NOISE REDUCTION IN CHAOTIC TIME SERIES BY A LOCAL PROJECTION WITH NONLINEAR CONSTRAINTS

Krzysztof Urbanowicz[†], Janusz A. Hołyst[‡],

Faculty of Physics and Center of Excellence Complex Systems Research
Warsaw University of Technology
Koszykowa 75, 00-662 Warsaw, Poland

Thomas Stemler and Hartmut Benner

Institute of Solid-State Physics
Darmstadt University of Technology
Hochschulstr. 6, D-64289 Darmstadt, Germany

(Received May 28, 2004)

On the basis of a local-projective (LP) approach we develop a method of noise reduction in time series that makes use of nonlinear constraints appearing due to the deterministic character of the underlying dynamical system. The Delaunay triangulation approach is used to find the optimal nearest neighboring points in time series. The efficiency of our method is comparable to standard LP methods but our method is more robust to the input parameter estimation. The approach has been successfully applied for separating a signal from noise in the chaotic Henon and Lorenz models as well as for noisy experimental data obtained from an electronic Chua circuit. The method works properly for a mixture of additive and dynamical noise and can be used for the noise-level detection.

PACS numbers: 05.45.Tp, 05.40.Ca

1. Introduction

It is common that observed data are contaminated by noise (for a review of methods of nonlinear time series analysis see [1–3]). The presence of noise can substantially affect such system parameters as dimension, entropy or Lyapunov exponents [4]. In fact noise can completely obscure or even

[†] e-mail: urbanow@mpipks-dresden.mpg.de
[‡] e-mail: jholyst@if.pw.edu.pl

destroy the fractal structure of a chaotic attractor [5] and even 2% of noise can make a dimension calculation misleading [6]. It follows that both from the theoretical as well as from the practical point of view it is desirable to reduce the noise level. Thanks to noise reduction [5, 7–18] it is possible *e.g.* to restore the hidden structure of an attractor which is smeared out by noise, as well as to improve the quality of predictions.

Every method of noise reduction assumes that it is possible to distinguish between noise and a *clean signal* on the basis of some objective criteria. Conventional methods such as linear filters use a power spectrum for this purpose. Low pass filters assume that a clean signal has some typical low frequency, respectively it is true for high pass filters. It follows that these methods are convenient for a regular source which generates a periodic or a quasi-periodic signal. In the case of chaotic signals linear filters cannot be used for noise reduction without a substantial disturbance of the clean signal. The reason is the broad-band spectrum of chaotic signals. It follows that for chaotic systems we make use of another generic feature of dissipative motion located on attractors that are smooth submanifolds of an admissible phase space. As results corresponding state vectors reconstructed from time delay variables are limited to geometric objects that can be locally linearized. This fact is a common background of all local projective (LP) methods of noise reduction.

Besides the LP approach there are also noise reduction methods that approximate an unknown equation of motion and use it to find corrections to state vectors. Such methods make use of neural networks [11] or a genetic programming [12] and one has to assume some basis functions *e.g.* radial basis functions [19] to reconstruct the equation of motion. Another group of methods are modified linear filters *e.g.* the Wiener filter [13], the Kalman filter [14], or methods based on wavelet analysis [15]. Applications of these methods are limited to systems with large sampling frequencies, and they are confined to the neighborhood of every point in phase space.

The method described in this paper can be considered as an extension of LP methods by taking into account constraints that occur due to the local linearization of the equation of motion of the system. We call our method the local projection with nonlinear constraints (LPNC).

The paper is organized as follows. In the following section we shall present the general background of LP methods. The LPNC method is introduced in Sec. 3 and compared with LP methods in Sec. 4. In Sec. 5 we present methods how to find the nearest neighborhood, and examples of noise reduction and estimation are introduced in Secs. 6 and 7. In the Appendix A one can find the multidimensional generalization of the solution presented in Sec. 3.

2. Local projective methods of noise reduction

Let us consider a scalar time series $\{\tilde{x}_n\}$, $n = 1, 2, ..., N$ corresponding to an experimentally accessible component of the system trajectory. We assume that in the presence of *measurement noise* instead of the clean time series $\tilde{x}_n$ we observe a noisy series x_n: $x_n = \tilde{x}_n + \eta_n$ where η_n is the noise variable. The aim of noise reduction methods is to estimate the set $\{\tilde{x}_n\}$ from the observed noisy data set $\{x_n\}$, *i.e.* to find corrections δx_n such that $x_n + \delta x_n \approx \tilde{x}_n$. The corrections δx_n can be estimated on the assumption that $\tilde{x}_n$ belongs to a clean deterministic trajectory. Let us create vectors of the system state $\tilde{\boldsymbol{x}}_n$ using the Takens Theorem [2] $\tilde{\boldsymbol{x}}_n = \{\tilde{x}_n, \tilde{x}_{n-\tau}, ..., \tilde{x}_{n-(d-1)\tau}\}$, where d is the embedding dimension, and τ is the embedding delay that further will be just 1. Now the simple approach is to use a linear approximation for the nearest neighborhood $\tilde{\boldsymbol{X}}_n^{NN}$ of a vector $\tilde{\boldsymbol{x}}_n$ and then to estimate an unknown equation of motion in the embedded space by a linear fit: $\tilde{\boldsymbol{x}}_{n+1} = \boldsymbol{A}\tilde{\boldsymbol{x}}_n + \boldsymbol{b}$. The matrix $\boldsymbol{A}$ is the corresponding Jacobi matrix and $\boldsymbol{b}$ is a constant vector.

In LP methods the local linearity of the system dynamics plays the crucial role. The unknown equation of motion of a deterministic systems $\tilde{x}_{n+1} = F(\tilde{x}_n, \tilde{x}_{n-1}, \ldots, \tilde{x}_{n-d+1})$ is equivalent to the presence of a constraint $H(\tilde{x}_{n+1}, \tilde{x}_n, \ldots, \tilde{x}_{n-d+1}) = 0$. If the embedding dimension d is larger than the dimension of the attractor then Q constraints appear:

$$H_q^{(n)}(\tilde{x}_{n+1}, \tilde{x}_n, \ldots, \tilde{x}_{n-d+1}) = 0 \quad \text{and} \quad q = 1, \ldots, Q \leq d\,, \tag{1}$$

where Q depends on the rounded up dimension d_a of the attractor, $Q = d + 1 - d_a$. Since we apply a linear approximation for vectors $\tilde{\boldsymbol{x}}_i \in \tilde{\boldsymbol{X}}_n^{NN}$ the constraints (1) can be written as

$$H_q^{(n)}(\tilde{x}_{n+1}, \tilde{x}_n, \ldots, \tilde{x}_{n-d+1}) = \sum_{j=1}^{d-Q} a_j^{q,(n)} \tilde{x}_{i-j+1-q} + b^{q,(n)} - \tilde{x}_{i+1-q} = 0\,, \tag{2}$$

where $a_j^{q,(n)}$ and $b^{q,(n)}$ are elements of $\boldsymbol{A}$ and $\boldsymbol{b}$ respectively. The main problem of LP methods is to find a tangent subspace determined by the linear constraints $H_q^{(n)}(\tilde{x}_{n+1}, \tilde{x}_n, \ldots, \tilde{x}_{n-d+1}) = 0$ and to perform an appropriate projection on this subspace. Different LP approaches make use of different projecting methods, however tangent subspaces are found in the same manner by all methods, *i.e.* the subspace should fulfill the condition (2) and the condition $\left\langle |x_i - \tilde{x}_i|^2 \right\rangle = \min$.

