Neural Network-Based PCA: An Application to Approximation of a Distributed Parameter System

Krzysztof Bartecki

Opole University of Technology, Institute of Control and Computer Engineering, ul. Sosnkowskiego 31, 45-272 Opole, Poland
k.bartecki@po.opole.pl

Abstract. In this article, an approximation of the spatiotemporal response of a distributed parameter system (DPS) with the use of the neural network-based principal component analysis (PCA) is considered. The presented approach is carried out using two different neural structures: single-layer network with unsupervised, generalized Hebbian learning (GHA-PCA) and two-layer feedforward network with supervised learning (FF-PCA). In each case considered, the effect of the number of units in the network projection layer on the mean square approximation error (MSAE) and on the data compression ratio is analysed.

Keywords: principal component analysis, neural networks, distributed parameter systems.

1 Introduction

Principal Component Analysis (PCA) is one of the main approaches to reduce the dimensionality of data, losing the least amount of information. It can be applied in many fields such as pattern recognition, computer vision, signal processing, data compression, etc. The advantages of PCA result from its optimality properties in maximization of variance as well as in minimization of mean square error [5,7]. However, numerical calculations of the data covariance matrix and its eigenvectors being the main feature of the PCA 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 properties of artificial neural networks. Their learning algorithms directly process the input vectors, which can be delivered either off- or on-line [2,10,15,17,19]. 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 good solution [8,13,14,18].

In this paper neural networks are proposed to be used as a tool for the PCA-based approximation of spatiotemporal responses of a distributed parameter system (DPS). A mathematical description of this class of systems takes most often the form of partial differential equations (PDEs), which lead to the infinite-dimensional state space and irrational transfer function representations. Therefore, due to the mathematical complexity, these 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. A significant role is played here by the reduction methods based on the PCA approach [6,11,12,16]. This paper proposes PCA-based DPS approximation to be carried out using two different neural network structures: single-layer network with unsupervised, generalized Hebbian learning (GHA-PCA) and two-layer feedforward network with supervised autoassociative learning (FF-PCA).

2 Neural Network-Based PCA Schemes

In this section, the abovementioned neural network-based PCA techniques are introduced, with particular emphasis on their use in the approximation of the DPS spatiotemporal response.

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 $t_n = n \cdot \mathit{\Delta} t$ for $n = 1, 2, \ldots, N$ and $\mathit{\Delta} t = T/N$ is a discrete independent variable representing time, $l_m = m \cdot \mathit{\Delta} l$ for $m = 1, 2, \ldots, M$ and $\mathit{\Delta} l = L/M$ is a discrete independent variable representing spatial position. $T \in \mathbb{R}^+$ and $L \in \mathbb{R}^+$ denote temporal and spatial observation horizons, while $N \in \mathbb{N}$ and $M \in \mathbb{N}$ are number of observations and number of spatial positions, respectively. After initial processing, involving subtracting from each sample $y(l_m, t_n)$ the time average for the m-th spatial position, given by

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

the DPS response will be represented by the matrix $Y = [y(l_m, t_n) - \bar{y}(l_m)] \in \mathbb{R}^{M \times N}$.

PCA can be seen as optimal factorization of Y into two matrices:

$$\hat{Y} = \mathit{\Phi}_K \mathit{\Psi}_K \tag{2}$$

where $\hat{Y} \in \mathbb{R}^{M \times N}$ denotes the approximated matrix Y, $\mathit{\Phi}_K \in \mathbb{R}^{M \times K}$ is matrix consisting of $K < M$ orthogonal column eigenvectors $\varphi_1, \varphi_2, \ldots, \varphi_K \in \mathbb{R}^M$ of the response covariance matrix C calculated as:

$$C = \frac{1}{N} Y Y^T, \tag{3}$$

corresponding to its K largest eigenvalues, $\lambda_1, \lambda_2, \ldots, \lambda_K$. The matrix $\mathit{\Psi}_K \in \mathbb{R}^{K \times N}$ in (2) can be determined from the following relationship [3]:

$$\mathit{\Psi}_K = {\mathit{\Phi}_K}^T Y. \tag{4}$$

Optimality condition for the factorization (1) means that the Frobenius norm $\|E\|_{\mathrm{F}}$ of the approximation error matrix $E = Y - \hat{Y}$ must be minimized for the given value of the model order K.

In the following subsections two neural network-based PCA techniques are discussed: a single-layered neural network with unsupervised Hebbian learning and a feedforward neural network with supervised autoassociative learning.

