Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Hydrology xx (2005) 1–17
Journal of Hydrology
www.elsevier.com/locate/jhydrol

Forecasting daily streamflow using hybrid ANN models

Wen Wang [a,b,*], Pieter H.A.J.M. Van Gelder [b], J.K. Vrijling [b], Jun Ma [c]

[a] State Key Laboratory of Hydrology-Water Resources and Hydraulig Engineering, Hohai University, Nanjing 210098, China
[b] Section of Hydraulic Engineering, Faculty of Civil Engineering and Geosciences, Delft University of Technology, P.O.Box 5048, 2600 GA Delft, The Netherlands
[c] Yellow River Conservancy Commission, Hydrology Bureau, Zhengzhou 450004, China

Received 1 November 2004; revised 23 September 2005; accepted 27 September 2005

Abstract

In this paper, the classic 'divide and conquer (DAC)' paradigm is applied as a top-down black-box technique for the forecasting of daily streamflows from the streamflow records alone, i.e. without employing exogenous variables of the runoff generating process such as rainfall. To this end, three forms of hybrid artificial neural networks (ANNs) are used as univariate time series models, namely, the threshold-based ANN (TANN), the cluster-based ANN (CANN), and the periodic ANN (PANN). For the purpose of comparison of forecasting efficiency, the normal multi-layer perceptron form of ANN (MLP–ANN) is selected as the baseline ANN model. Having first applied the MLP–ANN models without any data-grouping procedure, the influence of various data preprocessing procedures on the MLP–ANN model forecasting performance is then investigated. The preprocessing procedures considered are: standardization, log-transformation, rescaling, deseasonalization, and combinations of these. In the context of the single streamflow series considered, deseasonalization without rescaling was found to be the most effective preprocessing procedure. Some discussions are presented (i) on data preprocessing and (ii) on selection of the best ANN model. Overall, among the three variations of hybrid ANNs tested, the PANN model performed best. Compared with the MLP–ANN fitted to the deseasonalized data, the PANN based on the soft seasonal partitioning performed better for short lead times ($\leq$3 days), but the advantage vanishes for longer lead times.

Keywords: Streamflow forecast; Hybrid artificial neural networks; Periodic ANN; Threshold ANN; Cluster-based ANN; Fuzzy c-means clustering

1. Introduction

It is generally accepted that streamflow generation processes, especially daily streamflow processes, are seasonal and nonlinear, since the processes usually have pronounced seasonal means, variances, and dependence structures, and the under-lying mechanisms of streamflow generation are likely to be quite different during low, medium, and high flow periods, especially when extreme events occur. For instance, low-flow events are mainly sustained by base flow, whereas high-flow events are typically generated by intense storm rainfall.

* Corresponding author. Address: State Key Laboratory of Hydrology-Water Resources and Hydraulic Engineering, Hohai University, Nanjing, 210098, China. Tel.: +86 25 83787530.
E-mail address: w.wang@126.com (W. Wang).

0022-1694/$ - see front matter © 2005 Published by Elsevier B.V.
doi:10.1016/j.jhydrol.2005.09.032

Many models can be found in the literature for modeling the complex nonlinearity of streamflow processes, among which are those based on the principle of divide-and-conquer (DAC). DAC algorithms deal with a complex problem by dividing it into simple problems whose solutions can be combined to yield a solution to the complex problem (Jordan and Jacobs, 1994). Depending on the feature of nonlinearity, usually a process could be divided, for example through using thresholds, into a number of regimes and fit a linear or nonlinear model for each regime. Correspondingly, the DAC algorithms could be roughly categorized into two types, i.e. local-linear models and local-nonlinear models.

Local-linear DAC models approximate a complex problem locally with linear models. They are fundamentally derivatives of the threshold regression models, such as the constrained linear system with thresholds (CLS-Ts) of Todini and Wallis (1977) and the multi-linear approach of Becher and Kundzewicz (1987), or the threshold autoregressive (TAR) model (Tong and Lim, 1980), which treats a nonlinear process piecewise with linear regression models according to the value of some explanatory variable or the preceding value of the process itself. Local-linear DAC models are still used for modeling rainfall-runoff process (e.g. Solomatine and Dulal, 2003). The periodic autoregressive moving average (PARMA) model, as well as its abbreviated version periodic autoregressive (PAR) model, is widely used to model hydrologic time series (e.g. Hipel and Mcleod, 1994). It is the extension of the ARMA model that allows periodic (seasonal) parameters, which can be perceived as a modification of the threshold regression model. Instead of using the value of some explanatory variable or the preceding value of the process itself as the threshold, the PARMA model treats seasonal process piecewise with linear ARMA models according to which season the process is operating in. When fitting a PARMA model to a seasonal series, a separate ARMA model is fitted for each season of the year. Literature on PARMA (including PAR) models has abounded since the late 1960s' (e.g. Jones and Brelsford, 1967; Pagano, 1978; Salas et al., 1982; Vecchia, 1985; Bartolini et al., 1988; Salas and Abdelmohsen, 1993).

Local-nonlinear DAC models approximate a complex problem locally with nonlinear models, such as nonlinear regression models or artificial neural network (ANN) models. Local-nonlinear DAC models are increasingly popular for dealing with complex nonlinear processes, mainly owing to the rapid development of ANN techniques. ANNs are known as having the ability of modeling nonlinear mechanisms. They have been increasingly applied to various hydrological problems in the past decade (Maier and Dandy, 2000; Dawson and Wilby, 2001). Nonetheless, some studies have suggested that a single ANN cannot predict the high- and low-runoff events satisfactorily (e.g. Minns and Hall, 1996) since, as aforementioned, the under-lying mechanisms of streamflow generation can be quite different during low, medium, and high flow periods. The mapping ability of a single ANN is limited when faced with complex problems like rainfall-runoff processes. To resolve such complex processes, tree-structured neural networks, such as the modular neural network (MNN) (Jacobs et al., 1991; Jacobs and Jordan, 1993; Jordan and Jacobs, 1994), could be employed to model the nonlinear systems by dividing the input space into a set of regions, each of which is approximated with a single ANN model. In general, a MNN is constructed from two types of network, namely expert networks and a gating network. Expert networks may be of a variety of different types of neural networks. Each network is designed for a particular task. A gating network receives the input vector and produces as many outputs as there are expert networks. These outputs must be nonnegative and sum to unity, representing the weights of the output of each expert network. A weighted sum of the outputs of the experts forms the MNN output. During training, the weights of the expert and gating networks are adjusted simultaneously using the backpropagation algorithm.

There are many local-nonlinear type DAC models similar to MNN but with different names, such as hybrid ANNs, integrated ANNs, threshold (or domain-dependent, range-dependent) ANNs, committee machine and so on. Some of them could be considered as special cases of MNN (e.g. threshold ANN), and some others have different ways of combining the separate expert neural networks. For example, instead of using a gating network to mediate the competition of expert networks, some hybrid neural networks use a fuzzy logic model to link

individual expert networks into an integrated modeling system, or use the cluster analysis technique in which the input space is first divided into several clusters, and then a separate expert network is fitted to each cluster. There are already some examples of applications of modular or hybrid ANN models in the field of hydrological modeling. Zhang and Govindaraju (2000) examined the performance of modular networks in predicting monthly discharges based on the Bayesian concept. See and Openshaw (1999) developed a fuzzy-logic based hybrid model to forecast river level, in which the forecasting data set is split into subsets before training with a series of neural networks. Abrahart and See (2000) presented a hybrid network solution based the clustering of the hydrological records with a self-organizing map (SOM) neural network. Hu et al. (2001) developed range-dependent hybrid neural networks (RDNN), which are virtually threshold ANNs, to forecast annual and daily streamflows. Pal et al. (2003) proposed a hybrid ANN model that combines the self-organizing feature map (SOFM) and the MLP network for temperature prediction, where the SOFM serves to partition the training data.

In addition, perhaps the most thorough DAC algorithm is the nearest neighbor method (NNM), in which a time series is reconstructed in a multi-dimensional state-space and then local approximation models (parametric or non-parametric) are fitted to the nearest neighbors in the space. Parametric NNM could be linear or nonlinear depending on what type of local parametric models are used. NNM has been applied to streamflow forecasting by many researchers (e.g. Yakowitz and Karlsson, 1987; Bordignon and Lisi, 2000; Sivakumar et al., 2001).