2.1. Cawley–Hsu–Sauer method (CHS)

The method makes use of a perpendicular projection on a subspace corresponding to the constraints (2) [16,17]. Since there are several constraints (2) and the same data will occur in several Takens vectors $\boldsymbol{x}_n$ there are many possible corrections $\delta x_{n,q}$ to the same observed data x_n. In the CHS method one makes a compromise between different corrections by taking the average

$$\tilde{\boldsymbol{x}}_n = \boldsymbol{x}_n + \alpha \sum_{q=1}^{Q} \delta \boldsymbol{x}_{n,q} \,, \tag{3}$$

where

$$\delta \boldsymbol{x}_{n,q} = -H_q^{(n)} \left(x_{n+1}, x_n, \ldots, x_{n-d+1}\right) \frac{\bigtriangledown_n H_q^{(n)}}{\left\| \bigtriangledown_n H_q^{(n)} \right\|^2} \tag{4}$$

is the correction of $\boldsymbol{x}_n$ obtained due to the constraint

$$H_q^{(n)} \left(\tilde{x}_{n+1}, \tilde{x}_n, \ldots, \tilde{x}_{n-d+1}\right) = 0 \,.$$

α is some constant $0 < \alpha < 1$ and $\bigtriangledown_n H_q^{(n)} = \bigtriangledown H_q^{(n)} \left(x_{n+1}, x_n, \ldots, x_{n-d+1}\right)$ is the gradient of the constraint function.

2.2. Schreiber–Grassberger method (SG)

Instead of the perpendicular projection on the subspace defined by (2) one can perform a projection by correcting only one variable [5]. If we choose x_{n+1-r} as the corrected variable where $r \approx d/2$ then the corrections are

$$\tilde{x}_n = x_n - \alpha \frac{H_s^{(n)} \left(x_{n+1+r}, x_{n+r}, \ldots, x_{n-d+1+r}\right)}{\partial H_s^{(n)} \left(x_{n+1+r}, x_{n+r}, \ldots, x_{n-d+1+r}\right) / \partial x_n} \,, \tag{5}$$

where $s \approx \frac{Q}{2}$. The approach can be justified as follows. If the largest (unstable) Lyapunov exponent is $\lambda_u > 0$ and the smallest (stable) Lyapunov exponent is $\lambda_s < 0$ we can write $\frac{\partial x_{n+r}}{\partial x_n} \sim \mathrm{e}^{\lambda_u r}$ and $\frac{\partial x_{n-d+1+r}}{\partial x_n} \sim \mathrm{e}^{|\lambda_s|(r-d+1)}$. If $\lambda_u \approx |\lambda_s|$ then the highest precision for determining the denominator of the rhs of (5) is usually obtained for $r = d/2$:

$$\sum_{l=0}^{d/2} \left| \mathrm{e}^{\lambda_u l} + \mathrm{e}^{|\lambda_s| l} \right| |\Delta x_n| = \min_{r=0,\ldots,d} \left\{ \sum_{l=0}^{r} \left| \mathrm{e}^{\lambda_u l} \right| |\Delta x_n| + \sum_{l=0}^{d-r} \left| \mathrm{e}^{|\lambda_s| l} \right| |\Delta x_n| \right\} \,, \tag{6}$$

where Δx_n is the error connected with the variable x_n.

2.3. The optimal method of local projection (GHKSS)

In the GHKSS method [18,20] developed by Grassberger *et al.* one looks for a minimization functional that fulfills the linear constraints (2) by corresponding corrections received in a one-step procedure. The constraints (2) can be written in the equivalent form $\left(\boldsymbol{a}^{q,(n)} \cdot \tilde{\boldsymbol{y}}_n\right) + \boldsymbol{b}^{q,(n)} = 0$, where a new vector $\tilde{\boldsymbol{y}}_n = \{\tilde{x}_{n+1}, \tilde{x}_n, \ldots, \tilde{x}_{n-d+1}\}$ is introduced, the dimension of which is larger by one than the dimension of the vector $\boldsymbol{x}_n$. Vectors $\boldsymbol{a}^{q,(n)}$ should be linearly independent and appropriately normalized, so that multiple corrections of the variables are eliminated, *i.e.* $\boldsymbol{a}^{q,(n)} \cdot P\boldsymbol{a}^{q',(n)} = \delta_{qq'}$ where P is the matrix describing the metric of the system. Let $\boldsymbol{Y}_n^{NN}$ be a set corresponding to the nearest neighborhood of the vector $\boldsymbol{y}_n$. Minimizing the functional $\sum\limits_k \delta\boldsymbol{x}_k \cdot P^{-1}\delta\boldsymbol{x}_k$ for $\left\{k : \boldsymbol{y}_k \in \boldsymbol{Y}_n^{NN}\right\}$ under the above conditions we get a system of coupled equations. The next step is to consider all vectors of $\boldsymbol{Y}_n^{NN}$ and to calculate the average $\xi_i^{(n)} = \frac{1}{\left|\boldsymbol{Y}_n^{NN}\right|} \sum\limits_k x_{k+i}$, $i = 0, 1, \ldots, d$ as well as corresponding $(d+1) \times (d+1)$ covariance matrix

$$C_{ij}^{(n)} = \frac{1}{\left|\boldsymbol{Y}_n^{NN}\right|} \sum_k x_{k+i} x_{k+j} - \xi_i^{(n)} \xi_j^{(n)}. \tag{7}$$

Here $\left|\boldsymbol{Y}_n^{NN}\right|$ means the number of elements in the set. Defining $R_i = \frac{1}{\sqrt{P_i}}$ and $\Gamma_{ij}^{(n)} = R_i C_{ij}^{(n)} R_j$ one can find Q orthonormal eigenvectors of the matrix $\Gamma^{(n)}$ corresponding to its smallest eigenvalues $\boldsymbol{e}^{q,(n)}$ for $q = 1, \ldots, Q$.

Let the matrix $\Pi_{ij}^{(n)} = \sum\limits_{q=1}^{Q} e_i^{q,(n)} e_j^{q,(n)}$ define a subspace spanned by the eigenvectors $\boldsymbol{e}^{q,(n)}$. Now the corrections to the observed signal can be written as follows

$$\delta x_{n+i} = \frac{1}{R_i} \sum_{j=0}^{d} \Pi_{ij}^{(n)} R_j \left(\xi_j^{(n)} - x_{n+j}\right). \tag{8}$$

We see that the GHKSS method does not employ multiple corrections resulting from constraints (2), but only performs a smaller number of corrections following the multiple occurrence of the same variable x_n in various vectors $\boldsymbol{y}_i : x_n \in \boldsymbol{y}_i$.

The solution (8) is a generalization of the CHS and SG methods. The main difference between the CHS method and the GHKSS method is in the subspace of projection. While a perpendicular projection of points is used in the first case, projection is on a tangent subspace defined by the matrix P

in the second case. The matrix P should be diagonal and such that the first and the last component of the vector $\boldsymbol{y}_n$ have only small weights *e.g.* $P_i = 1$ for $i = 1, 2, \ldots, d-1$ and $P_i = 1000$ for $i = 0, d$.

The efficiency of noise reduction methods can be measured by the gain parameter, defined as

$$\mathcal{G} = 10 \log \left(\frac{\sigma^2_{\text{noise}}}{\sigma^2_{\text{red}}} \right) , \tag{9}$$

where $\sigma^2_{\text{noise}} = \left\langle (x_n - \tilde{x}_n)^2 \right\rangle$ is the variance of added noise and σ^2_{red} is the variance of noise left after noise reduction. The last value is calculated as the square of the distance between the vector of noise-reduced data and the vector of clean data divided by the dimension of these vectors. The definition of the gain presumes the knowledge of the clean data $\tilde{X}_n = \{\tilde{x}_n\}$.

The noise level parameter $\mathcal{N}$ can be defined as the ratio of standard noise deviation σ_{noise} to standard data deviation σ_{data}

$$\mathcal{N} = \frac{\sigma_{\text{noise}}}{\sigma_{\text{data}}} . \tag{10}$$

3. The principle of LPNC method

The LP methods described in the previous section make use of linear constraints that appear due to linear approximation of the system dynamics. Such a linear approximation has only a local character and corresponding coefficients depend, in fact, on the position in phase space. If we assume that the nearest neighborhood of every point $\tilde{\boldsymbol{x}}_n$ is characterized by the same coefficients then nonlinear constraints appear that can be used for reconstruction of the unknown deterministic trajectory. The basic advantage of the local projection with nonlinear constraints (LPNC) method introduced here as compared to LP methods is its smaller sensitivity to the input parameters estimation. A weak point of the LPNC method is its slower convergence rate with respect to the standard LP approach. The LPNC algorithm can be accelerated but at the cost of decreasing the gain parameter. Like other LP methods the LPNC method belongs to the iterative approaches. A single iteration provides only a partial noise reduction and a corrected data set serves as an input for the next iteration.

