Journal of Hydrology, Vol. 358, No. 1-2, 2008, pp 96-111

River Stage Prediction Based on a Distributed Support Vector Regression

C. L. Wu; K. W. Chau*; and Y. S. Li

Dept. of Civil and Structural Engineering, Hong Kong Polytechnic University,
Hung Hom, Kowloon, Hong Kong, People's Republic of China
(*Email: cekwchau@polyu.edu.hk)

Abstract:

An accurate and timely prediction of river flow flooding can provide time for the authorities to take pertinent flood-protection measures such as evacuation. Various data-derived models including LR (linear regression), NNM (the nearest-neighbor method) ANN (artificial neural network) and SVR (support vector regression), have been successfully applied to water level prediction. Of them, SVR is particularly highly valued, because it has the advantage over many data-derived models in overcoming overfitting of training data. However, SVR is computationally time-consuming when used to solve large-size problems. In the context of river flow prediction, equipped with LR model as a benchmark and genetic algorithm-based ANN (ANN-GA) and NNM as counterparts, a novel distributed SVR (D-SVR) model is proposed in the present study. It implements a local approximation to training data because partitioned original training data is independently fitted by each local SVR model. ANN-GA and LR models are also used to help determine input variables. A two-step GA algorithm is employed to find the optimal triplets (C, ε, σ) for D-SVR model. The validation results reveal that the proposed D-SVR model can carry out the river flow prediction better in comparison with others, and dramatically reduce the training time compared with the conventional SVR model. The pivotal factor contributing to the performance of D-SVR may be that it implements a local approximation method and the principle of structural risk minimization.

Keywords: Water level prediction; D-SVR; Input selection; Parameter optimization

Introduction

As one of a number of nonstructural flood protection measures, an accurate and timely prediction of water levels in the station of interest is of great importance in helping the authorities determine whether to take measures and if do which measures would best mitigate potential flood damage. In the last two decades, with the development of software technology, many approaches affiliated to 'black box' techniques including NNM (nearest neighbor method), ANN (artificial neural network), and SVR (support vector regression) have been widely applied to flood prediction.

NNM has been reported in the literature to analyze rainfall-runoff and runoff/discharge processes and has been compared with ARX (autoregressive model with exogenous inputs), or ARMAX (autoregressive moving average model with exogenous inputs). NNM yielded satisfactory results (Yakowitz*, 1987; Karlsson and Yakowitz, 1987; Galeati, 1990*). The technique was extended to NNLPW (nearest neighbor linear perturbation model) for rainfall-runoff prediction (*Shamseldin and O'Connor, 1996*). Feature selection is one of the most important aspects of pattern recognition, as used in the nearest neighbor method. In the context of univariate time series such as discharge, the feature vector can consist of several previous values (*Karlsson and Yakowitz, 1987; Galeati, 1990*).

Since the renaissance of ANNs in the late of 1980s, they have become the preferred prediction approach for many researchers and have been applied to a variety of issues. While some researchers in the literature employed ANNs alone for river flow forecasts (*Prochazka, 1997; Thirumalaiah and Deo, 1998; Sheta and El-Sherif, 1999; Liong et al.,2000; Salas et al., 2000; Qin et al., 2002; Cannon and Whitfield, 2002; Li and Gu, 2003; Huang et al., 2004; Cheng et al., 2005; García-Pedrajas et al., 2006*), many other researchers compared ANNs with traditional statistical techniques for river flow flood predictions. Comparisons between ANNs and AR (autoregressive) approaches appeared in the work of Raman and Sunilkumar (*1995*), Elshorbagy and Simonovic (*2000*), Thirumalaiah and Deo (*2000*) and Kişi (*2003*). Likewise, some studies were focused on comparisons between ANNs and ARMA (*Jain et al., 1999; Abrahart and See, 2000; Castellano-Me´ndeza et al., 2004*). The majority of studies have proven that ANNs are able to outperform traditional statistical techniques. Further, the superiority of ANNs over nonlinear regression in predicting river flows has been attributed to the possible existence of nonlinear dynamics, which are not well captured by the regression technique. A hybrid ANN model developed by Wang et al. (*2006*) was used to predict daily stream flow.

SVR, with highly similar structures to ANN, can learn from experimental data. SVR performs structural risk minimization (SRM) that aims at minimizing a bound on the generalization error (*Kecman, 2001*). In this way, it creates a model with a minimized VC-dimension (named after the authors, Vapnik and Chervonenkis), which means good generalization. Since SVR generalization performance does not depend on the dimensionality of input space, it can be used with small data sets. However, ANN is data intensive, and has to cover as many patterns as possible in order to perform well, and the generality of ANN is difficult to control as a result of implementing the empirical risk minimization (ERM) principle. Recently, some applications of SVR have been seen in the prediction of rainfall-runoff process, rainfall, and river flow. For example, Sivapragasam et al. (*2001*) performed one-lead-day rainfall forecasting and runoff forecasting using SVR, in which the input data are pre-processed by singular spectrum analysis, resulting in a high-dimensional input space. Yu et al. (*2004*) proposed a scheme that combined chaos theory and SVM to forecast daily runoff. Bray and Han (*2004*) applied SVM to forecast runoff, focusing on the identification of an appropriate model structure and relevant parameters. Sivapragasam and Liong (*2004*) used the sequential elimination approach to identify the optimal training data set and then performed SVR to forecast the water level. Sivapragasam and Liong (*2005*) divided the flow range into three regions, and employed different SVR models to predict daily flows in high, medium and low regions. Lin et al. (*2006*) presented a SVR model to predict long-term monthly flow discharge series, and a comparison with results of appropriate ARMA and ANN models demonstrated the better performance of SVR. Yu et al. (*2006*) carried out a real-time flood stage forecasting based on SVR in which a hydrological concept of the time of response was employed to identify lags of inputs and a two-step grid search method was used for finding optimal parameters.

