# Modeling parameter dependence from time series

G. Langer[*] and U. Parlitz[†]
*Drittes Physikalisches Institut, Universität Göttingen, Bürgerstraße 42-44, D-37073 Göttingen, Germany*[‡]

(Received 5 May 2004; published 22 November 2004)

Two approaches for modeling of parameter dependence of dynamical systems from time series are investigated and applied to different examples. For both methods it is assumed that a few time series are available that have been measured for different (known) parameter values of the underlying (experimental) dynamical system. The objective is to model the changing dynamics of the system as a function of its parameters and to use this for experimental bifurcation analysis. Using *parametrized families* the tasks of modeling the dynamics and of modeling its parameter dependence are separated. Technical difficulties that may occur with this approach are discussed and illustrated. An alternative are *extended state space models* where both modeling tasks are treated simultaneously. To obtain reliable models from a few time series only, ensembles of models are employed that show very good extrapolation and generalization properties.

DOI: 10.1103/PhysRevE.70.056217 PACS number(s): 05.45.Tp, 82.40.Bj, 82.20.Wt

## I. INTRODUCTION

During the past two decades considerable progress has been achieved in modeling and predicting time series from complex systems. The machine learning community has developed general frameworks such as statistical learning theory [1] and sophisticated modeling and evaluation techniques [2] that are applicable to the special task of time series prediction. Particular interest is paid to the generalization features of models, i.e., their ability to correctly describe data (far) away from the training set used to generate the model. A promising approach for deriving robust and general methods are *ensemble methods*, where a superposition of many different models provides the desired description of the given data set [3–10]. Such ensemble methods are very useful if only a few (training) data are available and will be employed in the following.

In nonlinear dynamics, research on data analysis was initiated by the seminal work of Packard *et al.* [11] and Takens [12] who introduced the concept of state space reconstruction based on (scalar) time series [13]. To model the flow in reconstructed state space almost any method for function approximation (on scattered data) can and has been used [2]. In this way the temporal evolution of a deterministic (chaotic) system can be modeled and predicted. However, the dynamics of a nonlinear system depends not only on initial conditions and corresponding attractors but also on the current values of relevant parameters. Varying system parameters may result in bifurcations and transitions to coexisting attractors. Except for very few cases [14–21] such a parameter dependence has not been taken into account yet when deriving black-box models from measured data.

Given a physical system or process that depends on some system parameters that can be varied by an experimentalist, the task is to generate a time series based model of this system that describes not only the temporal dynamics but also the parameter dependence. To do this we assume that for (a few) different parameter values time series of the system are available and that these parameter values are known.

To motivate and illustrate the task of modeling parameter dependence and to present different approaches for solving it we shall consider now a simple example given by a one-dimensional iterated map (1) that depends on a single *system parameter* $p$,

$$x_{t+1} = f(x_t,p) = p \exp[-(x_t-1)^2]. \tag{1}$$

Figure 1(a) shows a bifurcation diagram of this discrete dynamical system vs system parameter $p$. To model the (chaotic) dynamics and the dependence on the parameter $p$ several time series are “measured” at some fixed parameter values $p_i$. Each of these time series consists of $N=10\,000$ samples where transients have been discarded. Then a polynomial model of sixth order,

$$y_{t+1} = \sum_{m=0}^{6} q_m y_t^m, \tag{2}$$

is fitted to each data set individually. Due to the varying system parameter $p$ the *model parameters* $q_m$ also change. Figure 1(b) shows model parameter $q_0$ vs system parameter $p$. The dashed line indicates the theoretical result $q_0=p/e$ ($e=2.7182\cdots$ Euler’s number) that is valid for a complete power series expansion of the function $f$ in Eq. (1). In those parameter regions where periodic orbits occur not enough *different* data points exist and the least squares problem for estimating the polynomial coefficients is ill posed. As a result the estimates for $q_0$ fluctuate strongly and deviate from the true values. This problem may be overcome if transients are available, in particular for (low) periodic attractors. Figure 1(c) shows the estimated model parameter $q_0$ based on approximations including transient data. The dependence of the other parameters $q_1-q_6$ on $p$ is very similar. The reason why transient data stabilize the model is illustrated in Fig. 2. On the left hand side a model (solid curve) based on a period-4 orbit (symbols) is shown that deviates significantly

[*]Electronic address: gerrit@dpi.physik.uni-goettingen.de
[†]Electronic address: parlitz@dpi.physik.uni-goettingen.de
[‡]URL:http://www.physik3.gwdg.de/~ulli

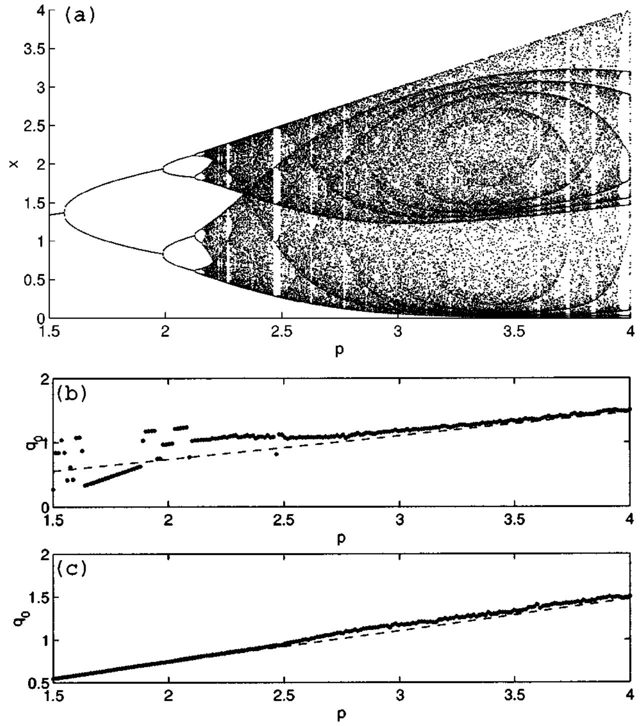

FIG. 1. (a) Bifurcation diagram of the iterated map (1). (b) Model parameter $q_0$ vs system parameter $p$. The dashed line $q_0 = p/e$ indicates the theoretical relation. (c) Same as (b) but for a data set including transients.