For the one-dimensional case the Jacobi matrix $\boldsymbol{A}$ and the additive vector $\boldsymbol{b}$ describing the locally linearized dynamics at point $\tilde{x}_n$ reduce to scalar coefficients $\boldsymbol{A} = a_1(\tilde{x}_n)$, $\boldsymbol{b} = b(\tilde{x}_n)$, and the linearized equation of motion at $\tilde{x}_n$ reads $\tilde{x}_{n+1} = a_1(\tilde{x}_n)\,\tilde{x}_n + b(\tilde{x}_n)$. Let us consider the nearest neighborhood $\tilde{\boldsymbol{X}}_n^{NN}$ of $\tilde{x}_n$. We assume that the set $\tilde{\boldsymbol{X}}_n^{NN}$ consists of three points $\left\{\tilde{x}_n, \tilde{x}_k, \tilde{x}_j \in \tilde{\boldsymbol{X}}_n^{NN}\right\}$ which are so *close* to each other that their locally linearized dynamics can be approximately described by *the same* pair

of coefficients $\boldsymbol{A} = a_1(\tilde{x}_n)$, $\boldsymbol{b} = b(\tilde{x}_n)$. When we write down three linear equations of motion for $\tilde{x}_n, \tilde{x}_k, \tilde{x}_j$

$$\begin{aligned} \tilde{x}_{n+1} &= a_1(\tilde{x}_n)\,\tilde{x}_n + b(\tilde{x}_n)\,, \\ \tilde{x}_{k+1} &= a_1(\tilde{x}_n)\,\tilde{x}_k + b(\tilde{x}_n)\,, \\ \tilde{x}_{j+1} &= a_1(\tilde{x}_n)\,\tilde{x}_j + b(\tilde{x}_n) \end{aligned} \tag{11}$$

the coefficients $a_1(\tilde{x}_n)$ and $b(\tilde{x}_n)$ can be eliminated. After elimination we get a constraint that has to be fulfilled by the system variables for consistency reasons.

$$\begin{aligned} G\left(\tilde{\boldsymbol{X}}_n^{NN}\right) \equiv\ & \tilde{x}_n(\tilde{x}_{k+1} - \tilde{x}_{j+1}) \\ & + \tilde{x}_k(\tilde{x}_{j+1} - \tilde{x}_{n+1}) + \tilde{x}_j(\tilde{x}_{n+1} - \tilde{x}_{k+1}) = 0\,. \end{aligned} \tag{12}$$

In the case of a higher dimension $d > 1$ we have three equations of motions but the number of unknown constants is larger than two, *i.e.*

$$\begin{aligned} \tilde{x}_{n+1} &= \sum_{i=1}^{d} a_i(\tilde{x}_n)\,\tilde{x}_{n-i+1} + b(\tilde{x}_n)\,, \\ \tilde{x}_{k+1} &= \sum_{i=1}^{d} a_i(\tilde{x}_n)\,\tilde{x}_{k-i+1} + b(\tilde{x}_n)\,, \\ \tilde{x}_{j+1} &= \sum_{i=1}^{d} a_i(\tilde{x}_n)\,\tilde{x}_{j-i+1} + b(\tilde{x}_n)\,, \end{aligned} \tag{13}$$

where $a_i(\tilde{x}_n)$ are elements of the first row of Jacobi matrix and $b(\tilde{x}_n)$ is a constant. The corresponding constraint G^d for higher dimensional case is as follows

$$\begin{aligned} G^d\left(\tilde{\boldsymbol{X}}_n^{NN}\right) \equiv\ & \left(\sum_{i=1}^{d} a_i x_{n-i+1}\right)(\tilde{x}_{k+1} - \tilde{x}_{j+1}) \\ & + \left(\sum_{i=1}^{d} a_i x_{k-i+1}\right)(\tilde{x}_{j+1} - \tilde{x}_{n+1}) \\ & + \left(\sum_{i=1}^{d} a_i x_{j-i+1}\right)(\tilde{x}_{n+1} - \tilde{x}_{k+1}) = 0\,. \end{aligned} \tag{14}$$

The extended constraints and the corresponding calculations that are valid for all rows of Jacobi matrix $\boldsymbol{A}$ are presented in Appendix A. The condition (12) and (14) should be fulfilled for every point $\tilde{x}_n$ and its nearest neighborhood $\tilde{\boldsymbol{X}}_n^{NN}$. Similarly as in LP methods these constraints are ensured in

the LPNC approach by application of the method of Lagrange multipliers to an appropriate cost function. Since we expect that corrections to noisy data should be as small as possible, the cost function can be assumed to be the sum of squared corrections $S = \sum_{s=1}^{N} (\delta x_s)^2$.

It follows that we are looking for the minimum of the functional

$$\tilde{S} = \sum_{n=1}^{N} (\delta x_n)^2 + \sum_{n=1}^{N} \lambda_n G^d \left(\tilde{\boldsymbol{X}}_n^{NN} \right) = \min . \tag{15}$$

After finding zero points of $2N$ partial derivatives one gets $2N$ equations with $2N$ unknown variables δx_n and λ_n. However, in such a case the derivatives of the functional (15) are nonlinear functions of these variables. For simplicity of computing we are interested to pose our problem in such a way that linear equations appear which can be solved by standard matrix algebra. To understand the role of nonlinearity let us write the constraint $G\left(\tilde{\boldsymbol{X}}_n^{NN} \right)$ in such a way that explicit dependence on the unknown variables is seen (the corresponding equations for $G^d(\tilde{\boldsymbol{X}}_n^{NN})$ have a similar form)

$$\begin{aligned} G\left(\tilde{\boldsymbol{X}}_n^{NN} \right) \cong\ & G\left(\boldsymbol{X}_n^{NN}, \boldsymbol{X}_{n+1} \right) + G\left(\delta \boldsymbol{X}_n, \boldsymbol{X}_{n+1} \right) \\ & + G\left(\boldsymbol{X}_n^{NN}, \delta \boldsymbol{X}_{n+1} \right) + G\left(\delta \boldsymbol{X}_n, \delta \boldsymbol{X}_{n+1} \right) . \end{aligned} \tag{16}$$

Here we introduced the following notation

$$\begin{aligned} G\left(\boldsymbol{X}_n^{NN}, \boldsymbol{X}_{n+1} \right) \equiv\ & x_n \left(x_{k+1} - x_{j+1} \right) + x_k \left(x_{j+1} - x_{n+1} \right) \\ & + x_j \left(x_{n+1} - x_{k+1} \right) , \\ G\left(\delta \boldsymbol{X}_n, \boldsymbol{X}_{n+1} \right) \equiv\ & \delta x_n \left(x_{k+1} - x_{j+1} \right) + \delta x_k \left(x_{j+1} - x_{n+1} \right) \\ & + \delta x_j \left(x_{n+1} - x_{k+1} \right) , \\ G\left(\boldsymbol{X}_n^{NN}, \delta \boldsymbol{X}_{n+1} \right) \equiv\ & x_n \left(\delta x_{k+1} - \delta x_{j+1} \right) + x_k \left(\delta x_{j+1} - \delta x_{n+1} \right) \\ & + x_j \left(\delta x_{n+1} - \delta x_{k+1} \right) , \\ G\left(\delta \boldsymbol{X}_n, \delta \boldsymbol{X}_{n+1} \right) \equiv\ & \delta x_n \left(\delta x_{k+1} - \delta x_{j+1} \right) + \delta x_k \left(\delta x_{j+1} - \delta x_{n+1} \right) \\ & + \delta x_j \left(\delta x_{n+1} - \delta x_{k+1} \right) , \end{aligned} \tag{17}$$

where $\boldsymbol{X}_n^{NN} = \{x_n, x_k, x_j\}$, $\boldsymbol{X}_{n+1} = \{x_{n+1}, x_{k+1}, x_{j+1}\}$, $\delta \boldsymbol{X}_n = \{\delta x_n, \delta x_k, \delta x_j\}$, $\delta \boldsymbol{X}_{n+1} = \{\delta x_{n+1}, \delta x_{k+1}, \delta x_{j+1}\}$ and x_k, x_j are the near neighbors of x_n. Indices are defined as $\left\{ n, j, k : x_n, x_k, x_j \in \boldsymbol{X}_n^{NN} \right\}$. Note that elements of the set $\boldsymbol{X}_{n+1}$ are not necessarily near neighbors to each other.