However, a major drawback of SVR is that training time tends to increase exponentially with the number of training samples. For example, according to the algorithm presented in this paper below, the time required is about two days for a magnitude of 1000 training data whereas it is only 40 minutes for a magnitude of 100 training data. Moreover, using a single model to learn large-size data may well lead to mismatch as there are different noise levels in different input regions (*Cheng et al., 2006b*), which is a normal scenario for those rivers characterized by seasonal flooding.

This paper mainly aims at developing a distributed SVR (D-SVR) model with a two-step GA parameter optimization method to carry out a prediction of river flow. In order to evaluate the performance of D-SVR, prediction is also arrived at via linear regression (LR), NNM, and ANN-GA (genetic algorithm-based ANN). As an extension of the previous study (*Chau et al., 2005*), some of the background on LR and ANN-GA will be set aside in the present paper. Thus, the paper is constructed as follows: firstly, the principle of SVR and D-SVR is introduced and following this NNM is briefly described. Secondly, in the section on construction of models, an emphasis is placed to input selection, and parameter k in NNM and parameters (C,ε,σ) in D-SVR are optimized. In the results and discussion section, results reveal that D-SVR model outperforms the other three models, but with a larger training time except for the conventional SVR. In the conclusion, it is suggested that nonlinear models may achieve more notable advantages over LR in the case of rainfall-runoff mapping.

SVR and Distributed SVR

Unlike classical adaptation algorithms that work in an L_1 or L_2 norm and minimize the absolute value of an error or of an error square with ERM, SVR performs SRM (*Kecman, 2001*). In this way, it creates a model with good generalization. The SRM induction principle and the methodology of SVR are briefly described below (*Gunn, 1998; Dibike et al., 2001; Kecman, 2001; Sivapragasam et al., 2001, Liong and Sivapragasam, 2002; Cherkassky and Ma, 2004; Yu et al., 2006*).

Statistical learning theory

We consider here standard regression formulation in general settings for predictive learning. The goal is to estimate an unknown real-valued function in the relationship:

$$y = r(X) + \delta \qquad (1)$$

where δ is independent and identically distributed (i.i.d) zero mean random error (noise), X is a multivariate input and y is a scalar output. The estimation is made based on a finite number of samples (training data): (X_i, y_i), ($i = 1,\cdots,N$). The training data are i.i.d. samples generated according to some (unknown) joint probability density function

$$p(X,y) = p(X)p(y|X) \qquad (2)$$

The unknown function in (1) is the mean of the output conditional probability (aka regression function)

$$r(X) = \int yp(y|X)dy \qquad (3)$$

A class of functions $f(X,\omega)$ can be formulated to approximate the relationship between input vector and the output variable, where ω is the parameter vector of the function. The problem of learning is to select the best function $f(X,\omega_0)$ (learning machine) from $f(X,\omega)$ that can predict the output y as accurately as possible. Generally, the quality of an approximation is measured by the loss or discrepancy measure $L(y, f(X,\omega))$. Therefore, the best approximation function is that for which the following expected risk function $R(\omega)$ is as small as possible:

$$R(\omega) = \int L(y, f(X,\omega))dp(X,y) \tag{4}$$

It is known that the regression function (3) is the one minimizing prediction risk (4) with the familiar squared loss function loss:

$$L(y, f(X,\omega)) = (y - f(X,\omega))^2 \tag{5}$$

Note that the set of functions $f(X,\omega)$, $\omega \in \Lambda$ supported by a learning method may or may not contain the regression function (3). Thus, the problem of regression estimation is the problem of finding the best approximation function that minimizes the prediction risk function

$$R(\omega) = \int (y - f(X,\omega))^2 dp(X,y) \tag{6}$$

using only the training data. This risk function measures the accuracy of the learning method's predictions of unknown target function $r(X)$.

A difficulty arises in the process of calculating (6) because the probability distribution $p(X,y)$ is unknown. Therefore, it is necessary an induction principle for risk minimization. One such principle is the ERM inductive principle. A straightforward method is to replace the expected risk $R(\omega)$ by the empirical risk $R_{emp}(\omega)$

$$R_{emp}(\omega) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - f\left(X_i,\omega\right)\right)^2 \tag{7}$$

However, the ERM principle does not guarantee that the function $f_{emp}(X,\omega)$ that minimizes the empirical risk $R_{emp}(\omega)$ converges to the true (or best) function $f(X,\omega_0)$ that minimizes the expected risk $R(\omega)$ when the number of training data is limited, such that the sample is small. In other words, a smaller error on the training set does not necessarily imply higher generalization ability (i.e., a smaller error on an independent test set). To make the most out of a limited amount of data, a novel statistical technique called SRM has been developed (*Vapnik 1995, 1998*). The theory of uniform convergence in probability provides bounds on the deviation of the empirical risk from the expected risk. This theory shows that it is crucial to restrict the class of functions that the learning machine can implement to one with a capacity that is suitable for the amount of available training data.

The SRM principle theoretically minimizes the expected risk based on the simultaneous minimization of both the empirical risk and the confidence interval Ω . Therefore, SRM can maintain a trade off between the accuracy of the training data and the capacity of the learning machine so as to improve generalization of the model.