from the graph of the true function (dashed curve). With a few transient data points the model agrees much better with the true function as shown in the right diagram of Fig. 2. This ansatz for modeling parameter dependence using *parametrized families* will be discussed in more detail in Sec. III A. If no transient data are available one may prefer another approach for modeling the parameter dependence and include the varying parameter(s) in the input vector of the dynamical model. For the one-dimensional map the function to be learned or approximated thus reads

$$(x_t,p) \mapsto x_{t+1}. \quad (3)$$

In general, the input vector consists of a delay vector augmented by the value(s) of the system parameter(s) $p_1,p_2,\ldots$ for which the corresponding time series $\{x_t\}$ have been measured,

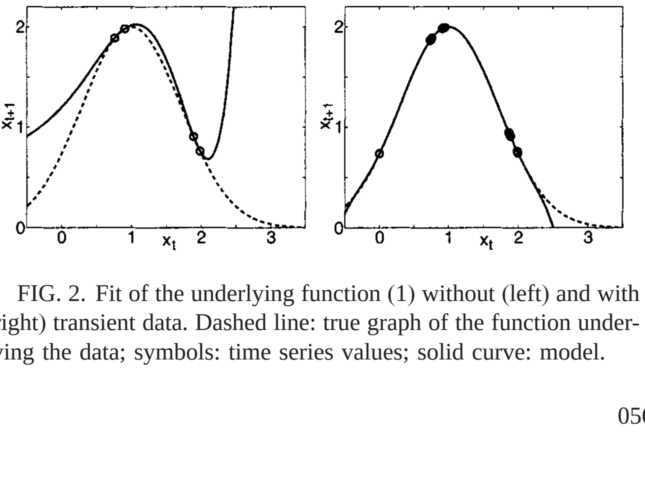

FIG. 2. Fit of the underlying function (1) without (left) and with (right) transient data. Dashed line: true graph of the function underlying the data; symbols: time series values; solid curve: model.

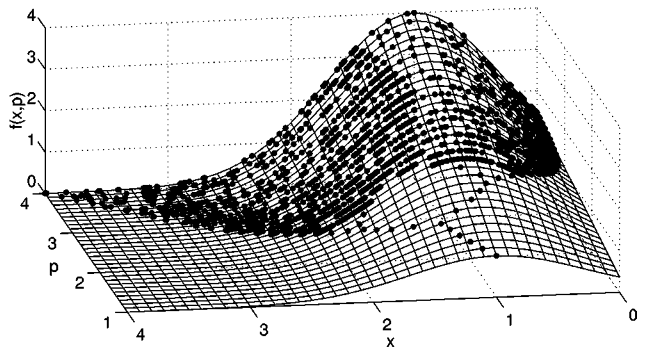

FIG. 3. Modeling parameter dependence in extended state space. Time series generated by the iterated map (1) for different values of the system parameter $p$ constitute the data base for approximating the underlying functional relation $x_{t+1}$ vs $(x_t,p)$. The grid indicates the graph of Eq. (1).

$$(x_t,x_{t-1},x_{t-2},\ldots,p_1,p_2,\ldots) \mapsto x_{t+1}. \quad (4)$$

Figure 3 shows such a representation of both the dynamics and its parameter dependence for the iterated map (1). The symbols denote the location of the points $(x_{t+1},x_t,p)$ that are given by the samples from the time series. The grid indicates the graph of the underlying function (1) that has to be approximated in order to model both the dynamics and its parameter dependence. Using all time series (with different values of the system parameter $p$) simultaneously stabilizes and improves the resulting regression function in particular in those regions of input space where only few data points (fixed points and periodic orbits of the map) are available. Such *extended state space models* will be discussed in more detail in Sec. III B. In the following Sec. II we shall briefly summarize fundamentals of nonlinear regression and ensemble modeling. Section III contains the main results for modeling parameter dependences using parametrized families and extended state space models.

## II. MODELING

### A. Nonlinear regression

To model the dynamics underlying some given scalar time series $\{s_t\}$ (with discrete time $t$) we first reconstruct state vectors $\mathbf{x}_t$ using time delay embedding $\mathbf{x}_t = (s_t, s_{t-\tau}, \ldots, s_{t-(D-1)\tau})$ where $\tau$ denotes the delay (or lag) and $D$ is the reconstruction dimension [12,13]. Regression modeling aims to predict future values of the time series from past samples. Therefore we introduce a second variable $y_t = s_{t+T}$ which is to be predicted with $T$ being the length of the prediction step. The task of regression modeling is to give the most probable representation of the underlying deterministic rule of the time series. Here we assume that additive Gaussian noise $\epsilon_t$ with zero mean perturbs the observed output values $y_t = f(\mathbf{x}_t) + \epsilon_t$ [22]. The model representation of $f$ can be accomplished for example by neural networks trained on the time series or a series expansion (also called *pseudolinear models*) with an appropriately chosen set of basis functions, which can be polynomials, splines, radial functions, etc. In the latter case the model $g: \mathbb{R}^D \to \mathbb{R}$ is a linear superposition of nonlinear basis functions $\phi_m$,

$$g(\mathbf{x}) = \sum_{m=1}^{M} q_m \phi_m(\mathbf{x}). \tag{5}$$

The coefficients $q_m$ are calculated for a fixed set of basis functions by minimization of the mean square error (MSE)

$$E_{\mathrm{MSE}} = \langle (y_t - g(\mathbf{x}_t))^2 \rangle_t$$

computed for a subset of the available data called *training set*. The minimization can be done by solving the normal equations

$$\mathbf{G}^T\mathbf{G}\hat{\mathbf{q}} = \mathbf{G}^T\mathbf{y}, \tag{6}$$

where $\mathbf{G}$ is a matrix with $G_{tm} = \phi_m(\mathbf{x}_t)$ and $\hat{\mathbf{q}}$ is the vector of estimated model parameters. These equations can be solved by singular value decomposition or other linear methods [23]. Furthermore, it is desired that the final model contains only the most significant basis functions in terms of their individually achieved error reduction on the training data. Techniques for term selection have been developed to either start with a full set of functions and to remove them from the model one by one—the most redundant in each step—or—computationally less costly—to start with the function that describes the data best, i.e., achieves lowest error as a single term model, and build in more functions, which in each step maximize error reduction. An efficient method for the second approach is *forward orthogonal regression* introduced by Korenberg [24], which we use here in a slightly modified version.