The approximation we use in (16) follows from the fact that in general the nearest neighborhood $\tilde{\boldsymbol{X}}_n^{NN}$ does not include the same indices as the

nearest neighborhood $\boldsymbol{X}_n^{NN}$, *i.e.*

$$\left\{k : \tilde{x}_k \in \tilde{\boldsymbol{X}}_n^{NN}\right\} \neq \left\{j : x_j \in \boldsymbol{X}_n^{NN}\right\}. \tag{18}$$

In the case of not correlated noise and under the assumption that the introduced corrections completely reduce the noise effect $\delta x_s = -\eta_s \quad (\forall_{s=1,\ldots,N})$ one can neglect the nonlinear terms in Eqs. (17) *i.e.*

$$G\left(\delta \boldsymbol{X}_n, \delta \boldsymbol{X}_{n+1}\right) \cong 0 \quad (\forall_{n=1,\ldots,N}). \tag{19}$$

In equation (19) we use the fact that $\langle \eta_i \rangle = 0$ and $\langle \eta_i \eta_j \rangle \sim \delta_{ij}$.

Taking into account the approximation (19) one can write the following *linear equation* for the problem (15)

$$\boldsymbol{M} \cdot \delta \boldsymbol{X} = \boldsymbol{B}, \tag{20}$$

where $\boldsymbol{M}$ is a matrix containing constant elements, $\boldsymbol{B}$ is a constant vector, and $\delta \boldsymbol{X}^{\mathrm{T}} = \{\delta x_1, \delta x_2, \ldots, \delta x_N, \lambda_1, \lambda_2, \ldots, \lambda_N\}$ is a vector of dependent variables (T — transposition). In practice it is very difficult or even impossible to find the solution of equation (20) for large N. First, it is time consuming to solve a linear equation with a matrix $2N \times 2N$ matrix for $N > 1000$. Second, when $\boldsymbol{M}$ becomes singular the estimation error of the inverse matrix M^{-1} is very large. Third, we cannot always find the true near neighbors (the set $\tilde{\boldsymbol{X}}_n^{NN}$) from the noisy data $\{x_i\}$. Taking into account the above reasons it is useful to replace the global minimization problem (15) by N local minimization problems related to the nearest neighborhood $\boldsymbol{X}_n^{NN}$. The corresponding local functionals to be minimized are

$$\tilde{S}_n^{NN} = \sum_s \left(\delta x_s\right)^2 + \lambda_n G^d\left(\boldsymbol{X}_n^{NN}\right) = \min \quad (\forall_{n=1,\ldots,N}),$$

where

$$\left\{s : x_s \in \boldsymbol{X}_n^{NN} \text{ or } x_s \in \boldsymbol{X}_{n+1} \text{ or } \ldots \; x_s \in \boldsymbol{X}_{n-d+1}\right\}. \tag{21}$$

We can consider the minimization problem (21) as a certain approximation of (15). Functionals (21) are linked to each other due to the fact that the same variable δx_n appears in $6 \cdot d$ different minimization problems (21). The global problem (15) is equivalent to Eq. (20) with $2N$ unknown variables that should be found single-time. The problem (21) is equivalent to a system of coupled equations that should be solved several times and as a result one gets an approximate global solution. Writing Eq. (21) in the linear form *i.e.* calculating zero sites of corresponding derivatives and using Eq. (19) one gets N linear equations as follows

$$\boldsymbol{M}_n \cdot \delta \boldsymbol{X}_n^{\lambda} = \boldsymbol{B}_n \quad (\forall_{n=1,\ldots,N}), \tag{22}$$

where $\left(\delta \boldsymbol{X}_n^{\lambda}\right)^{\mathrm{T}} = \{\delta x_n, \delta x_k, \delta x_j, \delta x_{n+1}, \delta x_{k+1}, \delta x_{j+1}, \lambda_n\}$. The matrices $\boldsymbol{M}_n$ corresponding to (21) avoid the disadvantages of (20), *i.e.* they are not singular, their dimension is smaller and they do not substantially depend on the initial approximation of near neighbors. The matrix $\boldsymbol{M}_n$ for one-dimensional case is given by

$$\boldsymbol{M}_n = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & x_{k+1} - x_{j+1} \\ 0 & 2 & 0 & 0 & 0 & 0 & x_{j+1} - x_{n+1} \\ 0 & 0 & 2 & 0 & 0 & 0 & x_{n+1} - x_{k+1} \\ 0 & 0 & 0 & 2 & 0 & 0 & x_j - x_k \\ 0 & 0 & 0 & 0 & 2 & 0 & x_n - x_j \\ 0 & 0 & 0 & 0 & 0 & 2 & x_k - x_n \\ x_{k+1} - x_{j+1} & x_{j+1} - x_{n+1} & x_{n+1} - x_{k+1} & x_j - x_k & x_n - x_j & x_k - x_n & 0 \end{bmatrix} \tag{23}$$

Vector $\boldsymbol{B}_n$ has the form $\boldsymbol{B}_n^{\mathrm{T}} = \left\{0, 0, 0, 0, 0, 0, -G\left(\boldsymbol{X}_n^{NN}, \boldsymbol{X}_{n+1}\right)\right\}$.

4. Comparing LPNC method to local projection methods

Let us illustrate the LPNC method by taking into account the cost functional (21) (it will be written as S^{LPNC})

$$\begin{aligned} S^{\mathrm{LPNC}} &= \sum_i \delta x_i^2 + \lambda[\tilde{x}_n\left(\tilde{x}_{k+1} - \tilde{x}_{j+1}\right) \\ &\quad + \tilde{x}_k\left(\tilde{x}_{j+1} - \tilde{x}_{n+1}\right) + \tilde{x}_j\left(\tilde{x}_{n+1} - \tilde{x}_{k+1}\right)] = \min. \end{aligned} \tag{24}$$

The corresponding cost function S^{GHKSS} that is used in the standard local projection method *e.g.* in the GHKSS method [18] is

$$\begin{aligned} S^{\mathrm{GHKSS}} &= \sum_i \delta x_i^2 + \lambda_1\left(\tilde{x}_n a + b - \tilde{x}_{n+1}\right) \\ &\quad + \lambda_2\left(\tilde{x}_j a + b - \tilde{x}_{j+1}\right) + \lambda_3\left(\tilde{x}_k a + b - \tilde{x}_{k+1}\right) = \min. \end{aligned} \tag{25}$$

If we were in the position to find exact solutions for the minimization problems (24) and (25) then both results would be the same since (24) can be obtained from (25) after elimination of the parameters a and b.

In both cases the variables $\left\{\tilde{x}_n, \tilde{x}_k, \tilde{x}_j \in \tilde{\boldsymbol{X}}_n^{NN}\right\}$ belong to the nearest neighborhood of the variable $\tilde{x}_n$. The index $i = \left\{k, k+p : \tilde{x}_k \in \tilde{\boldsymbol{X}}_n^{NN}\right\}$ runs through all indices of the variables appearing in (24) and (25) while

the variable $\tilde{x}_{k+p} = \left\{\tilde{x}_l : \tilde{x}_{l-p} \in \tilde{\boldsymbol{X}}_n^{NN}\right\}$ corresponds to the p-iterate of $\tilde{x}_k$. Parameters a and b can be calculated from a linearized form of the equations of motion at the point $\tilde{x}_n$.

In practice the minimization problems S^{LPNC} and S^{GHKSS} are not equivalent because in both cases different approximations are used. These differences are: *(i)* Eq. (24) is nonlinear against corrections δx_i. In this case the approximation consists in a linearization. *(ii)* For Eq. (25) the exact values of the parameters a and b are unknown. The approximation means that a and b are estimated from noisy data.

