Reconstruction of Vector Fields of Dynamical Systems from Time Series Data: a Neural Network Approach

V. AVRUTIN, M. SCHANZ, F. SCHREIBER, G. WACKENHUT
Institute of Parallel and Distributed High-Performance Systems,
University of Stuttgart,
Breitwiesenstrasse 20 - 22, 70565 Stuttgart, GERMANY
{vravrout,schanzml,fkschrei,wackengg} @informatik.uni-stuttgart.de

Abstract: – A novel, neural neural network based approach for the reconstruction of vector fields of dynamical systems from time series data is presented. A perceptron–like fully connected two–layer neural network is used to adapt the unknown parameters of the vector field, whereby this work is focussed on the case where the vector field can be expresses as a multivariate polynomial in the state variables. The approach is applicable in the case of time discrete processes as well as in the case of continuous processes.

Key–Words: – neural networks, dynamical systems, vector field reconstruction.

1 Introduction

The reconstruction of vector fields of deterministic dynamical systems discrete or continuous in time from times series data is important whenever dynamical processes are observed, where the underlying mathematical description is incomplete or even lacking [1], [2]. Here one can distinguish mainly between two cases: On the one hand there are unknown phenomenological models, such as EEG, MEG or ECG models in physiology ([3], [4]), stock index models in finance theory or for instance net traffic models in the field of computer science. On the other hand there are partially known models like some problems in geophysics, meteorology or biology. In both cases a reconstruction or at least an approximation of the corresponding vector field from time series data can be useful, because the reconstructed vector field can be investigated further in more detail either by numerical simulation or in some cases even analytically. In this work we present a novel approach for the reconstruction of vector fields of dynamical systems from time series data, based on a special constructed neural network.

2 Reconstruction of vector fields

At present the work focuses on the sub-classes of dynamical systems discrete or continuous in time, i.e. maps:

$$\underline{q}(t+1) = \underline{f}\left(\underline{q}(t)\right) = \underline{q}'(t) \tag{1}$$

and ordinary differential equations (ODEs):

$$\dot{\underline{q}}(t) = \underline{f}\left(\underline{q}(t)\right) = \underline{q}'(t) \tag{2}$$

whereby $\underline{q}(t)$ represents the n-dimensional state vector at time t and $\underline{f}$ the corresponding n-dimensional vector field of the dynamical system depending on the state vector. We remark, that the vector $\underline{q}'(t)$ have not the same meaning in equations (1) and (2), but was introduced to have a consistent notation for maps and ODEs. In the first case it means the next state vector of the dynamical system and in the second one the derivative of the state vector with respect to time. However, we have used the same notation because these vectors play the same role in the reconstruction approach for dynamical systems continuous and discrete in time (see Section 3 and Fig. 1 for more details).

We assume the vector field to be of a specific type. Firstly the dimension n of the vector field or state vector is a priori known and secondly the vector field is assumed to be of a pure multivariate polynomial form[1] up to a certain order. In this case the right hand side of a map or ODE, i.e., the vector field $\underline{f}(\underline{q})$, can be written as a polynomial of the state variables $(q_1, \ldots, q_n)$ with an unknown matrix of parameters $\underline{\underline{P}}$:

$$\underline{f}\left(\underline{q}\right) = \underline{\underline{P}}\,\underline{Q} \tag{3}$$

For instance, if one uses for the reconstruction a polynomial of order m, the matrix $\underline{\underline{P}}$ and the vector $\underline{Q}$ are given by

$$\underline{\underline{P}} = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,n+1} & p_{1,n+2} & \cdots & p_{1,\binom{n+m}{m}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ p_{n,1} & p_{n,2} & \cdots & p_{n,n+1} & p_{n,n+2} & \cdots & p_{n,\binom{n+m}{m}} \end{pmatrix}$$

$$\underline{Q} = (1 \quad q_1 \quad \ldots \quad q_n \quad \ldots \quad {q_1}^m \quad \ldots \quad {q_n}^m)^T$$