Another important feature of models is their *generalization capability*, i.e., a measure of prediction accuracy on *test data* that originate from the same source under the same conditions but that were not used for training the model. The generalization performance of a model can be assessed by evaluating training error and test error (generalization error) on independent splits of the data by means of cross-validation methods [2]. Since the model complexity increases when adding basis functions it can be observed that the training error decreases monotonously when increasing the number $M$ of basis functions $\phi_m$ in Eq. (5). The test error first follows this tendency but it increases again from some point onwards. The reason for this is called *overfitting* and it has a simple explanation: Any finite data set is a random realization of the underlying process (because the time of measurement and/or initial values are chosen randomly). Furthermore, data are usually contaminated with noise. As the model complexity increases, the model does not only describe the underlying deterministic structure but also these random features of the available data. Since only the deterministic content of the training data is reproduced in the test data, a too complex model has a lower predictive power there. Besides that a training error below the noise variance $\sigma_\epsilon^2 = \langle \epsilon^2 \rangle_t$ does not make sense. These insights can be expressed in terms of *bias* and *variance* according to the error decomposition introduced by Geman *et al.* [25]. The *bias* accounts for the deviation of the expected regression from the true regression and the *variance* captures the scattering of individual models around their expectation. Too simple models have a high bias (because of the lack of approximative power) and low variance. On the other hand, very (too) complex models possess a small bias but high variance. These statements on model behavior cannot be assigned to individual models. Therefore it is still possible that by chance a (too) complex model has good generalization properties (although in principle it should suffer from overfitting) while a model with same complexity, that has been trained on a different fraction of the data, generalizes badly.

Different methods have been devised to cope with overfitting. An important technique is *regularization* [2] where penalty terms are added to the MSE that depend upon the model complexity and/or the size of the coefficients $q_m$. Consequently, a slightly different optimization problem occurs that can again be solved with linear methods (see, for example, Ref. [26] on the application of the *Tikhonov-Phillips* regularization method). It achieves variance reduction through a smoothing effect on the model function $g$. Regularization of the model can also be achieved by using a multistep mean square error (MMSE) as cost function. In this case for each initial condition a number $U$ of free iterations of the model are computed and compared with the observed time series

$$E_{\mathrm{MMSE}} = U^{-1}\left\langle \sum_{u=1}^{U} (y_{t+u-1} - g^{(u)}(\mathbf{x}_t))^2 \right\rangle_t, \tag{7}$$

where $g^{(u)}$ denotes the $u$th iterate of the model function $g$ [27]. The coefficients $q_m$ of the model are adjusted iteratively by minimising the multistep mean square error (7) using any appropriate nonlinear optimization method. This guarantees that the resulting free run trajectory segments stay closer to the orbits reconstructed from the data. A similar method was proposed in Ref. [28] for the construction of noise-free trajectories. Minimizing the MMSE (7) results in a further decrease of bias and variance compared to the single step MSE.

As model terms we employ in the following Gaussian radial basis functions of width $d_m$:

$$\phi_m(\mathbf{x}) = \phi(\mathbf{x}, \mathbf{c}_m, d_m) = \exp[-(\mathbf{x} - \mathbf{c_m})^2/d_m^2] \tag{8}$$

centered at $\mathbf{c_m}$ in the input (state) space plus a constant term and linear functions of the components of $\mathbf{x}$.

### B. Ensemble models

One of the major issues of this paper are the generalization capabilities of models used for modeling parameter dependencies. As it is standard procedure to apply the learning scheme multiple times to diverse initial conditions including different fractions of the data, ensembling the resulting models relief the difficulty of selecting the single best model from that population and rather effectively combine some of them without spending too much effort on coping with features of randomness of the individual models. Ensemble methods make no assumptions on the best model structure to use especially when mixtures of different model topologies are concerned. When using cross validation for assessing the generalization capabilities of our single models we always have to have a test data set that is not directly used for generating the model. In contrast to that an ensemble as a whole may have seen during training of the participating models all of the data covered by the individual training sets. So, all data can enter the final (ensemble) model and no data are "wasted" for testing, only. It has been shown that ensembles consisting of different models yield better descriptions of the underlying deterministic structure of the data than single models do. In this way ensembles generalize better and give better predictions even far away from the training data.

At the beginning of the 1990s Hansen and Salamon [3] showed that this works for an ensemble of diverse neural networks. Research activities on ensemble methods have mainly focused on the use of neural networks and decision trees applied to classification tasks (e.g., Dietterich [6]), but ensemble methods are applicable to regression problems as well (e.g., Perrone *et al.* [4]). Whereas in classification a majority vote of single hypotheses/models is used, regression ensembles come along with weighted averages of their model outputs. For the choice of the participating models it is crucial that they are diverse to the extent that the errors they make are uncorrelated.

For generating ensembles we have to introduce variance in the models involved. *Computational variance* is achieved by randomizing the models. Prior to the creation of each single model participating in the ensemble a basis function pool is made up from a number of randomly drawn reconstructed state space vectors defining the centers $\mathbf{c}_m$ of the radial basis functions (8). Additionally the selected centers may be shifted by random variates [15] to increase computational variance of the resulting models. The widths $d_m$ of the Gaussian functions are chosen to be at maximum of the order of the time series' variance. From this pool the most significant functions are chosen by the forward orthogonal regression algorithm [24], which establishes the model in its first form. To improve it we apply a local search scheme (Levenberg-Marquardt method, for a description see Ref. [23]) to iteratively optimize those parameters (centers and widths) on which the model output of the series expansion (5) depends nonlinearly. The model coefficients $q_m$ are updated simultaneously in that process by linear methods. A third learning step may consist of a multistep error reduction and adjustments to the coefficients. Since in the multistep case the output depends nonlinearly on the coefficients $q_m$ the Levenberg-Marquardt method is used again to solve the minimization problem. The data presented to this learning scheme for each model are randomly drawn from the original data without checking for multiple occurrences. If the training data sets so generated had the same length as the original, this would resemble the bootstrap method from statistics (for a thorough discussion see Ref. [29]). Compared to a split of the data leading to training and test data in the classical fashion, the data used in each modeling run have a random distribution which enables the (unstable) learning algorithm to reach diverse solutions. In this way we introduce *realizational variance* into our models. In such a way an ensemble represents an effective way of dealing with small amounts of data.

