Neural networks for prediction and control of chaotic fluidized bed hydrodynamics: a first step

Citation for published version (APA):

Bakker, R., Korte, de, R. J., Schouten, J. C., Bleek, van den, C. M., & Takens, F. (1997). Neural networks for prediction and control of chaotic fluidized bed hydrodynamics: a first step. *Fractals*, 5(3), 523-530.
<https://doi.org/10.1142/S0218348X97000413>

DOI:

[10.1142/S0218348X97000413](https://doi.org/10.1142/S0218348X97000413)

Document status and date:

Published: 01/01/1997

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

NEURAL NETWORKS FOR PREDICTION AND CONTROL OF CHAOTIC FLUIDIZED BED HYDRODYNAMICS: A FIRST STEP

R. BAKKER*, R. J. DE KORTE,
J. C. SCHOUTEN, and C. M. VAN DEN BLEEK
*Department of Chemical Process Technology,
Julianalaan 136, 2628 BL Delft, The Netherlands*

F. TAKENS
*Department of Mathematics, University of Groningen,
P.O. Box 800, 9700 AV Groningen, The Netherlands*

Received August 15, 1996; Accepted January 5, 1997

Abstract

A neural-network-based model that has learnt the chaotic hydrodynamics of a fluidized bed reactor is presented. The network is trained on measured electrical capacitance tomography data. A training algorithm is used that does not only minimize the short-term prediction error but also the information needed to synchronize the model with the real system. This forces the model to focus more on learning the longer term dynamics of the system, expressed in the average multi-step-ahead prediction error and dynamic invariants such as correlation entropy and dimension. The availability of the model is an important step towards control of chaos in gas-solid fluidized beds.

1. INTRODUCTION

In the last 10 years, many physical systems that exhibit seemingly random behavior have been demonstrated to be low-dimensional chaotic. In practice, presence of chaos may be undesirable, and insights

from nonlinear dynamics can be used to account for chaos in process design. Ott, Grebogi and Yorke (OGY)¹ developed a control scheme that can be used to exploit chaotic behavior. The OGY method is based on the observation that the attractor of a chaotic system typically contains an infinite number

*E-mail: r.bakker@stm.tudelft.nl

of unstable periodic orbits (UPOs). All that is needed to change the system behavior from chaotic to periodic is to select one of these UPOs and stabilize it. Due to the sensitivity of chaotic systems for small perturbations, this can be achieved by using just very small control actions. The possibility to select different kinds of periodic behavior by just selecting different UPOs makes this type of control very appealing.

Recent research in our group has focused on the characterization of the hydrodynamics of gas-solids fluidized beds. Fluidized beds are among the most important reactor systems in chemical industry, offering excellent fluid mixing and high heat and mass transfer rates. Experimental evidence exists that fluidized beds can exhibit low-dimensional chaotic behavior.^{2,3} With the new possibilities to control chaos, offered by the OGY method, we envisage a way to make use of chaos in fluidized beds. It may be possible to find different kinds of periodic behavior embedded in the attractor of a fluidized bed, and to stabilize these using an OGY type of control. This way, the bubble size distribution in the bed may be optimized without much control effort, to improve conversion and selectivity of the chemical process in the reactor.

As a first step towards chaos control of the fluidized bed, we have chosen to test control algorithms on a comprehensive experimental system, a driven and damped pendulum. We controlled this system using a semi-continuous extension of the original OGY scheme, using delay coordinates of a single measured variable,⁴ and a neural-network-based process model.⁵ The neural network model opens the possibility to apply chaos control to systems with unknown chaotic dynamics, and consequently, the next step of our research has been to develop such a model for a fluidized bed.

In this paper, we present an example of a neural network based model of a gas-solid fluidized bed, trained with a new algorithm that does not only optimize the predictive capabilities of the neural network model, but also minimizes the information that the model uses to follow the real system (Sec. 2). The model is trained to learn the chaotic dynamics of electrical capacitance tomography (ECT) time series, measured in a 28.4 cm internal diameter fluidized bed (Sec. 3), and the chaotic features of the resulting model are discussed. The paper ends with concluding remarks

about the training algorithm and the resulting fluidized bed model (Sec. 4).