For $\omega \in \Lambda$ and $N > h$, a typical uniform VC bound on the expected risk (also called generalization bound R), which holds with probability $1 - \eta$, has the following form (*Vapnik, 1995, 1998*):

$$R(\omega) \le R_{emp}(\omega) + \Omega(N,h,\eta) \tag{8}$$

$$\Omega(N,h,\eta) = \sqrt{\frac{h\left(\log\frac{2N}{h} + 1\right) - \log\left(\frac{\eta}{4}\right)}{N}} \tag{9}$$

The parameter h is called the VC-dimension, and it describes the capacity of a set of functions to represent the data set. The VC dimension is a measure of the model complexity and is often proportional to the number of free parameters in the function $f(X,\omega)$. Particularly when N/h is small, a small empirical risk does not guarantee a small value of the actual risk. In this case, in order to minimize the actual risk $R(\omega)$, one has to minimize the right-hand side of the inequality in (8) simultaneously over both terms. In order to do this, one has to make the VC dimension a controlling parameter. Therefore, the SRM inductive principle is intended to minimize the risk functional with respect to both terms: the empirical risk $R_{emp}(\omega)$ and the confidence interval Ω. The VC confidence term in (8) depends on the chosen class of functions, whereas the empirical risk depends on the one particular function chosen by the training procedure. The objective here is to find that subset of the chosen set of functions, such that the risk bound for that subset is minimized. This is done by introducing a "structure" by dividing the entire class of functions into nested subsets (Fig. 1). SRM then consists of finding that subset of functions which minimizes the bound on the actual risk. This is done by simply training a series of machines, one for each subset, where for a given subset the goal of training is simply to minimize the empirical risk. One then takes that trained machine in the series whose sum of empirical risk and VC confidence is minimal (*Burges, 1998*).

Fig. 1 should be put here

Nonlinear support vector regression

In the real hydrological world, most issues of interest tend to be nonlinear. A linear SVR is extremely limited. In order to deal with the nonlinearity, the input data, X, in input space is mapped to a high dimensional feature space via a nonlinear mapping function, $\phi(X)$. Hence, the underlying function becomes

$$f(X,\omega) = \omega \cdot \phi(X) + b \qquad (10)$$

Therefore, the objective of the SVR is to find optimal ω , b and some parameters in kernel function $\phi(X)$ so as to construct an approximation function of the underlying function.

When introducing Vapnik's ε-insensitivity error or loss function (see Fig. 2), the loss function $L_\varepsilon(y, f(X,\omega))$ on the underlying function can be defined as

$$L_\varepsilon(y, f(X,\omega)) = \left|y - f(X,\omega)\right|_\varepsilon = \begin{cases} 0 & if \left|y - (\omega\cdot\phi(X)+b)\right| \le \varepsilon \\ \left|y-(\omega\cdot\phi(X)+b)\right| - \varepsilon & otherwise \end{cases} \qquad (11)$$

where y represents observed value. Fig. 2 presents the concept of nonlinear SVR, corresponding to Eq. (11). Similar to linear SVR (*Kecman, 2001; Yu et al., 2006*), the nonlinear SVR problem can be expressed as the following optimization problem:

$$\begin{aligned} &minimize \quad R_{W,\xi_i,\xi_i^*} = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \\ &subject \;\; to \quad \begin{cases} y_i - f(\phi(X_i),\omega) - b \le \varepsilon + \xi_i \\ f(\phi(X_i),\omega) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} \end{aligned} \qquad (12)$$

where, the term of $\frac{1}{2}\|\omega\|^2$ reflects generalization, and the term of $C\sum_{i=1}^{l}(\xi_i + \xi_i^*)$ stands for empirical risk. The objective in Eq. (12) is to minimize them simultaneously, which implements SRM to avoid underfitting and overfitting the training data. ξ_i and ξ_i^* are slack variables, shown in Fig. 2 for measurements "above" and "below" an ε tube. Both slack variables are positive values. C is a positive constant that determines the degree of penalized loss when a training error occurs.

By introducing a dual set of Lagrange Multipliers, α_i and α_i^* , the minimization problem can be solved in a dual space. The objective function in dual form can be represented as (*Gunn, 1998*):

$$\begin{aligned} &\mathit{maximize}\ \ L_d\left(\alpha,\alpha^*\right) = -\varepsilon\sum_{i=1}^{N}\left(\alpha_i^*+\alpha_i\right)+\sum_{i=1}^{N}\left(\alpha_i^*-\alpha_i\right)y_i-\frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)\left(\phi(X_i)\cdot\phi(X_j)\right) \\ &\mathit{subject\ \ to}\ \begin{cases} \sum_{i=1}^{N}(\alpha_i-\alpha_i^*)=0 \\ 0\le\alpha_i^*\le C, \qquad i=1,\cdots,N \\ 0\le\alpha_i\le C, \qquad i=1,\cdots,N \end{cases} \end{aligned} \tag{13}$$

There is no fixed guideline how to select an appropriate nonlinear function $\phi(X_i)$. Furthermore, the computation of $\left(\phi(X_i)\cdot\phi(X_j)\right)$ in the feature space may be too complex to perform. An advantage of SVR is that the nonlinear function $\phi(X)$ need not be used. The computation in input space can be performed using a ''kernel'' function $K(X_i,X_j)=\left(\phi(X_i)\cdot\phi(X_j)\right)$ to yield inner products in feature space, avoiding having to perform a mapping $\phi(X)$. In utilizing kernel functions, the key issue is to select admissible kernel functions. The admissible kernel function should be any symmetric function in input space which can represent a scalar product in feature space. The Mercer kernel functions belonging to a set of reproducing kernels (*Vapnik, 1999; Kecman, 2001*) can be proven admissible. Therefore, any functions that satisfy Mercer's theorem can be used as a kernel. A couple of commonly used kernels in SVR include: (1) linear $K(X_i,X_j)=X_i\cdot X_j$; (2) polynomial with degree d $K(X_i,X_j)=\left[(X_i\cdot X_j)+1\right]^d$;(3) multilayer perceptron $K(X_i,X_j)=\tanh[(X_i\cdot X_j)+b]$;(4) Gaussian RBF $K(X_i,X_j)=\exp(-\frac{\left\|X_i-X_j\right\|^2}{2\sigma^2})$. After obtaining parameters α_i , α_i^* , and b_0 , the final approximation function of the underlying function is