Here the value $p_{1,1}$ represents a constant term in the equation of motion of the first state variable, i.e. the constant term for the first variable in the multivariate

[1] We remark, that the second restriction is not essential for the approach, because also other assumptions about the form of the vector field are possible and hence other expansions with different basis functions can be used as well. Nevertheless the multivariate polynomial represents a more or less generic form, because it can be regarded as a Taylor–expansion.

polynomial, the values $p_{1,2}$ up to $p_{1,n+1}$ correspond to the linear terms in this equation, etc.
The task of reconstruction of a multivariate polynomial vector field is defined by the determination of the matrix $\underline{\underline{P}}$ by fitting a given times series data, i.e. a given set of N vectors:

$$\{\underline{q}(t_i) \,|\, i = 1, \ldots, N\} \tag{4}$$

The fitting can be performed using either a least mean square approach, whereby the following expression has to be minimized:

$$\sum_{i=1}^{N} \left[\underline{q}'(t_i) - \underline{f}\left(\underline{q}(t_i)\right)\right]^2 \quad \stackrel{!}{=} \quad \min \tag{5}$$

or a neural network based approach presented in this work. The reason to develop this approach is the well–known fault–tolerance of neural networks. This characteristic property of neural networks is important when dealing with experimental and therefore often noisy data.

3 Neural network approach

Using the assumptions about the form of the vector field (see Eq. (3)), a specific two–layer neural network is constructed (see Fig. 1). The network consists of neurons with the identity as input and output function and a linear activation function without threshold. The network is thereby fully connected, i.e. each neuron of the input layer is connected with each neuron of the output layer. The number of neurons in the input layer n_i is given by the number of all possible polynomial terms up to the given order and is equal to the number of columns of matrix $\underline{\underline{P}}$. The number of neurons in the output layer n_o is given by the dimension of the vector field and corresponds to the number of rows of matrix $\underline{\underline{P}}$. The connections are weighted by weights w_{jk}, hence the activation a_j of each output neuron o_j is given by:

$$a_j \quad = \quad \sum_{k=1}^{n_i} w_{jk} x_k \tag{6}$$

Hereby x_k is the value of input neuron i_k which corresponds to one of the polynomial terms. Thus the weights w_{jk} of the network represent directly the values of the unknown parameters of the vector field. In order to fit these parameters the network has to be trained. Starting with arbitrary initialized values for the weights the network is given successively input vectors of dimension n_i which are calculated directly from the time series data. The corresponding output of the neural network $o_j(t_i) = \tilde{q}'_j(t_i)$ will be compared with the expected output $o_j^{\text{target}}(t_i) = q'_j(t_i)$. In the case of dynamical systems discrete in time the values $q'_j(t_i)$ are directly given by the input data at the next time step:

$$q'_j(t_i) = q_j(t_{i+1}) \tag{7}$$

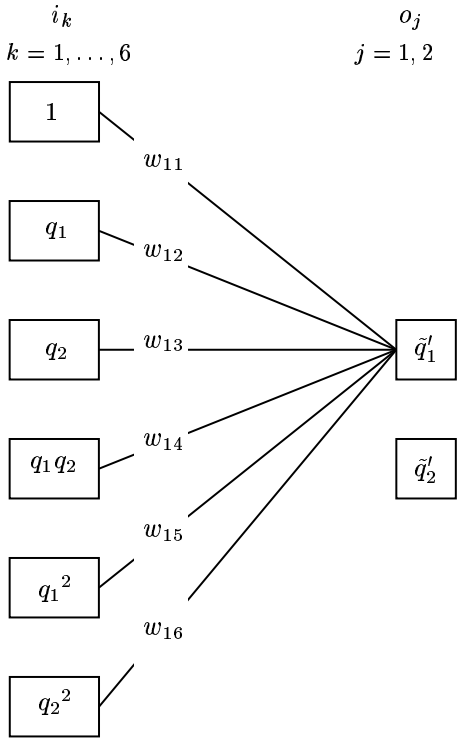


Figure 1: *Network architecture: an example for a two–dimensional dynamical system and vector field reconstruction up to the second order. The network is fully connected, but only the upper part is shown here.*