A reasonable number of models to be combined ranges from 3 to about 20. Investigations of the role of the ensemble size can be found in Krogh and Sollich [9]. In the following we shall use uniform ensemble weights $w_k = K^{-1}$ where $K$ is the ensemble size. Methods for improving the performance of the ensemble model using nonuniform weights have been proposed, for example, by Merz and Pazzani [30], Perrone and Cooper [4], and Zhou *et al.* [7]. For further discussion and reference on the ensemble effect, see Appendix A.

## III. PARAMETER DEPENDENCES

In this section both approaches for modeling parameter dependencies, *parametrized families* and *extended state space models*, are discussed in detail and illustrated with examples.

### A. Parametrized families

The crucial first step is a proper choice of the parametrized family on the basis of the given time series. The basis functions $\phi_m(\mathbf{x}) = \phi(\mathbf{x}, \mathbf{c}_m, d_m)$ have to be the same for all (different) time series. Only the coefficients $q_m$ are assumed to depend on the system parameters and a different set of them corresponds to each of the $l = 1, \ldots, L$ parameter settings $\mathbf{p}_l$,

$$g(\mathbf{x}|\mathbf{q}(\mathbf{p}_l)) = \sum_{m=1}^{M} q_m(\mathbf{p}_l)\,\phi_m(\mathbf{x}). \tag{9}$$

In this way all dynamics are described in the same function basis. The difficulty lies in the choice of a useful set of basis functions that allows capturing different dynamics associated with different time series. Here the construction of a model family from Gaussian radial basis functions is done in the following steps. First a pool of centers $\mathbf{c}_\mu$ is generated that are selected randomly from amongst *all* reconstructed state space vectors and therefore follow their empirical distribution (see also Ref. [15]). The widths $d_\mu$ are chosen to be at maximum of the order of magnitude of the *average* time series variance. In accordance with the chosen procedure terms are selected from the pool one after the other. Further details and considerations on this are given in Appendix B.

It is still possible then that not all models $g_l(\cdot) = g(\cdot|\mathbf{q}_l)$ achieve equally low error rates on their respective time series or do not reproduce the original dynamics with the same accuracy when run freely. This to a large extent depends on the spatial distribution of training examples given by the different time series. If the reconstructed attractors differ largely in shape, size, and position they fill different parts of state space. This may lead to the occurrence of basis functions $\phi_j$ in the parametrized family that are not equally important for the modeling of all of the participating time series. Like with the problem of a lack of transients, for some time series the corresponding coefficients may be difficult to estimate reliably. Generally, this method works well if transients are included or if the attractors underlying the different time series are (geometrically) similar to some degree.

Once the selection of basis functions is finished and the relevant model parameters are computed for the given time series, the relation between (physical) system parameters $\mathbf{p}$ and model parameters $\mathbf{q}$ has to be modeled. In general, let $B$ system parameters be mapped to $M$ model parameters

$$\mathbf{q}:\mathbb{R}^B \to \mathbb{R}^M, \quad \mathbf{p} \mapsto q_m(\mathbf{p}), m = 1 \cdots M. \tag{10}$$

Describing this relation is again a matter of interpolation or extrapolation. For example, a polynomial model (order $n$) could be fitted to the samples $(q_{ml}, \mathbf{p}_l)$, whose $(n+B)!/n!B!$ coefficients are solution to a set of linear equations ( $L$ rows). For given $B$ the polynomial order $n$ and number $L$ of models should be put into sensible proportions to avoid overfitting.

Figure 4 shows bifurcation diagrams of the Rössler oscillator,

$$\dot{x}_1 = -x_2 - x_3,$$

$$\dot{x}_2 = x_1 + p x_2,$$

$$\dot{x}_3 = 2 + x_3(x_1 - 4), \tag{11}$$

for $p_1 = 0.3 \cdots 0.45$. Figure 4(a) shows a bifurcation diagram computed with original model equations (11). The vertical dashed lines are plotted at those five parameter values $p \in \{0.31, 0.34, 0.37, 0.4, 0.43\}$ for which times series have been taken to construct a parametrized family of Gaussian radial basis function networks with 15 terms. To obtain a continuous parameter dependence of the coefficients a third order polynomial was fitted to the set of model parameters $(q_{jl}, \mathbf{p}_l), j = 1 \cdots 15, l = 1 \cdots 5$. Afterwards the model can be tuned to any value of the parameter $p$ within a sensible range to query the dynamics. This approach was used to compute the reconstructed bifurcation diagram shown in Fig. 4(b), where the model is used to generate time series in dependence of the control parameter $p$. As can be seen by comparing the results with Fig. 4(a) many details of the bifurcation structure are reproduced correctly.

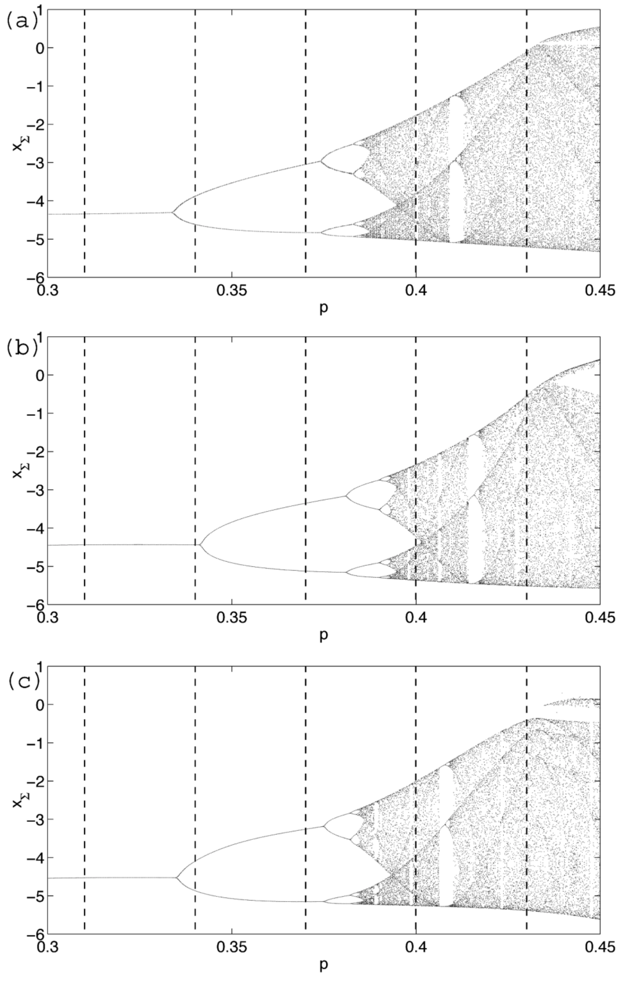

FIG. 4. Bifurcation diagrams of the Rössler oscillator. (a) Original using Eq. (11), (b) reconstructed with parametrized family of models (see Sec. III A), and (c) reconstructed with extended state space model (see Sec. III B).

