BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 60, No. 3, 2012
DOI: 10.2478/v10175-012-0077-7

PCA-based approximation of a class of distributed parameter systems: classical vs. neural network approach

K. BARTECKI*

Institute of Control and Computer Engineering, Opole University of Technology, 31 Sosnkowskiego St., 45-272 Opole, Poland

Abstract. In this article, an approximation of the spatiotemporal response of a distributed parameter system (DPS) with the use of the principal component analysis (PCA) is considered. Based on a data obtained by the numerical solution of a set of partial differential equations, a PCA-based approximation procedure is performed. It consists in the projection of the original data into the subspace spanned by the eigenvectors of the data covariance matrix, corresponding to its highest eigenvalues. The presented approach is carried out using both the classical PCA method as well as two different neural network structures: two-layer feed-forward network with supervised learning (FF-PCA) and single-layer network with unsupervised, generalized Hebbian learning rule (GHA-PCA). In each case considered, the effect of the approximation model structure represented by the number of eigenvectors (or, in the neural case, units in the network projection layer) on the mean square approximation error of the spatiotemporal response and on the data compression ratio is analysed. As shown in the paper, the best approximation quality is obtained for the classical PCA method as well as for the FF-PCA neural approach. On the other hand, an adaptive learning method for the GHA-PCA network allows to use it in e.g. an on-line identification scheme.

Key words: distributed parameter system, principal component analysis, artificial neural network, supervised learning, unsupervised learning.

1. Introduction

Many industrial processes are characterized by variables which can vary both temporally and spatially. Mathematical models of these processes are commonly known as spatiotemporal dynamical systems or distributed parameter systems (DPS). Phenomena taking place in the biotechnology, chemical engineering, semiconductor manufacturing as well as those associated with fluid flow, heat transfer or distillation processes are good examples here [1–10]. A mathematical description of this class of systems in the continuous case takes the form of partial differential equations (PDEs), which lead to the infinite-dimensional state space as well as to irrational transfer function representations. Therefore, due to the mathematical complexity, analysis of DPS is much more complex than in the case of lumped parameter systems (LPS), where spatial effects are averaged. Consequently, infinite-dimensional DPS models are often approximated by finite-dimensional ones. Among many approximation techniques, an important role is played by the so-called reduction methods, consisting in the replacement of the high-order model of DPS by a lower-order one, mapping the most relevant aspects of the dynamical properties of the system. The high-order model is usually derived by solving the PDE with the finite difference method or based on a measurement data set obtained from the identification experiment [6, 7, 11–13].

A significant role is played here by the principal component analysis (PCA) – a statistical method of factor analysis, depending on the application area also known as proper orthogonal decomposition, Hotelling transform or Karhunen-Loève transform [7, 14–19]. This technique is well-known from years and is used very successfully in many applications, e.g. in structural dynamics [20] and in model reduction of microelectromechanical systems [21]. The examples of the PCA-based approximations of DPS reported in the literature are related to different systems, mainly of parabolic type, e.g. [18,22–24]. In addition to the classical PCA method, based on the numerical eigenvalue/eigenvector decomposition of the data covariance matrix, one can find examples in the literature where the well-known properties of artificial neural networks are used for this purpose [13, 21, 25–30].

Unlike the previously mentioned, this paper focuses on the PCA-based approximation of the spatiotemporal responses of a certain class of DPS described by PDEs of hyperbolic type. The classical PCA method, involving numerical computation and analysis of the covariance matrix eigenvectors, as well as its neural counterparts are discussed and compared here. The software implementation of neural networks performing the task of the PCA approximation is realized based on an original C++ coded library, designed and implemented from scratch using object-oriented techniques. Thus, the remainder of the paper is organized as follows. Section 2 deals with the theoretical aspects of the PCA with a particular view to its possible use in the considered approximation task. This approach is implemented first using the classical PCA method and then employing two different neural structures: a two-layer feed-forward neural network with supervised learning, for which the abbreviation FF-PCA is introduced, as well as a single-layered structure with unsupervised, generalized Hebbian learning rule, later denoted as the GHA-PCA network. Section 3 introduces a widely encountered class of DPS, described by a system of two first-order hyperbolic PDEs. Exem-

*e-mail: k.bartecki@po.opole.pl

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

plary spatiotemporal responses obtained from the numerical solution of the PDEs by the use of the *method of lines* are also presented here. In Sec. 4 approximation results for both classical and neural PCA approaches are presented and compared. Finally, conclusions and suggestions for further research are given in Sec. 5.

2. PCA-based approximation of the DPS spatiotemporal response

In this section, theoretical aspects of the PCA method are presented, with a strict focus on the approximation of the DPS spatiotemporal response. The following two subsections discuss the use of the classical PCA method as well as two neural network-based PCA schemes.

2.1. Classical PCA method. Assume that as a result of the measurement or numerical simulation experiment, we have obtained a discrete set of values $y(l_m, t_n)$, representing the spatiotemporal distribution of a one-dimensional DPS process variable $y \in \mathbb{R}$, where $t_n = n \cdot \Delta t$ for $n = 1, 2, \ldots, N$ and $\Delta t = T/N$ is a discrete independent variable representing time, $l_m = m \cdot \Delta l$ for $m = 1, 2, \ldots, M$ and $\Delta l = L/M$ is a discrete independent variable representing spatial position. $T \in \mathbb{R}^+$ and $L \in \mathbb{R}^+$ denote temporal and spatial observation horizons of the process variable y, while $N \in \mathbb{N}$ and $M \in \mathbb{N}$ are number of observations and the number of spatial positions, respectively.

After initial processing, which involves subtracting from each sample $y(l_m, t_n)$ the time average for the m-th spatial position, equal to

$$\bar{y}(l_m) = \frac{1}{N} \sum_{n=1}^{N} y(l_m, t_n), \tag{1}$$

the DPS response will be represented by the matrix $Y \in \mathbb{R}^{M \times N}$ of the following form:

$$Y = \begin{bmatrix} y(l_1,t_1) - \bar{y}(l_1) & \cdots & y(l_1,t_N) - \bar{y}(l_1) \\ y(l_2,t_2) - \bar{y}(l_2) & \cdots & y(l_2,t_N) - \bar{y}(l_2) \\ \vdots & \ddots & \vdots \\ y(l_M,t_M) - \bar{y}(l_M) & \cdots & y(l_M,t_N) - \bar{y}(l_M) \end{bmatrix}. \tag{2}$$

The task consists in determining such an approximated matrix $\hat{Y} \in \mathbb{R}^{M \times N}$ for which the Frobenius norm of the approximation error matrix $E = Y - \hat{Y}$, given by

$$\begin{aligned} \|E\|_F &= \left\| Y - \hat{Y} \right\|_F = \sqrt{tr\left(\left(Y - \hat{Y} \right)^T \left(Y - \hat{Y} \right) \right)} \\ &= \sqrt{\sum_{m=1}^{M} \sum_{n=1}^{N} \left(y\left(l_m, t_n\right) - \hat{y}\left(l_m, t_n\right) \right)^2} \end{aligned} \tag{3}$$

reaches a given (possibly small) value, using the smallest possible data set.