In the case of dynamical systems continuous in time the values $q'_j(t_i)$ represent the derivative of component j of the state vector with respect to time. The derivative can be approximated numerically from the time series data in a different way. For instance, we have used the three following numerical differentiation rules:

$$q'_j(t_i) \quad = \quad \frac{1}{\Delta t}\left(q_j(t_i + \Delta t) - q_j(t_i)\right) \tag{8}$$

$$q'_j(t_i) \quad = \quad \frac{1}{2\,\Delta t}\left(q_j(t_i + \Delta t) - q_j(t_i - \Delta t)\right) \tag{9}$$

$$q'_j(t_i) \quad = \quad \frac{1}{12\,\Delta t}\Big(-q_j(t_i + 2\Delta t) + 8q_j(t_i + \Delta t) - 8q_j(t_i - \Delta t) + q_j(t_i - 2\Delta t)\Big) \tag{10}$$

Here the question arises, whether there is an influence of the applied differentiation rule on the obtained reconstruction results.

During the iterations of the training phase, the learning algorithm adapts then the weights w_{jk}, such that the deviation of the output vector from the reference vector of the time series data tends to zero. After the training the parameters of the vector field can be read off from the weights of the neural network. We remark, that this modus operandi is not very typical for neural applications: in most cases the weight matrix of the network can not be directly interpreted.

	1	q_1	q_2	q_1^2	$q_1 q_2$	q_2^2
p_{jk}	1	2	3	4	5	6
1	1.0	0.0	1.0	−1.4	0.0	0.0
2	0.0	0.3	0.0	0.0	0.0	0.0
w_{jk}	1	2	3	4	5	6
1	1.0	0.0	1.00001	−1.4	0.00001	0.00001
2	0.0	0.3	0.0	0.0	0.0	0.0

Table 1: *Hénon system. Parameter setting $a = 1.4$, $b = 0.3$. The matrix $\underline{\underline{P}}$ of parameters (see Eq. (3)) of the original system (upper part, values p_{jk}) and the weights of the two–layer neural network which correspond directly to the parameters of the reconstructed system (lower part, values w_{jk}).*

4 Validation

For the validation of the approach, our aim was to use in a first step test systems in order to prove, whether the vector field of the dynamical systems under consideration can be reconstructed from pure time series data, obtained by the numerical simulation of the systems. Hereby we investigated dynamical systems discrete in time as well as continuous in time to demonstrate the applicability of the presented approach. In all examples presented here we have used the classical vanilla back-propagation method for the training of the networks. This is due to the fact, that this learning method is on the one hand a fast method, due to its simplicity, and on the other hand sufficient enough for the required accuracy as it turned out. We examined also other, more sophisticated learning methods like enhanced back-propagation, batch back-propagation, back-propagation with momentum term or Rprop with adaptive weight–decay ([5]). However, all these methods did not lead to more accurate results. It turned also out, that the usage of the above mentioned differentiation rules (Eqs. (8), (9), (10)) has no significant influence on the accuracy of the results.

Dynamical systems discrete in time.
Reconstruction up to second order terms.

As an example for dynamical systems discrete in time we have chosen the well–known *Hénon system* [6]:

$$\begin{aligned} q_1(t_{i+1}) &= 1 + q_2(t_i) - a\, {q_1}^2(t_i) \\ q_2(t_{i+1}) &= b\, q_1(t_i) \end{aligned}$$

It is known, that this system shows a chaotic dynamics at the parameter settings $a = 1.4$, $b = 0.3$. The results of the reconstruction using all terms up to the second order in the vector field, 20 000 input vectors and 10 000 training cycles are presented in Table 1. As one can see, the approximation error lies in the range of 10^{-5}, i.e., quite precise values of the parameters of the vector field are determined by the neural network.