In this study, three types of hybrid ANN models, namely, the threshold ANN (TANN) and the cluster-based ANN (CANN), and periodic ANN (PANN), are used to forecast daily streamflows, and the model performance is compared with normal ANN models that are fitted to the data without any grouping. The organization of this paper is as follows. Section 2 gives a brief description of the study area and data used, followed by the introduction to the concept of state space reconstruction, a method used here to determine the number of inputs of ANN models for univariate time series. A brief review of ANN modeling methods is presented in Section 3, then normal ANNs are applied to forecasting 1- to 10-day ahead daily streamflows. In Section 4, different ANN hybridization approaches are presented, and forecasts are made with these hybrid ANNs. Some discussions and conclusions of the study are given in Sections 5 and 6, respectively.

2. State-space reconstruction of the daily streamflow series

2.1. Study area and data used

The case study area is the headwater region of the Yellow River, located in the northeastern Tibet Plateau in China. In this area, the discharge gauging station Tangnaihai has a 133,650 km^2 drainage area, including a permanently snow-covered area of 192 km^2. The length of main channel of this watershed is over 1500 km. Most of the watershed is 3000–4000 m above sea level. Snowmelt water composes about 5% of total runoff. Most rain falls in summer. Because the watershed is partly permanently snow-covered and sparsely populated, having no large-scale hydraulic works, it is fairly pristine. The average annual runoff volume (1956–2000) at Tangnaihai gauging station is 20.4 billion cubic meters, about 35% of the whole Yellow River Basin. Therefore, it is the main runoff generation area of the Yellow River basin. Discharges at Tangnaihai has been recorded since January 1, 1956. In this study, daily average discharges from January 1, 1956 to December 31, 2000 are used. The mean values as well as standard deviations of the streamflow series in each day over the year are plotted in Fig. 1.

2.2. State-space reconstruction

To describe the temporal evolution of a dynamical system in a multi-dimensional state space with a scalar time series, one needs to employ some techniques to unfold the multi-dimensional structure using the available data. Packard et al. (1980) and Takens (1981) proposed the time delay coordinate method to reconstruct the state space from a scalar time series. According to the method, the state vector X_i in a new space, the embedding space, is formed

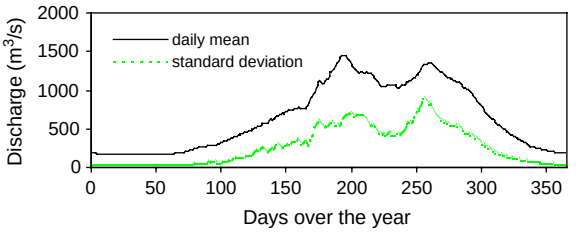

Fig. 1. Variation in daily means and standard deviations of the streamflows at Tangnaihai.

from time delayed values of the scalar measurements $\{x_i\}$ as $X_i=[x_i,x_{i-\tau},\ldots,x_{i-(m-1)\tau}]$, where x_i is the observed value of the time series at time i, m is the embedding dimension, and τ is the delay time.

There are lots of discussions about the parameter options for state-space reconstruction (e.g. Kantz and Schreiber, 2004, p 30–47). An optimum value of τ should give the best separation of neighboring trajectories within the minimum embedding phase-space. Popular methods for determining the optimum value of τ include the autocorrelation function method and the mutual information method. Due to the strong annual seasonality, according to the autocorrelation function method and the mutual information method, the τ value would be about 1/4 of the annual period of the daily streamflow process, namely about 91 days (Wang et al., 2005b). However, for the purpose of forecasting, although the reconstructed phase-space is redundant with a value of $\tau=1$, we tolerate having some information redundancy in preference to losing any useful information.

A method to determine the minimal sufficient embedding dimension m was proposed by Kennel et al. (1992), called the 'false nearest neighbor' method. The minimal embedding dimension *m* for a given time series means that *m* is the minimal value that is sufficient to insure that in a *m*-dimensional embedded space the reconstructed attractor is a one-to-one image of the attractor in the original phase space. If the series is embedded in a m'-dimensional space with $m'<m$, then some points that are actually far from each other will appear as neighbors because the geometric structure of the attractor has been projected down to a smaller space. These points are called false neighbors. For false neighbors, their trajectories will move far away as the embedding dimension increases. Suppose the point $X_i=[x_{i-p+1},\ldots,x_i]$ has a neighbor $X_j=[x_{j-p+1},\ldots,x_j]$ in a p-dimensional space. Calculate the distance $\|Xi-Xj\|$ and compute

$$R_i=\frac{|x_{i+1}-x_{j+1}|}{\|X_i-X_j\|}. \quad (1)$$

If R_i exceeds a given threshold R_T (say, 10 or 15), the point X_i is marked as having a false nearest neighbor. We say the embedding dimension p is high enough if the fraction of points that have false nearest neighbors is actually zero, or sufficiently small, say, smaller than a criterion R_f.

Setting the false neighbor threshold $R_T=10$, we calculate the fraction of false nearest neighbors as a function of the embedding dimension for daily streamflow series at Tangnaihai, shown in Fig. 2. If we set the fraction criterion $R_f=0.01$, then the embedding dimension is five, which means that the state of streamflow process is determined by five lagged observed values. Correspondingly, when fitting an ANN model to the series, we use five lagged values as input to forecast the one-step-ahead value.

3. Fitting normal ANN models to daily streamflow series

3.1. ANN structure

When building a neural network model, a number of decisions must be made, including the neural network type, network structure, methods of pre- and post-processing of input/output data, training algorithm and training stop criteria.

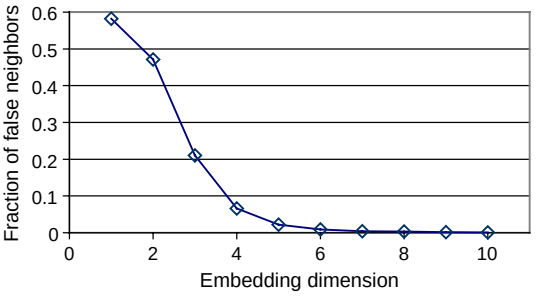

Fig. 2. The fraction of false nearest neighbors as a function of the embedding dimension for daily streamflow series at Tangnaihai.

The feed-forward multi-layer perceptron (MLP) ANN is the most widely used type of ANN in hydrological modeling, and is also adopted in this study. Network architecture mainly denotes the number of input/output variables, the number of hidden layers and the number of neurons in each hidden layer. It determines the number of connection weights and the way information flows through the network. The major concern of the designer of an ANN structure is to determine the appropriate number of hidden layers and the number of neurons in each layer. There is no systematic way to establish a suitable architecture, and the selection of the appropriate number of neurons is basically problem specific. Hornik et al. (1989) proved that a single hidden layer network containing a sufficiently large number of neurons can be used to approximate any measurable functional relationship between the input data and the output variable to any desired accuracy. De Villars and Barnard (1993) showed that an ANN comprised of two hidden layers tends to be less robust and converges with less accuracy than its single hidden layer counterpart. Furthermore, some studies indicate that the benefits of using a second hidden layer are marginal to the rainfall-runoff modeling problem (e.g. Minns and Hall, 1996; Abrahart and See, 2000). Taking recognizance of the above studies, a single hidden layer is used in this study.

There are some algorithms, including pruning and constructive algorithms, to determine an 'optimum' number of neurons in the hidden layer(s) during training. However, a trial and error procedure using different number of neurons is still the preferred choice of most users (e.g. Shamseldin, 1997; Zealand et al., 1999; Abrahart and See, 2000) and is the method used also in this research.