$$f(X_i)=\sum_{i=1}^{N}(\alpha_k-\alpha_k^*)K(X_k\cdot X_i)+b_0\,,k=1,\cdots,n \tag{14}$$

where X_k stands for the support vector, α_k and α_k^* are parameters associated with support vector X_k , N and n represent the number of training samples and support vectors, respectively.

Fig. 2 should be put here

SVR expressed in matrix notation

The standard quadratic optimization problem for an ε-insensitive function can be expressed in matrix notation as (*Gunn, 1998; Kecman, 2001*)

$$minimize \quad L_d(x) = \frac{1}{2}x^T Hx + C^T x \qquad (15)$$

where, H is Hessian matrix, x stands for Lagrangian Multipliers. They are expressed as

$$H = \begin{bmatrix} G & -G \\ -G & G \end{bmatrix}, \ C = \begin{bmatrix} \varepsilon - Y \\ \varepsilon + Y \end{bmatrix}, \text{ and } x = \begin{bmatrix} \alpha \\ \alpha * \end{bmatrix}$$

with constraints

$$x \cdot (1,\cdots,1,-1,\cdots,-1) = 0\,,$$

$$\alpha_i, \alpha_i * \geq 0, i = 1,\cdots,l.$$

G is an (l,l) matrix with entries $G_{ij} = [X_i^T X_j]$ for a linear regression, and $\alpha = [a_1,\cdots,\alpha_l]$, $\alpha* = [a_1*,\cdots,\alpha_l*]$, $\varepsilon - Y = [\varepsilon - y_1,\cdots,\varepsilon - y_l]$, $\varepsilon + Y = [\varepsilon + y_1,\cdots,\varepsilon + y_l]$. (Note that G_{ij}, as given above, is a badly conditioned matrix and we rather use $G_{ij} = [X_i^T X_j + 1]$ instead).

In the case of the nonlinear regression, the learning problem is again formulated as the maximization of a dual Lagrangian (15). A similar matrix notation as Eq. (15) is expressed. However, H here is with the changed Grammian matrix G that is now given as

$$G = \begin{bmatrix} G_{11} & \cdots & G_{1l} \\ \vdots & G_{ii} & \vdots \\ G_{l1} & \cdots & G_{ll} \end{bmatrix}$$

where the entries $G_{ij} = \phi^T(X_i)\phi(X_j) = K(X_i)(X_j), i, j = 1,\cdots,l.$ Based on the above matrix form, a SVR programming is easy to make.

D-SVR Configuration

Fig. 3 should be put here

A primitive idea of D-SVR is to partition the original training set into a couple of subsets and then generate a local SVR for each subset independently. Further, an appropriate data fusion approach (sometimes called aggregation) is employed to combine local predictions into a hybrid output. Fig. 3 displays the configuration of D-SVR. First of all, fuzzy c-means clustering algorithm is employed to split the original training set into L training subsets. In the present study, water level variables are characterized by clear seasonal variability, and so the raw training set is clustered into eight sub regions. Thus, each subset further serves for training L SVRs. For a new input X, L outputs ($\mathrm{Y_i}$,i=1,$\cdots$,L) will be generated by the D-SVR model and are associated with L degrees of membership (μ_i,i=1,$\cdots$,L). Degree of membership can be determined via the inverse of Square Euclidean Distance between the new input X and C_i which is the center of i-th subset. Calculation is formulated as follows:

$$\begin{cases} \mu_{\mathrm{i}}=1 & \mathit{if}\ d_i=0,\quad i=1,\cdots,L \\ \mu_{\mathrm{i}}=\left(\dfrac{1}{\mathrm{d_i}}\right)\Big/\left(\sum_{\mathrm{i=1}}^{\mathrm{L}}\dfrac{1}{\mathrm{d_i}}\right) & \mathit{otherwise} \end{cases} \tag{16}$$

where, $\mathrm{d_i}=\|\mathrm{X\text{-}C_i}\|^2$ and $\sum_{\mathrm{i=1}}^{\mathrm{L}}\mu_{\mathrm{i}}=1, \mu_{\mathrm{i}}\in[0,1]$.

After L outputs ($\mathrm{Y_i}$,i=1,$\cdots$,L) and their degrees of membership (μ_{i},i=1,$\cdots$,L) are achieved, the combined output Y is

$$\mathrm{Y}=\sum_{\mathrm{i=1}}^{\mathrm{L}}\mu_{\mathrm{i}}\mathrm{Y_i} \tag{17}$$

However, we found experimentally that there are some drawbacks in this D-SVR: when training data is partitioned into several independent subsets without any overlapping, a large prediction error occurs. Generally, the error is larger than that obtained by using a SVR model alone. Analysis also found that the SVR is weak at extrapolation. When an input is far from its clustering center, the SVR will generate a weird prediction, usually quite large although associated with a small degree of membership. In view of this, we attempted to make the following improvement. We set the nearest neighboring two training subsets to overlap one input region by one in the entire input space, thus the number of training data in all sub models will be twice that of the original training data. Furthermore, only two maximum degrees of membership are activated to contribute to the combined output Y. Therefore, the third box in Fig. 3 addresses this task, where μ_j ($j=1,2$) is the first two maximum degree of membership in μ_i ($i=1,\cdots,n$), and Y_j ($j=1,2$) are corresponding outputs as listed in the fourth box of Fig. 3. Finally, a combined output for D-SVR model is

$$\mathrm{Y}=\sum_{\mathrm{j=1}}^{2}(\mu_{\mathrm{j}}\Big/\sum_{j=1}^{2}\mu_{\mathrm{j}})\mathrm{Y_j} \tag{18}$$