### B. Extended state space models

One major shortcoming of the modeling ansatz discussed above is the need of transients or chaotic data to obtain good estimations of the model parameters. However, measured data often lack transient information and therefore are not well suited for that method. Another strategy of modeling the dynamics and the parameter dependence from time series goes back to Casdagli [14] in the late 1980s and was later used by Judd and Mees [15]. The novelty of this ansatz was the extension of the reconstructed state space to include all changing system parameters

$$\tilde{\mathbf{x}}_t = (\mathbf{x}_t, \mathbf{p}) = (s_t, s_{t-\tau}, \ldots, s_{t-(D-1)\tau}, p_1, \ldots, p_B). \tag{12}$$

The input space of the model now has dimension $D+B$ and the model approximates a function $\mathbb{R}^D \times \mathbb{R}^B \to \mathbb{R}$. The reconstructed time series along with their individual parameter extensions (parameters are constant for a given time series) are merged together and form a single data set. This data set is used to train the model as if it was one time series only. This approach is illustrated in Fig. 3. There $x_{t+1}$ is plotted vs the extended state vector $(x_t, p)$. By modeling we now seek the hypersurface containing the array of curves $x_{t+1} = f(x_t, p)$, instead of a model family based on single curves. The model comprises both dynamics and parameter dependence. As the parameter components are different from the other elements of the input vectors, dynamics takes place only in $x$-component slices $(\equiv \mathbb{R}^D)$ of the extended state space. This has to be taken into account, when the model is used for the simulation of time series. For example, when computing bifurcation diagrams those components of the input vector that correspond to parameters are kept fixed during the free run of the model.

One major advantage of this method is, that the demands on the distribution of training examples in reconstructed state space are reduced compared to training on a single time series. The reason for that is, that by aggregating the training examples of many time series there is more information for the modeling process, especially when time series are treated that contain no transient information. Consider again Fig. 3 as an example: In the region of low periodic orbits $(p_l < 2)$ there is not enough information to even fit a second order

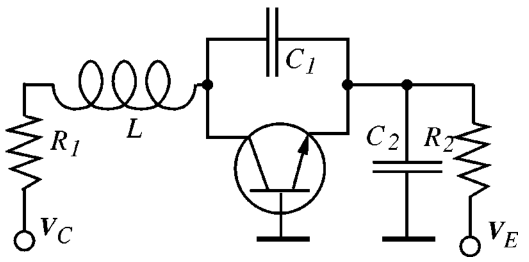

FIG. 5. Circuit diagram of the Colpitts oscillator.

polynomial. However, through the inclusion of data in a neighborhood (with respect to the parameter) the amount of training data now suffices for that purpose. This insight can be applied to the modeling of data of higher dimensional (and continuous) systems as well. The generalization capabilities of the model at a particular parameter value profit from additional information contained in time series nearby.

The training procedure here is simpler than that of a parametrized family because all time series are treated as a single data set. Centers for radial basis functions are again chosen randomly from all reconstructed state vectors in the extended state space. Term selection is performed as described before including nonlinear optimization of center positions and widths. The influence of the terms on error reduction is shared among the individual time series segments of the data set. If one time series requires $M$ terms to yield a good model, then $L$ time series modelled this way require significantly less than $L \times M$ terms.

When using radial basis functions the range of all elements of the input vector $(x_t, x_{t-1}, x_{t-2}, \ldots, p_1, p_2, \ldots)$ should be similar. This is guaranteed for the time series components $x_t, x_{t-1}, x_{t-2}, \ldots$, but the parameter components $p_1, p_2, \ldots$ may lie in a completely different range. In such cases the (generalization) performance of the model may be improved by introducing additional scaling factors for the parameters. With polynomial basis functions no such scaling factors are required.

The above described modeling approach has been used to obtain the result shown in Fig. 4(c). There we used the same time series as in Fig. 4(b) (see Sec. III A), but with no transients included. The model consisted of 55 Gaussian radial basis functions and the parameter $p$ was scaled upwards by a factor of 20.8. Note that the bifurcation diagram generated with the extended state space method [Fig. 4(c)] matches even better the original diagram [Fig. 4(a)] than the diagram computed with the parametrized families approach [Fig. 4(b)]. Our experience with examples indicate that in general the extended state space method is superior to parametrized families.

Furthermore, the method is well suited for the application of an ensemble strategy especially to improve the generalization capabilities of models based on noisy real world time series. To demonstrate this we measured time series from an electronic Colpitts oscillator (see circuit diagram in Fig. 5). The measured signal $s_t$ is the voltage over the capacitor $C_2$ and as the variable system parameter we chose the positive supply voltage $V_C$. Figure 6(a) shows an experimental bifurcation diagram that has been recorded for reference using a