Dynamical systems continuous in time.
Reconstruction up to second order terms.

An example for a dynamical system continuous in time is the well–known Lorenz–63 system [7].

$$\begin{aligned} \dot{x}(t) &= \sigma(y(t) - x(t)) \\ \dot{y}(t) &= x(t)(R - sz(t)) - y(t) \\ \dot{z}(t) &= sx(t)y(t) - bz(t) \end{aligned}$$

At the parameter settings $\sigma = 10.0$, $R = 175.0$ and $b = \frac{8}{3} \approx 2.667$ the system shows a chaotic dynamics. The parameter $s = 200$ is an arbitrary scaling factor, which is necessary for the usage of the SNNS–Tool (software package for the simulation of neural networks, [8], [5]). The results of the reconstruction using all terms up to the second order in the vector field, 60 000 input vectors and 10 000 training cycles are presented in the Table 2. In this example the approximation error for the reconstructed parameters of the vector field lies in the range of 10^{-2}. Although this error is much larger than in the case of the time discrete Hénon system, this result is precise enough. This means that the reconstructed system shows not only the typical chaotic dynamics of the Lorenz–63 system but has also a similar geometric structure.

In both examples presented so far, the basic approach for the vector field, which have to be reconstructed, is up to the second order, which coincides with the a priori known maximum order of the vector field of the considered system. Therefore the question arises, whether the approach works also in the case where there is no knowledge about the maximum order of the vector field and probably more high order terms than necessary will be used. This topic will be treated in more detail in the next section.

Dynamical systems continuous in time.
Reconstruction up to third order terms.

As test systems we have chosen another dynamical system continuous in time, namely the Lorenz–84 system [9]:

$$\begin{aligned} \dot{x}(t) &= \frac{aF}{s} - ax - s(y^2(t) + z^2(t)) \\ \dot{y}(t) &= \frac{G}{s} - y(t) + sx(t)y(t) - sbx(t)z(t) \\ \dot{z}(t) &= -z(t) + sbx(t)y(t) + sx(t)z(t) \end{aligned}$$

Here the parameters $a = 0.25, b = 4.0, F = 8.0, G = 1.0$ and the scaling factor $s = 4.0$ are used. At this parameter setting this system shows a chaotic behavior. The geometric structure of the corresponding chaotic strange attractor is shown in Fig. 2. The results of the reconstruction using all terms up to the third order in the vector field, 80 000 input vectors and 100 000 training cycles are presented in the Table 3. In this example the approximation error for the reconstructed parameters of the vector field is even larger than in the case of

the Lorenz–63 system. Moreover, there are some third order terms (see the terms xy^2 and xz^2 in Table 3) in the vector field which are significantly larger than zero, although these terms do not occur in the vector field of the original system. However this terms do not disturb the dynamics of the reconstructed system, so that the geometric structure is similar to the original one (see Fig. 2).

Dynamical systems continuous in time. Reconstruction using data with noise.

An important aspect of reconstruction of the vector field from data is the robustness of the approach with respect to noise. Therefore the presented approach was proved using noisy data with different noise levels (see Table 4 and Fig. 3). Instead of the input vector $\underline{q}(t_i)$ we have used vectors $\underline{q}(t_i) + \underline{\xi}$, whereby the components of the noise vectors $\underline{\xi}$ are randomly distributed corresponding to Gaussian white noise $N(\mu, \sigma)$ with mean value $\mu = 0$ and standard deviation σ. It turned out, that the dynamical system will be reconstructed for comparatively large values of the noise level σ. The reconstruction is possible until the noise level is lower than the distance between two sub–sequential states of the system:

$$||\underline{\xi}|| \lesssim ||\underline{q}(t_i) - \underline{q}(t_{i+1})|| \tag{11}$$

In Fig. 3.(e) and 3.(f) a case is presented, when this condition is not fulfilled. An one can see, the reconstruction fails in this case.

