Chemometrics and Intelligent Laboratory Systems 103 (2010) 152–159

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemolab

Hybrid artificial neural network based on BP-PLSR and its application in development of soft sensors

Yan Xuefeng *

Key Laboratory of Advanced Control and Optimization for Chemical Processes of Ministry of Education, East China University of Science and Technology, P.O. BOX 293, MeiLong Road 130, Shanghai 200237, PR China

ARTICLE INFO

Article history:
Received 22 May 2010
Accepted 8 July 2010
Available online 22 July 2010

Keywords:
Artificial neural network
Error back propagation
Partial least square regression
Correlation
Soft sensor

ABSTRACT

A novel hybrid artificial neural network (HANN) integrating error back propagation algorithm (BP) with partial least square regression (PLSR) was proposed to overcome two main flaws of artificial neural network (ANN), i.e. tendency to overfitting and difficulty to determine the optimal number of the hidden nodes. Firstly, single-hidden-layer network consisting of an input layer, a single hidden layer and an output layer is selected by HANN. The number of the hidden-layer neurons is determined according to the number of the modeling samples and the number of the neural network parameters. Secondly, BP is employed to train ANN, and then the hidden layer is applied to carry out the nonlinear transformation for independent variables. Thirdly, the inverse function of the output-layer node activation function is applied to calculate the expectation of the output-layer node input, and PLSR is employed to identify PLS components from the nonlinear transformed variables, remove the correlation among the nonlinear transformed variables and obtain the optimal relationship model of the nonlinear transformed variables with the expectation of the output-layer node input. Thus, the HANN model is developed. Further, HANN was employed to develop naphtha dry point soft sensor and the most important intermediate product concentration (i.e. 4-carboxybenzaldehyde concentration) soft sensor in p-xylene (PX) oxidation reaction due to the fact that there exist many factors having nonlinear effect on them and significant correlation among their factors. The results of two HANN applications show that HANN overcomes overfitting and has the robust character. And, the predicted squared relative errors of two optimal HANN models are all lower than those of two optimal ANN models and the mean predicted squared relative errors of HANN are lower than those of ANN in two applications.

1. Introduction

In response to demands for increasing petrochemical production levels and more stringent product quality specifications, the intensity and complexity of process operations at petrochemical industries have been exponentially increasing during the last three decades. To alleviate the operating requirements associated with these rising demands, plant designers and engineers are increasingly relying upon automatic control systems. However, the old inferred property predictors are neither sufficiently accurate nor reliable for utilization of advanced control applications [1]. Considering the existence of highly nonlinear relationships between the process variables (inputs) and the product stream properties (outputs), the implementation of intelligent control technology based on neural network soft sensor can remarkably enhance the regulatory and advanced control capabilities of the petrochemical industries [2,3]. There are various reasons [4–6]. First of all, neural networks are a good nonlinear function approximator. A number of researches have proven that neural networks can approximate any continuous function with arbitrary accuracy [7,8]. Secondly, neural networks can be trained to learn a chemical process by using process data. With plenty of data available from distributed control systems (DCS) in petrochemical processes, building a neural network based on process data is cost-effective [4,6].

However, there exist two main flaws in regular multi-layer neural networks [9,10,11], i.e. the tendency of overfitting and the difficulty to determine the optimal number of neurons for the hidden layer(s). In order to overcome these two flaws, a common approach is to train successively smaller networks until the smallest one is found which can learn the data [12–14]. This straightforward pruning algorithm, however, is rather inefficient, since a number of networks must be trained. To improve efficiency, principal component analysis (PCA) is often used to determine the optimal number of the hidden-layer neurons based on the first neural network model with excessive hidden-layer neurons, which is trained to sufficient precision [15,16]. In the pruning algorithm based on PCA, two networks must

* Tel.: +86 021 64252557; fax: +86 021 64253078.
E-mail addresses: yan_xuefeng@hotmail.com, xfyan@ecust.edu.cn.

0169-7439/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.chemolab.2010.07.002

be trained at least and PCA is carried out. To improve the predicted performance of neural networks, partial least square (PLS) method and PCA method are often combined with neural networks [17–20]. In these mixed algorithms combined with neural networks and regression, there are often two phases, which are the preprocessing phase and the neural network phase. In the preprocessing phase, PCA or PLS is applied to calculate the feature projection vector of sample data. Then, the feature projection vector is used as the input vector for neural networks. Thus, the preprocessing phase is used to extract variable components from the sample data, remove the correlation among the initial variables and reduce the input variable dimension of neural networks. But, the two main flaws in regular multi-layer neural networks are not solved in these mixed algorithms.

In this paper, a hybrid artificial neural network (HANN) integrating error back propagation algorithm (BP) with partial least square regression (PLSR) was proposed. First, a single-hidden-layer network consisting of an input layer, a single hidden layer and an output layer is selected because a neural network with a single hidden layer is sufficient for solving most functional approximation problems, and a larger number of the hidden-layer neurons are employed by HANN. Then, the neural network is trained to extract the quantitative information from the training samples using BP and the hidden layer is applied to carry out the nonlinear transformation for independent variables. Finally, the inverse function of the output-layer node activation function is applied to calculate the expectation of the output-layer node input, and PLSR is employed to identify PLS components from the nonlinear transformed variables, remove the correlation among the nonlinear transformed variables and obtain the optimal relationship model of the nonlinear transformed variables with the expectation of the output-layer node input. In the two illustrated examples of this paper, HANN was employed to develop two soft sensor models and satisfactory results were obtained.

The paper is organized as follows. Section 2 describes feedforward neural network structure, error back propagation algorithm, and partial least square regression. A systematic procedure to construct a HANN model based on BP-PLSR is also proposed in this section. The naphtha dry point soft sensor and the most important intermediate product concentration (i.e. 4-carboxybenzaldehyde concentration) soft sensor in p-xylene (PX) oxidation reaction were developed using HANN in Section 3. Moreover in Section 3 the performances of HANN are demonstrated and also compared with those of ANN obtained by BP algorithms. The paper is concluded with comments on the results found.

2. Hybrid artificial neural network

2.1. Error back propagation algorithm

The feedforward neural network is the most common neural network structure. Assume that each of input patterns is a vector represented by $\boldsymbol{x}$ consisting of p elements, and each of desired output patterns is a vector represented by $\boldsymbol{y}$ consisting of q elements. The number of layers for the network is $(L+1)$ and the number of neurons for the ith layer is N_i. The zeroth layer is the input layer, the Lth layer is the output layer, and the other layers, i.e. 1~$(L-1)$th layers, are the hidden layers. The number of input and output neurons corresponds to the number of the input and desired output elements, respectively, i.e. $N_0=p$ and $N_L=q$.