The number of ANN input/output variables is comparatively easy to determine. According to the result of state space reconstruction discussed in Section 2.2, we use the known discharges of Q_{t-4}, $Q_{t-3}, \ldots, Q_t$ of day $t-4$ to day t, respectively, as the inputs. The output of the ANN is the predicted discharge Q_{t+1} of day $t+1$. Before fitting an ANN model, the data should be preprocessed. There are basically two reasons for preprocessing. Firstly, preprocessing can ensure that all variables receive equal attention during the training process. Otherwise, input variables measured on different scales will dominate training to a greater or lesser extent because initial weights within a network are randomized to the same finite range (Dawson and Wilby, 2001). Secondly, preprocessing is important for the efficiency of training algorithms. For example, the gradient descent algorithm (error backpropagation) used to train the MLP is particularly sensitive to the scale of data used. Due to the nature of this algorithm, large values slow training because the gradient of the sigmoid function at extreme values approximates zero (Dawson and Wilby, 2001). In general, there are fundamentally two types of preprocessing methods. The first is to rescale the data to a small interval (referred to as rescaling), such as $[-1, 1]$ or $[0, 1]$, depending on the transfer (activation) function used in the neurons, because some transfer functions are bounded (e.g. logistic and hyperbolic tangent function). Another is to standardize the data by subtracting the mean and dividing by the standard deviation to make the data have a mean of 0 and variance 1 (referred to as standardization).

3.2. ANN training

The ANN training is fundamentally a problem of nonlinear optimization, which minimizes the error between the network output and the target output by repeatedly changing the values of ANN's connection weights according to a predetermined algorithm. Error backpropagation (e.g. Rumelhart et al., 1986) is by far the most widely used algorithm for optimizing feedforward ANNs. In this study, training was implemented using the *traingdm* function in Matlab Neural Network Toolbox, which uses the error backpropagation algorithm by updating the weight and bias values according to gradient descent with momentum. Networks trained with the backpropagation algorithm are sensitive to initial conditions and susceptible to local minima in the error surface. On the other hand, there may be many parameter sets within a model structure that are equally acceptable as simulators of a dynamical process of interest. Consequently, instead of attempting to find a best single ANN model, we may make predictions based on an ensemble of neural networks trained for the same task (see e.g. Sharkey, 1996). In this present study, the idea of

ensemble prediction is adopted, and the simple average ensemble method is used. For each ANN model, we train it ten times so as to get 10 networks, and choose five best ones according to their training performances. With the five selected networks, we get five outputs. Then we take a simple average of the five outputs to be the final output.

A very important issue about training an ANN is how to decide when to stop the training because ANNs are prone to either underfitting or overfitting if their trainings are not stopped appropriately. The cross-validated early stopping technique is most commonly used (e.g. Braddock et al., 1998) to avoid stopping too late (i.e. resulting in overfitting). However, Amari et al. (1997) showed that overfitting would not occur if the ratio of the number of training data sets to the number of the weights in the network exceeds 30. In such cases, training can be stopped when the training error has reached a sufficiently small value or when changes in the training error remain small. In fact, some experiments with simulated time series show that (Wang et al., 2005a), even when the ratio is as high as 50, very slight overfitting still can be observed in some cases. But cross-validation generally does not help to improve, in many cases even degrades, the generality of ANNs when the ratio is larger than 20. In our study, the ratio of the number of input data sets and the number of network weights is far larger than 50, therefore, the use of cross-validation data was not considered necessary. The training is stopped after 3000 epochs.

3.3. Model performance measures

There is an extensive literature on model forecasting evaluation indices (e.g. Nash and Sutcliffe, 1970; Garrick et al., 1978; Wilmott et al., 1985; Legates et al., 1999; Kneale et al., 2001). Despite its crudeness and identified weaknesses (Kachroo and Natale, 1992), the coefficient of efficiency (CE) introduced by Nash and Sutcliffe (1970) is still one of the most widely used criteria for the assessment of model performance. The CE, which provides a measure of the ability of a model to predict values that are different from the mean, has the form

$$\mathrm{CE} = 1 - \frac{\sum_{i=1}^{n} (Q_i - \hat{Q}_i)^2}{\sum_{i=1}^{n} (Q_i - \bar{Q})^2}, \tag{2}$$

where Q_i is the observed value, $\hat{Q}_i$ is the predicted value, $\bar{Q}$ is the mean value of the observed data. A CE of 0.9 and above is generally considered very satisfactory, 0.8–0.9 represents a fairly good model, and below 0.8 is considered unsatisfactory (Shamseldin, 1997). Because CE is a global measure of comparing the predicted value with the overall mean value, it is not efficient enough to evaluate the predictions for those series whose mean values change with seasons, which is almost always the case for hydrological processes. Therefore, a seasonally-adjusted coefficient of efficiency (SACE) (Wang et al., 2004b), which is originally named as adjusted coefficient of efficiency (ACE), is also used here for evaluating the model performance. SACE is calculated by

$$\mathrm{SACE} = 1 - \frac{\sum_{i=1}^{n} (Q_i - \hat{Q}_i)^2}{\sum_{i=1}^{n} (Q_i - \bar{Q}_m)^2}, \tag{3}$$

where m = i mod S (mod is the operator calculating the remainder), ranging from 0 to S − 1; and S is the total number of 'season' (Note that, a 'season' here is not a real season. It may be a month or a day depending on the timescale of the time series. For daily streamflow series, one season is one day over the year.); $\bar{Q}_m$ is the mean value of season m.

Besides the above two measures, the mean squared error (MSE) or equivalently root mean squared error (RMSE), is another popular measure because it is very sensitive to even small errors, which is good for comparing small differences of model performances. RMSE is calculated by

$$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (Q_i - \hat{Q}_i)^2}. \tag{4}$$

3.4. Forecasting with normal ANN models

For comparison with the hybrid ANN modeling approaches, the normal MLP–ANN models are first

fitted to the daily flow series without any data grouping or clustering procedure. The models are fitted using the daily discharge data at Tangnaihai from 1956 to 1995, and 1- to 10-day ahead forecasts are made for year 1996–2000. The purpose of making 1- to 10-day forecasts is to explore the differences of model behavior when making forecasts for short to long lead times. By splitting the total available data into a training data set of 40 years and a validation (i.e. forecasting) data set of 5 years, we have enough data to avoid the problem of overfitting when training the ANN models and have enough data to make the performance evaluation results on the basis of validation data statistically meaningful. To predict daily discharges up to 10 days of lead time, a simple recursive algorithm will be used to obtain forecasts for successive lead times. An ANN model will first predict Q_{t+1}, and the predicted Q_{t+1} will then be used to update the input variables into the ANN so as to predict Q_{t+2}. This procedure is thus repeated until the forecasted values ranging from Q_{t+1} to Q_{t+10} are made.

Matlab Neural Network Toolbox is used to construct different one-hidden-layer MLP networks with a range of 3–10 hidden neurons in the hidden layer. The model forecasting results with different architectures indicate that 3-node networks generally perform best. Therefore, the chosen configuration for MLP–ANNs is 5–3–1, namely, five inputs, one hidden layer with three hidden neurons and one output. This MLP structure is adopted throughout all the ANN models in this study.

To compare the influence of different preprocessing procedures on model performance, six different preprocessing procedures are applied:

- Standardizing the raw data series;
- Rescaling the raw data series;
- Standardizing the log-transformed data series;
- Rescaling the log-transformed data series;
- Deseasonalizing the log-transformed data series;
- Deseasonalizing the log-transformed data series and then rescaling the deseasonalized series.

The deseasonalization is a special type of standardization, which is commonly used when fitting time series model to streamflow series (e.g. Hipel and Mcleod, 1994). But instead of using the overall mean value and the overall deviation to make the standardization, the deseasonalization is accomplished by subtracting the seasonal (e.g. daily, monthly) means and dividing by the seasonal standard deviations. Because the activation (or transfer) function in the hidden neurons used in this study is the tan-sigmoid function, which is mathematically equivalent to the hyperbolic tangent function, in the form of $f(x) = 2/(1 + e^{-2x}) - 1$, and the output value of the tan-sigmoid function is bounded between -1 and 1, so when rescaling the data, we rescale both the input and output data to $[-1, 1]$ with

$$x = \frac{2(x - x_{\min})}{(x_{\max} - x_{\min})} - 1 \tag{5}$$

where $x_{\min}$ and $x_{\max}$ are the minimum and maximum values in the data set, respectively.