5 Summary

First results with time series data from several test systems (Hénon [6], Rössler [10], Rikitake [11], Lorenz–63 [7], Lorenz–84 [9], ...) show, that it is possible to derive the vector field of a dynamical system by the presented approach. However, due to the comparatively large dimension of the input space and hence the large number of parameters to be fitted by the network, a unique solution cannot be guaranteed. This means, if there exist several dynamical systems showing approximately the same dynamical behavior, the network determines the parameter vector of one of them. Nevertheless in the investigated cases so far, the dynamic and geometric properties of the reconstructed systems are in good coincidence with the test systems.

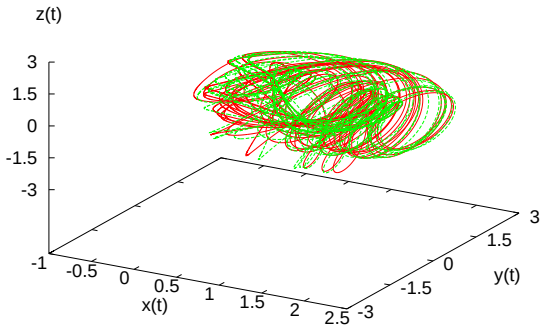


Figure 2: *Lorenz–84 system. Trajectories of the original and the reconstructed system*

6 Outlook

A lot of future work has to be done. Of course it has to be investigated whether it is possible to reconstruct the vector field on the basis of experimentally observed data, although there should be no problems concerning this topic at least if the data are sampled with an appropriate accuracy.
Furthermore a suitable measure of quality of approximation is required and has to be developed. One straight forward possibility is to use a weighted scalar product instead of the usual one applied in this work to measure the distance between the reconstructed parameter vector and the original one. However, this measure is too naive, because it is based on the parameter vector and not on the dynamic behavior of the reconstructed system. Therefore it is necessary to develop more sophisticated measures of quality of approximation which consider the similarity of the dynamic and geometric properties of the system. First of all one can think about the averaged distance between the trajectories of the reconstructed and the original system. This measure is only applicable if one scales and translates the systems appropriate. Moreover it must be remarked, that at least for dynamical systems showing a chaotic behavior the distance mentioned above has to be calculated in the sense of measurable sets. But even more is possible if one takes into account characteristic dynamic properties like Lyapunov exponents, fractal dimensions and entropies. Additionally it is to investigate the dependency of the reconstruction quality with respect to the noise level σ and also the step size h in the case of simulated systems and the sampling rate in the case of experimental data. Finally it is interesting to prove the behavior of the system reconstructed at a specific parameter setting by varying one or several significant parameters.

References:

[**1**] Y. Kuroe, H. Kawakami, T. Mori. In Proc. of the 8$^{\text{th}}$ Japanese–German Seminar on Nonlinear Problems in Dynamical Systems – Theory and Applications –, 1998, p. 217.

[**2**] G. Gouesbet, C. Letellier. Global vector–field reconstruction by using a multivariate polynomial L_2 approximation on nets. Phys. Rev. E, 49, 1994, p. 4955.

[**3**] A. Fuchs, J.A.S. Kelso and H. Haken. Phase transitions in the human brein: spatial mode dynamics, Internatonal Journal of Bifurcation and Chaos 2, 1992, p. 917.

[4] P.L. Nunez. Electric fields of the brain. Oxford University Press, Oxford, New York, 1981.
[5] SNNS user manual, (version 4.2) http://www-ra.informatik.uni-tuebingen.de/SNNS/
[6] M. Hénon. A two-dimensional mapping with a strange attractor, Commun. Math. Phys., 50, 1976, p. 69.
[7] E.N. Lorenz. Deterministic non-periodic flows, J. Atmos. Sci., 20, 1963, p. 130.
[8] A. Zell. Simulation Neuronaler Netze. Addison-Wesley, Bonn, 1994.
[9] E. N. Lorenz. Irregularity: A Fundamental Property of the Atmosphere, Tellus A, 1984, p. 36.
[10] O.E. Rössler. Electric fields of the brain. Phys. Lett. 57A, 1976, p. 397.
[11] T. Rikitake. Oscillations of a system of disk dynamos. Proc. Cambridge Philos. Soc. 54, 1958, p. 89.