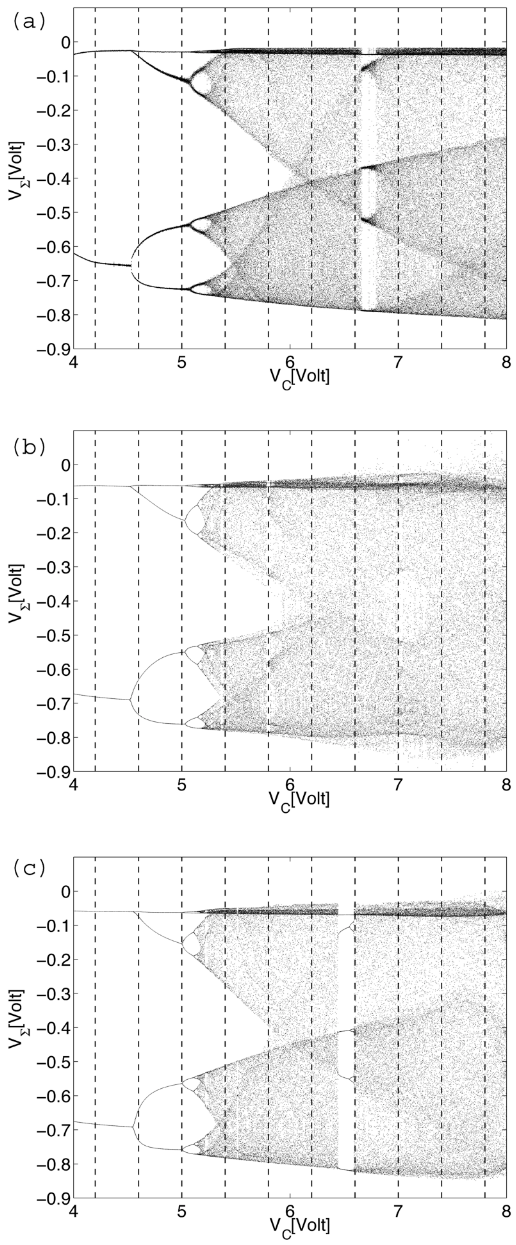

FIG. 6. Bifurcation diagrams of the Colpitts oscillator. (a) Experimental result, (b) single model, and (c) ensemble consisting of three models (see text).

small step size in the parameter dimension. There one coordinate of a Poincaré section is plotted vs the voltage $V_C$. For modeling the dynamics and parameter dependence only ten time series are used whose parameter values are given by the vertical dashed lines.

All models to participate in an ensemble were trained individually on randomized subsets of the original data. A typical single model result is shown in Fig. 6(b) and it should be compared to the reference diagram [Fig. 6(a)]. Here 160 Gaussian radial basis functions were used and a parameter scale factor of 2. As mentioned above (see Sec. II B) the single best model is hard to find and randomness in the whole modeling procedure leads to a distribution of models, which becomes obvious if one compares results from different modeling attempts that fail to reproduce each other. So we applied the ensemble strategy [see Fig. 6(c)] and combined three models of comparable quality with equal weights. The individual models contained 155–160 terms and the same scale factor was used. The ensemble in this case outperforms any single model. As the diagram shows, many detailed features of the original [Fig. 6(a)] are recovered.

## IV. CONCLUSION

Two approaches for modeling parameter dependence of the dynamics of a given (experimental) system have been presented. Both methods require as input a few time series measured for different values of the physical control parameters. Using *parametrized families* the problem of modeling parameter dependence is separated from modeling the dynamics. This approach yields satisfying results but problems may occur with simple attractors such as fixed points or periodic orbits filling a very small fraction of the input space, only. These difficulties can be avoided by including transients or using the *extended state space* approach where modeling dynamics and its parameter dependence are treated as a single task. In this framework fixed point solutions or periodic data are only a part of the whole modeling problem and thus no ill-posed approximation tasks occur. The given examples and experience with other data show that the *extended state space* method is in general superior to models based on parametrized families.

In this paper we have assumed that the variable physical parameters are known and that their values can be directly incorporated in the modeling ansatz. If (some of) the changing physical parameters are not known (or cannot be measured) information about the parameter dependence is only implicitly given in terms of the time series reflecting different dynamics. This case has been investigated by Tokunaga *et al.* [16], Tokuda *et al.* [17], and Bagarinao *et al.* [18–20] using (different types of) parametrized families. When fitting the parametrized familiy each time series provides a vector of model parameters defining a point in the coresponding model parameter space. Then linear (principle component analysis) [16,17,20,21] or nonlinear (principal curves) [18] approximation methods are used to extract a compact representation of the distribution of model parameters. Bifurcation sets computed with the resulting models were compared with those obtained with the original model that was used to generate the time series. Another approach for treating the case of unknown parameters [19] is based on a superposition of the different models where the weights are used as bifurcation control parameters.

If only a few time series for different parameter values are available modeling methods are needed that are robust and possess very good generalization abilities. The generated models have to be able to describe correctly also the dynamics and possible bifurcations (far) away from the parameter values where time series have been measured. To achieve this goal we use *ensemble methods* where a superposition of the output of different individually trained models is considered as result of the (ensemble) model. Ensembles perform better than their members if the errors of the individual models are small and not, or weakly, correlated. The key to ensemble learning methods are techniques to effectively produce diverse model populations. Such methods include the use of variable model topologies and types as well as randomizing initial parameters of training algorithms (computational variance) or some sort of manipulations of the data presented to these algorithms (realizational variance). The (dis-)agreement of the different models constituting the ensemble can be used to determine locations in (model) parameter space where the desired description of the parameter dependence is not yet reliable. In this way error bounds for predicted bifurcations can be estimated. Furthermore, parameter values can be identified for which additional measurements would be desirable resulting in an *active learning* scheme for improving the model.

## ACKNOWLEDGMENTS

We thank our colleagues at the Third Physical Institute for many interesting discussions. G.L. thanks G. Schewe and G. Dietz at the Institute for Aeroelasticity (DLR Göttingen) as well as the Deutsche Forschungsgemeinschaft (Graduiertenkolleg *Strömungsinstabilitäten und Turbulenz*) for continuous support.

## APPENDIX A: ENSEMBLE METHODS

For regression problems the ensemble method may be defined and described as follows. Let $f:\mathbb{R}^D \to \mathbb{R}$ be the true regression to be modeled. Assume we have trained $K$ predictor models $f_k$ to be combined in an ensemble. The ensemble output is the convex sum

$$\overline{f}(\mathbf{x}) = \sum_{k=1}^{K} w_k f_k(\mathbf{x}), \tag{A1}$$

where the ensemble weights $w_k$ have to obey $\Sigma_{k=1}^{K} w_k = 1$ and $0 \leqslant w_k \leqslant 1 \,\forall k$. The assessment of the generalization capabilities of the ensemble, i.e., the quadratic error of the ensemble $e(\mathbf{x}) = [y(\mathbf{x}) - \overline{f}(\mathbf{x})]^2$, involves the quadratic error $e_k(\mathbf{x}) = [y(\mathbf{x}) - f_k(\mathbf{x})]^2$ of model $f_k$, the average quadratic error $\overline{e}(\mathbf{x}) = \Sigma_{k=1}^{K} w_k e_k(\mathbf{x})$ of the participating models $f_k$ and their scattering around the ensemble mean $\overline{f}$ given by the so-called *ensemble ambiguity* [8,9],

$$\overline{a}(x) = \sum_{k=1}^{K} w_k [(\overline{f}(\mathbf{x}) - f_k(\mathbf{x})]^2. \tag{A2}$$

From these definitions the relation

$$e(\mathbf{x}) = \overline{e}(\mathbf{x}) - \overline{a}(\mathbf{x}) \tag{A3}$$

can be derived [9].