In total, six MLP models are fitted to the daily streamflow data, each of which has a different preprocessing procedure. When making forecasts, post-processing is needed to inversely transform the outputs to their original scale. Table 1 lists model performance evaluation results of the 1- to 10-day ahead forecasts with the MLP models. It is shown that: (i) MLP models fitted to standardized data perform better than those fitted to rescaled data (note that deseasonalization without rescaling is a special type of standardization); (ii) in the case of the longer lead time forecasts, it is better to do a log-transformation before standardizing of the data; and (iii) Overall, the MLP model fitted to the deseasonalized data, without rescaling, performs best, and the advantage becomes more evident as the lead time increases.

4. Forecasting with hybrid ANN models

In this study, three hybrid ANN models, namely, the periodic ANN model (PANN), the threshold ANN (TANN), and the cluster-based hybrid ANN (CANN), are built. The three hybrid ANN models are briefly described as follows.

4.1. Threshold ANN (TANN)

The threshold ANN (TANN), analogous to the threshold regression model, divides the streamflow series into several regimes according to some

Table 1
Forecasting performances with normal MLP–ANN models for daily flows at Tangnaihai

Lead time (days)		1	2	3	4	5	6	7	8	9	10
CE	Raw_std	0.989	0.966	0.938	0.910	0.881	0.849	0.814	0.776	0.739	0.699
	Raw_rescale	0.981	0.955	0.918	0.887	0.851	0.815	0.774	0.735	0.694	0.655
	Ln_std	0.988	0.965	0.942	0.919	0.895	0.870	0.843	0.815	0.788	0.760
	Ln_rescale	0.963	0.942	0.919	0.894	0.864	0.834	0.806	0.778	0.750	0.723
	Ln_DS	0.989	0.967	0.943	0.92	0.897	0.873	0.848	0.823	0.799	0.775
	Ln_DS_rescale	0.985	0.964	0.939	0.916	0.891	0.866	0.838	0.811	0.785	0.758
SACE	raw_std	0.980	0.936	0.885	0.834	0.779	0.721	0.654	0.585	0.516	0.442
	raw_rescale	0.965	0.916	0.848	0.791	0.724	0.657	0.582	0.508	0.434	0.36
	Ln_std	0.978	0.935	0.892	0.849	0.806	0.759	0.708	0.658	0.607	0.555
	Ln_rescale	0.932	0.893	0.849	0.803	0.748	0.692	0.641	0.588	0.536	0.487
	Ln_DS	0.980	0.938	0.894	0.852	0.810	0.765	0.719	0.672	0.628	0.582
	Ln_DS_rescale	0.972	0.933	0.887	0.845	0.799	0.751	0.700	0.65	0.601	0.551

Note: 'Raw' denotes the raw data; 'ln' denotes log-transformation; 'std' denotes the standardization by subtracting the mean and dividing by the standard deviation; 'DS' denotes the deseasonalization by subtracting seasonal means and dividing by seasonal standard deviations; 'rescale' denotes rescaling to −1 to 1.

threshold values, and then builds one ANN model for each regime.

To fit a threshold model, whether it is a threshold linear model or a threshold nonlinear model, the main concern is to determine the threshold value. Tong (1983) suggested that the location of modes and antimodes of the univariate histogram for x_t and the bivariate histogram for (x_t, x_{t-i}) (i = 1, 2,…,p, say) may assist in the identification of the threshold parameters. Histogram is the simplest estimator of the probability density function (pdf). An improved histogram estimate is the kernel density estimate, which can overcome the histogram's sensitivity to choice of data origin and bin (i.e. class interval) width (Silverman, 1986). The kernel pdf is estimated as

$$f(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \quad (6)$$

where n is the sample size, $x_i = 1,\ldots,n$ are the data, h is the bandwidth, and K(u) is the kernel function, in this case the Gaussian kernel function.

The kernel density estimate of pdf is calculated for the daily streamflow of the Yellow River at Tangnaihai. Fig. 3 shows that the streamflow series has a bimodal pdf, whose two modes may indicate two regimes of the flow process dynamics, and the antimode may correspond to a point of separation of the two regimes. The low-flow regime is around 195 m^3/s ($\approx \exp(5.273)$), and the high-flow regime around 671 m^3/s($\approx \exp(6.508)$), the two regimes being separated at the antimode, near 381 m^3/s($\approx \exp(5.942)$).

Therefore, we divide the streamflow states in the reconstructed state-space (described in Section 2.2) into two regimes, namely, one regime composed of the streamflow states whose average discharges are larger than 381 m^3/s, and another one composed of the streamflow states less than 381 m^3/s. Then one MLP network is fitted to the streamflows in each regime.

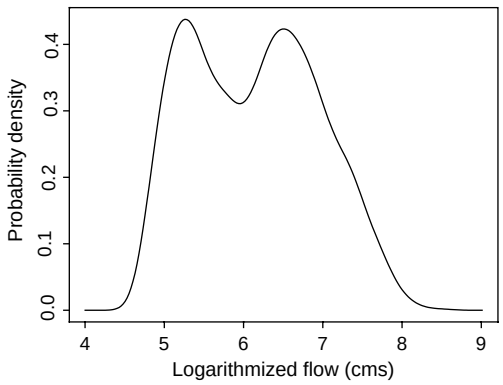

Fig. 3. Probability density estimate of log-transformed daily streamflows at Tangnaihai.

4.2. Cluster-based ANN (CANN)

For the cluster-based ANN (CANN) model, we divide the streamflow state vectors in the reconstructed state space into several clusters based on cluster analysis techniques, and then build one ANN model for each cluster. The fuzzy C-means (FCM) clustering technique is applied in the present study to do the clustering, so that we can cluster the streamflow state vectors both softly and crisply.

FCM clustering is proposed by Bezdek (1981) as an improvement over the hard k-means clustering algorithm. The FCM method partitions a set of n vector x_j, $j=1,\ldots,n$, into c fuzzy clusters, and each data point belongs to a cluster to a degree specified by a membership grade u_{ij} between 0 and 1. We define a matrix U consisting of the elements u_{ij}, and assume that the summation of degrees of belonging for a data point is equal to 1, i.e. $\sum_{i=1}^{c} u_{ij} = 1, \quad \forall j = 1, \ldots, n$. The goal of the FCM algorithm is to find c cluster centers such that the cost function of dissimilarity (or distance) measure is minimized. The cost function is defined by

$$J(U, v_{1,}\ldots, v_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m d_{ij}^2 \tag{7}$$

where v_i is the cluster center of the fuzzy group *i*; $d_{ij} = \|v_i - x_j\|$ is the Euclidean distance between the *i*th cluster center and the *j*th data point; and $m \geq 1$ is a weighting exponent, taken as 2 here. The necessary conditions for Eq. (7) to reach its minimum are:

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{8}$$

and

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}}\right)^{2/(m-1)}\right]^{-1} \tag{9}$$

The fuzzy C-means algorithm is an iterative procedure that satisfies the preceding two necessary conditions as follows.

(1) Initialize the membership matrix U with random values between 0 and 1.
(2) Calculate c fuzzy cluster centers vi, $i=1,\ldots,c$, using Eq. (8).
(3) Compute the cost function according to Eq. (7). Stop if either it is below a tolerance value or its improvement over the previous iteration is below a certain threshold.
(4) Compute a new U using Eq. (9). Return to step (2) .

The streamflow states in the reconstructed state space of the Yellow River at Tangnaihai are grouped into three clusters using the FCM clustering method. The FCM clustering result of the streamflow states in a typical year is shown in Fig. 4. Compared with Fig. 1, it is seen that, when grouping the streamflow states into three clusters, the three groups basically represent three different daily flow regimes, i.e. low flow, medium flow and high flow. However, for some years, three groups may be much more fragmented.