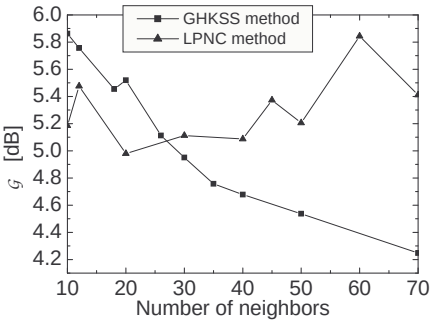

Fig. 1. The plot of the gain parameter $\mathcal{G}$ *versus* number of iterations of the GHKSS method (squares) and LPNC method (triangles). Lorenz system $\mathcal{N} = 78\%$, $N = 1000$.

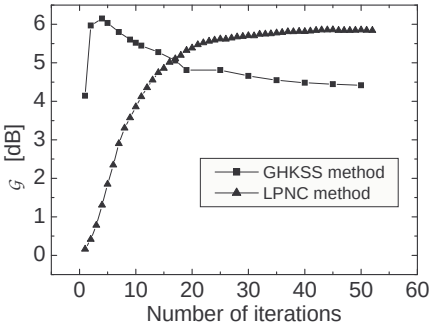

Fig. 2. The plot of the gain parameter $\mathcal{G}$ *versus* number of neighbors of the GHKSS method (squares) and LPNC method (triangles). Lorenz system $\mathcal{N} = 78\%$, $N = 1000$.

Figs. 1 and 2 present a comparison between results received by the GHKSS and LPNC methods. Fig. 1 shows that the gain parameter $\mathcal{G}$ depends on the number of neighbors, which is an input parameter of both methods. One can see that for LPNC method the gain parameter is more

robust to changes of the number of neighbors than for the GHKSS method. In Fig. 2 the dependence of the gain parameter on the number of iteration steps of the methods is shown. One can see that LPNC method finished reduction at the maximal efficiency what is not the case of GHKSS method, so the former method is easier to use since it does not need estimation of the iteration number.

If we consider uniformly distributed stochastic variables (see Fig. 3) the LPNC method reduces the noise very well, and as a result all data are represented as a neighborhood of a point attractor (see Fig. 4) while a complete noise reduction would correspond to a phase portrait consisting of a single point. In fact, for the case considered we observed for the LPNC method a noise reduction of about 96% of data variance.

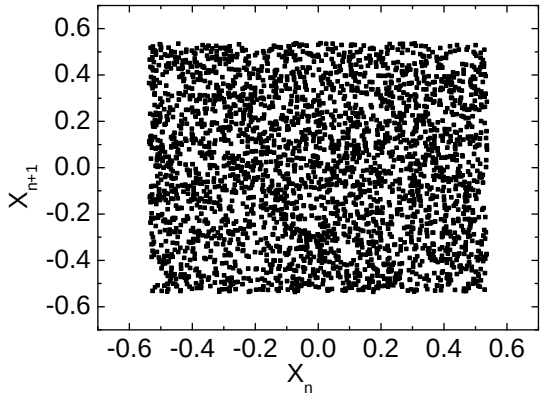

Fig. 3. The random data from uniform distribution.

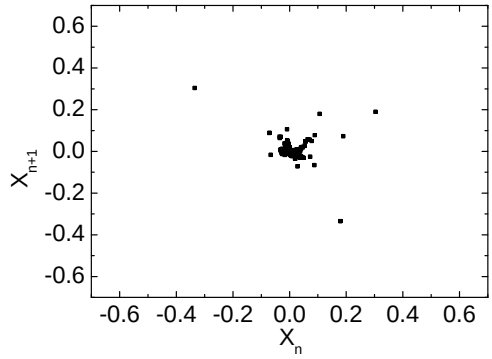

Fig. 4. The random data shown in the Fig. 3 after noise reduction with the LPNC method.

5. The nearest neighborhood assessment

The LP methods are local. It follows that features of the nearest neighborhood $\boldsymbol{X}_n^{NN}$ of every point $\boldsymbol{x}_n$ in the phase space play an important role. Usually the nearest neighborhood is estimated by the smallest distance approach that makes use of the standard Euclidian geometry. We have found, however, that our LPNC method works much better when the Delaunay triangulation approach [21] is applied for the nearest neighborhood estimation.

5.1. The smallest distance approach (SD)

In the smallest distance approach the Euclidian metric is used, *i.e.* first the distance between every pair of points in the Takens embedded space is calculated as $d_{i,j} = \sqrt{(x_i - x_j)^2 + \ldots + \left(x_{i-(d-1)\tau} - x_{j-(d-1)\tau}\right)^2}$ and then the nearest neighborhood $\boldsymbol{X}_n^{NN}$ of a point $\boldsymbol{x}_n$ is defined as a set of ν points fulfilling the relation

$$\left\{\boldsymbol{x}_j \in \boldsymbol{X}_n^{NN}, \forall_k \boldsymbol{x}_k \notin \boldsymbol{X}_n^{NN} : d_{n,j} \leq d_{n,k}\right\}. \tag{26}$$

Let us stress that this definition depends on the chosen value of the ν parameter. *i.e.* on the assumed number of near neighbors, $\nu = 2, 3, \ldots$.

5.2. The Delaunay triangulation approach (DT)

To find the nearest neighborhood relations for the LPNC method we have used the Delaunay triangulation [21]. In general the triangulation of any set of points $\boldsymbol{X} = \{\boldsymbol{x}_i\} \in \mathbb{R}^d$ is a collection of d-dimensional simplices with disjoint interiors and vertices chosen from $\boldsymbol{X}$. There are many triangulation of the same set of points $\boldsymbol{X}$. One of the best known is the Delaunay triangulation (see Fig. 5). Let $T(\boldsymbol{x}_n)$ be a part of the space $\mathbb{R}^d$ that contains all points that are closer to $\boldsymbol{x}_n$ than any other point $\boldsymbol{x}_j$ from the set $\boldsymbol{X}$

$$T(\boldsymbol{x}_n) = \left\{z \in \mathbb{R}^d, \boldsymbol{x}_j \in \boldsymbol{X} : \forall_{j \neq n} \|z - \boldsymbol{x}_n\| \leq \|z - \boldsymbol{x}_j\|\right\}. \tag{27}$$

If $\boldsymbol{m}_{i,j} = (\boldsymbol{x}_i + \boldsymbol{x}_j)/2$ belongs to both sets $T(\boldsymbol{x}_i)$ and $T(\boldsymbol{x}_j)$ then by definition the point $\boldsymbol{x}_j$ is the nearest neighbor of $\boldsymbol{x}_i$ received due to the Delaunay triangulation. By the above definition every point $\boldsymbol{x}_n$ belongs to its nearest neighborhood $\boldsymbol{x}_n \in \boldsymbol{X}_n^{NN}$.

In practice the Delaunay approach can be performed as follows. A pair of points $\boldsymbol{x}_n$ and $\boldsymbol{x}_j$ are near neighbors provided that there are no other points $\boldsymbol{x}_k$ $(k \neq j, n)$ belonging to the hypersphere centered at the point $\boldsymbol{m}_{n,j}$ and of the radius $r_{n,j} = \|\boldsymbol{x}_n - \boldsymbol{x}_j\|/2$.

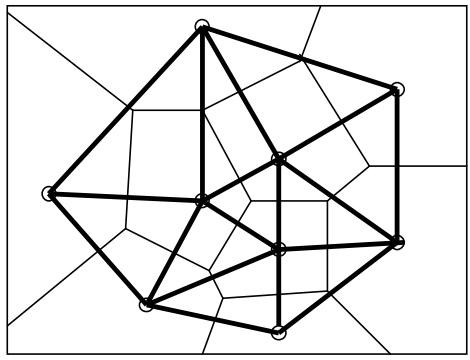

Fig. 5. Delaunay triangulation for a set of nine points in a two-dimensional space. The near neighbors are connected by bold lines. Sets $T(\boldsymbol{x}_i)$, $i = 1, 2, \ldots, 9$ are limited by thin lines.