The application of the PCA in the above task allows to determine the approximation matrix $\hat{Y}$ based on the following relationship [14]:

$$\hat{Y} = \Phi_K \Psi_K \tag{4}$$

where the reduced matrix $\Phi_K \in \mathbb{R}^{M \times K}$ consists of $K < M$ first columns of the orthogonal matrix $\Phi \in \mathbb{R}^{M \times M}$ including M column eigenvectors $\varphi_1, \varphi_2, \ldots, \varphi_M \in \mathbb{R}^M$ of the input data covariance matrix C, calculated as:

$$C = \frac{1}{N} Y Y^T. \tag{5}$$

The matrix Φ can thus be computed as the solution of the following eigenvalue problem:

$$C\Phi = \Phi\Lambda, \tag{6}$$

where $\Lambda \in \mathbb{R}^{M \times M}$ is the diagonal matrix of (real) eigenvalues of C, arranged in descending order:

$$\Lambda = diag\left(\lambda_1, \lambda_2, \ldots, \lambda_M\right), \quad \lambda_1 \geq \lambda_2 \geq \ldots \lambda_M. \tag{7}$$

The reduced orthogonal matrix $\Psi_K \in \mathbb{R}^{K \times N}$ appearing in Eq. (4) can be determined from the following relationship [14]:

$$\Psi_K = {\Phi_K}^T Y. \tag{8}$$

Hence, based on Eqs. (4) and (8) we obtain the relationship between the "original" and the approximated system response matrices:

$$\hat{Y} = \Phi_K {\Phi_K}^T Y. \tag{9}$$

As can be seen from the Eqs. (4)-(9), the PCA method consists in the projection of the original data set Y into the subspace spanned by the eigenvectors $\varphi_1, \varphi_2, \ldots, \varphi_K$ of the spatial covariance matrix C, corresponding to its K highest eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_K$. As mentioned earlier, for a given value of K, later referred to as the order of the approximation model, it can be proved (see e.g. [15]) that this projection is optimal in the sense of the minimum of the mean square error (MSE):

$$\text{MSE} = \frac{1}{M \cdot N} \|E\|_F^2. \tag{10}$$

The presented PCA approach can also be considered as a lossy compression problem, for which the data compression ratio C_K can be calculated as the ratio of the number of elements in the spatiotemporal response matrix Y to the total number of elements in Φ_K, Ψ_K and $\bar{y}$, required for the approximation of the response matrix:

$$C_K = \frac{M \cdot N}{M \cdot K + K \cdot N + M}. \tag{11}$$

A separate question concerns selecting the appropriate order for the approximation model. It can be based on the determination of such a value of K, for which the relative "energy" $E_{K\%}$ of the model, expressed as the ratio of the sum of the K largest eigenvalues of the covariance matrix C to the sum of all its eigenvalues:

$$E_{K\%} = \sum_{i=1}^{K} \lambda_i \Big/ \sum_{i=1}^{M} \lambda_i \cdot 100\%, \tag{12}$$

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

exceeds a given value, e.g. 99% [11]. Therefore, the initial data set can be reduced using only K most "energetic" eigenvectors of the covariance matrix. Selection of K can therefore be thought as a tradeoff between a suitably small value of the approximation error (10) and a sufficiently high value of the data compression ratio (11).

2.2. Neural PCA methods. The PCA-based approximation method described in the previous subsection, including numerical calculations of the covariance matrix and its eigenvectors, can achieve considerable computational complexity, particularly at the high dimensionality of the input data. In this case, it may be preferable to employ methods that do not require explicit determination of the covariance matrix. Such an approach can rely e.g. on the well-known approximation and adaptive properties of artificial neural networks. Their learning algorithms directly process the input vectors, which can be delivered either off- or on-line [28, 30–33]. Therefore, when the online scheme is taken into account, or when only a few principal components are required, the neural network-based PCA technique tends to be the best solution [25–27, 29].

In the subsequent paragraphs, two neural network structures are discussed, that can be used in the task of PCA-based approximation of the spatiotemporal response of a distributed parameter system.

Two-layer feed-forward network with supervised learning (FF-PCA). The first case considered concerns a feed-forward, two-layer linear neural network of the structure shown in Fig. 1. The number of network outputs (i.e. number of neurons in the second layer, hereinafter referred to as a *reconstruction layer*) is equal to the number of the network inputs and corresponds to the number of M spatial positions of the process variable y. Furthermore, the number of neurons in the first network layer, called the *projection layer*, representing the order K of the approximation model, is selected based on criteria similar to those presented in the case of the classical PCA method (see Subsec. 2.1). For the structure presented here, the acronym FF-PCA (*Feed-Forward Principal Component Analysis*) neural network will be used later in the article.

The role of the network input patterns will be taken over by the vectors representing distribution of the process variable y at the successive time instants t_n, i.e. subsequent columns of the matrix Y (2). In the considered case of auto-associative network learning, the output patterns are equal to the corresponding input ones, and the learning procedure consists in the iterative modifications of all weight coefficients in order to minimize the network error function of the following form [28, 30, 32, 33]:

$$E(w) = \frac{1}{M \cdot N} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(y(l_m, t_n) - \hat{y}(l_m, t_n) \right)^2, \quad (13)$$

being the direct equivalent of the approximation error function (10).