Nearest-Neighbor Method (NNM)

The following is a brief review of the NN method (*Galeati, 1990; Shamseldin and O'Connor, 1996*). Let $\{\mathrm{X}(i), i=1,N\}$ be a set of rainfall measurements or parameters related to the forecasting process being studied (e.g., temperature, soil saturation, etc.) expressed as $\mathrm{X}(i)=(P_i,P_{i-1},P_{i-2},\cdots,P_{i-m+1})^T$ where P stands for feature information (various hydro-meteorological factors affecting runoff prediction (*Galeati, 1990; Yakowitz, 1987*), m is the number of feature information contributing to feature vector or the vector dimension and $\{Q(i), i=1,N\}$ a set of discharges. Here, X and Q may be single or multiple variables. For each feature vector $\mathrm{X}(i)$, there is an associated discharge $Q(i)$ observed at the same time instant. Thus, the available historical data may be summarized into a set of pairs of feature vectors $\mathrm{X}(i)$ and scalar discharges $Q(i)$, as $\{\mathrm{X}(i), Q(i): i=1,N\}$, where n is the total number of the data in the whole historical record. Thus, the NN prediction of $Q(N+1)$ is obtained as:

$$\hat{Q}(N+1)=\frac{1}{k}\sum_{i\in\mathrm{S(X,N)}}Q(i+1) \tag{19}$$

where $\mathrm{S(X,N)}$ denotes the indices of k, the nearest neighbors to the feature vector $\mathrm{X(N)}$. The meaning of “nearest neighbors” has to be interpreted according to the Euclidean distance: if

$d(n)$ represents a vector of coordinates d_1 , d_2 ,…, d_m , the differences between the current feature vector and past data, the Euclidean distant is defined as:

$$\|d\| = \left(\sum_{i=1}^{m} d_i^2\right)^{1/2} \qquad (20)$$

Therefore, if i is in S and j is not in S , then $\|\mathrm{X}(N) - \mathrm{X}(i)\| \le \|\mathrm{X}(N) - \mathrm{X}(j)\|$. Intuitively speaking, the forecast $\hat{Q}(N+1)$ by the k nearest neighbor method is the sample average of succeeding runoff of the k nearest neighbors in the database.

As an example from the work of Karlsson and Yakowitz (*1987*) displayed in Fig.4, for simplicity, it is supposed that the feature vector depends only on three values of past discharges ($m=3$) i.e. $\mathrm{X(N)} = [Q(N), Q(N-1), Q(N-2)]$, and it is assumed that $k=4$. The NN algorithm searches through all the consecutive triplets of the historical record for the four triplets closest (in a Euclidean sense) to the present feature vector. The predicted discharge is the mean of successive outflows (shown in Fig.4 as circles) from the four closest historical events.

Standardization of X and Q is usually necessary because it eliminates the units from components or elements and reduces any differences in the range of values amongst components such as rainfall and discharge with their different units and scales. In order to reflect the relative importance because the more recent measurements in the feature vector generally have a greater weight towards predicted values, the Euclidean distance can be computed as a weighted Euclidean norm, i.e., $\|d\|_w = \left(\sum_{i=1}^{m} w_i \cdot d_i^2\right)^{1/2}$ where $w = (w_1, w_2, \cdots, w_m)$ is a fixed sequence of positive numbers (weights). In the present study, an equivalent weight is assigned to each dimension in the feature vector because all variables are water levels.

Thus, the prediction model is $\hat{Q}(N+1) = \frac{1}{k}\sum_{i \in \mathrm{S(X,N)}} Q(i+1)$. In order to reflect the relative contribution to prediction value, each of all k neighbors is set to a weight factor ω_i which is based on the Euclidean distance. The prediction model becomes

$$\hat{Q}(N+1) = \frac{1}{k}\sum_{i \in \mathrm{S(X,N)}} Q(i+1) \cdot \omega_i \qquad (21)$$

where, $\omega_i = \|d_i\|^{-2} \Big/ \sum_{i=1}^{k} \|d_i\|^{-2}$. Then, an optimal k has to be determined by calibration. Generally, the data set is divided into two parts: one is used to construct the NN-predictors (constructing patterns); the other is used to calibrate parameters. Objective function optimizing k is set up as $J(k) = \sum (Q(i+1) - \hat{Q}(i+1))^2, i = 1, \cdots N$, where $Q(i+1)$ is observed value.

Fig. 4 should be put here

Construction of Models

Study Area

The channel reach studied is in the middle stream of the Yangtze River, which is the largest river in China. It passes through Wuhan City, which is the capital of the Hubei Province (see Fig. 5). The flow of the Yangtze River is quite unsteady and exhibits a seasonal behavior. The flow is low during the winter months, and peak flow occurs during August and September. A hydrological year is often classified into a flooding period and a nonflooding period, which are from June to October and from November to May, respectively. The water level at the Luo-Shan station can be as low as 17.35 m during the nonflooding period and as high as 31.04 m during the flooding period. The average water levels are 20.8 and 27.1 m during the nonflooding and flooding periods, respectively. The purpose of this study is to predict water levels of the downstream station, Han-Kou, by known water levels of the upstream station, Luo-Shan. The lateral inflow is neglected, because it is very small in comparison with the discharge of the main stream.