BP is one of the most popular algorithms for training a network due to its success from both simplicity and applicability viewpoint. The algorithm consists of two phases: training phase and recall phase. Assume that training sample set $\boldsymbol{S}$ consists of A pairs of the input and output patterns, and that the input and output data have been normalized to have the same order of magnitude. The standard procedure of the training phase for BP is described as follows.

1) Initialize the weights (and bias weights) of the neural network randomly.
2) Select the mth pair of the input and output pattern from the training sample set, i.e. $\boldsymbol{x}_m$ and $\boldsymbol{y}_m$, for the neural network. The inputs of each neuron are the weighted sum of the outputs from the previous layer, namely

$$z_{mjk}=\sum_{i=0}^{N_{j-1}} w_{jki}(t)\hat{y}_{m(j-1)i} \tag{1}$$

and

$$\begin{cases}\hat{y}_{m(j-1)i}=f\left(z_{m(j-1)i}\right) & j=2,3,\cdots,L\\ \hat{y}_{m(j-1)i}=x_{mi} & j=1\end{cases} \tag{2}$$

in which t denotes the number of iterations for BP, w_{jki} is the weight between the ith neuron of the $(j-1)$th layer and the kth neuron of the jth layer and w_{jko} represents the kth neuron bias weight of jth layer, $\hat{y}_{m(j-1)i}$ is the ith neuron output of the $(j-1)$th layer and $\hat{y}_{m(j-1)0}=1$, $f(\bullet)$ is the activation function, and x_{mi} is the ith element of $\mathbf{x}_m$. Thus, the outputs of the last layer $\hat{\mathbf{y}}_{mL}$ are calculated as below and considered the outputs of the network.

$$\hat{y}_{mLi}=f(z_{mLi}),\ i=1,2,\cdots,N_L$$

in which $\hat{y}_{mLi}$ is the ith output of the last layer, i.e. the ith element of $\hat{\mathbf{y}}_{mL}$.

3) Calculate the error E_m between network output $\hat{\mathbf{y}}_{mL}$ and the desired output $\mathbf{y}_m$ as

$$\begin{cases}E_m=\dfrac{1}{2}\sum_{k=1}^{N_L} e_{mLk}^2\\ e_{mLk}=y_{mk}-\hat{y}_{mLk}\end{cases} \tag{3}$$

in which y_{mk} is the kth element of $\mathbf{y}_m$.

4) Propagate error backward and adjust the weights (and bias weights) in such a way that minimizes the error E_m. Start from the output layer and go backward to the input layer. Namely, update the weights (and bias weights) of neural network as

$$w_{jki}(t+1)=w_{jki}(t)+\eta\delta_{mjk}\hat{y}_{m(j-1)i} \tag{4}$$

$$\begin{cases}\delta_{mjk}=f'\left(z_{mjk}\right)\sum_{i=1}^{N_{j+1}}\delta_{m(j+1)i}\,w_{(j+1)ik}(t) & j=L-1,\cdots,1\\ \delta_{mjk}=e_{mjk}f'\left(z_{mjk}\right) & j=L\end{cases} \tag{5}$$

in which $f'(\bullet)$ is the derivative of the activation function, η is a positive constant, called learning factor.

5) Repeat steps 2–5 for each pair of the input and output patterns in the training sample set and $t=t+1$ until the error for the training sample set is lower than the required minimum error $E_{\min}$, i.e.

$$E=\sum_{m=1}^{A} E_m\le E_{\min}.$$

In order to improve the learning speed and avoid the local minima, various modified BP algorithms [21] were proposed. After enough repetitions of these steps, the error between the neural network outputs and target outputs should be reduced to an acceptable value.

The network can be used in the recall or generalization phases where the weights are not changed.

2.2. Partial least square regression

Assume that the data for modeling consist of n pairs of observation data each of which is composed of a p-dimensional independent vector $\boldsymbol{x}$ and a q-dimensional dependent vector $\boldsymbol{y}$. Thus, the dimension of the independent variable matrix $\boldsymbol{X}$ is $n\times p$, and that of the dependent variable matrix $\boldsymbol{Y}$ is $n\times q$. The partial least square regression, whose goal is to optimize two objective functions, i.e. to minimize the variance of the prediction, while maximizing the covariance of $\boldsymbol{X}$ and $\boldsymbol{Y}$, can be applied to model matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$. The linear PLSR model is set up as

$$\mathbf{Y} = \mathbf{TC} + \mathbf{E} = \mathbf{XUC} + \mathbf{E} \tag{6}$$

in which $\boldsymbol{T}$ is the low-dimensional score matrix of $\boldsymbol{X}$ with the dimension of $n\times k$ (k is the number of loading vectors), $\boldsymbol{C}$ represents the regression coefficient matrix with the dimension of $k\times q$, $\boldsymbol{U}$ is the transformation matrix of $\boldsymbol{X}$ with the dimension of $p\times k$, and $\boldsymbol{E}$ is the residuals matrix with the dimension of $n\times q$. $\mathbf{C}$ and $\boldsymbol{U}$ can be determined by NIPALS algorithm [22,23] based on $\boldsymbol{X}$ and $\boldsymbol{Y}$. The details of PLSR can be referred to [22,23].

Due to the fact that $\boldsymbol{T}$ is the linear combination of the row vectors of $\boldsymbol{X}$ and will maximize the covariance between $\boldsymbol{X}$ and $\boldsymbol{Y}$, k plays an important role in the precision of PLSR model. The optimal k used in the score matrix $\boldsymbol{T}$ is often determined by the leave-one-out cross-validation method, in which one sample is left out in turn to test the prediction ability of the model based on the remaining samples. When the optimal k is determined, the PLSR model can be obtained and used for prediction or other purposes. For the predicted independent variable vector $\boldsymbol{x}$, the predicted dependent vector $\hat{y}$ is calculated as

$$\hat{\mathbf{y}} = \mathbf{C}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{x}. \tag{7}$$

2.3. Hybrid artificial neural network algorithm

To overcome the two main flaws of the neural network, HANN was proposed to train the neural network via BP-PLSR. Suppose that a three-layer neural network with N_1 hidden neurons is selected by HANN. Assume that each of the input patterns is a vector represented by $\boldsymbol{x}$ consisting of p elements, and each of the desired output patterns is a vector represented by $\boldsymbol{y}$ consisting of q elements. Therefore, the number of the input-layer neurons N_0 equals p, and the number of the output-layer neurons N_2 equals q. Sigmoid function is used as the activation function in the trial due to the fact that it has been successfully and widely applied to model complex and nonlinear systems. The sigmoid function is given by:

$$f(z) = \frac{1}{1+e^{-z}}.$$

The overall samples collected from the plant process are represented by $\boldsymbol{S}_{ALL}$, and are randomly divided into the training sample set $\boldsymbol{S}$ and the testing sample set $\boldsymbol{S}_{\boldsymbol{T}}$. $\boldsymbol{S}$ contains the A pairs of the input and output patterns, and $\boldsymbol{S}_{\boldsymbol{T}}$ consists of the $\boldsymbol{A}_{\boldsymbol{T}}$ pairs of the input and output patterns. Assume that the input and output data have been normalized to have the same order of magnitude. The training procedure for HANN based on BP-PLSR is described as follows.