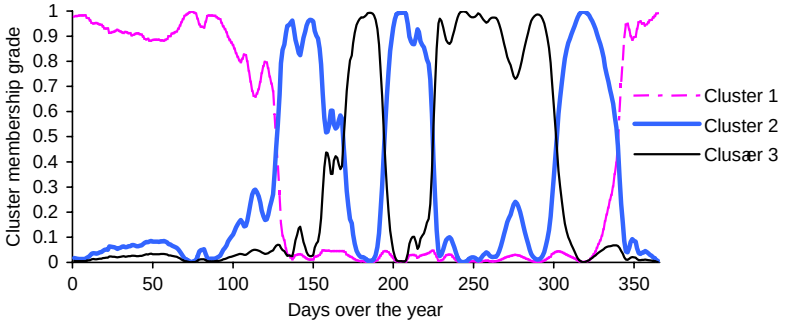

Fig. 4. Membership grades of the daily streamflows in a typical year with the FCM of three clusters.

After the streamflow state vectors in the reconstructed state space are grouped into three clusters, we then fit one ANN to the streamflow states in each cluster, three ANN models in total. When making a forecast from a current streamflow state, the final output of the overall CANN model is a weighted average of the outputs of the three ANN models. The weights are determined according to the distance between the current streamflow state and each cluster center.

4.3. Periodic ANN (PANN) model

The periodic ANN (PANN) model essentially is a group of ANN models, each of which is fitted to the streamflows that occurred in a separate 'season' (notice that, season here does not mean a real season. It may be a group of neighbouring days or months over the year). The idea of fitting PANN model to daily streamflows is adopted from that of fitting the periodic autoregressive (PAR) model (Wang et al., 2004a). When we build a PAR model for monthly flows, one AR model may be built for each month over the year. However, it is unfeasible to fit one AR model for each day of the year when building a PAR model for daily flows. In order to make the periodic model 'parsimonious', an approach was proposed by Wang et al. (2004a) whereby a periodic AR model is fitted to daily streamflows based on partitioning of days over the year with clustering techniques. This approach is followed here to build the periodic ANN model.

The PANN differs from the CANN in that, in the construction of the CANN we clusters the discharges in the reconstructed phase-space, whereas in the construction of the PANN we cluster the days over the year according to the characters of the discharges of each day. When partitioning the days over the year with the clustering analysis method, the raw average daily discharge data and the autocorrelation values at different lag times (1–10 days, as shown in Fig. 5) are used. The daily discharge data and the autocorrelation coefficients are organized as a matrix of the size $(N+10)\times 365$, where N is the number of years (in this case, $N=45$) and '10' represents the autocorrelation values at 10 lags. To eliminate the influence of big differences among data values on cluster analysis result, the log-transformation is first applied to the daily discharges before making the cluster analysis.

Then the 365 days over the year are partitioned with the fuzzy c-means clustering described in Section 4.2. The cluster result is shown in Fig. 6. Comparing Figs. 1 and 6, we see that if we just follow the clustering result to partition the days over a year into five groups, the dynamics of streamflow is not well captured because cluster two and three in Fig. 6 mix the streamflow rising limb and falling limb shown in Fig. 1. Therefore, according to the FCM clustering result, and considering the dynamics of the streamflow process, we partition the 365 days over the year into seven hard segments, as listed in Table 2.

The days over the year can also be softly partitioned, such that each day could belong to several partitions. The essence of soft partitioning is the determination of the membership grade (or membership function) of each partition, which is one of the most crucial issues in the foundation of fuzzy reasoning. In this study, following the pattern of the FCM clustering result, the membership grade

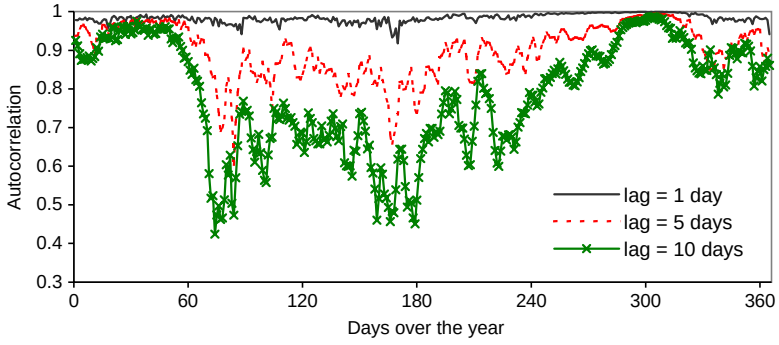

Fig. 5. Daily autocorrelations at different lag days for daily flow series at Tangnaihai.

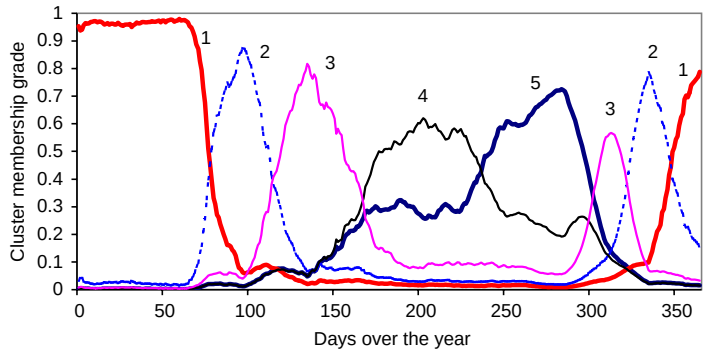

Fig. 6. Membership grades of the days over the year for the daily streamflows at Tangnaihai with the FCM of five clusters.

Table 2
Hard partitioning of the days over the year for the daily streamflows at Tangnaihai

Partition	1	2	3	4	5	6	7
Day span	1–77, 349–365/366	78–114	115–167	168–237	238–302	303–322	323–348

is formed intuitively for the days over the year, as shown in Fig. 7.

Based on the partitioning results, the PANN model is built, whereby one MLP model is fitted to each (hard) seasonal partition. When forecasting, different MLP models are used depending on what seasonal partition the date for which the forecasted is made lies in. The fitted PANN model can be applied to forecasting in two ways: based on hard partitioning (referred to as *hard PANN*) and based on soft partitioning (referred to as *soft PANN*). When soft partitioning is applied, one day could belong to several season partitions. Correspondingly, the final output would be a weighted average of the outputs of the several ANN models fitted for these seasonal partitions. The weight is equal to the membership grade obtained with the FCM clustering results.

4.4. Performances of hybrid ANN models

The components of all the hybrid ANN models have the same 5–3–1 MLP structure, as identified for normal MLP–ANN models considered in Section 3.4. According to the comparison among different data preprocessing methods described in Section 3, deseasonalization is the best choice of preprocessing method. However, because TANN and CANN divide up the streamflow states into different regimes according to their values, it is more convenient to standardize the data than to deseasonalize the data

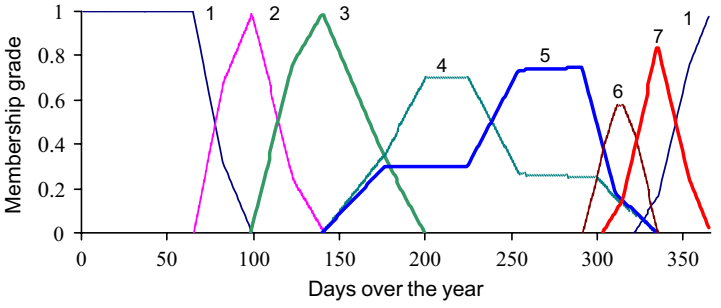

Fig. 7. Soft partitioning of the days over the year for the daily streamflows at Tangnaihai according to the FCM of five clusters.

Table 3
Forecasting performances with hybrid ANN models for daily flows at Tangnaihai

Lead time (days)		1	2	3	4	5	6	7	8	9	10
CE	CANN	0.984	0.942	0.898	0.856	0.815	0.774	0.735	0.698	0.665	0.633
	TANN	0.989	0.967	0.941	0.916	0.892	0.867	0.839	0.811	0.784	0.755
	Hard PANN	0.989	0.966	0.942	0.919	0.895	0.870	0.844	0.818	0.793	0.769
	Soft PANN	0.990	0.967	0.943	0.920	0.897	0.873	0.848	0.823	0.799	0.776
SACE	CANN	0.969	0.893	0.810	0.733	0.657	0.581	0.508	0.441	0.378	0.320
	TANN	0.980	0.938	0.891	0.845	0.799	0.753	0.702	0.650	0.599	0.546
	Hard PANN	0.980	0.937	0.892	0.849	0.805	0.759	0.711	0.663	0.617	0.571
	Soft PANN	0.981	0.940	0.895	0.852	0.808	0.765	0.718	0.671	0.628	0.584

for building TANN and CANN models. Therefore, the input/output data are log-transformed and then standardized before fitting the TANN model and the CANN model, and log-transformed and then deseasonalized before fitting the PANN model.