2. NEURAL NETWORK FOR LEARNING CHAOTIC DYNAMICS

2.1 Neural Network Modeling of Chaotic Systems

The evolution of a deterministic chaotic system in discrete time is given by the map

$$\mathbf{z}_{t+1} = \mathbf{F}(\mathbf{z}_t), \quad (1)$$

where $\mathbf{z}_t$ is the state of the system at discrete time t . The modeling of an experimental chaotic system consists of two steps:

- (i) find a representation of the state $\mathbf{z}$ in terms of variables that can be measured;
- (ii) approximate the nonlinear function $\mathbf{F}$, using information from the measured time series only.

For the first step, the measurement of at least one variable, representative for the system, is needed.⁶ The system's state can be represented by a sequence of delayed values of this variable. For the second step, we can use a neural network. A neural network consists of a number of simple basis functions connected to each other. Together they form a nonlinear function $\mathbf{f}_w(\mathbf{z})$. The shape of this function can be varied to minimize the difference between $\mathbf{F}$ and $\mathbf{f}_w$, by adjusting the weights $\mathbf{w}$ of the network. The neural network we use is the standard multi-layer perceptron (MLP) with sigmoidal nodes, see Fig. 1 and Bakker et al.⁵ for details. This type of network has been used in many studies⁷⁻⁹ to model chaotic systems.

When building a neural network model of a chaotic system, choices to make are what kind of measurements to use, how many delayed values of each variable to take into account, and how big a network to use. This requires trial and error, see for an example the fluidized bed model in Sec. 3. In the remainder of this section we describe two extensions of the neural network modeling approach that make it easier to handle complex and noisy data. These are (i) input data reduction with principal component analysis (Sec. 2.2), and (ii) the application of a training objective that improves the multi-step-ahead prediction error of the model (Sec. 2.3).

Fig. 1 Multilayer perceptron (MLP) neural network having six inputs ($x_1, \dots, x_6$), one output (y_1) and eight sigmoidal nodes in a single hidden layer.

2.2 Input Data Reduction with PCA

To reduce the complexity and number of adjustable parameters of the model, it is important to keep the number of inputs of the network and the number of nodes as small as possible. A technique that we have found very useful here is principal component analysis¹⁰ (PCA). PCA reduces the number of (probably correlated) input variables by creating a new set of uncorrelated variables such that the original input variables can be reconstructed without losing much accuracy, expressed as a percentage of the original data's variance. PCA can be efficiently implemented in a dynamic neural network. Figure 2 illustrates how the old state z_{t-1} of the system (consisting of principal components), is put into the model together with current measurements y_t . In the update block, the new principal

Fig. 2 Neural network prediction model using principal component inputs. In the update block, the new principal component state z_t is computed from the previous state z_{t-1} and new measured inputs y_t .

component state z_t is computed. See the appendix for details of this computation. Note that the network architecture in Fig. 2 is very similar to that proposed by Elman.¹¹ The difference is that here the state-update is a linear operation and the weights that connect the update block to the network are fixed, pre-computed with PCA.

2.3 Training Objective to Learn Chaotic Dynamics

Training of the neural network consists of the following three steps:

- (i) The network output is computed for a number of consecutive time steps. At each time step, the network state is updated with new data from the measured time series.
- (ii) The prediction error, i.e., the squared difference between predicted and measured output, is backpropagated through the network and through time¹² to compute the gradient of the prediction error with respect to all the adjustable parameters in the network.
- (iii) The gradient information is used to adjust the weights of the neural network in order to minimize the prediction error.