1) BP is employed to train the neural network using the training sample set $\boldsymbol{S}$, so the weights (and bias weights) between the input layer and the hidden layer, i.e. $w_{1ki}, k=1,2,\cdots,N_1, i=0,1,2,\cdots,N_0$, and those between the hidden layer and the output layer, i.e. w_{2ki}, $k=1,2,\cdots,N_2, i=0,1,2,\cdots,N_1$, are determined.
2) For each input pattern $\mathbf{x}_m, m=1,2,\cdots,A$ of the training sample set $\boldsymbol{S}$, the hidden-layer output vector $\hat{y}_{m1}$ is calculated as follows according to Eqs. (1) and (2).

$$\begin{cases} z_{m1k} = \sum_{i=0}^{N_0} w_{1ki}x_{mi}, \ k=1,2,\cdots,N_1 \\ \hat{y}_{m1k} = f(z_{m1k}) \end{cases}$$

in which z_{m1k} is the input of the kth neuron of the hidden layer, x_{mi}, $i=1,2,\cdots,N_0$ is the ith element of $\mathbf{x}_m$ and $x_{m0}=1$, $\hat{y}_{m1k}$ is the kth element of $\hat{\mathbf{y}}_{m1}$ and regarded as a nonlinear transformed variable. The nonlinear transformed variable matrix $\overline{\mathbf{X}}$ is defined as follows.

$$\overline{\mathbf{X}} = \begin{bmatrix} 1 & \hat{y}_{111} & \hat{y}_{112} & \cdots & \hat{y}_{11N_1} \\ 1 & \hat{y}_{211} & \hat{y}_{212} & \cdots & \hat{y}_{21N_1} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \hat{y}_{A11} & \hat{y}_{A12} & \cdots & \hat{y}_{A1N_1} \end{bmatrix} \tag{8}$$

3) With the same method as step 2), calculate the nonlinear transformed variable matrix $\overline{X}_T$ for the testing sample $\boldsymbol{S}_{\boldsymbol{T}}$.
4) Employ the inverse of the output-layer neuron activation function for each desired output pattern $\mathbf{y}_m, m=1,2,\cdots,A$ of the training sample $\boldsymbol{S}$ to calculate the desired input $\overline{\mathbf{y}}_m$ of the output-layer neurons as follows.

$$\overline{y}_{mi} = f^{-1}(y_{mi}), i=1,2,\cdots,N_2 \tag{9}$$

in which $\overline{y}_{mi}$ is the desired input of the ith output-layer neuron, i.e. the ith element of $\overline{\mathbf{y}}_m$, y_{mi} is the ith element of $\mathbf{y}_m$, and $f^{-1}(\bullet)$ is the inverse of the output-layer neuron activation function. If sigmoid function is used as activation function, so the ith element $\overline{y}_{mi}$ of $\overline{\mathbf{y}}_m$ is calculated as

$$\overline{y}_{mi} = \ln\left(\frac{y_{mi}}{1-y_{mi}}\right), i=1,2,\cdots,N_2 \tag{10}$$

5) The desired dependent matrix $\overline{Y}$ for the training sample set is defined as $\begin{bmatrix} \overline{\mathbf{y}}_1 \\ \overline{\mathbf{y}}_2 \\ \vdots \\ \overline{\mathbf{y}}_A \end{bmatrix}$.
6) With the same method as step 4) and 5), calculate the desired dependent matrix $\overline{Y}_T$ for the testing sample $\boldsymbol{S}_{\boldsymbol{T}}$.
7) Based on $\overline{X}$ and $\overline{Y}$, PLSR is employed to obtain the relationship model between nonlinear transformed variables and the desired inputs of the output-layer neurons. Suppose the number of loading vectors is k, i.e. PLSR identifying k PLS components from $\overline{\mathbf{X}}$, the regression coefficient matrix is $\boldsymbol{C}_k$, and the transformation matrix is $\boldsymbol{U}_k$. Thus, the relationship model is represented as

$$\tilde{\mathbf{y}} = \mathbf{C}_k^{\mathrm{T}}\mathbf{U}_k^{\mathrm{T}}\,\overline{\mathbf{x}}$$

in which $\overline{\mathbf{x}}$ is the independent vector consisting of nonlinear transformed variables, $\tilde{\mathbf{y}}$ is the predicted vector for the desired inputs of the output-layer neurons. Then, the prediction value $\tilde{\mathbf{Y}}_T$ for $\overline{\mathbf{Y}}_T$ is calculated as follows.

$$\tilde{\mathbf{Y}}_T = \overline{\mathbf{X}}_T\mathbf{U}_k\mathbf{C}_k$$

Thus, the mean square prediction errors MSE_k corresponding to the value k (the number of loading vectors) is defined as

$$MSE_k = \frac{1}{A_T}\sum_{i=1}^{A_T}\sum_{j=1}^{N_2}\left(\frac{\tilde{y}_{T,ij}-\overline{y}_{T,ij}}{\overline{y}_{T,ij}}\right)^2$$

in which $\bar{y}_{T,ij}$ is the ith row and jth column element of $\bar{\mathbf{Y}}_T$ and $\tilde{y}_{T,ij}$ is the predicted value of $\bar{y}_{T,ij}$. Therefore, MSE_k is a function of k. Calculate MSE_k $k = 1, 2, \cdots, (N_1 + 1)$ and the optimal k denoted as k_{opt} has the minimal MSE_k. The optimal k_{opt} can also be determined by the leave-one-out cross-validation method. When the optimal k_{opt} is obtained, the HANN model is developed. HANN structure is shown as Fig. 1.