	1	x	y	z	x^2	xy	xz	y^2	yz	z^2
p_{jk}	1	2	3	4	5	6	7	8	9	10
1	0.0	−10.0	10.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	175.0	−1.0	0.0	0.0	0.0	−200.0	0.0	0.0	0.0
3	0.0	0.0	0.0	−2.667	0.0	200.0	0.0	0.0	0.0	0.0
w_{jk}	1	2	3	4	5	6	7	8	9	10
1	0.00002	−10.011	10.006	0.00005	−0.0002	−0.00004	0.0118	−0.00007	−0.008	−0.00005
2	0.0004	174.915	−0.993	−0.0005	−0.00006	0.0006	−199.915	0.0002	−0.006	−0.0007
3	−0.004	−0.0001	0.0001	−2.647	0.26227	199.865	−0.001	0.006	−0.00009	−0.02

Table 2: *Lorenz–63 system. Reconstructed with terms of second order.*

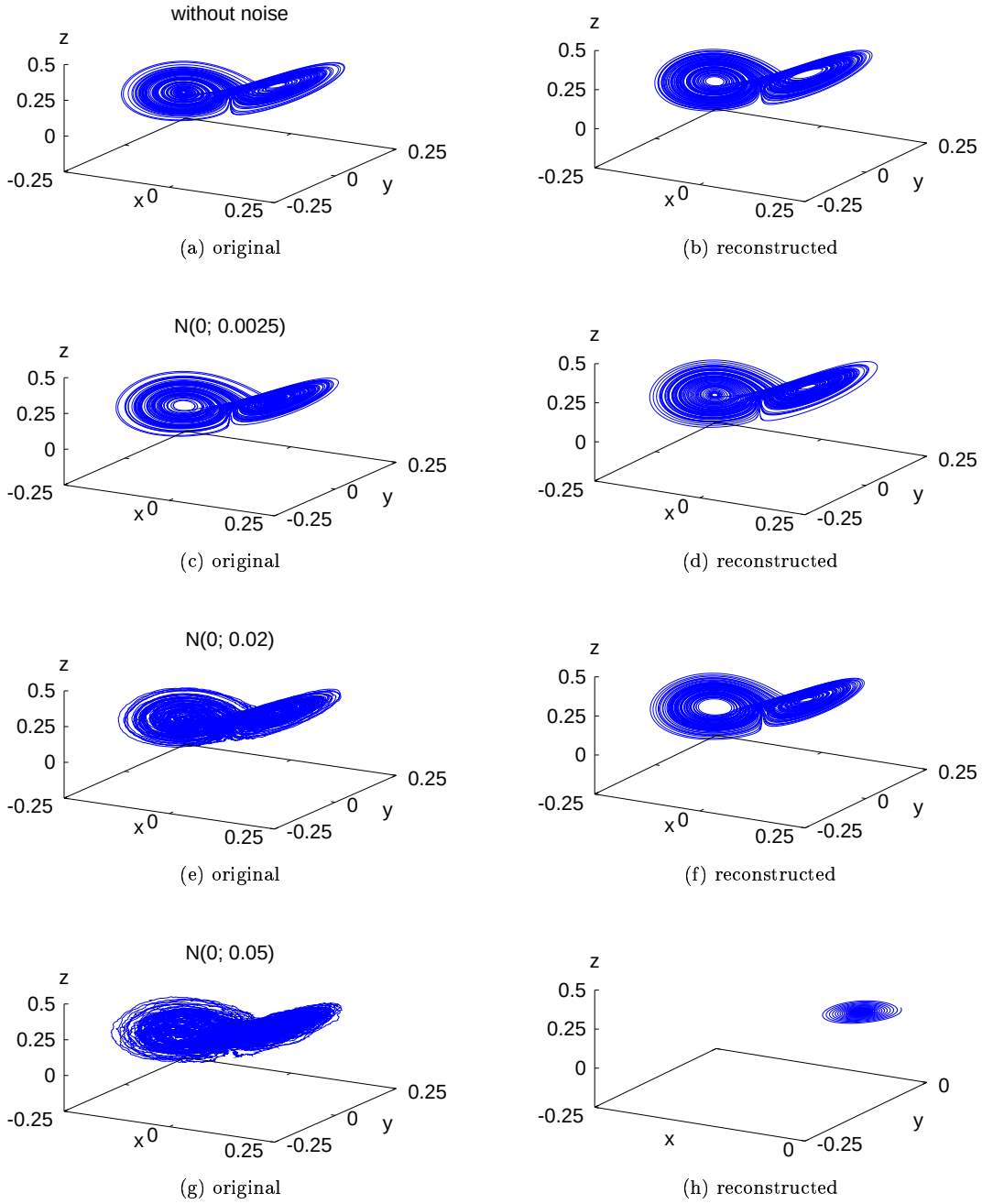
	1	x	y	z	x^2	xy	xz	y^2	yz	z^2
p_{jk}	1	2	3	4	5	6	7	8	9	10
1	0.5	−0.25	0.0	0.0	0.0	0.0	0.0	−4.0	0.0	−4.0
2	0.25	0.0	−1.0	0.0	0.0	4.0	−16.0	0.0	0.0	0.0
3	0.0	0.0	0.0	−1.0	0.0	16.0	4.0	0.0	0.0	0.0
w_{jk}	1	2	3	4	5	6	7	8	9	10
1	0.50005	−0.25023	−0.00011	−0.00009	0.00003	−0.00006	−0.00018	−3.99982	0.00003	−3.99998
2	0.24991	0.00029	−0.99988	0.00013	−0.00005	4.00017	−15.99996	0.00014	0.00004	0.00015
3	0.00014	−0.00005	−0.00021	−0.99993	−0.00001	15.99992	4.00012	0.00003	0.00000	−0.00001