This training procedure minimizes the one-step-ahead prediction error of the neural network model. Deco and Schürmann⁷ investigated whether this also implies that the model learns the system's dynamic invariants, a requirement for a model in order to find UPOs.⁵ They compared the largest Lyapunov exponent of measured and model-generated time series and found that during training, the difference gets small first but eventually, when the prediction error is still decreasing, it starts to increase. This "dynamical overtraining" indicates that if the objective is to learn the global dynamics (expressed in invariants such as correlation dimension and entropy) of the system, minimizing the one-step-ahead error is not very appropriate. Kuo and Principe⁹ proposed to let the model predict more than one step ahead and minimize the error up to a prediction horizon of fixed length. This fixed length should be chosen such that it does not exceed the predictability limit of the chaotic system. We have found⁵ for the chaotic pendulum that the predictability highly depends on the state of the system, and therefore we here propose an alternative algorithm.

Fig. 3 Illustration of the alternative training objective. In the standard approach (a), only the one-step-ahead prediction error is minimized, whereas in the approach adopted here (b), also the synchronization effort is minimized.

Instead of either resetting the neural network predicted output to the measured output at each time step [standard approach, Fig. 3(a)], or not resetting the neural network predicted output at all for a fixed number of time steps (approach of Kuo and Principe⁹), we adjust the neural network output partially towards the measured output at each time step [Fig. 3(b)] and call this partial reset a correction. Figure 3(a) shows that if the model is completely reset towards the measured time series, it will closely follow the measured time-series even if it has only learnt the *short-term* dynamics of the system. However, if the corrections are made smaller, the model can only follow the measured time series if it has learnt the *long-term* dynamics of the system. Therefore, we propose here to use the following training objective:

$$E = \sum_{t=1}^T \|d_t\|^2 + \gamma \sum_{t=1}^T \|c_t\|^2, \quad (2)$$

where E is the error function to be minimized, c_t is the correction at time t , d_t is the error that remains after this correction, T is the total length of the measured time series, and γ is a parameter that determines to what extent learning of *long-term* dynamics is stressed, $\gamma \geq 0$. We let the corrections c_t be a fraction of e_t according to

$$c_t = s e_t^T, \quad (3)$$

where the factors s are additional outputs of the neural network predictor, restricted to lie between 0 and 1. This way, s depends on the state of the system and during training of the neural network it is decided how the size of the corrections should depend on the state of the system.

3. NEURAL NETWORK MODEL OF FLUIDIZED BED

3.1 Fluidized Bed Time Series Data

Information about the hydrodynamics of fluidized beds can be obtained from a time-series of some characteristic property. In our experimental setup we can measure pressure and/or voidage at some location in the reactor. The bed has an internal diameter of 28.4 cm, is filled with 560 μm polystyrene particles, has a settled bed height of 39 cm and is fluidized with a gas velocity of 3.25 times the minimum fluidization velocity. For the neural network model presented in this paper, we use electrical capacitance tomography (ECT) measurements. This technique can be used to reconstruct images of material distributions over a cross section at a fixed height in the reactor. The measurement system consists of a tomographic ring containing 12 electrodes, mounted at a fixed height (39 cm) to the reactor, see Fig. 4 and Kühn et al.¹³ The capacitance between each pair of electrodes is measured, so 66 capacitances are obtained, at a frequency of 100 Hz. The computation of material distributions from the capacitance data is complicated,¹³ but since we are only looking for a variable that reflects the chaotic dynamics of the system, it is sufficient just to know that the data contains information about the material distribution and simply use the 'raw' capacitance data. It is known that the data contain much more information about the material distribution near the wall (where the electrodes are) than in the center of the column. To keep the model simple, we do not want to take the detailed information near the wall into account and therefore we apply

Fig. 4 Cross section of the 12-electrode capacitance sensor as mounted to the 28.4 cm (inner diameter) fluidized bed. The axial electrode length equals 5 cm. [Figure adapted from Kühn et. al.¹³]

principal component analysis (PCA) to reduce the 66 capacitances to only three principal components, explaining almost 90% of the variance of the data (this application of PCA should not be confused with that mentioned in Sec. 2.2). Figure 5(a) shows 20 seconds of the three time series thus obtained. Recall that the data has now lost its physical meaning, but nevertheless reflects the chaotic dynamics of the system.