Fig. 5 should be put here

Data Preparation

A remarkable property of ANNs or SVRs is their ability to handle nonlinear, noise, and non-stationary data. However, with suitable data preparation beforehand, it is possible to improve the performance further (*Maier and Dandy, 2000; Bray and Han, 2004*). Data preparation involves a number of processes such as data collection, data division and data-preprocessing. Here, data division and data standardization belonging to data preprocessing will be covered.

Many research papers have discussed data division in the process of application of ANN (*ASCE, 2000; Chau et al., 2005*). Typically, ANNs are unable to extrapolate beyond the range of the data used for training. Consequently, poor forecasts/predictions can be expected when the validation data contains values outside of the range of those used for training. It is also imperative that the training and validation sets are representative of the same population. Often statistical properties (mean, variance, range) from them are compared in order to measure the representatives. The similar data handling can be applied to SVR in order to obtain the same baseline of comparison. Taking the same data splitting way as that in Chau et al. (*2005*), the data are randomly divided into three sets: training, testing, and validation. While 75% of the data are used for training, 25% are used for validation. The training data are further divided into 2/3 for the training set and 1/3 for the testing set.

In the present study we extract 1,448 input-output data pairs of the following format from the data record:

$$[X(t-4),X(t-2),X(t),Y(t+1)]$$

which shows that the water level of Y at Han-Kou for the next day can be mapped by water levels of X at Luo-Shan at the present day, two-day ahead and four-day ahead. A detailed description for the mapping format can be found in the section on inputs selection. It was ensured that the data used for training, testing, and validation represents the same population so there is no need to extrapolate beyond the range of their training data. Table 1 shows the statistical parameters, including the mean, standard deviation, minimum, maximum, and range, for the training, testing, and validation sets.

Table 1 should be put here

Generally, original data for different variables span different ranges. In order to ensure that all variables receive equal attention during the training process, they should be normalized. In this regard, it is not true for this case as shown in Table 1. However, due to restricted domain of independent variables of transfer functions in ANN and kernel functions in SVR, the raw data normalization is required. Additionally, normalization will improve the condition number of the Hessian in the optimization problem (*Gunn, 1998*). All data are scaled to the interval 0.1–0.9. The advantage of using [0.1, 0.9] rather than [0, 1] is that extreme (high and low) water levels, occurring outside the range of the calibration data, may be accommodated (*Hsu et al., 1995*). The scaling and reserve scaling processes are formulated below:

$$X_{norm} = 0.1 + 0.8 \times \left(\frac{X_i - X_{min}}{X_{max} - X_{min}} \right) \qquad (22)$$

$$Y_{norm} = 0.1 + 0.8 \times \left(\frac{Y_i - Y_{min}}{Y_{max} - Y_{min}} \right) \qquad (23)$$

$$\hat{Y}_i = Y_{min} + \left(\frac{1.0}{0.8} \right) \times (\hat{Y}_{i,norm} - 0.1) \times (Y_{max} - Y_{min}) \qquad (24)$$

where X_{norm} and Y_{norm} denote scaled appearance of the raw data X_i and Y_i , $\hat{Y}_{i,norm}$ stands for the scaled prediction corresponding to Y_i , and $\hat{Y}_i$ is the prediction of Y_i in original scale.

Inputs Selection

In model development the selection of appropriate input variables is important since it provides the basic information about the system being modeled. However, determining appropriate inputs is not an easy task. Generally, input determination can be divided into two broad stages (*Bowden et al., 2005*). In the first stage, the objective is to reduce the dimensionality of the original set of inputs, resulting in a set of independent inputs, which are not necessarily related to the model output. As a matter of fact, the addition of unnecessary variables would create a more complex model than is required. Moreover, the complex model is susceptible to overfitting of training data. Therefore, it is imperative that variables are independent of each other as system inputs. This subset of inputs can then be used in the second stage to determine which of these inputs are related in some way to the output.

Bowden et al. (*2005)* presented a comprehensive review of approaches on input determination in the water resources and those approaches are broadly classified into five groups. In the present paper, a mixed approach is employed to find optimal inputs.

Usually, the number of input variables is not known a priori. A firm understanding of the hydrologic system under consideration plays an important role in the successful implementation of the model. For the present case, the travel time of flood between Luo-Shan and Han-Kou is determined to be about 24 hrs using the Muskingum method. In other words, the flood at Han-Kou has a phase lag of approximately one day with that at Luo-Shan. So X(t) as an input is reasonable. In order to reduce the dimensionality of inputs, an autocorrelation analysis on water levels on Luo-Shan was performed and is shown in Fig. 6. An extreme good autocorrelation exists in water level series and any one input at least in the first ten lags cannot be deleted according to this chart. A linear relation on water levels exists between Luo-Shan and Han-Kou. A stepwise linear model analysis on inputs (Luo-Shan water levels) and output (Han-Kou water level) can help determine optimal inputs from a viewpoint of the linear relationship. Fig. 7 is the result of a stepwise linear model. The optimal linear mapping format between two hydrology stations is with three inputs X_{10}, X_8, and X_6(corresponding to X (t), X (t-2), and X (t-4)) and one output Y (t+1).