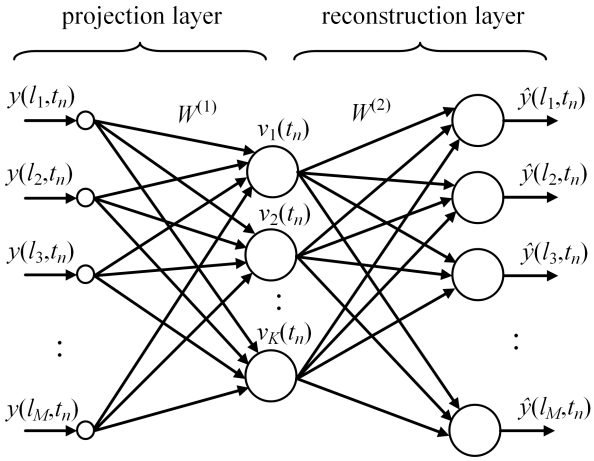

Fig. 1. Structure of the FF-PCA neural network

Denoting by $W^{(1)} \in \mathbb{R}^{K \times M}$ the weight coefficient matrix of the projection layer, by $W^{(2)} \in \mathbb{R}^{M \times K}$ the weight coefficient matrix of the reconstruction layer and by $V \in \mathbb{R}^{K \times N}$ the matrix of the projection layer responses to the input patterns Y, we obtain the following relationships describing the operation of the network of Fig. 1:

$$V = W^{(1)} Y, \quad (14)$$

and

$$\hat{Y} = W^{(2)} V = W^{(2)} W^{(1)} Y. \quad (15)$$

As can be easily seen, Eq. (15) is equivalent to the formula (9) for the classical PCA method, wherein the network weight matrix $W^{(1)}$ corresponds to the transposed matrix Φ_K and projection layer response matrix V corresponds to the matrix Ψ_K.

In order to determine the optimal (i.e. minimizing the error function given by Eq. (13)) values of the weight coefficients, an iteratively performed procedure, commonly known as the network learning process, has to be applied. However, the results obtained by the use of the learning procedure are ambiguous with respect to the solutions provided by the classical PCA method, since each matrix being a linear transformation of $W^{(1)}$ and $W^{(2)}$, respectively, will be considered to meet the presented auto-association task [28].

Among many different supervised learning algorithms for neural networks, two are mentioned below: gradient descent and Levenberg-Marquardt algorithms, the first with regard to its simplicity and the second considering its effectiveness. According to the first method, the network weights are updated in the i-th learning epoch in the direction of the negative gradient of the network error function (13). Taking additionally into account recent trends in modification of weights (i.e. including the so-called *momentum* term) and allowing the learning rate to change during the network training proces, the gradient descent learning algorithm can be described by the following updating formula [28, 30, 32, 33]:

$$\begin{aligned} w(i+1) = w(i) - \eta(i) \nabla E(w(i)) \\ + \alpha(i) (w(i) - w(i-1)), \end{aligned} \quad (16)$$

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

where $w(i)$ denotes the vector of network weights, $\nabla E(w(i))$ – gradient of the network error function, $\eta(i)$ – learning rate and $\alpha(i)$ – *momentum* rate.

However, significantly better results, both in terms of the error function value and the number of required learning epochs, can be achieved by the use of the quasi-Newton optimization methods (e.g. Levenberg-Marquardt algorithm). According to these methods, network weight coefficients are updated in the i-th learning epoch along with the following formula [28, 33]:

$$w\left(i+1\right) = w\left(i\right) - \eta\left(i\right)\tilde{H}^{-1}\left(w\left(i\right)\right)\nabla E\left(w\left(i\right)\right), \quad (17)$$

where $\tilde{H}^{-1}(w(i))$ is the inverse approximate Hessian matrix, i.e. square matrix of the second order partial derivatives of the network error function $E(w(i))$ with respect to all the weights. The Levenberg-Marquardt method, like other quasi-Newton algorithms, does not require calculation of all second derivatives, it rather updates Hessian matrix approximation at each iteration, computed as a function of the gradient.

The algorithms presented here can be implemented either in the batch mode, where the modification of the network weights is performed after the presentation of the entire training data set, or using the adaptive approach, in which the weights are modified after each presentation of a single learning pattern. The second approach allows the use of neural networks in the on-line approximation task, e.g. along with the incoming measurement data.

Single-layered network with unsupervised Hebbian learning (GHA-PCA). An alternative approach to extracting principal components from the data set can be based on a neural network with unsupervised learning, e.g. using the so-called Generalized Hebbian Algorithm (GHA). This method for a single neuron acting as a principal component analyzer was first proposed by Oja in [26]. Its extension to a network consisting of many neurons, known as the GHA or Sanger's rule, enabling the estimation of the subsequent principal components, was presented in the works of Oja and Sanger [27, 29].

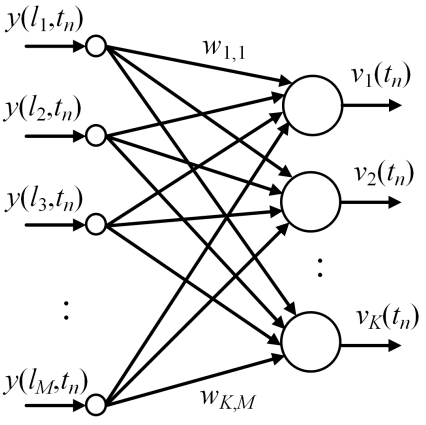

Fig. 2. Structure of the GHA-PCA neural network

In this case, the PCA task is performed using a single-layer neural network consisting of K linear neurons, corresponding to the subsequent principal components. The structure of the GHA-PCA network used in the approximation of the spatio-temporal DPS response is presented in Fig. 2 and corresponds to the projection layer of the feed-forward network of Fig. 1. However, unlike in the case of the FF-PCA network, the current learning procedure does not use the output learning patterns but only the input ones, as well as network output signals. Employing the notation of Fig. 2 and assuming the on-line network learning mode, where the modification of the weights takes place after each presentation of the input pattern corresponding to the time sample t_n, the k-th neuron generates an output signal according to the following relationship:

$$v_k(t_n) = \sum_{m=1}^{M} w_{k,m}\left(t_n\right) y\left(l_m, t_n\right), \quad (18)$$

where $w_{k,m}(t_n)$ denotes the value of the weight coefficient connecting the k-th neuron with the m-th network input, calculated for the time instant t_n.

The modification of the weight coefficients according to the GHA is performed based on the following expression [28, 29]:

$$\begin{aligned} w_{k,m}(t_{n+1}) &= w_{k,m}\left(t_n\right) + \\ &+ \eta v_k(t_n)\left[y(l_m,t_n) - \sum_{h=1}^{k} w_{h,m}(t_n)v_h(t_n)\right], \end{aligned} \quad (19)$$

for $m = 1, 2, \ldots, M$ and $k = 1, 2, \ldots, K$.