3.2 Neural Network Model Trained on ECT Data

A neural network model was trained on 4000 points of the time series from Fig. 5(a). The network input consists of 30 delays of each of the three variables. This number of delays is about as large as the average number of points between two zero crossings of the first time series. The total number of 90 inputs was reduced with PCA to 15 principal components (see Sec. 2.2), explaining 99.5% of the variance. The neural network predictor (Fig. 2) had a single hidden layer containing 20 sigmoidal nodes. The model was trained with the algorithm from Sec. 2.3, using for γ the value 1. Figure 5(b) shows the time series generated by the neural network model. In Table 1, some statistical and chaotic characteristics of the measured and model-generated time-series are compared. The numbers show that the behavior of the deterministic neural network model is chaotic, with approximately the same entropy and

Fig. 5 Time series data: (a) obtained from tomography measurements in the fluidized bed. The 66 measured capacitance time series were each scaled to zero mean and unit variance, then reduced with PCA to 3 principal component (pc) time-series which were again scaled to zero mean and unit variance; (b) generated by neural network model, starting 4000 steps after the (random) initial state.

dimension as the real system. In future work, a more elaborate comparison will be made by looking at the distribution of points in state space, for example by comparing the correlation integrals of both cases. The practical use of the model will be evaluated by testing its use for chaos control.

In Fig. 6, the multi-step-ahead prediction error curves for networks trained with different values of γ [Eq. (2)] are shown. It can be seen that the algorithm yields a better multi-step-ahead prediction error compared to the standard approach, at the cost of a lower one-step-ahead accuracy.

4. CONCLUDING REMARKS

A neural-network-based model for the chaotic hydrodynamics in a gas-solids fluidized bed has been developed. The model is trained on a multi-variate time series, obtained from electrical capacitance tomography measurements. Since it has been observed that training a model to make short-term predictions does not imply that the model learns the dynamic invariants of the system, a new algorithm

Table 1 Comparison between characteristic numbers of measured and model generated time series.

	AAD			cycle frequency			KML			DML		
	y^1	y^2	y^3	y^1	y^2	y^3	y^1	y^2	y^3	y^1	y^2	y^3
measured	0.80	0.62	0.64	2.5	2.7	2.4	12.1	12.1	13.0	5.3	5.1	5.7
model	0.81	0.80	0.79	2.4	2.2	2.2	10.6	9.6	11.0	5.1	5.4	5.5

Measured and model generated time-series consist of 16000 samples each.

AAD — absolute average deviation from signal mean,

cycle frequency — average number of upward zero-crossings per second,

KML — maximum likelihood estimate of correlation entropy¹⁴,

DML — maximum likelihood estimate of correlation dimension¹⁵.

All numbers were computed with the time-series analysis software package RRCHAOS (Reactor Research Foundation, Delft, The Netherlands, 1994), using default parameter settings.

Fig. 6 Multi-step-ahead prediction error as a function of prediction time. The mean squared error (MSE) is defined as the sum of squared errors divided by the number of points and the number of variables. Shown are curves for the standard and alternative training objective, with two different values for γ [Eq. (3)].

is tested that minimizes both the prediction error and the information that the model needs to reach this error. This information is made adjustable by resetting the state of the model only partially towards the measured state after each prediction. The multi-step-ahead error curves (Fig. 6) indicate that the alternative training objective can be very useful. The concepts of the algorithm can be further verified by investigating whether the size

of the corrections produced by the trained neural network does show a correlation with the local Lyapunov exponents of the chaotic system.