2.1 Single-Layered Network with Supervised Training (GHA-PCA)

A single neuron acting as a principal component analyzer was first proposed by Oja in [13]. Its extension to a network consisting of many neurons, known as Generalized Hebbian Algorithm (GHA) or Sanger's rule, enabling the estimation of the subsequent principal components, was presented in the works of Oja and Sanger [14,18]. In this case, the PCA task is performed by the use of a single-layered neural network consisting of K linear neurons, corresponding to the subsequent principal components.

The structure of the GHA-PCA network used for the approximation of the spatiotemporal DPS response is presented in Fig. 1. The number of network inputs is equal to the number of M spatial positions for which the value of the process variable y is determined, whereas number of network outputs is equal to the number of the K principal components.

According to the GHA, modification of the weight coefficients of the k-th neuron is performed after each presentation of the input patterns corresponding to the time instant t_n, based on the following expression [15,18]:

$$w_{k,m}(t_{n+1}) = w_{k,m}\,(t_n) + \eta v_k(t_n) \left[y(l_m, t_n) - \sum_{h=1}^{k} w_{h,m}(t_n) v_h(t_n) \right] \tag{5}$$

where $w_{k,m}(t_n)$ is the value of the weight coefficient connecting the k-th neuron

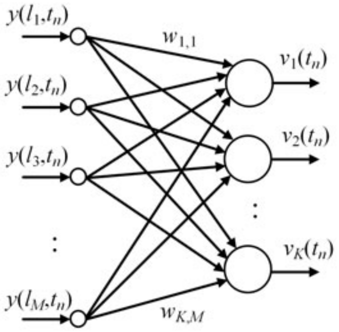


Fig. 1. Structure of the GHA-PCA neural network

with the m-th network input, $v_k(t_n)$ is the output signal of the k-th neuron, both calculated for the time instant t_n, and η is the network learning rate. Denoting by $w_k(t_n)$ vector containing all weight coefficients of the k-th neuron at the time instant t_n, i.e. vector of the following form:

$$w_k(t_j) = \left[w_{k,1}(t_n)\; w_{k,2}(t_n) \ldots w_{k,M}(t_n) \right], \tag{6}$$

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

$$y(t_n) = \left[y(l_1, t_n)\; y(l_2, t_n) \ldots y(l_M, t_n) \right]^T \tag{7}$$

and introducing the following notation:

$$y'(t_n) = y(t_n) - \sum_{h=1}^{k-1} \left(w_h(t_n)\right)^T v_h(t_n), \tag{8}$$

the relationship (5) can be written in the compact vector form:

$$w_k(t_{n+1}) = w_k\,(t_n) + \eta v_k(t_n) \left[\left(y'(t_n)\right)^T - w_k(t_n) v_k(t_n) \right], \tag{9}$$

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

As mentioned in Sec. 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 input data set represented by the matrix $Y \in \mathbb{R}^{M \times N}$ is replaced by the reduced data set consisting of the network weight matrix $W = [w_{k,m}] \in \mathbb{R}^{K \times M}$, the network responses matrix $V = [v_k(t_n)] \in \mathbb{R}^{K \times N}$ and the vector of time averages $\bar{y} = [\bar{y}(l_m)] \in \mathbb{R}^{M}$. The data compression ratio C_K can be thus calculated as:

$$C_K = \frac{M \times N}{M \times K + K \times N + M}\,. \tag{10}$$

Approximation of the spatiotemporal response is possible due to the "decompression", realized as simple multiplication of W^T by V and adding time-averaged values of $\bar{y}(l_m)$ to the result (see (1) and (2)).

2.2 Two-Layer Feedforward Network with Supervised Training (FF-PCA)

An alternative approach to extracting principal components from the data set is based on a feedforward, two-layer linear neural network of the structure shown in Fig. 2. The number of the network outputs (i.e. number of neurons in its second layer, hereinafter referred to as a *reconstruction layer*) is equal to the number of its inputs and represents the number M of spatial positions of the process variable y. Furthermore, the number of $K < M$ units in the first network layer, called the *projection layer*, represents the number of the principal components to be extracted. 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, as in the case of the GHA-PCA network, by the vectors representing distribution of the process variable y at the successive time instants t_n, i.e. by the subsequent columns of the matrix Y. In the considered case of the auto-associative network learning, the output patterns are equal to the 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 well-known form [10,15,17,19]:

$$E(w) = \frac{1}{M \cdot N} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(y(l_m, t_n) - \hat{y}(l_m, t_n)\right)^2. \tag{11}$$