Fig. 6. Illustration of nearest neighborhood search by DT approach (a) $\boldsymbol{x}_j$ and $\boldsymbol{x}_n$ are not near neighbors. (b) $\boldsymbol{x}_j$ and $\boldsymbol{x}_n$ are near neighbors.

In Fig. 6 two cases are presented when in the two-dimensional space a) the point $\boldsymbol{x}_j$ is not the nearest neighbor of $\boldsymbol{x}_n$ and b) the point $\boldsymbol{x}_j$ is the nearest neighbor of $\boldsymbol{x}_n$.

The DT method has the advantage that triangles appearing due to connections of near neighbors are almost equiangular (see Fig. 5). This property is the main reason for using the DT method in search of the near neighbors. The disadvantage of this method is a slowing down of numerical calculations.

6. Examples of noise reductions

The LPNC method has been applied to three systems: the Henon map, the Lorenz model [22] and the Chua circuit [23–25]. Figures 7–9 present the chaotic Henon map in the absence and in the presence of measurement noise as well as a result of the noise reduction. Table I presents the values of the gain parameter for the Henon map and for the Lorenz system.

To verify our method in a real experiment we have performed the analysis of data generated by a nonlinear electronic circuit. The Chua circuit in the chaotic regime [23, 24] has been used and we have added a measurement

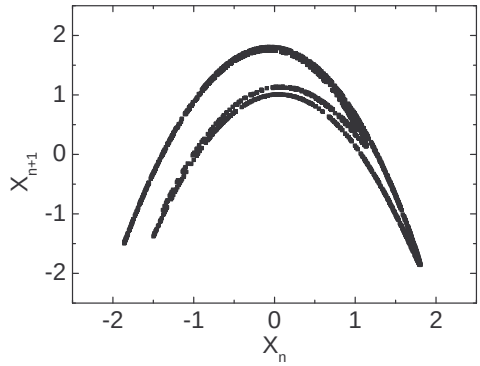

Fig. 7. Chaotic Henon map without noise

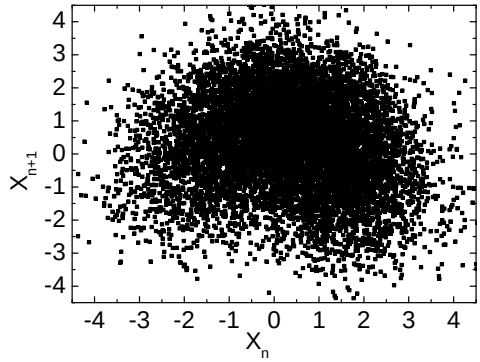

Fig. 8. Chaotic Henon map with a measurement noise $\mathcal{N} = 69\%$. Note the difference in scale.

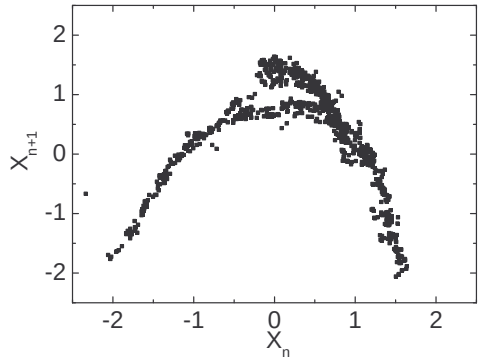

Fig. 9. Chaotic Henon map with a measurement noise after noise reduction with the LPNC method, $\mathcal{G} = 4.3$.

TABLE I

Results of noise reduction by the LPNC method for the Henon map and Lorenz model.

System	$\mathcal{N}$	$\mathcal{G}$	percent of eliminated noise
Henon $N = 1000$	10%	9.58	89%
Henon $N = 3000$	10%	10.04	91%
Henon $N = 1000$	66%	5.08	69%
Lorenz $N = 1000$	78%	5.85	74%
Lorenz $N = 3000$	76%	6.02	75%
Lorenz $N = 1000$	34%	7.21	81%

noise to the outcoming signal. The noise (white and Gaussian) came from an electronic noise generator. Figures 10–12 show a clean signal coming from this circuit, the signal generated by Chua circuit with measurement noise ($\mathcal{N} = 96.5\%$) and the same signal after the noise reduction with the LPNC method ($\mathcal{G} = 6.38$). Table II presents values of the $\mathcal{G}$ parameter and the percentage of eliminated noise for several values of the noise level in the Chua circuit.

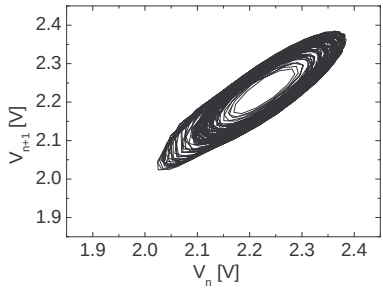

Fig. 10. The stroboscopic map corresponding to a clean trajectory in the Chua circuit.

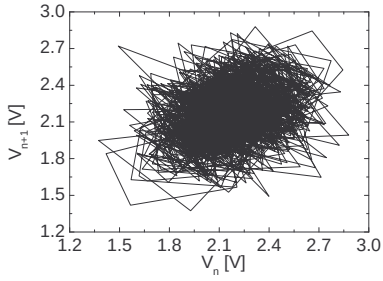

Fig. 11. The stroboscopic map received from the Chua circuit in the presence of a measurement noise $\mathcal{N} = 96.5\%$. Note the difference in scale.

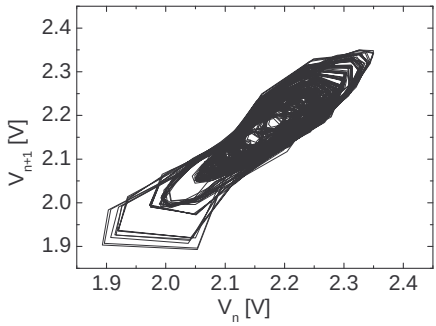

Fig. 12. The stroboscopic map received after the noise reduction by the LPNC method applied to data presented in Fig. 11.

TABLE II

Results of noise reduction by the LPNC method for the Chua circuit with a measurement noise ($N = 3000$).

$\mathcal{N}$	$\mathcal{G}$	percent of eliminated noise
24.9%	5.4	71%
28.3%	4.9	68%
46.1%	7.0	80%
73.7%	4.81	67%
90.6%	7.4	82%
96.5%	6.4	77%

All the above applications of the LPNC method consider the case of measurement noise that has been added to the signal in numerical or electronic experiments. However, our LPNC method can also be applied to dynamical noise *i.e.* to the noise which in experiments is included in the equations of motion [26]. In such a case one cannot compare the noisy data with the clean trajectory since the latter one does not exist anymore, and there are only ϵ-shadowed trajectories [8] that can be approximated by means of the LPNC method. Figure 13 shows a measured signal generated by a Chua circuit where a mixture of measurement noise and dynamical noise occurs. Figure 14 shows the result of noise reduction applied to such a signal.

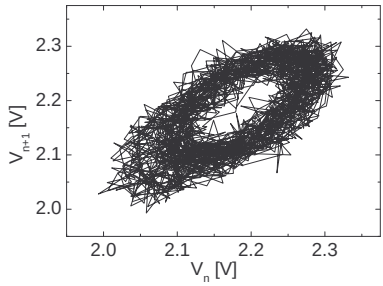

Fig. 13. Stroboscopic map received from the Chua circuit in the presence of a mixture of a measurement and dynamical noise $\mathcal{N} \approx 22\%$.

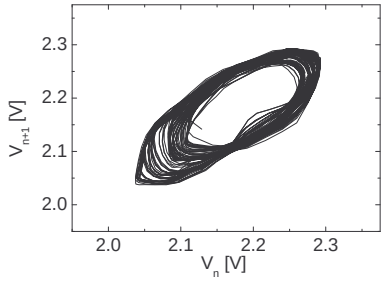

Fig. 14. Stroboscopic map received after noise reduction by the LPNC method applied to data presented at Fig. 13.