Obviously, autocorrelation analysis and stepwise linear regression analysis cannot capture any nonlinearity among inputs and between inputs and output. Further, sensitivity analyses (computing the contribution to variance) (*Nord and Jacobsson, 1998*) and weights analyses (*Muttil and Chau, 2006*) on inputs based on ANN are carried out to extract nonlinear information. Notably, as Nord and Jacobsson (1998) reported in the conclusion of their paper, due to the random starting conditions, important inputs remain changeable. In addition, according to their evaluation criteria, results from both methods on the ranking of inputs are not always consistent during training. An improvement is to employ the ANN-GA model, which is with the architecture 3-3-1 described in the section of results, to obtain the relative optimal initial weights and biases for an ANN model.

When ANN is initialized by weights and biases from GA optimization, a more stable ANN model can be achieved and has a good generalization. Due to the fixed initial weights and biases, evaluation results on input importance are steady, but results from two approaches are still inconsistent. Based on the approach of Nord and Jacobsson (1998), X_9, X_{10}, X_8, X_3, and X_6 are ranked in the top five important inputs. When adding X_3, X_9 respectively to initial linear model based on X_6, X_8, and X_{10}, several models are generated. The performances of these models are listed in Table 2. From the perspective of AIC and RMSE from LR and ANN-GA, choosing X_6, X_8, and X_{10}, i.e. X (t), X (t-2), and X (t-4), as the optimal inputs is tenable. Finally, the optimal linear regression (LR) model is

$$\mathrm{Y(t{+}1)} = 1.18\mathrm{X(t)} - 0.398\mathrm{X(t-2)} + 0.229\mathrm{X(t-4)} - 5.08 \qquad (25)$$

Fig. 6 should be put here

Fig. 7 should be put here

Table 2 should be put here

Parameters Tuning Strategy of D-SVM

Obtaining optimal α_i and α_i^* in Eq. (13) depends heavily on these parameters that dominate the nonlinear SVR including the cost constant C, the radius of the insensitive tube ε, and the kernel parameters. In the present study, the Gaussian RBF is employed as kernel function. So these parameters consist of a triplet (C, ε, σ), whose components are mutually dependent, and so changing the value of one parameter changes other parameters. Therefore, a simultaneous or global optimization scheme such as GA can be helpful (*Cheng et al., 2006a*). Due to lack of any a priori knowledge for their bounds, a two-step GA search algorithm is recommended here, which is inspired by a two-step grid search method (*Hsu et al., 2003*). First, a coarse range search was used to achieve the best region of these three-dimensional grids. In the present study, coarse range partitions for C are [10^{-2} 10^{0}], [10^{0} 10^{2}], [10^{2} 5.0×10^{2}], and [5.0×10^{2} 10^{3}]. Coarse range partitions for ε are [10^{-4} 10^{-3}], [10^{-3} 10^{-2}], [10^{-2} 10^{-1}], and [10^{-1} 10^{0}], and coarse range partitions for σ are [10^{-3} 10^{-2}], [10^{-2} 10^{-1}], [10^{-1} 10^{0}], and [10^{0} 10^{2}]. There are 4^3 grids, and one of them is selected as intervals of parameters for the next step. Then, in the second step a further GA search for the triplets (C, ε, σ) will be carried out in the selected intervals.

In order to avoid overfitting of training data, testing data and training data were evaluated at the same time according to GA's fitting degree function (i.e., RMSE), and weighted average of their fitting degrees was used as the fitting degree of each population in the process of GA operation.

Evaluation of Performance

Many measures for model evaluation have been documented in the literature of hydrology application (*Legates and McCabe, 1999*; *Elshorbagy and* Simonovic*, 2000;* Luchetta and Manetti, 2003; *Goswami et al., 2005*). Several conventional measures such as correlation coefficient (r or R^2), efficiency coefficient (E), index of agreement (d), RMSE, and so on, were critically reviewed by Legates and McCabe (*1999*), and the review suggested that correlation coefficient is inappropriate for model evaluation. Legates and McCabe (*1999)* suggested a complete assessment of model performance should include at least one 'goodness-of-fit' or relative error measure (e.g., E or d) and at least one absolute error measure (e.g., RMSE or MAE) with additional supporting information. Herein, two conventional evaluation criteria in hydrology, RMSE (root mean square error) and E (efficiency coefficient), are used to measure performances of models based on training data, testing data and validation data.

(1) RMSE

$$RMSE = \sqrt{\frac{1}{N}\sum\nolimits_{i=1}^{N}(Y_i - \hat{Y}_i)^2} \qquad (26)$$

(2) E

Nash and Stucliffe (*1970*) defined the model coefficient of efficiency which ranges from minus infinity to 1.0, with higher values indicating better agreement, as:

$$E = 1 - \frac{\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{N}(Y_i - \bar{Y})^2} \qquad (27)$$

where $\hat{Y}_i$=forecast water level, Y_i=observed water level, $\bar{Y}$ =average observed flow, and N=number of observations. RMSE provides a quantitative indication of the model absolute error in terms of the units of the variable, with the characteristic that larger errors receive greater attention than smaller ones. This characteristic can help eliminate approaches with significant errors. However, some studies (*Kachroo and Natale, 1992; Legates and McCabe, 1999*) have reported that the index E is a rather crude index, being overly sensitive to extreme values, because of the square differences in the definition, while being insensitive to additive and proportional differences between model predictions and observations. This feature will lead to the increasing influence of large floods on the calibrated parameter values and thereby enhance the forecast accuracy of the larger floods. In the present study, however, parameter calibration is not based on E, but rather on RMSE.

Results and Discussion

Results from NNM

The nearest-neighbor method belongs to typical pattern prediction. A good prediction can be achieved when testing or validation patterns are as similar as possible to those of the training data. In other words, a salient limitation of the NNM is that in no case can a value higher than the historical discharges be predicted. This is a deficiency which would severely limit the generality or even the plausibility of the NNM when used in real time forecasting (*Karlsson and Yakowitz, 1987*). However, for daily management purpose in which the interest is not centered on extreme values, it is viable. Therefore, it is viable for daily water level prediction in the present study.