Denoting by $W^{(1)} \in \mathbb{R}^{K \times M}$ the weight matrix of the projection layer, by $W^{(2)} \in \mathbb{R}^{M \times K}$ the weight 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. 2:

$$V = W^{(1)} Y \tag{12}$$

and

$$\hat{Y} = W^{(2)} V = W^{(2)} W^{(1)} Y. \tag{13}$$

As can be easily seen, (13) is equivalent to (2) and (4), wherein $W^{(1)}$ corresponds to ${\Phi_K}^T$ and V corresponds to Ψ_K. In order to determine the optimal values of the weight coefficients, a supervised learning procedure has to be applied – e.g. gradient descent or Levenberg-Marquardt algorithm [2,15,17,19].

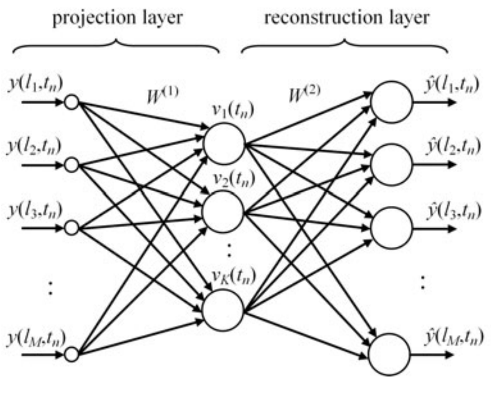


Fig. 2. Structure of the FF-PCA neural network

3 Example: Spatiotemporal Response of Hyperbolic DPS

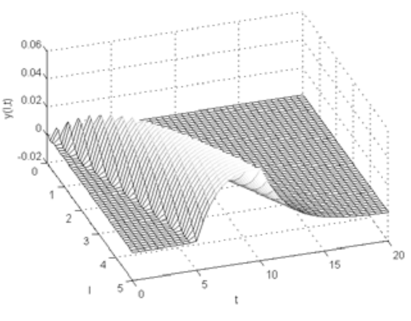
Among many different kinds of DPS, an important class is constituted by the processes in which the phenomena of mass and energy transport take place. One can mention here e.g. electrical transmission lines, transport pipelines or heat exchangers [1,3,4,9]. Their mathematical description takes the general form of the following two coupled partial differential equations of hyperbolic type:

$$\begin{aligned}\frac{\partial y_1(l,t)}{\partial t} + f_1 \frac{\partial y_1(l,t)}{\partial l} &= g_{11} y_1(l,t) + g_{12} y_2(l,t), \\ \frac{\partial y_2(l,t)}{\partial t} + f_2 \frac{\partial y_2(l,t)}{\partial l} &= g_{21} y_1(l,t) + g_{22} y_2(l,t),\end{aligned} \tag{14}$$

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, defined on the set $\Omega \times \Theta$, where $\Omega = [0, L]$ is the domain of the independent spatial variable l, while $\Theta = [0, T]$ is the domain of the independent variable t representing time. The constant coefficients $f_1, f_2 \in \mathbb{R}$ 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 determine the numerical solutions of (14), the *method of lines* has been applied for the following values of the equation parameters, the domain of the solution and the spatial discretization step: $f_1 = 1$, $f_2 = 0.5$, $g_{11} = -0.0638$, $g_{12} = 0.0638$, $g_{21} = -0.0359$, $g_{22} = 0.0359$, $L = 5$, $T = 50$, $\Delta l = 0.1$. The simulations were carried out assuming zero initial conditions, $y_1(l,0) = 0$ and $y_2(l,0) = 0$, as well as two different forms of boundary conditions: the Kronecker delta impulse and the Heaviside step function for the control variable $y_1(0,t)$. The solutions of (14) representing spatiotemporal distribution of the controlled variable $y_2(l,t)$ for both types of boundary conditions are shown in Fig. 3. In the following, the results of the response approximation using aforementioned neural PCA techniques are presented.

a)



b)

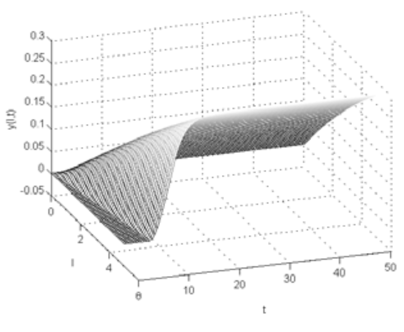


Fig. 3. Impulse (a) and step (b) spatiotemporal responses $y_2(l,t)$ of the system (14)

The results of the application of the GHA-PCA network with three neurons ($K = 3$) in the approximation of the step response of Fig. 3b are presented in Fig. 4. In Fig. 4a weight vectors of individual neurons are plotted, whereas Fig. 4b shows original step responses (solid line), obtained by the numerical solution of (14) and the responses of the GHA-PCA approximation model (dashed line), compared for six different spatial positions l.