7. Noise level estimation by LPNC method

The LPNC method introduced in the previous section can be used to quantify the noise level of data. The noise level, *i.e.* the standard deviation in noisy time series, may be approximated as the Euclidian distance between the vectors $\{x_i\}$ and $\{\bar{x}_i\}$ representing the time series before and after noise reduction [16]

$$\tilde{\sigma}_{\text{noise}} \approx \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x}_i)^2}\,. \tag{28}$$

The main disadvantage of the LPNC method used for the noise level estimation is its small rate of convergence with respect to other known methods [4, 27–29] and the fact that the method can be used only for low-dimensional systems. On the other hand the LPNC method can be applied for estimation of any noise level including a large one. In Table III we have presented the estimated noise level $\tilde{\sigma}_{\text{noise}}$ for the Chua circuit.

TABLE III

Noise level estimated by the LPNC method for the Chua circuit with measurement noise ($N = 3000$).

$\mathcal{N}$	σ_{noise} [mV]	$\tilde{\sigma}_{\text{noise}}$ [mV]
0%	0	5.5
3.1%	30.4	28.9
6.2%	60.8	53.7
12.3%	121.7	110
24.9%	243.4	235
28.3%	304	305
46.1%	486	454
73.7%	973	938
90.6%	1520	1375
96.5%	2120	1844

8. Conclusions

In conclusion we have developed a method of noise reduction that makes use of nonlinear constraints which occur in a natural way due to the linearization of a deterministic system trajectory in the nearest neighborhood of every point in the phase space. This neighborhood has been determined by Delaunay triangulation. The method has been applied to data from the Henon map, Lorenz model and electronic Chua circuit contaminated by measurement (additive) noise. The efficiency of our method is comparable to that of standard LP methods but it is more robust to input parameter adjustment.

We gratefully acknowledge helpful discussion with Holger Kantz and Rainer Hegger. KU is thankful to Organizers of the Summer School German–Polish Dialogue 2002 in Darmstadt. He has been partially supported by the Polish State Committee for Scientific Rsearch (KBN) under grant 2 P03B 032 24 and JAH has been supported by the special program Dynamics of Complex Systems of Warsaw University of Technology.

Appendix A

Multidimensional version of LPNC method

In Sec. 3 the LPNC method has been presented for one-dimensional systems. Here we show the generalization of this approach for d-dimensional dynamics. For one-dimensional problems the Jacobi matrix of the system does not appear explicitly in our method. For higher dimensional models the

corresponding Jacobian $\boldsymbol{A}$ has to be calculated but we manage to minimalize errors occurring by its estimation. The linearized equation of motion for vectors from the nearest neighborhood $\tilde{\boldsymbol{X}}_n^{NN}$ of a vector $\tilde{\boldsymbol{x}}_n$ can be written in the form

$$\tilde{\boldsymbol{x}}_{n+1} = \boldsymbol{A} \cdot \tilde{\boldsymbol{x}}_n + \boldsymbol{b}. \tag{A.1}$$

In such a case one needs three vectors $\tilde{\boldsymbol{x}}_n, \tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{x}}_j \in \tilde{\boldsymbol{X}}_n^{NN}$ to write constraints corresponding to Eq. (12). In comparison with the one-dimensional case the number of near neighbors *i.e.* the number of points in the set $\tilde{\boldsymbol{X}}_n^{NN}$ must be larger to allow a unique estimation of the Jacobian $\boldsymbol{A}$. We assume that the Jacobi matrix can be approximately received by minimalization of the following cost functional

$$\begin{aligned} &\sum_s (a_{m1}x_s + a_{m2}x_{s-\tau} + \ldots + a_{md}x_{s-(d-1)\tau} - x_{s+1-m\tau})^2 = \min \\ &(\forall_{m=1,\ldots,d}) \quad \text{and} \quad \left\{ s : \boldsymbol{x}_s \in \boldsymbol{X}_n^{NN} \right\}, \end{aligned} \tag{A.2}$$

where $a_{ij} = [\boldsymbol{A}]_{ij}$. By analogy with Eq. (12) we introduce the following:

$$\begin{aligned} &G_m^d\left(\tilde{\boldsymbol{X}}_n^{NN}\right) \equiv a_{m1} G\left(\tilde{\boldsymbol{X}}_n^{NN}, \tilde{\boldsymbol{X}}_{n+1-m\tau}\right) \\ &+a_{m2} G\left(\tilde{\boldsymbol{X}}_{n-\tau}, \tilde{\boldsymbol{X}}_{n+1-m\tau}\right) + \ldots \\ &\ldots + a_{md} G\left(\tilde{\boldsymbol{X}}_{n-(d-1)\tau}, \tilde{\boldsymbol{X}}_{n+1-m\tau}\right) = 0 \\ &(\forall_{m=1,\ldots,d}), \end{aligned} \tag{A.3}$$

where we used the notation corresponding to equation (16) *i.e.*

$$\begin{aligned} G\left(\boldsymbol{X}_n^{NN}, \boldsymbol{X}_{n+l}\right) &= x_n\left(x_{k+l} - x_{j+l}\right) + x_k\left(x_{j+l} - x_{n+l}\right) \\ &\quad + x_j\left(x_{n+l} - x_{k+l}\right), \\ G\left(\boldsymbol{X}_{n-s}, \boldsymbol{X}_{n+l}\right) &= x_{n-s}\left(x_{k+l} - x_{j+l}\right) + x_{k-s}\left(x_{j+l} - x_{n+l}\right) \\ &\quad + x_{j-s}\left(x_{n+l} - x_{k+l}\right), \end{aligned} \tag{A.4}$$

$$\boldsymbol{X}_{n+s} = \{\boldsymbol{x}_{n+s}, \boldsymbol{x}_{k+s}, \boldsymbol{x}_{j+s}\} \quad (\forall_{s=0,\pm1,\pm2,\ldots}),$$

where $\left\{n, j, k : \boldsymbol{x}_n, \boldsymbol{x}_k, \boldsymbol{x}_j \in \boldsymbol{X}_n^{NN}\right\}$. Since the clean trajectory is not known thus in Eq. (A.3) the observed variables $\boldsymbol{X}_n^{NN}$, $\boldsymbol{X}_{n+1-m\tau}$ *etc.* are used.

In such a way equation (19) can be written in a more general way as

$$\begin{aligned} &\sum_{l=1}^{d} a_{ml} G\left(\delta \boldsymbol{X}_{n-(l-1)\tau}, \delta \boldsymbol{X}_{n+1-m\tau}\right) \cong 0 \\ &(\forall_{n=1,\ldots,N}, \forall_{m=1,\ldots,d}), \end{aligned} \tag{A.5}$$

where we use

$$G\left(\delta \boldsymbol{X}_{n-s}, \delta \boldsymbol{X}_{n+l}\right) = \delta x_{n-s}\left(\delta x_{k+l} - \delta x_{j+l}\right) + \delta x_{k-s}\left(\delta x_{j+l} - \delta x_{n+l}\right) + \delta x_{j-s}\left(\delta x_{n+l} - \delta x_{k+l}\right), \tag{A.6}$$

$$\delta \boldsymbol{X}_{n+s} = \left\{\delta \boldsymbol{x}_{n+s}, \delta \boldsymbol{x}_{k+s}, \delta \boldsymbol{x}_{j+s}\right\} \quad \left(\forall_{s=0,\pm 1,\pm 2,\ldots}\right),$$

where $\left\{n, j, k : \boldsymbol{x}_n, \boldsymbol{x}_k, \boldsymbol{x}_j \in \boldsymbol{X}_n^{NN}\right\}$, $\delta \boldsymbol{x}_n = \left\{\delta x_n, \delta x_{n-\tau}, \ldots, \delta x_{n-(d-1)\tau}\right\}$ and $\left\{n : \boldsymbol{x}_n \in \boldsymbol{X}_n^{NN}\right\}$.