During the recall phase, for the predicted independent variable vector $\mathbf{x}_P$, the calculating procedure of the HANN model is as follows.

1) According to Eqs. (1) and (2), calculate $\hat{\mathbf{y}}_{P1}$, i.e. nonlinear transformed vector of $\mathbf{x}_P$.
2) Calculate the inputs of the output-layer neurons as $\tilde{\mathbf{y}}_P = \mathbf{C}_k^T \mathbf{U}_k^T \hat{\mathbf{y}}_{P1}$.
3) According to the activation function of the output-layer neuron, calculate the outputs of the HANN model as $\hat{y}_{p2i} = f\left(\tilde{y}_{pi}\right), i = 1, 2, \cdots, N_2$, in which $\tilde{y}_{pi}$ is the ith element of $\tilde{\mathbf{y}}_P$.

3. Application of hybrid artificial neural network

3.1. Development of naphtha dry point soft sensor using HANN

3.1.1. Preflash tower

A crude distillation unit is generally and widely utilized in chemical and petroleum industries for separation of crude oil. A preflash tower (PT) is one of the critical and important unit operations for petroleum industry. A typical PT consists of the main column with a large number of equilibrium stages and pump-arounds as shown in Fig. 2. The preheated crude oil enters the flash zone near the bottom of the main column. Pump-around flows are used to regulate the vapor traffic through the column as well as for heat recovery. A typical list of products from PT includes naphtha taken from the condenser at the top of the main column and preflash residue as the bottom product.

Thus, the function of PT is to separate the crude oil into two kinds of the fractionation products, i.e. naphtha and preflash residue. The separation process of PT involves many complex phenomena between input and output variables of the system, although only physical instead of chemical reaction presents in the unit. The input variables are usually manipulated variables of PT, such as energy supply inputs, reflux ratios, and product flow rates; while the output variables are the fractionation product qualities.

It was known that the mass flow rate of the crude oil (x_1,Kg/Hr), the temperature at the top of the main column (x_2, °C), the pressure at the top of the main column (x_3, MPa), the special energy taken away by the reflux (x_4, °C), the reflux ratio of the main column (x_5, %), the mass flow rate of naphtha (x_6, Kg/Hr), the special energy taken away by the first pump-around (x_7, °C), the special energy taken away by the second pump-around (x_8, °C), and the temperature at the flash zone (x_9, °C) are considered the main input variables influencing the naphtha dry point in PT. The values of all input variables except x_4, x_5, x_7 and x_8 can be real-time collected from distributed control systems (DCS) in PT directly. x_4, x_5, x_7 and x_8 are defined as

$$\begin{cases} x_4 = \dfrac{(x_2 - t_1)m_0}{x_1} \\ x_5 = \dfrac{m_0}{x_6} \\ x_7 = \dfrac{\Delta t_1 m_1}{x_1} \\ x_8 = \dfrac{\Delta t_2 m_2}{x_1} \end{cases} \tag{11}$$

in which t_1 denotes the temperature of the reflux (°C), m_0 denotes the mass flow rate of the reflux (Kg/Hr), Δt_i denotes the temperature change of the ith pump-around (°C), and m_i denotes the mass flow rate of the ith pump-around (Kg/Hr). Thus, according to Eq. (11), the four input variables (i.e. x_4, x_5, x_7 and x_8) can also be real-time calculated based on the other variables available from DCS. The naphtha dry point, denoted by y in (°C), is determined by off-line laboratory analysis every 4 hours. At the same time, the crude oil property has an important effect on the naphtha dry point, but the crude oil property is seldom known. Considering the effect of the crude oil property on the naphtha dry point only, Yan et al. [24] proposed to regard the previous analysis value of the naphtha dry point as the crude oil property indirectly, and thus the previous analysis value of the naphtha dry point is considered as the tenth input variable (x_{10}, °C) influencing the current naphtha dry point. Thus, developing the relationship model of the naphtha dry point with ten input variables can achieve the real-time measure of the naphtha dry point for further use.

3.1.2. Sample collected from preflash tower

Ten input variables, i.e. $x_1, x_2, x_3, \ldots\ldots, x_{10}$ influencing the naphtha dry point in PT are regarded as the independent variables of the naphtha dry point soft sensor, and thus independent vector $\mathbf{x} = [x_1, x_2, x_3, \cdots, x_{10}]$. The naphtha dry point y is considered a dependent variable. The sample $\boldsymbol{S}_{ALL}$ collected from PT consists of 275 pairs of the independent vectors and the dependent variables (i.e. observation variables). $\boldsymbol{S}_{ALL}$ is randomly divided into the modeling sample $\boldsymbol{S}$ containing 184 pairs of the observation variables and the test sample $\boldsymbol{S}_T$ consisting of 91 pairs of the observation variables.

3.1.3. Performances of HANN

In order to illustrate the performance of the model obtained by HANN, BP and BP-PLSR are applied to develop the naphtha dry point soft sensor, respectively. BP and BP-PLSR select the same neural network structure. The neural network developed by BP uses the weights (and bias weights) resulting from the first step of BP-PLSR. To illustrate the influence of the number of the hidden-layer neurons N_1 on the performances of the models (i.e. ANN obtained by BP and HANN obtained by BP-PLSR), N_1 is set in turn to be each value of {1 , 2 , 3 , 4 , ⋯ , 20}. Thus, twenty tests were executed in all for BP and BP-PLSR, respectively. To reduce the influence of incidental variation, BP and BP-PLSR were executed ten times for each test, respectively.

For the testing sample set S_T, the mean square prediction errors MSE_{N_1} of ANN or HANN with N_1 hidden-layer neurons is defined as

$$MSE_{N_1} = \frac{1}{A_T}\left(\mathbf{y}_T - \hat{\mathbf{y}}_T\right)^T \left(\mathbf{y}_T - \hat{\mathbf{y}}_T\right) \tag{12}$$

in which $\boldsymbol{A_T}$ is the number of pairs of the observation variables in the testing sample set $\boldsymbol{S_T}$, i.e. $\boldsymbol{A_T} = 91$, $\mathbf{y_T}$ is the dependent vector of the testing sample set $\boldsymbol{S_T}$, and $\hat{\mathbf{y}}_\mathbf{T}$ is the prediction vector for $\mathbf{y_T}$.