Denoting by $w_k(t_n)$ vector of the weight coefficients of the k-th neuron at the time instant t_n, i.e. vector of the following form:

$$w_k(t_n) = \left[\begin{array}{cccc} w_{k,1}(t_n) & w_{k,2}(t_n) & \ldots & w_{k,M}(t_n) \end{array}\right], \quad (20)$$

by $y(t_n)$ the input vector representing the distribution of the process variable for all M spatial positions at the time instant t_n:

$$y(t_n) = \left[\begin{array}{cccc} y(l_1,t_n) & y(l_2,t_n) & \ldots & y(l_M,t_n) \end{array}\right]^T, \quad (21)$$

and introducing the following notation:

$$y'(t_n) = y(t_n) - \sum_{h=1}^{k-1}\left(w_h(t_n)\right)^T v_h(t_n), \quad (22)$$

the relationship (19) can be written in the compact vector form:

$$\begin{aligned} w_k(t_{j+1}) &= w_k\left(t_j\right) + \\ &+ \eta v_k(t_j)\left[\left(y'(t_j)\right)^T - w_k(t_j)v_k(t_j)\right], \end{aligned} \quad (23)$$

analogous to the Oja algorithm for a single neuron, for which self-normalization of weight coefficients is carried out.

As mentioned in Subsec. 2.1, one of the main applications of PCA is lossy data compression. In the case under consideration, the compression task should be understood as follows: a large set of the input data, representing the distribution of the process variable y as the function of the temporal and spatial variables, is replaced by the reduced data set consisting of the weight coefficients and the network responses.

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

Approximation of the spatiotemporal response is possible due to the "decompression", carried out based on the matrices of the weight coefficients and network responses (see Eqs. (4) and (15)).

3. Case study – hyperbolic DPS

Among many different kinds of distributed parameter systems, an important class is constituted by the processes in which the phenomena of mass transport, as well as thermal and electrical energy transport take place. The following typical examples can be mentioned here [2–6, 10, 14, 34]:

- the voltage $U(l,t)$ and the current $I(l,t)$ in the electrical transmission line,
- the pressure $p(l,t)$ and the flow $q(l,t)$ of the medium transported through the pipeline,
- the temperatures $T_t(l,t)$ and $T_s(l,t)$ of the heating and the heated fluid in the case of a coaxial heat exchanger.

A mathematical description of the mentioned class of systems (after possible linearization at a fixed operating point) takes the general form of the following two coupled partial differential equations of hyperbolic type [14, 34, 35]:

$$\begin{aligned} \frac{\partial y_1(l,t)}{\partial t} + v_1 \frac{\partial y_1(l,t)}{\partial l} &= g_{11} y_1(l,t) + g_{12} y_2(l,t), \\ \frac{\partial y_2(l,t)}{\partial t} + v_2 \frac{\partial y_2(l,t)}{\partial l} &= g_{21} y_1(l,t) + g_{22} y_2(l,t), \end{aligned} \tag{24}$$

where $y_1(l,t) \in \mathbb{R}$ and $y_2(l,t) \in \mathbb{R}$ are functions representing spatiotemporal distribution of the process variables, square-integrable on the set $\Omega \times \Theta$, where $\Omega = [0, L]$ is the domain of the independent spatial variable l and $\Theta = [0, T]$ is the domain of the independent variable t representing time. The constant coefficients $v_1, v_2 \in \mathbb{R}$ usually represent the transport or wave propagation velocities, whereas the constants $g_{11}, g_{12}, g_{21}, g_{22} \in \mathbb{R}$ depend on the geometrical and physical parameters of the plant.

In order to obtain a unique solution to the PDE system (24), one must specify the initial and boundary conditions [35]. In the above case, the first represent the initial (i.e. corresponding to t=0) distribution of the process variables y_1 and y_2 over the spatial domain:

$$[y_1(l,0) = y_{01}(l)] \wedge [y_2(l,0) = y_{02}(l)], \tag{25}$$

where $y_{01}(l) \in \mathbb{R}$ and $y_{02}(l) \in \mathbb{R}$ are given functions, square-integrable on the set Ω.

On the other hand, the boundary conditions represent the requirements to be met by the solution to the Eqs. (24) at the boundary points of Ω. The general form of these conditions may include both the boundary reflection and feedback as well as the boundary control [34]. In the article, the Dirichlet boundary control is considered, specifying the values the solution needs to take on the boundary of the domain, e.g. for $l = 0$:

$$[y_1(0,t) = y_{b1}(t)] \wedge [y_2(0,t) = y_{b2}(t)], \tag{26}$$

where $y_{b1}(t) \in \mathbb{R}$ and $y_{b2}(t) \in \mathbb{R}$ are given functions, square-integrable on the set Θ.

Due to the complex form of the most analytical solutions to the PDEs, if available, often involving expressions containing integrals and infinite series, an important role is played by the numerical methods for solving them [6, 35]. Among many existing numerical approaches, one of the most powerful and generally applicable is the *finite difference method* (FDM), based on the approximation of the solutions using finite difference equations to approximate derivatives. Another frequently used numerical technique is the so-called *method of lines* (MOL), consisting in the discretization of the spatial derivatives only and leaving the time variable continuous. This approach leads to the system of ordinary differential equations to which any existing numerical method for the initial value problem can be applied.

In the case under consideration, the method of lines is used to determine the numerical solutions of Eqs. (24), representing the spatiotemporal responses of the hyperbolic DPS, for the initial and boundary conditions given by Eqs. (25) and (26), respectively. The simulation tests are performed for the following values of equation parameters, domain of the solution and the discretization step: $v_1 = 1$, $v_2 = 0.5$, $g_{11} = -0.0638$, $g_{12} = 0.0638$, $g_{21} = -0.0359$, $g_{22} = 0.0359$, $L = 5$, $T = 50$, $\Delta l = 0.1$. The simulations were carried out assuming zero initial conditions (25):

$$\forall l \in]0, L]: \quad [y_{01}(l) = 0] \wedge [y_{02}(l) = 0] \tag{27}$$

as well as for two different forms of boundary conditions (26):

$$\forall t \in [0, T]: \quad [y_{b1}(t) = \delta(t)] \wedge [y_{b2}(t) = 0] \tag{28}$$

and

$$\forall t \in [0, T]: \quad [y_{b1}(t) = H(t)] \wedge [y_{b2}(t) = 0] \tag{29}$$

where $\delta(t)$ denotes the delta Dirac function and $H(t)$ – the Heaviside step function, representing the impulse and the step input forcing, respectively.