Table 3 lists performance evaluation results of the 1- to 10-day ahead forecasts with these four ANN models (including hard PANN and soft PANN). It is shown that the PANN model performs best among three types of hybrid ANN models, and the soft PANN performs better than the hard PANN. The scatter plots of one-day ahead, 5-day ahead and 10-day ahead forecasted versus observed daily discharges for year 1996–2000 with the soft- partition

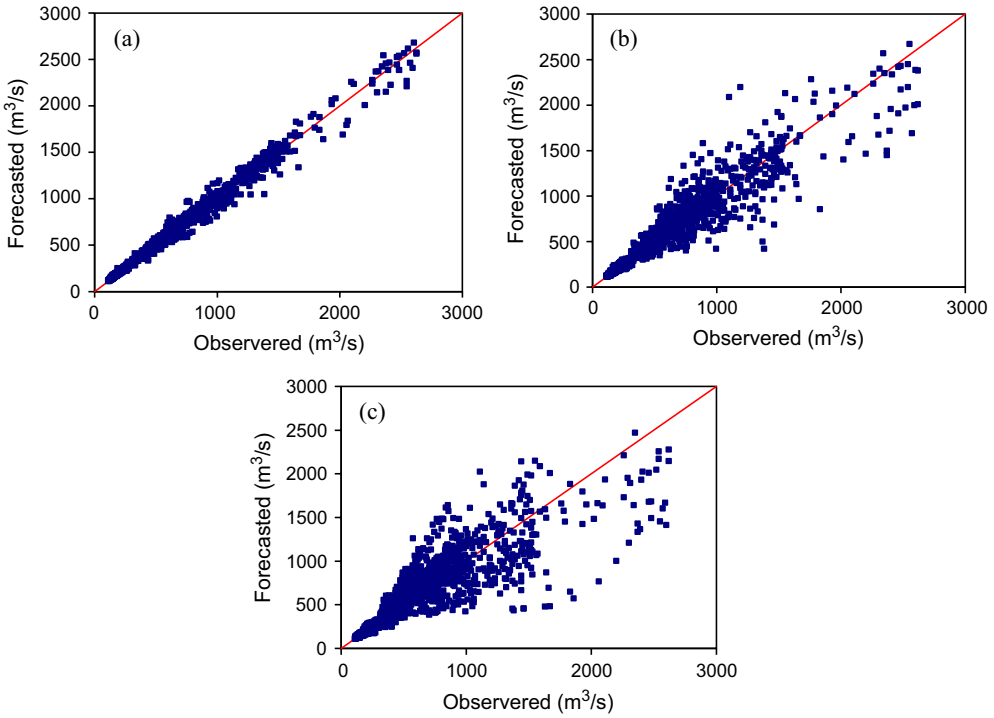

Fig. 8. Scatter plots of (a) 1-day; (b) 5-day and (c) 1-=day ahead discharge forecasts for year 1996–2000 with the PANN based on soft seasonal partitions.

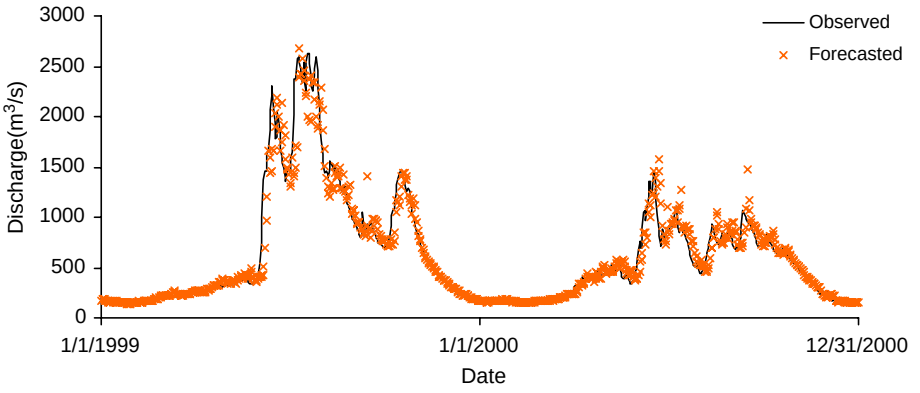

Fig. 9. Five-day ahead discharge forecasts with the PANN based on soft seasonal partitions.

based PANN are shown in Fig. 8. The five-day ahead forecasted hydrograph versus observed hydrograph of daily discharges in year 1999 and 2000 is shown in Fig. 9, and the 10-day ahead forecasted hydrograph shown in Fig. 10. Because the hydrograph of one-day ahead forecast is not very informative due to high forecast accuracy, it is not shown here to save the space.

Because the model performance difference measured with CE and SACE is not very informative (comparing Tables 1 and 3), in order to give a more clear comparison between the PANN models and the normal MLP–ANN model that is fitted to the deseansonalized data without rescaling or any grouping procedure, the RMSE measure is used to evaluate the model performance. The results are listed in Table 4, where it is seen that for the shorter lead times, the soft PANN performs better than the normal MLP model, but the advantage vanishes as the lead time increases ($\geq$4 days).

5. Some discussions

5.1. About the preprocessing of input/output data

It is recognized that data preprocessing can have a significant effect on model performance (e.g. Maier and Dandy, 2000). It is commonly considered that, because the outputs of some transfer functions are bounded, the outputs of an MLP ANN must be in the interval [0,1] or [−1,1] depending on the transfer function used in the neurons. Some authors suggest using even smaller intervals for streamflow modelling, such as [0.1, 0.85] (Shamseldin, 1997), [0.1, 0.9] (e.g. Hsu et al., 1995; Abrahart and See, 2000) and [−0.9, 0.9] (e.g. Braddock

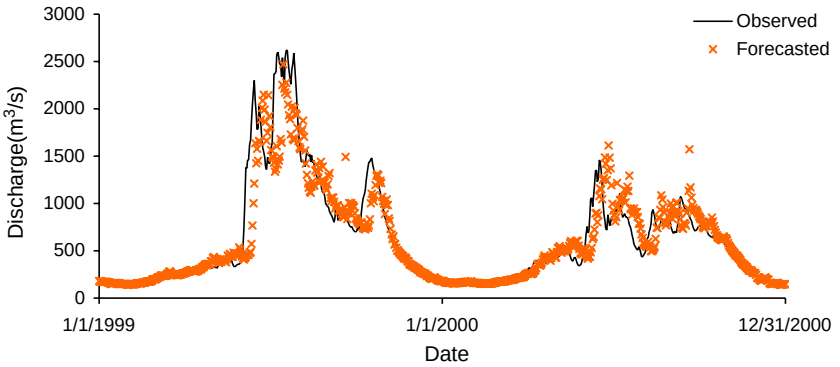

Fig. 10. Ten-day ahead discharge forecasts with the PANN based on soft seasonal partitions.

Table 4
Forecasting performances of the normal MLP–ANN model and PANN models

Lead time (days)		1	2	3	4	5	6	7	8	9	10
RMSE	MLP	45.9	80.3	105.0	123.9	140.6	156.2	171.0	184.5	196.6	208.3
	Hard PANN	45.5	80.6	105.7	125.3	142.4	158.1	173.3	187.2	199.5	211.0
	Soft PANN	44.8	79.2	104.4	124.1	141.1	156.3	171.1	184.8	196.7	207.9

Note: The normal MLP–ANN model is fitted to the deseasonalized and non-rescaled series.

et al., 1998), so that extreme (high and low) flow events occurring outside the range of the calibration data may be accommodated. For model building convenience, it is common that both input and output data are rescaled before fitting ANN models (e.g. Hsu et al., 1995; Rajurkar et al., 2004). However, the advantage of rescaling the data into a small interval is not supported by this study. In the case of forecasting for the daily flows at Tangnaihai, not only is the general performance of the MLP-ANN with standardization preprocessing better than the MLP-ANN with rescaling preprocessing, but the performance for low flow and high flow periods is also better.