Furthermore, Fig. 5 shows approximation results analogous to those presented in Fig. 4, obtained as a result of the use of the FF-PCA method discussed in Sec. 2.2. As can be seen, in contrast to the case of the GHA-PCA network, the weight vectors w_1, w_2 and w_3 of the projection layer of the FF-PCA network 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 GHA-PCA method.

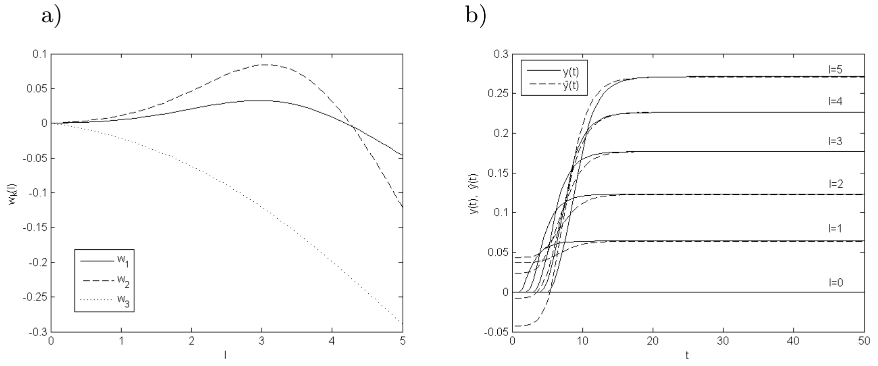


Fig. 4. GHA-PCA approximation results for the step response and $K = 3$: a) weight vectors w_1, w_2 and w_3, b) exact and approximate responses, $y(l,t)$ and $\hat{y}(l,t)$

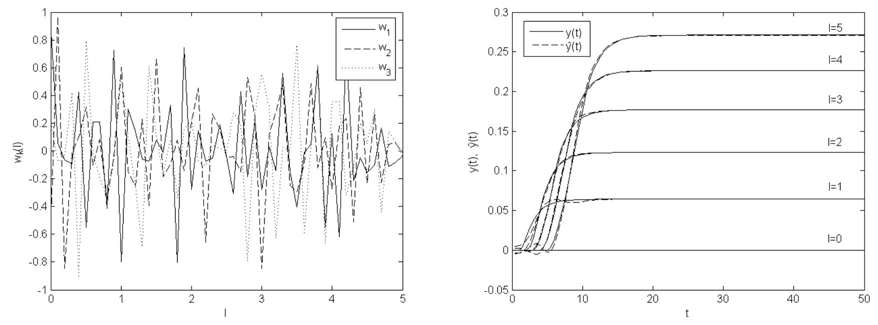


Fig. 5. FF-PCA approximation results for the step response and $K = 3$: a) weight vectors w_1, w_2 and w_3, b) exact and approximate responses, $y(l,t)$ and $\hat{y}(l,t)$

Table 1. Approximation results for GHA-PCA and FF-PCA neural networks (after 1000 training epochs)

		impulse response			step response		
		number of neurons K			number of neurons K		
		$K = 1$	$K = 3$	$K = 5$	$K = 1$	$K = 3$	$K = 5$
GHA-PCA	$\Vert E\Vert_{\mathrm{F}}$	0.335	0.111	0.074	0.820	0.689	0.680
	MSAE	$2.18\cdot10^{-5}$	$2.38\cdot10^{-6}$	$1.07\cdot10^{-6}$	$1.33\cdot10^{-4}$	$9.23\cdot10^{-5}$	$8.98\cdot10^{-5}$
	C_K	25.37	10.16	6.35	25.37	10.16	6.35
	T_t	1.9 s	2.5 s	3.1 s	1.9 s	2.5 s	3.1 s
FF-PCA	$\Vert E\Vert_{\mathrm{F}}$	0.337	0.147	0.143	0.773	0.724	0.650
	MSAE	$2.21\cdot10^{-5}$	$4.20\cdot10^{-6}$	$4.00\cdot10^{-6}$	$1.04\cdot10^{-4}$	$1.02\cdot10^{-4}$	$8.2\cdot10^{-5}$
	C_K	25.37	10.15	6.35	25.37	10.16	6.35
	T_t	1.0 s	1.7 s	2.4 s	1.0 s	1.7 s	2.4 s