After an additional discretization of the numerical solution in the time domain using the time step $\Delta t = 0.5$ and subtracting from each sample the time average (1) for the given spatial position l_m, two matrices of the form (2) are obtained, each consisting of M=51 rows, representing the discrete spatial positions, and N=101 columns, corresponding to the discrete time samples. The solutions $y_2(l,t)$ for both types of boundary conditions (28) and (29) are shown in Figs. 3 and 4, respectively. In the next section, the approximation results of these responses using the PCA method are presented, both for the classical and neural network-based approaches.

The approximation tests were conducted for different values of K, representing the number of eigenvectors of the spatial covariance matrix C included in the approximation model, or, in the case of the neural-based PCA, denoting the number of neurons in the network projection layer. Selected results are presented both as graphs and in the tabular form.

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

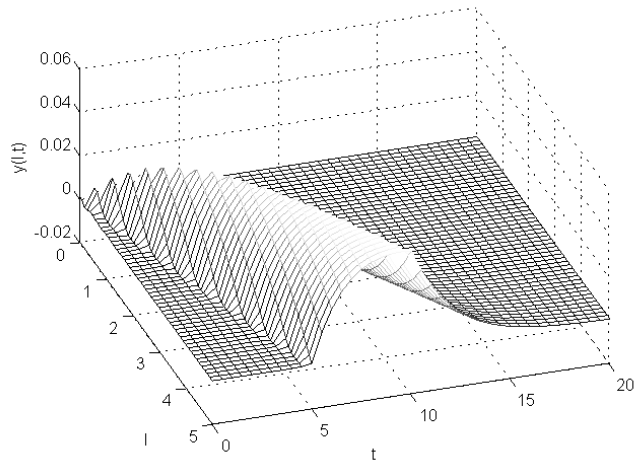

Fig. 3. Response $y_2(l,t)$ of the system (24) for initial conditions (27) and boundary conditions (28)

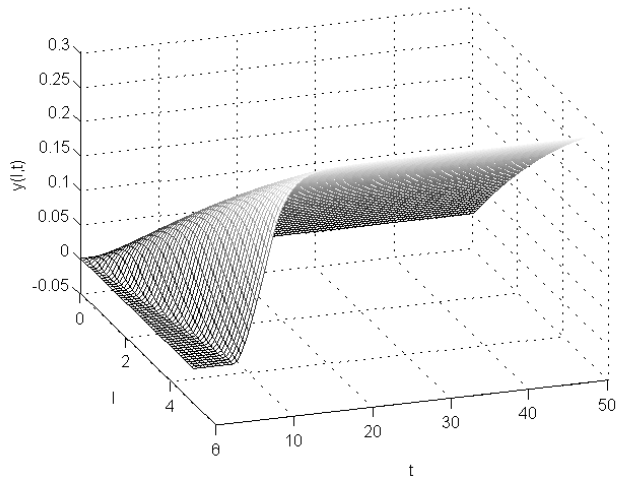

Fig. 4. Response $y_2(l,t)$ of the system (24) for initial conditions (27) and boundary conditions (29)

4. PCA-based approximation results

4.1. Classical PCA method. The results of the application of the classical PCA method in the task of approximation of the impulse response of Fig. 3 for the model order $K = 3$ are presented in Figs. 5 and 6. Figure 5 shows the graphs illustrating the eigenvectors φ_1, φ_2 and φ_3 of the spatial covariance matrix C, comprising the matrix Φ_K (see Eq. (4)). The original impulse responses (solid line), obtained by the numerical solution of the equation set (24), and the approximation model responses (dashed line), calculated based on Eq. (4), are compared for five different spatial positions in the Fig. 6.

The results for the classical PCA approximation for both step and impulse responses obtained for the model orders $K = 1$, $K = 3$ and $K = 5$ are summarized in Table 1 at the end of this section. For each of these cases, it contains the following results: K largest eigenvalues of the covariance matrix C of Eq. (5), the "energy" coefficient $E_{K\%}$ defined by Eq. (12), the Frobenius norm of the error matrix $\|E\|_F$ given by Eq. (3), the mean square approximation error (10) and the data compression coefficient C_K defined by Eq. (11).

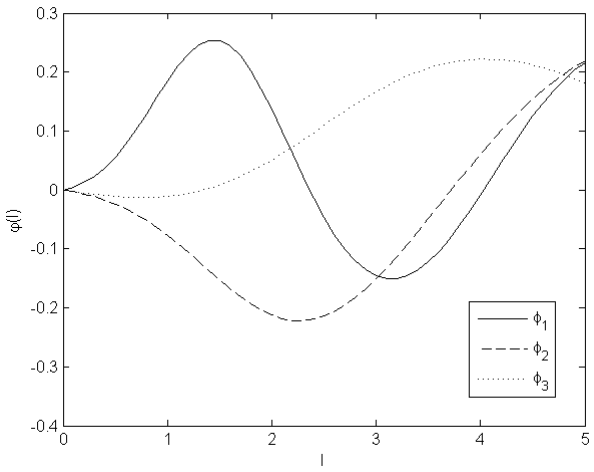

Fig. 5. Classical PCA approximation results for the impulse response and $K = 3$: eigenvectors φ_1, φ_2 and φ_3

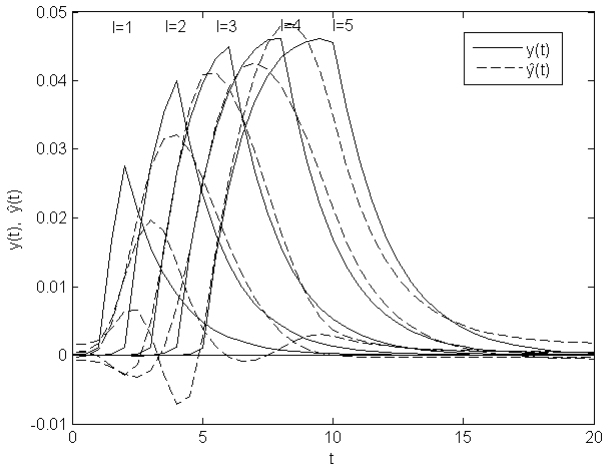

Fig. 6. Classical PCA approximation results for the impulse response and $K = 3$: $y(l,t)$ and $\hat{y}(l,t)$

As can be seen from the presented results, the increase in the approximation order K reduces the value of the approximation error, representing the sum-squared differences between the original and the approximated spatio-temporal responses of the system for all M spatial positions and N time samples. However, it also decreases the value of the data compression coefficient C_K, representing the ratio of the number of elements in the spatiotemporal response matrix Y to the total number of elements in matrices Φ_K and Ψ_K, required for the approximation. Therefore, selection of the appropriate value for K should take into account the tradeoff between an assumed (reasonably low) value for the approximation error, and a sufficiently high value for the compression ratio. As mentioned before, the number of the most "energet-

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

ic" eigenvectors of the spatio-temporal response data set can be selected based e.g. on Eq. (12).