Now the cost problem (21) can be transformed to the form

$$\tilde{S}_n^{NN} = \sum_s \left(\delta x_s\right)^2 + \lambda_n^m G_m^d\left(\boldsymbol{X}_n^{NN}\right) = \min$$
$$\left(\forall_{n=1,\ldots,N}, \forall_{m=1,\ldots,d}\right)$$

and

$$\left\{s : \boldsymbol{x}_s \in \boldsymbol{X}_n^{NN} \text{ or } \boldsymbol{x}_s \in \boldsymbol{X}_{n+1}\right\}. \tag{A.7}$$

Finding zeros of partial derivatives of the functional (A.7) one can linearize this problem and write it in the form similar to the Eq. (22)

$$\boldsymbol{M}_n \cdot \delta \boldsymbol{X}_n^{\lambda} = \boldsymbol{B}_n \quad \left(\forall_{n=1,\ldots,N}\right). \tag{A.8}$$

Vectors $\delta \boldsymbol{X}_n^{\lambda}$ and $\boldsymbol{B}_n$ occurring in Eq. (A.8) are equal to

$$\left(\delta \boldsymbol{X}_n^{\lambda}\right)^{\mathrm{T}} = \left\{\delta x_{n-(d-1)\tau}, \delta x_{n-(d-2)\tau}, \ldots \right.$$
$$\left. \ldots, \delta x_n, \delta x_{n+1}, \lambda_n^1, \lambda_n^2, \ldots, \lambda_n^d\right\}, \tag{A.9}$$

$$\boldsymbol{B}_n^{\mathrm{T}} = \left\{0, 0, \ldots, 0, -G_1^d\left(\boldsymbol{X}_n^{NN}\right), -G_2^d\left(\boldsymbol{X}_n^{NN}\right), \ldots, -G_d^d\left(\boldsymbol{X}_n^{NN}\right)\right\}, \tag{A.10}$$

where the number of zeros d_0 appearing in $\boldsymbol{B}_n^{\mathrm{T}}$ depends on the values of τ and d and for the case $\tau = 1$, $d_0 = d + 1$. Elements of the matrix $\boldsymbol{M}_n$ can be written as

$$\left[\boldsymbol{M}_n\right]_{mm} = 2 \quad \left(\forall_{m=1,\ldots,d_0}\right),$$
$$\left[\boldsymbol{M}_n\right]_{lm} = \left[\boldsymbol{M}_n\right]_{ml} + = \sum_{s=1}^{d} a_{ms}\left(x_{k+1} - x_{j+1}\right)$$
$$\left(\forall_{m=d_0+1,\ldots,d_0+d+1}\right) \text{ and } \left\{l : x_l \in \boldsymbol{x}_n\right\},$$

$$
\begin{aligned}
[\boldsymbol{M}_n]_{lm} &= [\boldsymbol{M}_n]_{ml} += \sum_{s=1}^{d} a_{ms}\,(x_{j+1} - x_{n+1}) \\
&\quad (\forall_{m=d_0+1,\ldots,d_0+d+1}) \text{ and } \{l : x_l \in \boldsymbol{x}_k\}\ , \\
[\boldsymbol{M}_n]_{lm} &= [\boldsymbol{M}_n]_{ml} += \sum_{s=1}^{d} a_{ms}\,(x_{n+1} - x_{k+1}) \\
&\quad (\forall_{m=d_0+1,\ldots,d_0+d+1}) \text{ and } \{l : x_l \in \boldsymbol{x}_j\}\ , \\
[\boldsymbol{M}_n]_{lm} &= [\boldsymbol{M}_n]_{ml} += \sum_{s=1}^{d} a_{ms}\,(x_{j-m\tau} - x_{k-m\tau}) \\
&\quad (\forall_{m=d_0+1,\ldots,d_0+d+1}) \text{ and } \{l : x_l \in \boldsymbol{x}_{n+1}\}\ , \\
[\boldsymbol{M}_n]_{lm} &= [\boldsymbol{M}_n]_{ml} += \sum_{s=1}^{d} a_{ms}\,(x_{n-m\tau} - x_{j-m\tau}) \\
&\quad (\forall_{m=d_0+1,\ldots,d_0+d+1}) \text{ and } \{l : x_l \in \boldsymbol{x}_{k+1}\}\ , \\
[\boldsymbol{M}_n]_{lm} &= [\boldsymbol{M}_n]_{ml} += \sum_{s=1}^{d} a_{ms}\,(x_{k-m\tau} - x_{n-m\tau}) \\
&\quad (\forall_{m=d_0+1,\ldots,d_0+d+1}) \text{ and } \{l : x_l \in \boldsymbol{x}_{j+1}\}\ , \qquad \text{(A.11)}
\end{aligned}
$$

where the remaining $\boldsymbol{M}_n$ elements vanish and $x_l \in \boldsymbol{x}_n$ means that the variable x_l is a component of the $\boldsymbol{x}_n$ vector.

The operator $+=$ in (A.11) has the same meaning as in the programming language C++, *i.e.* if elements of the matrix $[\boldsymbol{M}_n]_{ml}$ occur in a few places (*e.g.*: $x_n \in \boldsymbol{x}_n$ and $x_n \in \boldsymbol{x}_{n+1} \quad \forall_{d>1}, \vee_{\tau=1}$) then the elements at the rhs of such equations have to be summed up.

REFERENCES

[1] H. Kantz, T. Schreiber, *Nonlinear Time Series Analysis*, Cambridge University Press, Cambridge 1997.

[2] H.D.I. Abarbanel, *Analysis of Observed Chaotic Data*, Springer, New York 1996.

[3] T. Kapitaniak, *Chaos in Systems with Noise*, World Scientific, Singapore 1990.

[4] K. Urbanowicz, J.A. Hołyst, *Phys. Rev.* **E67**, 046218 (2003).

[5] E.J. Kostelich, T. Schreiber, *Phys. Rev.* **E48**, 1752 (1993).

[6] T. Schreiber, *Phys. Rev.* **E48**, 13(4) (1993).

[7] T. Schreiber, *Phys. Rev.* **E47**, 2401 (1992).

[8] J.D. Farmer, J.J. Sidorowich, *Physica D* **47**, 373 (1991).

[9] S.M. Hammel, *Phys. Lett.* **A148**, 421 (1990).

[10] M.E. Davies, *Physica D* **79**, 174 (1994).

[11] X.P. Zhang, *IEEE Trans. Neural Networ.* **12(3)**, 567 (2001).

[12] X.R. Chen, S. Tokinaga, *IEICE Trans. Fund. Electr.* **E85A**, 2107 (2002).

[13] P. Grassberger, I. Procaccia, *Phys. Rev. Lett.* **50**, 346 (1983).

[14] S.J. Julier, J.K. Uhlmann, H.F. Durrans-Whyte, *IEEE Trans. Autom. Contr.* **5**, 477 (2000).

[15] A. Effern, K. Lehnertz, T. Schreiber, T. Grunwald, P. David, C.E. Elger, *Physica D* **140**, 247 (2000).

[16] R. Cawley, G.H. Hsu, *Phys. Rev.* **A46**, 3057 (1992).

[17] T. Sauer, *Physica D* **58**, 193 (1994).

[18] P. Grassberger, R. Hegger, H. Kantz, C. Schaffrath, T. Schreiber, *Chaos* **3**, 127 (1993).

[19] D. Broomhead, D. Lowe, *Complex Syst.* **2**, 321 (1988).

[20] L. Matassini, H. Kantz, J. Holyst, R. Hegger, *Phys. Rev.* **E65**, 021102 (2002).

[21] S. Allie, A. Mees, K. Judd, D. Watson, *Phys. Rev.* **E55**, 87 (1997).

[22] E.N. Lorenz, *J. Atmos. Sci.* **20**, 130 (1963).

[23] S. Wu, *Proceedings of the IEEE*, vol. 75, No. 8 (1987).

[24] L.O Chua, G.-N. Lin, *IEEE Trans. Circuits Syst.* **37**, 885 (1990).

[25] V.S. Anishchenko, M.A. Safonova, L.O. Chua, *Journ. Circuits Syst. Comp.* **3**, 553 (1993).

[26] L. Jaeger, H. Kantz, *Physica D* **105**, 79 (1997).

[27] C. Diks, *Phys. Rev.* **E53**, 4263 (1996).

[28] D. Yu, M. Small, R.G. Harrison, C. Diks, *Phys. Rev.* **E61**, 3750 (2000).

[29] H. Oltmans, P.J.T. Verheijen, *Phys. Rev.* **E56**, 1160 (1997).