The approximation results obtained for both considered PCA neural networks and for both spatiotemporal responses of Fig. 3 are summarized in Table 1. For each of these cases, the table contains: the Frobenius norm $\Vert E\Vert_{\mathrm{F}}$ of the approximation error matrix, the mean square approximation error (11) and the data compression coefficient C_K (10). In order to enable a rough estimation of the computational cost of the proposed algorithms, the training time values T_t averaged for 10 simulations of 1000 learning epochs performed on a 2.27 GHz Intel Core i5 processor with 4 GB of RAM memory are also included here. As can be seen from the presented results, the increase in K reduces the value of MSAE, however, it also decreases the value of C_K as well as increases the value of T_t. 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 well as low computation time.

4 Summary

In this paper, neural network-based PCA techniques as applied to the approximation of the spatiotemporal responses of a distributed parameter system have been discussed. A positive aspect of using artificial neural networks as a tool for extracting principal components from a spatiotemporal data set is that they do not require calculating the correlation matrix explicitly, as in the case of the classical PCA approach. 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 approach presented in the paper may act as a good starting point for further research concerning, for example, approximation of nonlinear DPS using nonlinear PCA method, based on the function approximation properties of neural networks with nonlinear, sigmoidal units.

References

1. Bartecki, K.: Frequency- and time-domain analysis of a simple pipeline system. In: Proceedings of the 14th IEEE IFAC International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje (August 2009)
2. Bartecki, K.: On some peculiarities of neural network approximation applied to the inverse kinematics problem. In: Proceedings of the Conference on Control and Fault-Tolerant Systems, Nice, France, pp. 317–322 (October 2010)
3. Bartecki, K.: Approximation of a class of distributed parameter systems using proper orthogonal decomposition. In: Proceedings of the 16th IEEE International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje, pp. 351–356 (August 2011)
4. Bartecki, K., Rojek, R.: Instantaneous linearization of neural network model in adaptive control of heat exchange process. In: Proceedings of the 11th IEEE International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje, pp. 967–972 (August 2005)
5. Berkooz, G., Holmes, P., Lumley, J.L.: The proper orthogonal decomposition in the analysis of turbulent flows. Annual Review of Fluid Mechanics 25, 539–575 (1993)
6. Bleris, L.G., Kothare, M.V.: Low-order empirical modeling of distributed parameter systems using temporal and spatial eigenfunctions. Computers and Chemical Engineering 29(4), 817–827 (2005)
7. Chatterjee, A.: An introduction to the proper orthogonal decomposition. Current Science 78(7), 808–817 (2000)
8. Diamantaras, K.I., Kung, S.Y.: Principal Component Neural Networks - Theory and Applications. John Wiley, New York (1996)
9. Friedly, J.C.: Dynamic Behaviour of Processes. Prentice Hall, New York (1972)
10. Hertz, J., Krogh, A., Palmer, R.: Introduction to the Theory of Neural Computation. Addison Wesley, Redwood City (1991)
11. Hoo, K.A., Zheng, D.: Low-order control-relevant models for a class of distributed parameters system. Chemical Engineering Science 56(23), 6683–6710 (2001)
12. Li, H.X., Qi, C.: Modeling of distributed parameter systems for applications - A synthesized review from time-space separation. Journal of Process Control 20(8), 891–901 (2010)
13. Oja, E.: A simplified neuron model as a principal component analyzer. Journal of Mathematical Biology 15(3), 267–273 (1982)
14. Oja, E.: Neural networks, principal components, and subspaces. International Journal of Neural Systems 1(1), 61–68 (1989)
15. Osowski, S.: Neural Networks for Information Processing. Warsaw University of Technology Press, Warsaw (2000)
16. Qi, C., Li, H.X.: A time-space separation-based Hammerstein modeling approach for nonlinear distributed parameter processes. Computers and Chemical Engineering 33(7), 1247–1260 (2009)
17. Rojas, R.: Neural networks: a systematic introduction. Springer, Berlin (1996)
18. Sanger, T.D.: Optimal unsupervised learning in a single-layer linear feedforward neural network. Neural Networks 2(6), 459–473 (1989)
19. Zurada, J.: Introduction to artificial neural systems. West Publishing Company, St. Paul, MN (1992)