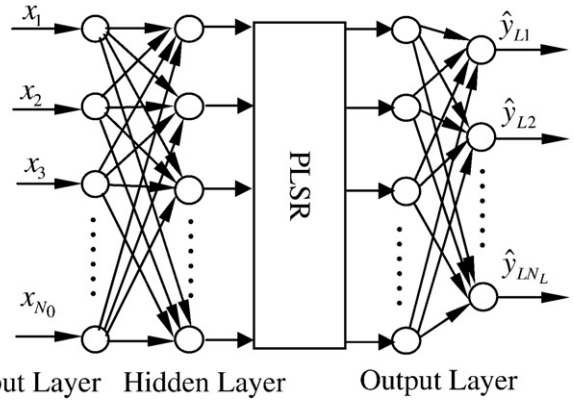

Fig. 1. Structure of hybrid artificial neural networks.

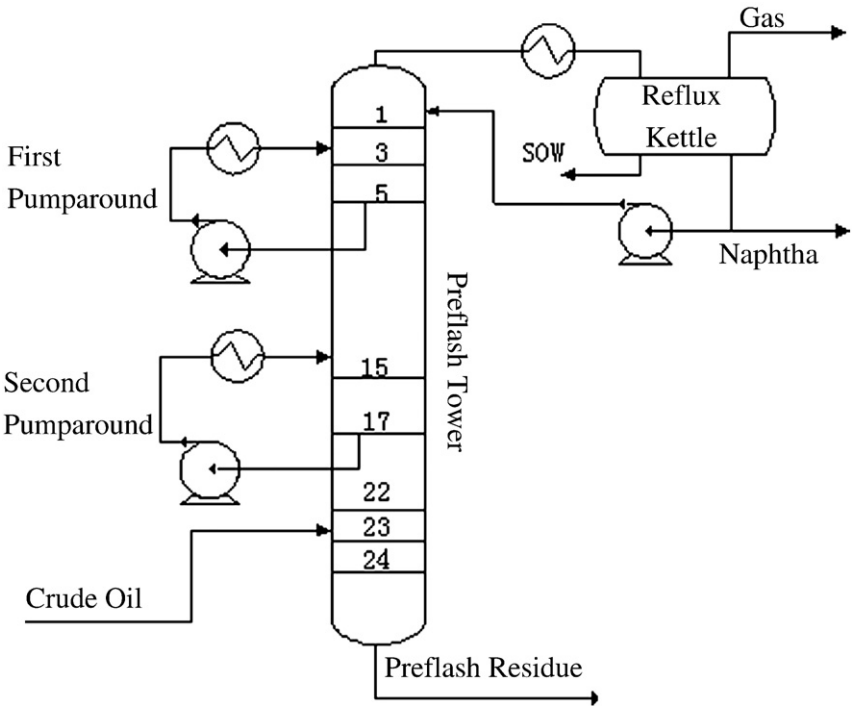

Fig. 2. Flow chart of the preflash tower.

For N_1 hidden-layer neurons, the mean MSE_{N_1}, represented by $\overline{MSE}_{N_1}$, is calculated as

$$\overline{MSE}_{N_1} = \frac{1}{10}\sum_{i=1}^{10} MSE_{N_1}^{i} \tag{13}$$

in which the superscript i denotes the ith executing trial for N_1 hidden-layer neurons test. And, the mean $\overline{MSE}_{N_1}$ is defined as

$$\overline{E} = \frac{1}{20}\sum_{N_1=1}^{20} \overline{MSE}_{N_1} \tag{14}$$

The MSE_{N_1} distributions of ANN and HANN with N_1 are shown in Figs. 3 and 4 respectively. In order to compare the MSE_{N_1} distributions of two approaches simply, Figs. 3 and 4 have the same range of y-axis. Owing to the main merit of HANN that PLSR is embedded in BP to obtain the optimal relationship model of the nonlinear transformed variables with the expectation of the output-layer neuron input, the range of MSE_{N_1} distributions shown in Fig. 3 is larger than that of HANN shown in Fig. 4. In other words, HANN has the better robust character than ANN. Figs. 3 and 4 also show $\overline{MSE}_{N_1}$ curves of ANN and HANN with respect to the number of the hidden-layer neurons N_1, respectively. As shown in Figs. 3 and 4, $\overline{MSE}_{N_1}$ of ANN is decreased along with N_1 due to the lack of the hidden-layer neurons when $N_1 \leq 2$, then increased along with N_1 due to the redundant hidden-layer neurons when $N > 2$; while $\overline{MSE}_{N_1}$ of HANN is slightly wavy within $1 \leq N_1 \leq 20$. The prediction abilities of HANN with the different number of the hidden-layer neurons N_1 are slightly different. Therefore, it is easy to determine the number of the hidden-layer neurons, only select a larger number of neurons for the hidden layer when HANN is applied. $\overline{E}$ of HANN is 0.026421 and 7.99% lower than that of ANN, which is 0. 028715. For HANN with the optimal number

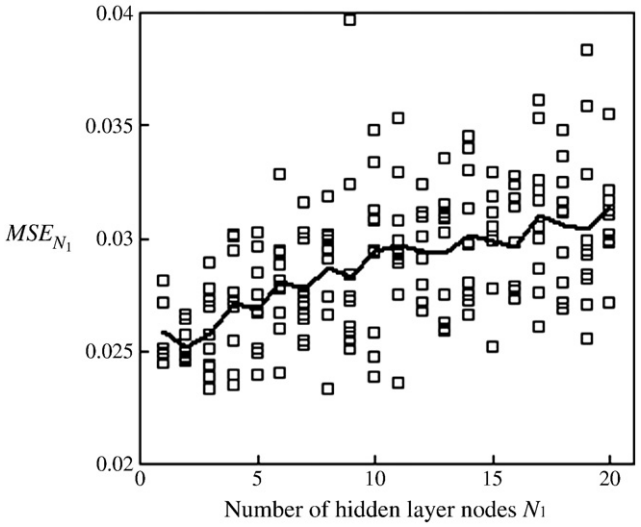

Fig. 3. The distribution of ANN MSE_{N_1} versus N_1.

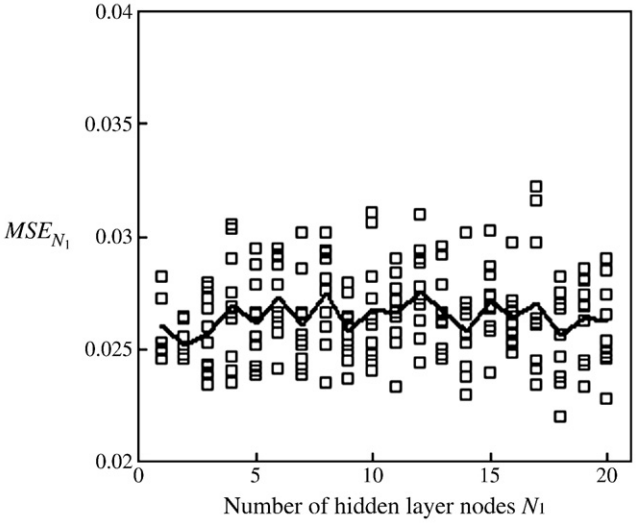

Fig. 4. The distribution of HANN MSE_{N_1} versus N_1.