4.2. FF-PCA neural network. Figures 7 and 8 show approximation results analogous to those presented in Figs. 5 and 6, obtained as a result of the use of FF-PCA method consisting in principal component extraction with the use of the two-layer neural network with supervised learning. The results presented here concern the approximation of the impulse response of Fig. 3 using a network with three neurons in the projection layer ($K = 3$). In Fig. 7 vectors of the weight coefficients w_1, w_2 and w_3 of this layer are presented, comprising the matrix $W^{(1)}$ (see Fig. 1). In contrast to the base vectors φ_1, φ_2 and φ_3 of Fig. 5, the weight vectors w_1, w_2 and w_3 obtained as a result of the network learning procedure are not orthogonal – their values have somewhat "chaotic" distribution. This is mainly due to the fact that the network learning algorithm generates random initial values of the weight coefficients, and, moreover, it does not impose the orthogonality condition on the weight vectors as opposed to the classical PCA method.

Figure 8 compares, similarly as Fig. 6, the numerically calculated impulse responses (solid line) and its FF-PCA approximations (dashed line) for five different spatial positions. Table 1 contains the results of the FF-PCA approximation obtained for the number of neurons in the projection layer equal to: $K = 1$, $K = 3$ and $K = 5$. From the presented results one can conclude that despite the different values of the model parameters, represented by the matrices Φ_K and Ψ_K for the classical PCA method and $W^{(1)}$ together with V for the neural FF-PCA model, identical values of the approximation error are obtained for each response type and for each value of K.

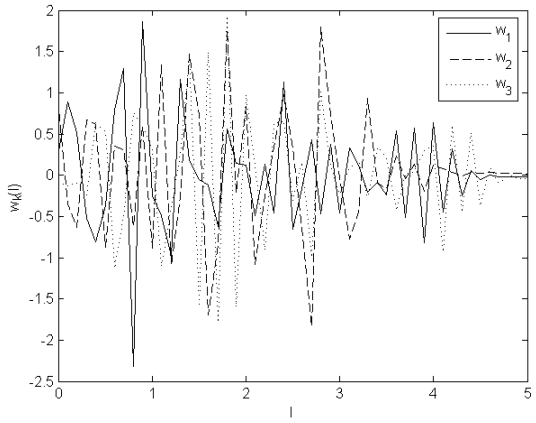

Fig. 7. FF-PCA approximation results for the impulse response and $K = 3$: weight vectors w_1, w_2 and w_3

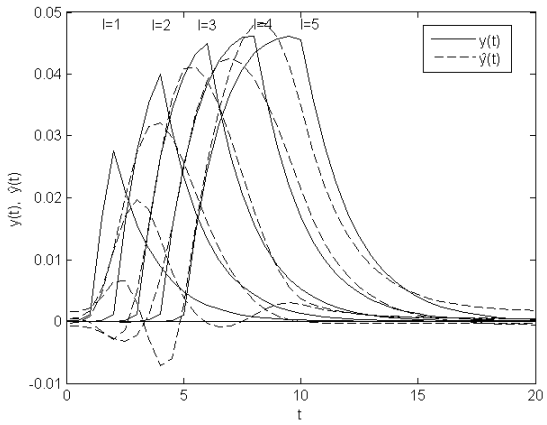

Fig. 8. FF-PCA approximation results for the impulse response and $K = 3$: $y(l,t)$ and $\hat{y}(l,t)$

Table 1
PCA-based approximation results

		impulse response			step response		
		number of eigenvectors (neurons)			number of eigenvectors (neurons)		
		$K=1$	$K=3$	$K=5$	$K=1$	$K=3$	$K=5$
classical PCA	$\lambda_1..\lambda_K$	$\lambda_1 = 4.71{\cdot}10^{-3}$	$\lambda_1 = 4.71{\cdot}10^{-3}$ $\lambda_2 = 1.95{\cdot}10^{-3}$ $\lambda_3 = 4.39{\cdot}10^{-4}$	$\lambda_1 = 4.71{\cdot}10^{-3}$ $\lambda_2 = 1.95{\cdot}10^{-3}$ $\lambda_3 = 4.39{\cdot}10^{-4}$ $\lambda_4 = 1.76{\cdot}10^{-4}$ $\lambda_5 = 5.23{\cdot}10^{-5}$	$\lambda_1 = 1.42{\cdot}10^{-1}$	$\lambda_1 = 1.42{\cdot}10^{-1}$ $\lambda_2 = 4.61{\cdot}10^{-3}$ $\lambda_3 = 5.13{\cdot}10^{-4}$	$\lambda_1 = 1.42{\cdot}10^{-1}$ $\lambda_2 = 4.61{\cdot}10^{-3}$ $\lambda_3 = 5.13{\cdot}10^{-4}$ $\lambda_4 = 8.41{\cdot}10^{-5}$ $\lambda_5 = 1.77{\cdot}10^{-5}$
	$E_{K\%}$	63.64%	96.00%	99.09%	96.44%	99.93%	99.99%
	$\|E\|_F$	0.328	0.109	0.052	0.724	0.105	0.029
	MSE	$5.14 \cdot 10^{-5}$	$5.67 \cdot 10^{-6}$	$1.29 \cdot 10^{-6}$	$1.02 \cdot 10^{-4}$	$2.16 \cdot 10^{-6}$	$1.59 \cdot 10^{-7}$
	C_K	14.62	6.39	4.09	25.37	10.16	6.35
FF-PCA	$\|E\|_F$	0.328	0.109	0.052	0.724	0.105	0.029
	MSE	$5.14 \cdot 10^{-5}$	$5.67 \cdot 10^{-6}$	$1.29 \cdot 10^{-6}$	$1.02 \cdot 10^{-4}$	$2.16 \cdot 10^{-6}$	$1.59 \cdot 10^{-7}$
	C_K	14.62	6.39	4.09	25.37	10.16	6.35
GHA-PCA	$\|E\|_F$	0.328	0.109	0.055	0.828	0.690	0.682
	MSE	$5.15 \cdot 10^{-5}$	$5.68 \cdot 10^{-6}$	$1.42 \cdot 10^{-6}$	$1.33 \cdot 10^{-4}$	$9.23 \cdot 10^{-5}$	$9.03 \cdot 10^{-5}$
	C_K	14.62	6.39	4.09	25.37	10.16	6.35

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

4.3. GHA-PCA neural network. The corresponding results for the GHA-PCA network, obtained for the impulse response and $K = 3$ neurons are presented in Figs. 9 and 10, whereas the results for both impulse and step responses for three different values of K are summarized, as in the previous cases, in Table 1. The initial visual assessment of graphs illustrating the parameters of the approximation models shows certain similarity of the results for the GHA-PCA (Fig. 9) and the classical PCA approaches (Fig. 5).

