only needed for data generation and as such the approach could well be applied to real world data where the underlying dynamics are not known. Adding noise to the input data naturally reduces prediction quality but in Sect. 6 it is shown that PCA can restore accuracy from a more redundant local delay coordinate map.

The currently presented method has its limitations, however. A core assumption of the process is that the dynamics of the system-to-be-predicted is homogeneous. This becomes evident from the fact that the reduced states of all pixels compose a single *k*–*d* tree. This limitation could potentially be resolved. It has been proposed in Parlitz and Merkwirth (2000) that simply extending the local states with an additional space-dependent entry on the vector could resolve the issue. Another alternative would include creating several parallel *k*–*d* trees instead of a single one for clearly separated domains with different yet locally homogeneous dynamics. Furthermore, nonlinear dimension reduction methods could be an improvement over PCA (which is a linear process).

In its present form, the presented prediction method is not yet fully competitive with recent machine learning approaches, like those presented in Pathak et al. (2018), Lu et al. (2017) and Vlachas et al. (2018). One attempt to improve the prediction performance could be to use a more sophisticated local function approximation scheme instead of the distance weighted averaging.

An advantage on the presented algorithm is that nearest neighbours modelling based on local delay coordinate vectors is conceptually simple and computationally efficient. In addition, as long as the homogeneity assumption holds, the method is scalable to systems with larger spatial extend (as not all pixels of the system need to be sampled for the creation of the *k*–*d* tree).

Acknowledgements Open access funding provided by Max Planck Society. The authors thank R. Zimmermann for allowing the use of BOCF model simulation and S. Luther for scientific discussions and continuous support.

Appendix A: Bueno-Orovio–Cherry–Fenton Model

$H(\cdot)$ denotes the Heaviside function and the currents J_{fi}, J_{so}, J_{si} of the Buono-Orovio-Cherry-Fenton (BOCF) model (5) (Bueno-Orovio et al. 2008) are defined as:

$$\begin{aligned} J_{\mathrm{fi}} &= \frac{-v}{\tau_{\mathrm{fi}}} H(u-\theta_{\mathrm{v}})(u-\theta_{\mathrm{v}})(u_{\mathrm{u}}-u) \\ J_{\mathrm{so}} &= \frac{1}{\tau_{\mathrm{o}}}(u-u_{\mathrm{o}})(1-H(u-\theta_{\mathrm{w}})) + \frac{1}{\tau_{\mathrm{so}}} H(u-\theta_{\mathrm{w}}) \\ J_{\mathrm{si}} &= \frac{-1}{\tau_{\mathrm{si}}} H(u-\theta_{\mathrm{w}}) ws. \end{aligned}$$

The τ parameters used above are not constant but rather a function of the cell membrane voltage variable u:

$$\begin{aligned} \tau_{\mathrm{v}}^{-} &= (1-H(u-\theta_{\mathrm{v}}^{-}))\tau_{\mathrm{v1}}^{-} + H(u-\theta_{\mathrm{v}}^{-})\tau_{\mathrm{v2}}^{-} \\ \tau_{\mathrm{w}}^{-} &= \tau_{\mathrm{w1}}^{-} + \frac{1}{2}(\tau_{\mathrm{w2}}^{-} - \tau_{\mathrm{w1}}^{-})(1+\tanh(k_{\mathrm{w}}^{-}(u-u_{\mathrm{w}}^{-}))) \\ \tau_{\mathrm{so}}^{-} &= \tau_{so1} + \frac{1}{2}(\tau_{\mathrm{so2}} - \tau_{so1})(1+\tanh(k_{\mathrm{so}}(u-u_{\mathrm{so}}))) \\ \tau_{\mathrm{s}} &= (1-H(u-\theta_{\mathrm{w}}))\tau_{\mathrm{s1}} + H(u-\theta_{\mathrm{w}})\tau_{\mathrm{s2}} \\ \tau_{\mathrm{o}} &= (1-H(u-\theta_{\mathrm{o}}))\tau_{o1} + H(u-\theta_{\mathrm{o}})\tau_{\mathrm{o2}} \end{aligned}$$

Table 3 Parameter set for the BOCF model (Bueno-Orovio et al. 2008) that imitates the Ten Tusscher–Noble–Noble–Panfilov model (ten Tusscher et al. 2004)

u_{o}	0	τ_{v2}^{-}	1150	τ_{fi}	0.11	τ_{s1}	2.7342
u_{u}	1.58	τ_{v}^{+}	1.4506	τ_{o1}	6	τ_{s2}	3
θ_{v}	0.3	τ_{w1}^{-}	70	τ_{o2}	6	k_{s}	2.0994
θ_{w}	0.015	τ_{w2}^{-}	20	τ_{so1}	43	u_{s}	0.9087
θ_{v}^{-}	0.015	k_{w}^{-}	65	τ_{so2}	0.2	τ_{si}	2.8723
θ_{o}	0.006	u_{w}^{-}	0.03	k_{so}	2	$\tau_{w\infty}$	0.07
τ_{v1}^{-}	60	τ_{w}^{+}	280	u_{so}	0.65	w_{∞}^{*}	0.94

and

$$
\begin{aligned}
v_\infty =& \begin{cases} 1, & u < \theta_\mathrm{v}^- \\ 0, & u \geq \theta_\mathrm{v}^- \end{cases} \\
w_\infty =& (1 - H(u - \theta_\mathrm{o})) \left(1 - \frac{u}{\tau_{w\infty}}\right) + H(u - \theta_\mathrm{o}) w_\infty^*.
\end{aligned}
$$

All other parameters are listed in Table 3.