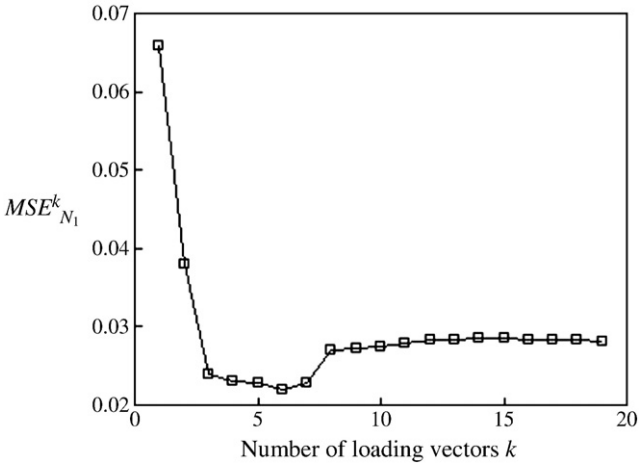

Fig. 5. Curve of HANN $MSE^k_{N_1}$ with k.

of loading vectors $k_{opt}=6$, the best $MSE^{opt}_{18}=0.021947$ is 5.64% lower than the best $MSE^{opt}_{8}=0.023259$ of ANN.

3.1.4. Analysis of loading vector number

In order to analyze the effect of the loading vector number k in PLSR on the predicting performance of HANN, Fig. 5 shows the relationship between $MSE^k_{N_1}$ and k in the obtained best HANN trial (i.e. $N_1=18$), in which the superscript k denotes the loading vector number. It can be seen from Fig. 5 that $MSE^{k_{opt}}_{N_1}$ has the minimum value 0.021947 when $k_{opt}=6$, which means that HANN shows the best predicting ability under the condition of $k_{opt}=6$.

3.2. Develop of 4-CBA concentration soft sensor using HANN

3.2.1. Amoco oxidation process

The Amoco oxidation process of PX to TA shown in Fig. 6 mainly consists of a feed mixing drum, an Amoco reactor and the first crystallizer for the crystallization process that includes three crystallizers. The feed including the liquid reactant (i.e. p-xylene) mixed with catalysts (i.e. cobalt and manganese), promoter (i.e. bromide) and acetic acid solvent is prepared in the feed mixing drum and then is fed to the Amoco reactor with air. The liquid-phase catalytic oxidation reaction takes place in the Amoco reactor to form TA. Then along with the air, the TA slurry from the Amoco reactor is supplied to the first crystallizer where some further oxidation occurs. The first crystallizer can be regarded as another Amoco reactor to a certain extent. TA from the first crystallizer is then supplied to a product recovery stage where it is separated and dried. Dry TA produced is then sent to storage.

In the Amoco oxidation process, the primary intermediate product is 4-carboxybenzaldehyde (4-CBA). The TA produced in the Amoco oxidation process is sent to AMOCO purification process to remove the impurities (primarily 4-CBA) and produce the final purified TA (PTA). The ability to control the 4-CBA concentration in the TA at its optimal value accurately and automatically is of considerable interest to many PTA industries since it can enable them to reduce their production costs and increase the yield while at the same time maintaining the quality of the final product PTA. However the control system design of the 4-CBA concentration is not straightforward due to the lack of the reliable on-line sensor that can detect the 4-CBA concentration, and the fact that the 4-CBA concentration is determined by off-line laboratory analysis with a 4 h delay. Therefore, the development of the 4-CBA concentration on-line soft sensor is required for the optimization and control of the TA quality.

It was known that the reaction temperature of the Amoco reactor (x_1, °C), the reaction temperature of the first crystallizer (x_2, °C), the weight percentage of cobalt in the feed (x_3, %), the weight percentage of manganese in the feed (x_4, %), the weight percentage of bromine in the feed (x_5, %), the residence time of the Amoco reactor (x_6, Hr), the residence time of the first crystallizer (x_7, Hr), and the ratio of the outlet stream from the Amoco reactor to the feed (x_8, %) are considered the main factors [25–28] influencing the 4-CBA concentration in the Amoco oxidation process. The values of these eight main factors can be real-time collected from DCS in the Amoco oxidation process. The 4-CBA concentration in the TA, denoted by y in [%], is determined by off-line laboratory analysis. Thus, developing the relationship model of the 4-CBA concentration with these eight main factors can estimate the real-time measure of the 4-CBA concentration for further use.

3.2.2. Sample collection in Amoco oxidation process

The eight main factors, i.e. x_1, x_2, x_3, x_4, x_5, x_6, x_7 and x_8, influencing the 4-CBA concentration in the Amoco oxidation process are regarded as the independent variables of the 4-CBA concentration on-line soft sensor, i.e., independent vector $\mathbf{x}=[x_1,x_2,x_3,\cdots,x_8]$. The 4-CBA concentration y is considered a dependent variable. The sample set $\boldsymbol{S}_{ALL}$ collected from the Amoco oxidation process consists of 197 pairs of the independent vectors and the dependent variables (i.e. observation variables), and assumes that the sample data have been normalized to the same order of magnitude. $\boldsymbol{S}_{ALL}$ is randomly divided into the modeling sample set $\boldsymbol{S}$ containing 132 pairs of the observation variables and the testing sample set $\boldsymbol{S}_T$ consisting of 65 pairs of the observation variables.