There are two explanations for this result. On the one hand, suppose we only consider the relationship between one input and one neuron with a hyperbolic tangent transfer function *tansig*(x). To rescale the input data to $[-1, 1]$ would limit the output range of the *tansig*(x) function approximately to $[-0.7616, 0.7616]$. To rescale the input range to $[-0.9, 0.9]$ would further shrink the output range approximately to $[-0.7163, 0.7163]$. Both 0.7616 and 0.7163 are still far away from the extreme limits of the *tansig*(x) function, whereas such a small output data range will make the output less sensitive to the change of the weights between the hidden layer and output layer, and will therefore possibly make the training process more difficult. On the other hand, because the neurons in an ANN are combined linearly with many weights (as in a MLP model), any rescaling of the input vector can be effectively offset by changing the corresponding weights and biases. Therefore, to standardize the input/output data may be a better choice than to rescale them into a small interval (e.g., $[-1,1]$), especially when the data size is large enough to include possible data extremes.

A related subject is the choice of transfer function. The most commonly used transfer functions are the logistic sigmoid function and hyperbolic tangent function, the logistic sigmoid function ($\text{logsig}(x) = 1/(1+e^{-x})$) being much more frequently used than the hyperbolic tangent function in hydrologic forecasting (e.g. Hsu et al., 1995; Minns and Hall, 1996; Zealand et al., 1999; Abrahart and See, 2000). However, Kalman and Kwasny (1992) argue that the hyperbolic tangent transfer function should be used, and the empirical results obtained by Maier and Dandy (1998) also indicate that not only is the training with the hyperbolic tangent function faster than the training with the logistic sigmoid transfer function, but the predictions obtained using networks with the hyperbolic tangent are slightly better than those with the logistic sigmoid transfer functions. Kalman and Kwasny (1992) show mathematically that the hyperbolic tangent function possesses particular properties that make it appealing for use while training. However, no comparison of the logistic sigmoid function and hyperbolic tangent function is carried out in this present study. We only make a heuristic analysis about this problem here. Suppose we only consider the relationship between one input and one neuron with a logistic sigmoid function. When the input is in the interval $[-1,1]$, then the output is in the range [0.26894–0.73106] ($\log \text{sig}(-1) \approx 0.26894$, $\log \text{sig}(1) \approx 0.73106$), less than 1/3 of that of the hyperbolic tangent function. It is possible that, because the output of the logistic sigmoid function is constrained into a much smaller range than that of the hyperbolic tangent function, this results in the output of the ANN with logistic sigmoid functions being less sensitive to the change of connection weights, consequently making the

training of ANNs with the logistic sigmoid function more difficult than that of ANNs with the hyperbolic tangent function.

5.2. About the selection of the optimal ANN model

It is well known that an ANN is sensitive to the initial weights. When building an ANN model, every time we train it, we may get a different set of parameters. One way to increase the likelihood of obtaining near-optimum set of parameters is to train the ANN with different independent initial weights, then the model builders choose the best ANN model according to the training or validation performance among many competitive ANN models (e.g. Rajurkar et al., 2004). However, in practical applications, real validation data are future values, which virtually do not exist when we fit an ANN model, therefore we cannot choose the best model according to the validation performance. On the other hand, if we choose the best ANN model according to the training performances, the best training performance actually does not guarantee best validation performance, especially for multi-step forecasting, although the best training performance mostly indicate the best validation performance. That means, suppose we split all the available data into three parts, namely one training data set, and two validation data sets (A and B), a model which is the best for validation set A does not necessarily perform the best for validation set B. This is because the generality of an ANN model may be limited even though the size of training data may be sufficiently large, especially in the case of a natural watershed system which may have more or less underlying changes due to climate changes and human activities. On the other hand, as mentioned in Section 3.2, there may be many parameter sets within a model structure that are equally acceptable as simulators of a dynamical process of interest. Therefore, the attempt to choose a best ANN model is not sound. Instead, it is much better to make the ensemble forecast. One robust way of making ensemble forecast is simply taking the average of the forecasts of an ensemble of ANN models. To minimize the possibility that some of the ensemble members are poorly trained due to the effects of local minima in the error surface, we may train a number of networks (say, 10) first, then choose several best ones (say, 5) as ensemble members according to their training performances. This is the approach taken in this study.

6. Conclusions

For modeling complicated streamflow processes efficiently, lots of models have been proposed based on the principle of divide-and-conquer (DAC), which deals with a complex problem by dividing it into simple problems, which are solved independently. In this study, three types of hybrid ANN models based on the DAC principle, namely, the threshold-based ANN (TANN), the cluster-based ANN (CANN), and the periodic ANN (PANN), are used as univariate streamflow time series models to forecast 1- to 10-day ahead daily discharges of the upper Yellow River at Tangnaihai in China. For the purpose of comparing the forecasting efficiency, the normal multi-layer perceptron form of ANN (MLP–ANN) is selected as the baseline ANN model. The model evaluation results indicate that, among three types of hybrid ANN models, the PANN model performs best. Furthermore, the PANN based on soft seasonal partitions performs better than the PANN based on hard seasonal partitions. Compared with the normal MLP models, the PANN models perform generally better than normal MLP–ANN models, although the advantage vanishes as the lead time increases ($\geq$4 days) compared with the normal MLP–ANN model that is fitted to the deseansonalized data without rescaling or any grouping procedure.

In addition, the influence of different data preprocessing procedures, namely, standardization, log-transformation, rescaling, deseasonalization, and their combinations, on the ANN model performance is analyzed. It is shown that, for MLP networks with a tan-sigmoid transfer function, standardizing the data by subtracting the mean value and dividing by the standard deviation is better than rescaling the data to a small interval of $[-1, 1]$. Furthermore, for seasonal data such as streamflow series, deseasonization accomplished by subtracting the seasonal (e.g. daily or monthly) means and dividing by the seasonal standard deviations, is a better choice than other preprocessing methods.

One limitation of the current study is that only one data series (the daily streamflow series of the Yellow River at the Tangnaihai gauging station) is

considered for analysis. To establish generality of the conclusions, more streamflow processes should be analysed, and exogenous variables of the runoff generating processes such as rainfall processes should be employed. But be careful that, when exogenous variables, the preprocessing procedure of deseasonalization may not work because the seasonality of different variables would not be identical, therefore seasonal information may be useful in forecasting. In consequence, the standardization may work better in the case of having exogenous variables involved. Besides cluster-based and threshold-based approaches, there is a variety of other techniques available now to break the complicated streamflow forecasting problem down before solving the resulting sub-problems with different neural networks. It would be interesting to further compare the season-based PANN approach with other techniques, such as Bayesian-concept based modular ANN (e.g. Zhang and Govindaraju, 2000), fuzzy-logic based hybrid modelling (e.g. See and Openshaw, 1999) and SOM-cluster based hybrid modelling (Abrahart and See, 2000).

Acknowledgements

We are very grateful to two anonymous reviewers. Their comments, especially the detailed comments from one of them, help to improve the paper considerably.

References

Abrahart, R.J., See, L., 2000. Comparing neural network and autoregressive moving average techniques for the provision of continuous river flow forecasts in two contrasting catchments. Hydrol. Process. 14, 2157–2172.

Amari, S.-I., Murata, N., Muller, K.-R., Finke, M., Yang, H.H., 1997. Asymptotic statistical theory of overtraining and cross-validation. IEEE Trans. Neural Netw. 8 (5), 985–996.

Bartolini, P., Salas, J.D., Obeysekera, J.T.B., 1988. Multivariate periodic ARMA(1,1) processes. Water Resour. Res. 24, 1237–1246.

Becker, A., Kundzewicz, Z.W., 1987. Nonlinear floodrouting with multilinear models. Water Resour. Res. 23, 1043–1048.

Bezdek, J.C., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Bordignon, S., Lisi, F., 2000. Nonlinear analysis and prediction of river flow time series. Environmetrics 11, 463–477.