Appendix B: Optimizing Parameters

In this section we discuss a few approaches on how to find local delay coordinate map parameters that produce decent prediction results. The parameters that can be optimized in the algorithm we presented are γ, τ, r, D_R, and the number of neighbours k for modelling. In principle one can also attempt to optimize the constant value used for the missing spatial neighbours at the domain boundary but we have observed that any value significantly larger than the system dynamics such as 100 times larger works fine. The obvious measure to minimise during the optimization procedure is the prediction error on some test set that is not contained in the training data.

Going back to the structure of the algorithm we note that number k of nearest neighbours used for modelling is only needed for prediction and does not operate on the local delay coordinates map directly. We therefore propose to optimize k independently from the remaining parameters and keep it constant at $k = 1$ during optimization of γ, τ, and r.

The choice of the dimension D_R of the dimension reduced local delay coordinate vectors also influences the predictiveness of the approach. However in Sect. 6 we showed that this can be a simple function of the retained variance in the local delay coordinate space. We therefore chose to automatically determine D_R such that it is the smallest number of dimensions that still retain 99% of the original variance but capped at a maximum of 15. The maximal D_R needs to be set as the memory usage grows linearly with D_R and runtime depends heavily on it.

Estimating γ, τ, and r is more difficult. The parameters are discrete and their effects on the prediction error are nonlinear. The most general approach would be an exhaustive grid search that may be computationally unfeasible depending on the system. Instead one can constrain the parameters significantly by estimating characteristic scales in the system of choice. In the case of the Barkley we computed the average time between two excitation waves in each pixel which is an estimate for the refractory phase of the hidden variables. This time window of around 500 timesteps was then used as an estimate for $\gamma\tau$.

The concept of a relevant time window also allows for consecutive grid searches that sample the parameter space in a coarse grained way.

References

Abarbanel, H.D.I.: Analysis of Observed Chaotic Data. Springer, New York (1996). https://doi.org/10.1007/978-1-4612-0763-4

Abarbanel, H.D.I., Carroll, T.A., Pecora, L.M., Sidorowich, J.J., Tsimring, L.S.: Predicting physical variables in time-delay embedding. Phys. Rev. E **49**(3), 1840–1853 (1994)

Atkeson, C.G., Moore, A.W., Schaal, S.: Locally weighted learning. Artif. Intell. Rev. **11**(1), 11–73 (1997). https://doi.org/10.1023/A:1006559212014

Barkley, D.: A model for fast computer simulation of waves in excitable media. Physica D **49**(1), 61–70 (1991). https://doi.org/10.1016/0167-2789(91)90194-E

Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9), 509–517 (1975). https://doi.org/10.1145/361002.361007

Bezanson, J., Edelman, A., Karpinski, S., Shah, V.: Julia: a fresh approach to numerical computing. SIAM Rev. **59**(1), 65–98 (2017). https://doi.org/10.1137/141000671

Bialonski, S., Ansmann, G., Kantz, H.: Data-driven prediction and prevention of extreme events in a spatially extended excitable system. Phys. Rev. E **92**, 042910 (2015). https://doi.org/10.1103/PhysRevE.92.042910

Bradley, E., Kantz, H.: Nonlinear time-series analysis revisited. Chaos **25**(9), 097610 (2015). https://doi.org/10.1063/1.4917289

Broomhead, D., King, G.P.: Extracting qualitative dynamics from experimental data. Physica D **20**(2), 217–236 (1986). https://doi.org/10.1016/0167-2789(86)90031-X

Bueno-Orovio, A., Cherry, E.M., Fenton, F.H.: Minimal model for human ventricular action potentials in tissue. J. Theor. Biol. **253**(3), 544–560 (2008). https://doi.org/10.1016/j.jtbi.2008.03.029

Carlsson, K.: NearestNeighbors.jl: High performance nearest neighbor data structures and algorithms for Julia (2018). https://github.com/KristofferC/NearestNeighbors.jl

Casdagli, M., Des Jardins, D., Eubank, S., Farmer, J., Gibson, J., Theiler, J.: Nonlinear modeling of chaotic time series: theory and applications. In: Kim, J., Stringer, J. (eds.) Applied Chaos, pp. 335–380. Wiley, New York (1992)

Coca, D., Billings, S.: Identification of coupled map lattice models of complex spatio-temporal patterns. Phys. Lett. A **287**(1), 65–73 (2001)

Engster, D., Parlitz, U.: Local and cluster weighted modeling for time series prediction. In: Schelter, B., Winterhalder, T., Timmer, J. (eds.) Handbook of Time Series Analysis, pp. 39–65. Wiley-VCH, New York (2006). https://doi.org/10.1002/9783527609970.ch3