Under the assumption that the inputs $\mathbf{x}$ have been chosen randomly according to a probability distribution $P(\mathbf{x})$ we can average (A3) over $P(\mathbf{x})$ [i.e., $e = \int e(x) P(x) dx$, etc.] which yields the *mean ensemble generalization error*,

$$e = \overline{e} - \overline{a}. \tag{A4}$$

The message of Eq. (A4) is the following: Make the individual models $f_k$ good (low $\overline{e}$) and make them differ in their estimates (large $\overline{a}$). These partly contradictory demands also can be expressed in terms of the error correlation (see Zhou *et al.* [7]) of the individual models,

$$c_{ij} = \int [y(\mathbf{x}) - f_i(\mathbf{x})][y(\mathbf{x}) - f_j(\mathbf{x})]P(\mathbf{x})d\mathbf{x}, \tag{A5}$$

because

$$e = \sum_{i=1}^{K}\sum_{j=1}^{K} w_i w_j c_{ij}. \tag{A6}$$

Therefore the ensemble generalization error is small when individual errors $e_k = c_{kk}$ are small and the errors of the participating models are uncorrelated (small $c_{ij} = c_{ji}$). This shows immediately that nothing can be gained from totally correlated models where $c_{ij} = e_i$, because the ensemble generalization error in this case is given by the average quadratic error of the individual models, $e = \Sigma_{i=1}^{K} w_i e_i = \overline{e}$. On the other hand, for vanishing correlations (i.e., $c_{ij} = 0$ for $i \neq j$) we have a theoretically minimal error of

$$e = \sum_{k=1}^{K} w_k^2 c_{kk}. \tag{A7}$$

For realistic scenarios the ensemble generalization error settles somewhere in between these limiting cases.

In recent years many explanations have been given why ensembles work and why they are often superior to single models (see, e.g., Refs. [5,8]). If, for instance, in classification the error rate of a single classifier is lower than 0.5, then, if a majority rule is applied, the ensemble classifier has an even lower error rate [5]. In both cases, classification and regression, we encounter variances of different origin in populations of single models. There is a so called *computational variance* which has its origin in unstable optimization algorithms like most neural network training algorithms or splitting rules of decision trees. These algorithms respond sensitively to different initial conditions for the model parameters such as random initial weights of a neural network. They never produce the same model in different runs with different initializations, because they converge to different local minima of some complex error landscape. Another type of variance is introduced by the data sample itself, its limited size and its noise content. This so-called *realizational variance* is closely related to overfitting, but in contrast to a single model an ensemble of overfitted models (with its resulting variety) may provide a robust description of the data with good generalization features. Realizational variance may be generated or enhanced by various data manipulation and resampling techniques such as *bagging* (bootstrap aggregating, see Breiman [31] and Efron [29]) and *boosting* (see Freund and Shapire [32] or Avnimelech and Intrator [33]). Whereas in bagging the bootstrap replicates of the data set are independent in different runs, the data presented to the learning algorithm in a boosting scheme depend on prior runs. A bootstrap replicate is obtained by random sampling from the empirical distribution of the data (i.e., for a data set of length $N$ each sample is drawn with probability $N^{-1}$). The $N$ drawings can result in multiple appearance of some and absence of other examples in the training data set. Diversity is therefore provoked via randomizing the distribution of the data without any feedback. The latter, on the other hand, comes into place in a boosting scheme, where examples that turned out to be difficult to predict (or classify) by a prior model receive a higher probability to be drawn as training examples for the next model.

## APPENDIX B: GENERATING PARAMETRIZED FAMILIES

Here we discuss in short what is necessary to consider, when selecting terms for a parametrized family. We consider here a general situation in the middle of the process. Let $\mathcal{I}^{(i)}$ be the set of indices corresponding to the pool of basis functions available in the $i$th step. Furthermore, by $\mathbf{y}_l$ we denote the vector of values $y_t^l$ of time series $l$ that its reconstructed state vectors $\mathbf{x}_t^l$ are supposed to be mapped to by the model, $\mathbf{x}_t^l \mapsto y_t^l$. Let $\mathbf{g}_{\mu l} = (g_\mu(\mathbf{x}_1^l)\ldots, g_\mu(\mathbf{x}_t^l), \ldots, g_\mu(\mathbf{x}_1^N))$ be a vector containing these mappings of state vectors of time series $l$ by basis function $g_\mu(\cdot)$ and let $\tilde{\mathbf{g}}_{\mu l}^{(i)}$ being its orthogonalized version with respect to all corresponding vectors based on basis functions that have been chosen during previous steps $1\cdots i-1$. Then in step $i$ the basis function $g_{\mu_i}(\cdot)$ is inserted into the model architecture which provides the largest total error reduction on all $L$ time series, i.e., explains more of them than any other term,

$$\mu_i = \mathrm{argmax}_{\mu \in \mathcal{I}^{(i)}} \sum_{l=1}^{L} \langle \mathbf{y}_l | \tilde{\mathbf{g}}_{\mu l}^{(i)} \rangle^2 / \| \tilde{\mathbf{g}}_{\mu l}^{(i)} \|^2, \tag{B1}$$

and the $i$th coefficient of the model of the $l$th time series is

$$\tilde{q}_{il} = \langle \mathbf{y}_l | \tilde{\mathbf{g}}_{\mu_i l}^{(i)} \rangle / \| \tilde{\mathbf{g}}_{\mu_i l}^{(i)} \|. \tag{B2}$$