Comparison between chaotic invariants (correlation entropy and dimension) of measured and model generated time series shows that the model has captured some important characteristics of the real system that are important for chaos control. The ultimate test of the model will be its application in chaos control of fluidized beds, using similar techniques as for the control of an experimental chaotic pendulum in previous work.⁵

We notice that the modeling approach adopted here yields a model that is a black-box mathematical description of observed behavior. This is satisfactory for our purposes, but it leaves no room for physical interpretation. The many parameters involved in the modeling procedure need to be set according to trial-and-error methods or experience with the problem at hand. For the fluidized bed we need to decide how to preprocess the measured data and how many delays of each input variable, how many principal components, how big a neural network, and what value for the parameter γ to use. The fact that the training of the neural network involves nonlinear minimization makes it difficult to see how the choices of these parameters affect the resulting model, since a different initialization of the neural network will give a (usually slightly) different result. We hope to find some rules of thumb for a good choice of parameter settings, based on experience from many training sessions that are currently running.

ACKNOWLEDGMENTS

This work is supported by the Netherlands Foundation for Chemical Research (SON) with financial aid from the Netherlands Organization for Scientific Research (NWO).

REFERENCES

1. E. Ott, C. Grebogi and J.A. Yorke, *Phys. Rev. Lett.* **64**(11), 1196 (1990).
2. J. C. Schouten, M. L. M. vander Stappen and C. M. van den Bleek, *Chem. Eng. Sci.* **51**(10), 1991 (1996).
3. C. S. Daw et. al., *Phys. Rev. Lett.* **75**(12), 2308 (1995).
4. R. J. de Korte, J. C. Schouten and C.M. van den Bleek, *Phys. Rev. E* **52**(4-A), 3358 (1995).
5. R. Bakker, J. C. Schouten, F. Takens, and C. M. van den Bleek, *Phys. Rev. E* **54**(4-A), 3545 (1996).
6. F. Takens, *Lecture Notes in Mathematics* **898**, 366 (1981).
7. G. Deco and B. Schürmann, *IEICE Trans. Fundamentals* **E77-A**, 1840 (1994).
8. J. L. Hudson, M. Kube, R. A. Adomaitis, I. G. Kevrekidis, A. S. Lapedes and R. M. Farber, *Chem. Eng. Sci.* **45**(8), 2075 (1990).
9. Jyh-M. Kuo and J. C. Principe, *Proc. IEEE Int. Conf. on Neural Networks* **5**, 3131 (1994).
10. J. E. Jackson, *A User's Guide to Principal Components* (John Wiley, 1991).
11. J. L. Elman, *Cognitive Science* **14**, 179–211 (1990).
12. P. J. Werbos, Ph. D. Thesis (Harvard University, 1974).
13. F. T. Kühn, J. C. Schouten, R. F. Mudde, C. M. van den Bleek and B. Scarlett, *Meas. Sci. Technol.* **7**, 361 (1996).
14. J. C. Schouten, F. Takens and C. M. van den Bleek, *Phys. Rev. E* **49**(1), 126 (1994).
15. F. Takens, in *Dynamical Systems and Bifurcations*, eds. B. L. J. Braaksma, H. W. Broer and F. Takens, *Lecture Notes in Mathematics*, Vol. 1125 (Springer-Verlag, Berlin, 1984), 990–106.

APPENDIX

Here we explain the method used when updating a principal component state from the previous state and a new, scalar, measurement. The method is easily extended to the case of multivariate time-series (used for the fluidized bed model), but to keep notation simple this is not shown here.

Principal component analysis (PCA) applied to a series of N vectors $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ yields a set of

N principal component vectors $\{\mathbf{z}_1, \dots, \mathbf{z}_N\}$ of reduced dimension. These vectors are computed with

$$\mathbf{z}_n = U^T \mathbf{x}_n, \quad (4)$$

where U is an orthonormal matrix obtained with PCA. The original vectors can be recovered, to some degree of accuracy, with

$$\mathbf{x}_n \approx U \mathbf{z}_n. \quad (5)$$

The aim of PCA is to maximize the data reduction (the dimension of $\mathbf{z}$ compared to that of $\mathbf{x}$) without losing much accuracy when recovering $\mathbf{x}$ by Eq. (5). See for example Jackson¹⁰ or a textbook on multivariate analysis for more details.