3.2.3. Performances of HANN

In order to illustrate the performance of the model obtained by HANN, BP and BP-PLSR are applied to develop the naphtha dry point

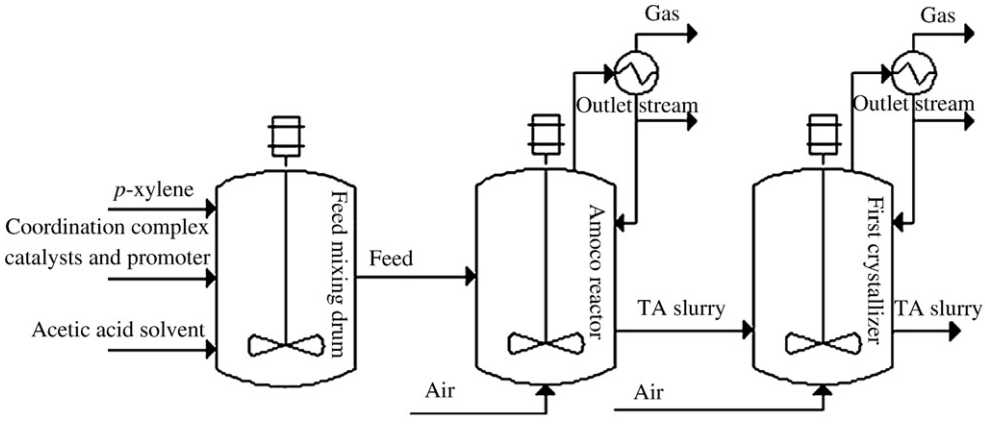

Fig. 6. Flow chart of the PX oxidation reaction process.

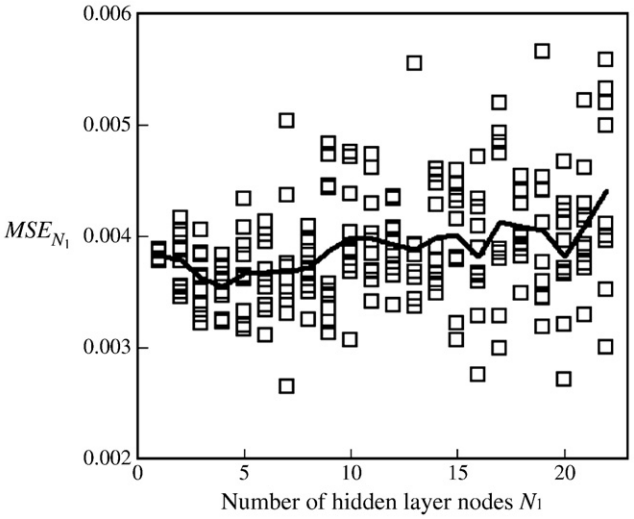

Fig. 7. The distribution of ANN MSE_{N_1} versus N_1.

soft sensor, respectively. BP and BP-PLSR select the same neural network structure. The neural network developed by BP uses the weights (and bias weights) resulting from the first step of BP-PLSR. To illustrate the influence of the number of the hidden-layer neurons N_1 on the performances of the models (i.e. ANN obtained by BP and HANN obtained by BP-PLSR), N_1 is set in turn to be each value of $\{1, 2, 3, 4, \cdots, 22\}$. Thus, twenty two tests were executed in all for BP and BP-PLSR, respectively. To reduce the influence of incidental variation, BP and BP-PLSR were executed ten times for each test, respectively.

The MSE_{N_1} distributions of ANN and HANN with N_1 are shown in Figs. 7 and 8 respectively. In order to compare the MSE_{N_1} distributions of the two approaches simply, Figs. 7 and 8 have the same range of y-axis. Owing to the main merit of HANN that the optimal relationship model of the nonlinear transformed variables with the expectation of the output-layer neuron input is obtained by BP-PLSR, the range of MSE_{N_1} distributions shown in Fig. 7 is larger than that of HANN shown in Fig. 8. In other words, HANN has the better robust character than ANN. Figs. 7 and 8 also show $\overline{MSE}_{N_1}$ curves of ANN and HANN with respect to the number of the hidden-layer neurons N_1, respectively. As shown in Figs. 7 and 8, $\overline{MSE}_{N_1}$ of ANN is decreased along with N_1 due to the lack of the hidden-layer neurons

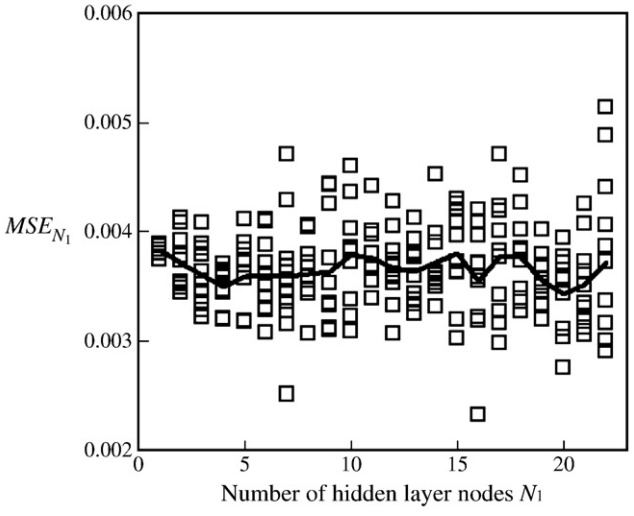

Fig. 8. The distribution of HANN MSE_{N_1} versus N_1.

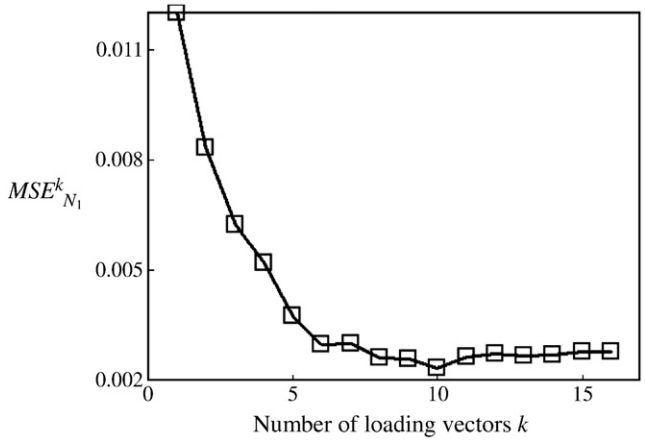

Fig. 9. Curve of HANN $MSE^k_{N_1}$ with k.

when $N_1 \leq 4$, then increased along with N_1 due to the redundant hidden-layer neurons when $N > 4$; while $\overline{MSE}_{N_1}$ of HANN is slightly wavy within $1 \leq N_1 \leq 22$. The prediction abilities of HANN with the different number of the hidden-layer neurons N_1 are slightly different. Therefore, it is easy to determine the number of the hidden-layer neurons, only select a larger number of neurons for the hidden layer when HANN is applied. $\overline{E}$ of HANN is 0.00365 and 6.2% lower than that of ANN, which is 0. 00389. For HANN with the optimal number of loading vectors $k_{opt} = 10$, the best $MSE^{opt}_{16} = 0.00232$ is 12.1% lower than the best $MSE^{opt}_{7} = 0.00264$ of ANN.