Braddock, R.D., Kremmer, M.L., Sanzogni, L., 1998. Feed-forward artificial neural network model for forecasting rainfall run-off. Environmetrics, 419–432.

Dawson, C.W., Wilby, R.L., 2001. Hydrological modelling using artificial neural networks. Prog. Phys. Geography 25 (1), 80–108.

De Villars, J., Barnard, E., 1993. Backpropagation neural nets with one and two hidden layers. IEEE Trans. Neural Netw. 4 (1), 136–141.

Garrick, M., Cunnane, C., Nash, J.E., 1978. A criterion of efficiency for rainfall-runoff models. J. Hydrol. 36, 375–381.

Hipel, K.W., Mcleod, A.I., 1994. Time Series Modelling of Water Resources and Environmental Systems. Elsvier, Amsterdam.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural Netw. 2 (5), 359–366.

Hsu, K., Gupta, H.V., Sorooshian, S., 1995. Artificial neural network modeling of the rainfall-runoff process. Water Resour. Res. 31, 2517–2530.

Hu, T.S., Lam, K.C., Ng, S.T., 2001. River flow time series prediction with a range-dependent neural network. Hydrol. Sci. J. 46 (5), 729–745.

Jacobs, R.A., Jordan, M.I., 1993. Learning piecewise control strategies in a modular neural network architecture. IEEE Trans. Syst. Man Cybern. 23 (2), 337–345.

Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E., 1991. Adaptive mixtures of local experts. Neural Comput. 3, 79–87.

Jones, R.H., Brelsford, W.M., 1967. Time series with periodic structure. Biometrika 54, 403–408.

Jordan, M.I., Jacobs, R.A., 1994. Hierarchical mixture of experts and the EM algorithm. Neural Comput. 6, 181–214.

Kachroo, R.K., Natale, L., 1992. Non-linear modelling of the rainfall-runoff relation. J. Hydrol. 135, 341–369.

Kalman, B.L., Kwasny, S.C., 1992. Why Tanh: choosing a sigmoidal function. In: Proceedings of the International Joint Conference on Neural Networks, Baltimore, vol. 4, pp. 578-581.

Kantz, H., Schreiber, T., 2004. Nonlinear Time Series Analysis. Cambridge University Press, Cambridge.

Kennel, M.B., Brown, R., Abarbanel, H.D., 1992. Determining embedding dimension for phase-space reconstruction using geometrical construction. Phys. Rev. A 45, 3403–3411.

Kneale, P.E., See, L., Smith, A., 2001. Towards defining evaluation measures for neural network forecasting models. Proceedings of the Sixth International Conference on GeoComputation, University of Queensland, Australia, available at http://www.geocomputation.org/2001.

Legates, D.R., Gregory, J., McCabe, G.J., 1999. Evaluating the use of "Goodness of Fit" measures in hydrologic and hydroclimatic model validation. Water Resour. Res. 35, 233–241.

Maier, H.R., Dandy, G.C., 1998. The effect of internal parameters and geometry on the performance of back-propagation neural networks: an empirical study. Environ. Model. Softw. 13 (2), 193–209.

Maier, H.R., Dandy, G.C., 2000. Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications. Environ. Model. Softw. 15, 101–123.

Minns, A.W., Hall, M.J., 1996. Artificial neural networks as rainfall-runoff models. Hydrol. Sci. J. 41, 399–417.

Nash, J.E., Sutcliffe, J.V., 1970. River flow forecasting through conceptual models, I. A discussion of principles. J. Hydrol. 10, 282–290.

Packard, N.H., Crutchfield, J.P., Farmer, J.D., Shaw, R.S., 1980. Geometry from a time series. Phys. Rev. Lett. 45 (9), 712–716.

Pagano, M., 1978. On periodic and multiple autoregressions. Ann. Stat. 6, 1310–1317.

Pal, N.R., Pal, S., Das, J., Majumdar, K., 2003. SOFM-MLP: a hybrid neural network for atmospheric temperature prediction. IEEE Trans. Geosci. Remote Sensing 41 (12), 2783–2791.

Rajurkar, M.P., Kothyari, U.C., Chaube, U.C., 2004. Modeling of the daily rainfall-runoff relationship with artificial neural network. J. Hydrol. 285, 96–113.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536.

Salas, J.D., Abdelmohsen, M.W., 1993. Initialization for generating single-site and multisite low-order periodic autoregressive and moving average processes. Water Resour. Res. 29 (6), 1771–1776.

Salas, J.D., Boes, D.C., Smith, R.A., 1982. Estimation of ARMA models with seasonal parameters. Water Resour. Res. 18, 1006–1010.

See, L., Openshaw, S., 1999. Applying soft computing approaches to river level forecasting. Hydrol. Sci. J. 44 (5), 763–778.

Shamseldin, A.Y., 1997. Application of a neural network technique to rainfall-runoff modelling. J. Hydrol. 199, 272–294.

Sharkey, A.J.C., 1996. On combining artificial neural nets. Connect. Sci. 8 (3–4), 299–313.

Silverman, B.W., 1986. Density Estimation. Chapman Hall, London.

Sivakumar, B., Berndtsson, R., Persson, M., 2001. Monthly runoff prediction using phase space reconstruction. Hydrol. Sci. J. 46 (3), 377–387.

Solomatine, D.P., Dulal, K.N., 2003. Model trees as an alternative to neural networks in rainfall-runoff modelling. Hydrol. Sci. J. 48 (3), 399–411.

Takens, F., 1981. Detecting strange attractors in turbulence. Lecture Notes in Mathematics 898, 366–381.

Todini, E., Wallis, J.R., 1977. Using CLS for daily or longer period rainfall-runoff modelling. In: Ciriani, T.A., Maione, U., Wallis, J.R. (Eds.), Mathematical Models for Surface Water Hydrology (Part 2: Flood Models). Wiley, pp. 149–168.

Tong, H., 1983. Threshold Models in Non-linear Time Series Analysis. Springer-Verlag, New York.

Tong, H., Lim, K.S., 1980. Threshold autoregression, limit cycles and cyclical data. J. R. Stat. Soc. B, Methodol. 42 (3), 245–292.

Vecchia, A.V., 1985. Maximum likelihood estimation for periodic autoregressive moving average models. Technometrics 27, 375–384.

Wang, W., Van Gelder, P.H.A.J.M., Vrijling, J.K., Ma, J., 2004a. Predictability of streamflow processes of the Yellow River. Proceedings of the Sixth International Conference on Hydroinformatics. World Scientific, Singapore, pp. 1261-1268.

Wang, W., Van Gelder, P.H.A.J.M., Vrijling, J.K., 2004b. Periodic autoregressive models applied to daily streamflow. Proceedings of the Sixth International Conference on Hydroinformatics. World Scientific, Singapore, pp. 1334-1341.

Wang, W., Van Gelder, P.H.A.J.M., Vrijling, J.K., 2005. Some issues about the generalization of neural networks for time series prediction. In: Duch, W. (Ed.), Artificial Neural Networks: Formal Models and Their Applications, Lecture Notes in Computer Science, vol. 3697, pp. 559–564.

Wang, W., Van Gelder, P.H.A.J.M., Vrijling, J.K., Ma, J., in press. Testing for nonlinearity of streamflow processes at different timescales. J. Hydrol.

Willmott, C.J., Ackleson, S.G., Davis, R.E., Feddema, J.J., Klink, K.M., Legates, D.R., O'Donnell, J., Rowe, C.M., 1985. Statistics for the evaluation and comparison of models. J. Geophys. Res. 90 (c5), 8995–9005.

Yakowitz, S., Karlsson, M., 1987. Neareast neighbour methods for time series with application to rainfall-runoff prediction. In: MacNeil, J.B., Umphrey, G.J. (Eds.), Stochastic Hydrology. D. Reidel, Dordrecht, pp. 149–160.

Zealand, C.M., Burn, D.H., Simonovic, S.P., 1999. Short term streamflow forecasting using artificial neural networks. J. Hydrol. 214, 32–48.

Zhang, B., Govindaraju, R.S., 2000. Prediction of watershed runoff using Bayesian concepts and modular neural networks. Water Resour. Res. 36 (3), 753–762.