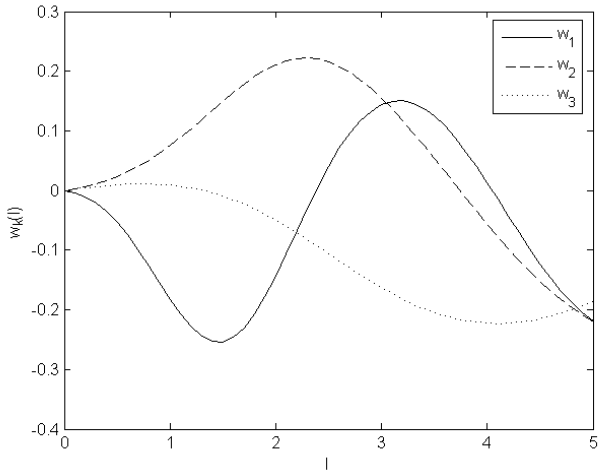

Fig. 9. GHA-PCA approximation results for the impulse response and $K = 3$: weight vectors w_1, w_2 and w_3

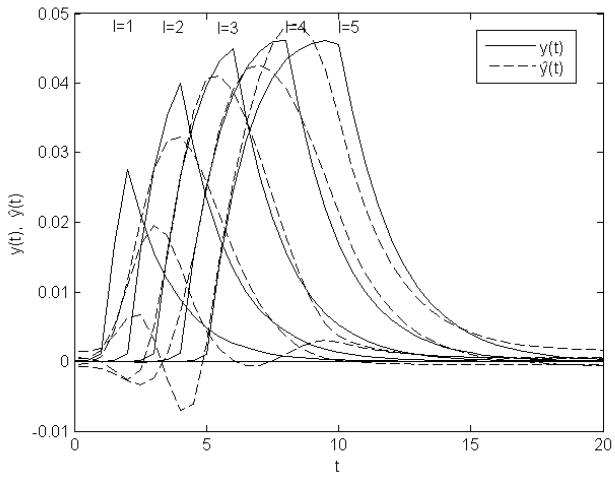

Fig. 10. GHA-PCA approximation results for the impulse response and $K = 3$: $y(l, t)$ and $\hat{y}(l, t)$

As a result of the adaptive network learning procedure, orthogonal normalized vectors of the weight coefficients are obtained, corresponding to the eigenvectors of the covariance matrix of the spatiotemporal response. However, after inspecting the results presented in Table 1, it can be concluded that only in the case of the impulse response the approximation error values are similar. Using as a reference the classical PCA algorithm, it can be noticed that with increasing K, better approximation quality in terms of the MSE is obtained for the FF-PCA network.

In the case of the approximation of the step response the differences are even greater: while in the case of FF-PCA approximation error reaches a value of $1 \cdot 10^{-7}$, the error for the GHA-PCA network is fixed at the level of $9 \cdot 10^{-5}$ – further increasing the number of learning iterations does not improve the approximation quality.

5. Conclusions

In this paper, approximation of the spatiotemporal response of a distributed parameter system with the use of the principal component analysis, also known as the proper orthogonal decomposition, Hotelling transform or Karhunen-Loève transform, has been discussed. This method has been applied to the data obtained from the numerical solution of the system of two partial differential equations of hyperbolic type, describing the phenomena often encountered in the industrial practice.

The approximation approach has been carried out adopting two different approaches: first, using the classical PCA method, based on the analysis of the eigenvectors of the response covariance matrix, and second, employing two different neural network structures. In the later case, two-layer feedforward neural network with supervised learning (FF-PCA) as well as single-layered neural network with unsupervised, generalized Hebbian learning (GHA-PCA) have been used.

The main idea of applying the PCA method in the considered approximation task can be summarized as follows: based on a data set obtained by the numerical solution of a given PDE, the projection of this data set into the subspace spanned by the eigenvectors of the spatial covariance matrix, corresponding to its highest eigenvalues is performed. This PCA-based approach can also be considered as a lossy compression problem, for which the data compression ratio is equal to the ratio of the number of elements in the spatiotemporal response matrix to the total number of elements in matrices required for its approximation. Therefore, selection of the model order can be regarded as a tradeoff between suitably low value of the approximation error and a sufficiently high value of the data compression ratio.

In the article, influence of the approximation order, representing the number of eigenvectors on the mean square response approximation error and on the data compression ratio, has been analyzed. When using a neural network, this parameter corresponds directly to the number of units in the network projection layer. It has also been demonstrated that the model order, necessary to achieve a good approximation quality, depends on the form of the boundary constraint, acting as an input forcing. Therefore, an important issue in the context of the use of this method for the identification purpose is choosing a suitable input signal, providing good excitation to the spatio-temporal dynamics of the system.

The values of the approximation error obtained for the FF-PCA method were identical as in the case of the classical PCA approach. The only drawback in the application of

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

neural techniques was a slightly longer time required to determine the appropriate values of model parameters (i.e. network weight coefficients), resulting from the iterative nature of the training procedure. On the other hand, results obtained for the PCA neural network with the generalized Hebbian learning rule were worse than those obtained for the two methods mentioned previously, particularly in the case of the approximation of the step response.

A positive aspect of using neural networks as a tool for extracting principal components from a data set is that they do not require calculating its correlation matrix explicitly. For this reason, they can be used e.g. in the on-line data acquisition scheme, when calculation of the data correlation matrix in the explicit form is impossible. The neural-based approach presented in the paper may act as a good starting point for further research concerning, for example, approximation of nonlinear distributed parameter systems using nonlinear PCA method, based on the function approximation properties of neural networks with nonlinear units.

Acknowledgements. The author gratefully acknowledges helpful comments from anonymous Reviewers.

REFERENCES

[1] K. Bartecki, "Optimization of the night setback process for a heated building using neural algorithms", *Proc. 5th IEEE Int. Conf. on Neural Networks and Soft Computing* 1, 675–680 (2000).