	x^3	x^2y	x^2z	xy^2	xyz	xz^2	y^3	y^2z	yz^2	z^3
p_{jk}	11	12	13	14	15	16	17	18	19	20
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
w_{jk}	11	12	13	14	15	16	17	18	19	20
1	−0.00002	−0.00007	−0.00005	−0.38192	−0.00003	0.38188	0.00004	0.00003	0.00001	−0.00004
2	−0.00011	0.00016	0.00003	−0.49224	0.00012	0.49236	0.00021	0.00003	−0.00014	0.00004
3	0.00006	0.00004	0.00006	0.41868	−0.00003	−0.41863	−0.00016	0.00003	0.00009	−0.00002

Table 3: *Lorenz–84 system. Reconstructed with terms of third order.*

	Gaussian white noise			
	$N(0;0)$	$N(0;0.0025)$	$N(0;0.02)$	$N(0;0.05)$
$\|\|\underline{\Delta}\|\|$	$0.11503 \cdot 10^{-5}$	0.05801	1.11438	5.00007

Table 4: Absolute value of deviation vector $\underline{\Delta}$ (difference between the reconstructed parameter vector and the original one) for different noise levels σ.



(a) original (b) reconstructed

(c) original (d) reconstructed

(e) original (f) reconstructed

(g) original (h) reconstructed

Figure 3: *Lorenz–63 system (Eq. (11), parameter setting* $\sigma = 16.0$, $R = 50.0$, $b = 4.0$, $s = 200.0$*): comparison between the data from the original system at different noise levels* σ *and data from the corresponding reconstructed systems.*