When applying PCA to delay vectors obtained from a time series $\mathbf{y} = \{y_1, \dots, y_T\}$, we write Eq. (4) in different notation

$$\mathbf{z}(t) = U^T \mathbf{x}(t) \quad (6)$$

where

$$\mathbf{x}(t) = (y_t, \dots, y_{t-m+1})^T \quad (7)$$

As mentioned in Sec. 2.2, it has many advantages to use the principal component vectors $\mathbf{z}(t)$ instead of the delay vectors $\mathbf{x}(t)$ as inputs for the neural network prediction model. The vector $\mathbf{z}(t)$ is then called the principal component state, and as shown in Fig. 2, we need an update formula to compute $\mathbf{z}(t+1)$ from the previous state $\mathbf{z}(t)$ and a new measurement y_{t+1} .

The most straightforward way to do this is to store measurements $\{y_t, \dots, y_{t-m+1}\}$ and compute the principal components directly with Eqs. (6) and (7).

Alternatively, we can first estimate $\mathbf{x}(t)$ from $\mathbf{z}(t)$

$$\mathbf{x}(t) \approx U \mathbf{z}(t), \quad (8)$$

then update $\mathbf{x}(t)$ with the new measurement

$$\mathbf{x}(t+1) = (y_{t+1}, x_1(t), \dots, x_{m-1}(t))^T, \quad (9)$$

and finally compute the updated principal component state

$$\mathbf{z}(t+1) = U^T \mathbf{x}(t+1). \quad (10)$$

The computations [Eqs. (8) to (10)] can be done very efficiently. The calculation of the i^{th} element of $\mathbf{z}(t+1)$,

$$z_i(t+1) = \sum_{k=1}^m U_{i,k}^T x_k(t+1), \quad (11)$$

can be split in two parts

$$z_i(t+1) = \sum_{k=2}^m U_{i,k}^T x_k(t+1) + U_{i,1}^T x_1(t+1). \quad (12)$$

Estimate, as in Eq. (8)

$$x_k(t+1) = x_{k-1}(t) \approx \sum_{j=1}^m U_{k-1,j} z_j(t) \quad (13)$$

and Eq. (13) becomes

$$z_i(t+1) = \sum_{k=2}^m U_{i,k}^T \left(\sum_{j=1}^m U_{k-1,j} z_j(t) \right) + U_{i,1}^T x_1(t+1) \quad (14)$$

or after changing the order of summation

$$z_i(t+1) = \sum_{j=1}^m \left(\sum_{k=2}^m U_{i,k}^T U_{k-1,j} \right) z_j(t) + U_{i,1}^T x_1(t+1) \quad (15)$$

This can be written in the form

$$\mathbf{z}(t+1) = \mathbf{A}\mathbf{z}(t) + \mathbf{b}y_{t+1} \quad (16)$$

with

$$A_{ij} = \sum_{k=2}^m U_{i,k}^T U_{k-1,j} \quad (17)$$

and

$$\mathbf{b}_i = U_{i,1}^T \quad (18)$$

Equation (16) shows that computing the principal component state $\mathbf{z}(t+1)$ from $\mathbf{z}(t)$ and y_t only requires a single matrix multiplication.

The update formula [Eq. (16)] is more efficient and more easily implemented in a neural network than the direct computation [Eq. (6)], but the results of the formulas differ. This is because in Eq. (13), $\mathbf{x}(t)$ is recovered only approximately, whereas in the direct computation $\mathbf{x}(t)$ is known exactly.

Fig. 7 shows a comparison between the principal components computed with Eq. (6) and with Eq. (16) for the fluidized bed neural network model from Sec. 3.2. The difference is the largest for the principal components with the lowest importance, therefore a few extra principal components may be needed when using the update formula.

Fig. 7 Comparison between principal components (pcs) computed directly [Eq. (6)] and with the update formula [Eq. (16)]. The figure shows 500 samples of the first, 8th and 15th principal component, obtained from the fluidized bed tomography data (Sec. 3.1).