[2] K. Bartecki, "Comparison of frequency responses of parallel- and counter-flow type of heat exchanger", *Proc. 13th IEEE IFAC Int. Conf. on Methods and Models in Automation and Robotics* 1, CD-ROM (2007).

[3] K. Bartecki, "Frequency- and time-domain analysis of a simple pipeline system", *Proc. 14th IEEE IFAC Int. Conf. on Methods and Models in Automation and Robotics* 1, CD-ROM (2009).

[4] K. Bartecki and R. Rojek, "Performance optimization of heat exchanging centre using neural algorithms", *Proc. 4th IEEE Int. Conf. on Neural Networks and Their Applications* 1, 643–648 (1999).

[5] K. Bartecki and R. Rojek, "Instantaneous linearization of neural network model in adaptive control of heat exchange process", *Proc. 11th IEEE Int. Conf. on Methods and Models in Automation and Robotics* 1, 967–972 (2005).

[6] J.C. Friedly, *Dynamic Behaviour of Processes*, Prentice Hall, New York, 1972.

[7] H.-X. Li and C. Qi, "Modeling of distributed parameter systems for applications – a synthesized review from time–space separation", *J. Process Control* 20, 891–901 (2010).

[8] B. Cichy, P. Augusta, E. Rogers, K. Gałkowski, and Z. Hurák, "Robust control of distributed parameter mechanical systems using a multidimensional systems approach", *Bull. Pol. Ac.: Tech.* 58 (1), 67–75 (2010).

[9] K. Murawski and D. Lee, "Numerical methods of solving equations of hydrodynamics from perspectives of the code flash", *Bull. Pol. Ac.: Tech.* 59 (1), 81–91 (2011).

[10] R. Rojek, *Modeling of Flow Networks*, Opole University of Technology Press, Opole, 2002.

[11] L.G. Bleris and M.V. Kothare, "Low-order empirical modeling of distributed parameter systems using temporal and spatial eigenfunctions", *Computers and Chemical Engineering* 29 (4), 817–827 (2005).

[12] K.A. Hoo and D. Zheng, "Low-order control-relevant models for a class of distributed parameters system", *Chemical Engineering Science* 56 (23), 6683–6710 (2001).

[13] Qi Ch. and H.-X. Li, "A time-space separation-based Hammerstein modeling approach for nonlinear distributed parameter processes", *Computers and Chemical Engineering* 33 (7), 1247–1260 (2009).

[14] K. Bartecki, "Approximation of a class of distributed parameter systems using proper orthogonal decomposition", *Proc. 16th IEEE Int. Conf. on Methods and Models in Automation and Robotics* 1, CD-ROM (2011).

[15] G. Berkooz, P. Holmes, and J.L. Lumley, "The proper orthogonal decomposition in the analysis of turbulent flows", *Annual Review of Fluid Mechanics* 25, 539–575 (1993).

[16] A. Chatterjee. "An introduction to the proper orthogonal decomposition", *Current Science* 78 (7), 808–817 (2000).

[17] Y.C. Liang, H.P. Lee, S.P. Lim, W.Z. Lin, K.H. Lee, and C.G. Wu, "Proper ortogonal decomposition and its applications – part I: Theory", *J. Sound and Vibration* 252 (3), 527–544 (2002).

[18] H.M. Park and D.H. Cho, "The use of the Karhunen-Loève decomposition for the modeling of distributed parameter systems", *Chemical Engineering Science* 51 (1), 81–98 (1996).

[19] C. Wolter, M.A. Trindade, and R. Sampaio, "Obtaining mode shapes through the Karhunen-Loève expansion for distributed-parameter linear systems", *Shock and Vibration* 9 (4), 177–192 (2002).

[20] G. Kerschen, F. Poncelet, and J.-C. Golinval, "Physical interpretation of independent component analysis in structural dynamics", *Mechanical Systems and Signal Processing* 21 (4), 1561–1575 (2007).

[21] Y.C. Liang, H.P. Lee, S.P. Lim, W.Z. Lin, K.H. Lee, and H. Sun, "Proper orthogonal decomposition and its applications – part II: Model reduction for MEMS dynamical analysis", *J. Sound and Vibration* 256 (3), 515–532 (2002).

[22] D. Zheng and K. A. Hoo, "Low-order model identifcation for implementable control solutions of distributed parameter systems", *Computers and Chemical Engineering* 26, 1049–1076 (2002).

[23] M. Li and P.D. Christofides, "Optimal control of diffusion-convection-reaction processes using reduced-order models", *Computers and Chemical Engineering* 32, 2123–2135 (2007).

[24] E. Aggelogiannaki and H. Sarimveis, "Nonlinear model predictive control for distributed parameter systems using data driven artificial neural network models", *Computers & Chemical Engineering* 32 (6), 1225–1237 (2008).

[25] K.I. Diamantaras and S.Y. Kung, *Principal Component Neural Networks – Theory and Applications*, John Wiley, New York, 1996.

[26] E. Oja, "A simplified neuron model as a principal component analyzer", *J. Mathematical Biology* 15 (3), 267–273 (1982).

[27] E. Oja, "Neural networks, principal components, and subspaces", *Int. J. Neural Systems* 1 (1), 61–68 (1989).

[28] S. Osowski, *Neural Networks for Information Processing*, Warsaw University of Technology Press, Warsaw, 2000.

[29] T.D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network", *Neural Networks* 2 (6), 459–473 (1989).

[30] J. Żurada, *Introduction to artificial neural systems*, West Publishing Company, St. Paul, 1992.

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM

[31] K. Bartecki, "On some peculiarities of neural network approximation applied to the inverse kinematics problem", *Proc. Conf. on Control and Fault-Tolerant Systems* 1, 317–322 (2010).
[32] J. Hertz, A. Krogh, and R. Palmer, *Introduction to the Theory of Neural Computation*, Addison Wesley, Redwood City, 1991.
[33] R. Rojas, *Neural Networks: a Systematic Introduction*, Springer, Berlin, 1996.
[34] M. Ziółko, *Modeling of Wave Phenomena*, AGH University of Science and Technology Press, Kraków, 2000.
[35] R.M.M. Mattheij, S.W. Rienstra, and J.H.M. ten Thije Boonkkamp, *Partial Differential Equations: Modeling, Analysis, Computation*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2005.

Unauthenticated | 89.67.242.59
Download Date | 5/19/13 8:37 PM