The $\tilde{q}_{il}$ have to be transformed to be coefficients of the nonorthogonal basis functions $\mathbf{g}_{\mu_i l}(\cdot)$. For simplicity the basis functions belonging to the model are now relabeled by $m = 1, \ldots, M$ (instead of $\mu_1, \ldots, \mu_M$). This choice of a subset of terms constitutes the initial condition for a nonlinear optimization of the model. The optimization aims at improving the Gaussian basis functions $\phi_m(\cdot)$ by moving their centers $\mathbf{c}_m$ and changing their widths $d_m$ to more favorable values. This is done iteratively, where in between consecutive iterations the $q_{ml}$ are updated via the orthogonal method of least squares. For the ease of notation the parameters $\mathbf{c}_m$ and $d_m$ are summarized in a vector $\boldsymbol{\theta}$ and by a $g_l(\cdot)$ we now denote the model with coefficients $q_{ml}$ fitted to the $l$th time series. So $L$ of them additively contribute to the total mean square error and to its derivatives with respect to $\boldsymbol{\theta}$ that are necessary to define the gradient and the pseudo-Hessian $\mathbf{D}$ (needed for the Levenberg-Marquardt method). A general iteration step $r$ of the minimization algorithm requires the following calculations:

$$E_{\mathrm{MSE}}(\boldsymbol{\theta}^{(r)}) = \frac{1}{LN}\sum_{l=1}^{L}\sum_{t=1}^{N} [y_{lt} - g_l(\mathbf{x}_{lt}|\boldsymbol{\theta}^{(r)})]^2, \tag{B3}$$

$$\frac{\partial}{\partial \theta_i} E_{\text{MSE}} = \frac{1}{LN} \sum_{l=1}^{L} \sum_{t=1}^{N} \frac{\partial}{\partial \theta_i} [y_{lt} - g_l(\mathbf{x}_{lt}|\theta^{(r)})]^2, \qquad \text{(B4)}$$

$$D_{ij} = \frac{1}{LN} \sum_{l=1}^{L} \sum_{t=1}^{N} \frac{\partial g_l(\mathbf{x}_{lt}|\boldsymbol{\theta}^{(r)})}{\partial \theta_i} \frac{\partial g_l(\mathbf{x}_{lt}|\boldsymbol{\theta}^{(r)})}{\partial \theta_j}. \qquad \text{(B5)}$$

Here we assume that all time series are of equal length $N$ as the length ratio now acts like a weight when computing each new $\Delta\boldsymbol{\theta}^{(r)}$ (resulting in a tradeoff among all participating time series).

Afterwards the coefficients $q_{ml}$ can be "polished" individually for all time series according to multistep error reduction, as was explained above Eq. (7).

[1] V. N. Vapnik, *The Nature of Statistical Learning Theory* (Springer, Berlin, 1995).

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (Springer, New York, 2001).

[3] L. K. Hansen and P. Salamon, IEEE Trans. Pattern Anal. Mach. Intell. **12**, 993 (1990).

[4] M. P. Perrone and L. N. Cooper, *Neural Networks for Speech and Image Processing*, edited by R. J. Mammone (Chapman and Hall, London, 1993), pp. 126–142.

[5] T. G. Dietterich, in *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science*, edited by J. Kittler and F. Roli (Springer, New York, 2000), pp. 1–15.

[6] T. G. Dietterich, in *The Handbook of Brain Theory and Neural Networks*, 2nd ed., edited by M. A. Arbib (MIT Press, Cambridge, MA, 2002).

[7] Z.-H. Zhou, J.-X. Wu, W. Tang, and Z.-Q. Chen, Int. J. Comput. Intell. Appl. **1**, 341 (2001).

[8] A. Krogh and J. Vedelsby, in *Advances in Neural Information Processing Systems*, edited by G. Tesauro, D. Touretzky, and T. Leen (MIT Press, Cambridge, MA, 1995), Vol. 7, pp. 650–659.

[9] A. Krogh and P. Sollich, Phys. Rev. E **55**, 811 (1997).

[10] P. Sollich and A. Krogh, in *Advances in Neural Information Processing Systems*, edited by D. Touretzky, M. Mozer, and M. Hasselmo (MIT Press, Cambridge, MA, 1996), Vol. 8, pp. 190–196.

[11] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, Phys. Rev. Lett. **45**, 712 (1980).

[12] F. Takens, in *Dynamical Systems and Turbulence*, edited by D. A. Rand and L.-S. Young (Springer, Berlin, 1981), pp. 366–381.

[13] T. Sauer, J. Yorke, and M. Casdagli, J. Stat. Phys. **65**, 579 (1991).

[14] M. Casdagli, Physica D **35**, 335 (1989).

[15] K. Judd and A. Mees, Physica D **92**, 221 (1996).

[16] R. Tokunaga, S. Kajiwara, and T. Matsumoto, Physica D **79**, 348 (1994).

[17] I. Tokuda, S. Kajiwara, R. Tokunaga, and R. Matsumoto, Physica D **95**, 380 (1996).

[18] E. Bagarinao, T. Nomura, K. Pakdaman, and S. Sato, Physica D **124**, 258 (1998).

[19] E. Bagarinao, K. Pakdaman, T. Nomura, and S. Sato, Physica D **130**, 211 (1999).

[20] E. Bagarinao, K. Pakdaman, T. Nomura, and S. Sato, Phys. Rev. E **60**, 1073 (1999).

[21] I. Tokuda, J. Kurths, and E. Rosa, Jr., Phys. Rev. Lett. **88**, 014101 (2002).

[22] The influence of noise on the input vectors $\mathbf{x}_t = (s_t, s_{t-\tau}, \ldots, s_{t-(D-1)\tau})$ is neglected here to simplify the parameter estimation. In principle it should be taken into account in terms of a modified cost function resulting in a nonlinear minimization problem for determining the model parameters.

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C* (Cambridge University Press, Cambridge, England, 1994).

[24] M. Korenberg, Biol. Cybern. **60**, 267 (1989).

[25] S. Geman, E. Bienenstock, and R. Doursat, Neural Comput. **4**, 1 (1992).

[26] U. Parlitz, Int. J. Bifurcation Chaos Appl. Sci. Eng. **2**, 155 (1991).

[27] To compute iterated multistep predictions vector valued models are required. A simple scheme can be devised to implement this with delay input vectors $\mathbf{x}$ and scalar outputs $g$.

[28] L. Jaeger and H. Kantz, Chaos **6**, 440 (1996).

[29] B. Efron, *The Jackknife, the Bootstrap and Other Resampling Plans* (SIAM, Philadelphia, 1982).

[30] C. J. Merz and M. J. Pazzani, Mach. Learn. **36**, 9 (1999).

[31] L. Breiman, Tech. Report No. 421, Department of Statistics, University of California, Berkeley, 1994.

[32] Y. Freund and R. Shapire, in *13th International Conference on Machine Learning*, 1996, pp. 148–156.

[33] R. Avnimelech and N. Intrator, Neural Comput. **11**, 499 (1999).