3.2.4. Analysis of loading vector number

In order to analyze the effect of the loading vector number k in PLSR on the predicting performance of HANN, Fig. 9 shows the relationship between $MSE^k_{N_1}$ and k in the obtained best HANN trial (i.e. $N_1 = 16$), in which the superscript k denotes the loading vector number. It can be seen from Fig. 9 that $MSE^{k_{opt}}_{N_1}$ has the minimum value 0.00232 when $k_{opt} = 10$, which means that HANN shows the best predicting ability under the condition of $k_{opt} = 10$.

4. Conclusion

Hybrid artificial neural network (HANN) introduces partial least square regression (PLSR) into artificial neural network, which is first trained by error back propagation algorithm (BP), to remove the multicollinearity among the hidden-layer-neuron output data and to improve the relationship model of the hidden-layer-neuron output with the output-layer-neuron input. Moreover, it is easy to determine the number of the hidden-layer neurons for HANN due to the fact that PLSR is able to remove the redundant information among the hidden-layer-neuron output data. Thus, a larger number of the hidden-layer neurons are employed by HANN and the number of the hidden-layer neurons can be determined according to the number of the modeling samples and the number of the neural network parameters. HANN is a generalized methodology for modeling the petrochemical process, in which there exist highly nonlinear relationships between input variables and output variables.

Acknowledgements

The authors gratefully acknowledge the supports from the following foundations: National Natural Science Foundation of China (20776042), National High-Tech Research and Development Program of China (863 Program: 2007AA04Z164), Doctoral Fund of Ministry of Education of China (20090074110005), Program for new century excellent talents in university, "Shu Guang" project (09SG29) supported by Shanghai Municipal Education Commission and Shanghai Education

Development Foundation, and Shanghai Leading Academic Discipline Project (B504).

References

[1] A. Zilouchian, M. Jamshidi, Intelligent Control Systems Using Soft Computing Methodologies, CRC Press, New York, 2001.
[2] K.H. Bawazeer, A. Zilouchian, Prediction of Crude Oil Production Quality Parameters Using Neural Networks, Proceedings of 1997 IEEE Int. Conference on Neural Networks, New Orleans.
[3] A. Draeger, S. Engell, H. Ranke, Model predictive control using neural networks, IEEE Control Magazine 15 (5) (1995) 61–67.
[4] A.J. de Assis, R.M. Filho, Soft sensors development for on-line bioreactor state estimation, Computers & Chemical Engineering 24 (2000) 1099–1103.
[5] L. Fortuna, S. Graziania, M.G. Xibilia, Soft sensors for product quality-monitoring in debutanizer distillation columns, Control Engineering Practice 13 (2005) 499–508.
[6] G.C. Goodwin, Predicting the performance of soft sensors as a route to low cost automation, Annual Reviews in Control 24 (2000) 55–66.
[7] K.J. Hunt, D. Sbarbaro, R. Zbikowski, P.J. Gawthrop, Neural networks for control systems — a survey, Automatica 28 (1992) 083–1099.
[8] B. Widrow, M.A. Lehr, 30 years of adaptive neural networks: perception, madaline and backpropagation, Proceedings of the IEEE 78 (1990) 1441–1457.
[9] M. Lehtokangas, Modeling with constructive back propagation, Neural Networks 12 (1999) 707–716.
[10] M.H. Nguye, A.A. Hussein, I.M. Robert, Stopping criteria for ensemble of evolutionary artificial neural networks, Applied Soft Computing 6 (2005) 100–107.
[11] R.M. Zur, Y. Jiang, C.E. Metz, Comparison of two methods of adding jitter to artificial neural network training, International Congress Series 1268 (2004) 886–889.
[12] S.E. Fahlman, C. Lebiere, The cascade correlation learning architecture, Advances in Neural Information Processing Systems 2 (1989) 519–526.
[13] R. Reed, Pruning algorithms. A survey, IEEE Transactions on Neural Networks 4 (5) (1989) 740–747.
[14] J. Sietsmas, R.J. Dow, Neural net pruning — why and now, Proceeding International Joint Conference Neural Networks, 1, 1988, pp. 325–333, San Diego, CA.
[15] C. Schittenkopf, G. Deco, W. Brauer, Two strategies to avoid overfitting in feedforward networks, Neural Networks 10 (3) (1997) 505–516.
[16] L. Zhang, et al., Multivariate nonlinear modeling of fluorescence data by neural network with hidden node pruning algorithm, Analytica Chimica Acta 344 (1997) 29–39.
[17] D. Wu, et al., BP neural networks combined with PLS applied to pattern recognition of Vis/NIRs, Lecture Notes in Computer Science 4224 (2006) 428–435.
[18] A. Debiolles, L. Oukhellou, P. Aknin, Combined use of partial least squares regression and neural network for diagnosis tasks, Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04), 4, 2004, pp. 573–576.
[19] A. Eleyan, H. Demirel, Face recognition system based on PCA and feedforward neural networks, Lecture Notes in Computer Science 3512 (2005) 935–942.
[20] R.H. Li, et al., Combined use of partial least-squares regression and neural network for residual life estimation of large generator stator insulation, Measurement Science and Technology 18 (2007) 2074–2082.
[21] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representation by back-propagating errors, Nature 323 (1986) 533–536 (London).
[22] P. Geladi, B.R. Kowalski, Partial least-squares regression: a tutorial, Anlytical Chemical Acta 185 (1986) 1–17.
[23] A. Hoskuldsson, PLS regression methods, Journal of Chemometrics 2 (1988) 211–228.
[24] X.F. Yan, J. Yu, F. Qian, Development of naphtha dry point soft sensor by adaptive partial least square regression, Journal of Chemical Industry and Engineering 56 (8) (2005) 1511–1515 (China).
[25] A. Saffer, R.S. Barker, Aromatic Polycarboxylic Acids, U.S. Pat. 2833816, 1958.
[26] X.F. Yan, W.L. Du, F. Qian, Development of a kinetic model for industrial oxidation of p-xylene by RBF-PLS and CGA, AIChE Journal 50 (6) (2004) 1169–1176.
[27] J. Zhang, Experimental Studies on Kinetics of p-Xylene Liquid-Phase Oxidation at High Temperature, Zhejiang University Dissertation for Master Degree (China), 2000.
[28] G.H. Zhou, Studies on kinetics of p-xylene liquid-phase catalytic oxidation, Tianjin University Dissertation for Master Degree (China), 1988.