Farmer, J.D., Sidorowich, J.J.: Predicting chaotic time series. Phys. Rev. Lett. **59**, 845–848 (1987). https://doi.org/10.1103/PhysRevLett.59.845

Gareth, J., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning. Springer, New York (2015). https://doi.org/10.1007/978-1-4614-7

Guo, L., Billings, S.: State space reconstruction and spatio-temporal prediction of lattice dynamical systems. IEEE Trans. Autom. Control **52**(4), 622–632 (2007)

Kantz, H., Schreiber, T.: Nonlinear Time Series Analysis. Cambridge University Press, Cambridge (2004) Google-Books-ID: RfQjAG2pKMUC

Kugiumtzis, D., Lingjærde, O., Christophersen, N.: Regularized local linear prediction of chaotic time series. Physica D **112**(3), 344–360 (1998). https://doi.org/10.1016/S0167-2789(97)00171-1

Kuramoto, Y.: Diffusion-induced chaos in reaction systems. Progr. Theor. Phys. Suppl. **64**, 346–367 (1978). https://doi.org/10.1143/PTPS.64.346

Lilienkamp, T., Christoph, J., Parlitz, U.: Features of chaotic transients in excitable media governed by spiral and scroll waves. Phys. Rev. Lett. **119**, 054101 (2017). https://doi.org/10.1103/PhysRevLett.119.054101

Lu, Z., Pathak, J., Hunt, B., Girvan, M., Brockett, R., Ott, E.: Reservoir observers: model-free inference of unmeasured variables in chaotic systems. Chaos **27**(4), 041102 (2017). https://doi.org/10.1063/1.4979665

Mandelj, S., Grabec, I., Govekar, E.: Statistical approach to modeling of spatiotemporal dynamics. Int. J. Bifurc. Chaos **11**(11), 2731–2738 (2001)

Mandelj, S., Grabec, I., Govekar, E.: Nonparametric statistical modeling of spatiotemporal dynamics based on recorded data. Int. J. Bifurc. Chaos **14**(6), 2011–2025 (2004)

Mc Names, J., Suykens, J., Vandewalle, J.: Time series prediction competition. Int. J. Bifurc. Chaos **9**(8), 1485–1500 (1999)

Packard, N.H., Crutchfield, J.P., Farmer, J.D., Shaw, R.S.: Geometry from a time series. Phys. Rev. Lett. **45**, 712–716 (1980). https://doi.org/10.1103/PhysRevLett

Parlitz, U.: Nonlinear time-series analysis. In: Suykens, J.A., Vandewalle, J. (eds.) Nonlinear Modeling-Advanced Black-Box Techniques, pp. 209–239. Springer, Boston, MA (1998). https://doi.org/10.1007/978-1-4615-5703-6

Parlitz, U., Merkwirth, C.: Prediction of spatiotemporal time series based on reconstructed local states. Phys. Rev. Lett. **84**(9), 1890–1893 (2000). https://doi.org/10.1103/PhysRevLett.84.1890

Pathak, J., Hunt, B., Girvan, M., Lu, Z., Ott, E.: Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach. Phys. Rev. Lett. (2018). https://doi.org/10.1103/PhysRevLett.120.024102

Rackauckas, C., Nie, Q.: DifferentialEquations.jl–A performant and feature-rich ecosystem for solving differential equations in Julia. J. Open Res. Softw. (2017). https://doi.org/10.5334/jors.151

Sauer, T., Yorke, J.A., Casdagli, M.: Embedology. J. Stat. Phys. **65**, 579–616 (1991). https://doi.org/10.1007/BF01053745

Sivashinsky, G.: On flame propagation under conditions of stoichiometry. SIAM J. Appl. Math. **39**(1), 67–82 (1980). https://doi.org/10.1137/0139007

Sivashinsky, G.I.: Nonlinear analysis of hydrodynamic instability in laminar flames—I. Derivation of basic equations. In: Pelcé, P. (ed.) Dynamics of Curved Fronts, pp. 459–488. Academic Press, San Diego (1988). https://doi.org/10.1016/B978-0-08-092523-3.50048-4

Takens, F.: Detecting strange attractors in turbulence. Dyn. Syst. Turbul. **898**, 366–381 (1981)

ten Tusscher, K.H.W.J., Noble, D., Noble, P.J., Panfilov, A.V.: A model for human ventricular tissue. Am. J. Physiol. Heart Circ. Physiol. **286**(4), H1573–H1589 (2004). https://doi.org/10.1152/ajpheart.00794.2003

Vlachas, P.R., Byeon, W., Wan, Z.Y., Sapsis, T.P., Koumoutsakos, P.: Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. Proc. R. Soc. A **474**(2213), 20170844 (2018). https://doi.org/10.1098/rspa.2017.0844

Zimmermann, R.S., Parlitz, U.: Observing spatio-temporal dynamics of excitable media using reservoir computing. Chaos **28**(4), 043118 (2018). https://doi.org/10.1063/1.5022276