According to the principle of NNM, a key step is to find the optimal k (the number of the nearest neighbors) based on the training data. An optimization process on k is graphed in Fig. 8. The optimal k is 7 with RMSE_tst of 0.234m and RMSE_vali of 0.242m.

Fig. 8 should be put here

Fig. 9 should be put here

The upper pane in Fig. 8 displays 362 validation samples and comparison of absolute errors between LR and NNW prediction models is exhibited in the lower pane of Fig. 9. As a whole, error curves from LR and NNW show the same trend. However, compared with LR, the NNM exhibits larger error amplification at some particular points where local extremum appear on the water level curve. Obviously, the performance of NNM is slightly poorer than that of LR, which seems to be discrepant with the recognized fact that NNM can be superior to some linear models. Two potential aspects can contribute to the present phenomenon: first, the prediction series are highly linear; second, training data is not enough for NNM which make it not be able to efficiently capture these patterns reflecting local extremum points.

Results from ANN-GA

In the present study, the ANN-GA model played dual roles both as a counterpart model and as helping determine inputs for all models. Table 3 shows the process determining optimal architecture of ANN based on a three-layer network assumption. So the main task of this experiment was to find the optimal number of hidden nodes and number of training epochs. Here, a testing set was employed to avoid overfitting of the training set based on the early stop method. These values highlighted by bold and italic typeface in 'Test' column exhibit optimal training epochs for different hidden nodes. Configuration of ANN corresponding to the minimum of them may be relatively optimal. Obviously, the minimum is 0.2285 corresponding to M =3 and epoch=7000. Further, based on the selected parameters M and epoch, inputs analysis can be performed as shown in the previous section of input selection. Finally, the determined architecture of ANN for the present case is 3-3-1 with optimal training epoch of 7000. Corresponding RMSE for training, testing and validation set are 0.213m, 0.223m, and 0.237m as shown in Table 6.

Table 3 should be put here

Fig. 10 should be put here

Similar to Fig. 9, Fig. 10 describes prediction error processes from LR and ANN-GA. While ANN-GA does not exhibit a good capturing capacity for local extremum points on the curve of validation samples, it seems to exhibit a better capacity for capturing other points than the LR model. Other than the small size of training samples, an unsteady prediction result can contribute to the poor performance due to the unstable parameter optimization method inherent in ANN although GA can lead to a relatively stable initial weights and biases. In other words, the present ANN may still not an optimal ANN for this case.

Results from D-SVR and Conventional SVR

According to previous partition of original data set, samples in training, testing and validation sets are, respectively, 724, 362, and 362. Experiment showed that computation time may vary from about a couple of seconds to nearly half an hour when the number of samples ranges from 50 up to about 300. The optimization process for C, ε, σ based on GA will have to run hundreds of times, which is extremely time-consuming for large-size training samples. Therefore, the present training data was partitioned into eight subsets with an average size of 181 (724/4=181) samples due to the overlapping between two nearest subsets. When adding testing data to the training set, the sample number employed in using GA to optimize parameters (C, ε, σ) for D-SVRs is 2172 in all, i.e., two times as the number of training and testing samples (2172=2×(724+362)). On the other hand, for conventional SVR model, GA is also employed to find optimal triplets (C, ε, σ) for training set with the help of testing set to control overfitting. Table 4 displays clustering centers and the size of training and testing data associated with each subset for D-SVR model.

Table 4 should be put here

Based on the two-step GA search approach, the optimal values of triplet parameter (C, ε, σ) for each subset are obtained as shown in Table 5. The composite training error (RMSE) is 0.21m with a training time of about 2hrs, and support vectors are 68.5%. Further, the testing error and validation error are 0.209m and 0.211m, as shown in Table 6. As a comparison, the training, testing and validation errors from conventional SVR are respectively 0.213m, 0.216m, and 0.236m, which are larger than those from D-SVR, in particular for the validation error. Meanwhile, the training time in conventional SVR is far larger than that in D-SVR, which is unaccepted for the current one-day-ahead prediction.

In addition, Fig. 11 displays the comparison of absolute errors between LR and D-SVR models. Their error curves exhibit similar trend, but D-SVR shows evident better prediction capacity than LR in terms of absolute errors although predictions on local extremum points are still not very good, which may be due to the property of the local approximation performed by D-SVR model.

Table 5 should be put here

Fig. 11 should be put here

Comparison among Models and Discussion

Table 6 summarizes performance of different models from RMSE, E of validation data, and training time. In view of its unacceptable training time, conventional SVR model will be put aside in the discussion. Three nonlinear models, NNM, ANN-GA, and D-SVM, show a better performance than that of LR in terms of RMSE of training and testing. However, only D-SVR exhibits a better generalization than LR in terms of RMSE of validation data. The value of E also proves that D-SVR's efficiency is the best. A drawback of D-SVR is computationally time-consuming due to hundreds of times parameters optimization via GA.

In order to display the performance from nonlinear models, absolute error curves of them were graphed in Fig. 12. Errors from these curves are with a very similar trend that predictions are underestimated at some points whereas predictions are overestimated at other points such as from 230 to 290 at the X-axis.

Although NNM, ANN-GA, and D-SVM are all nonlinear models, they are different in essence. NNM and ANN are generally called nonlinear and non-parameter models unlike LR with its fixed formula form. Therefore, their performance is related to many aspects including raw data quality, suitable data preprocessing, and even the ability of modelers, in particular for ANN. However, different from NNM and ANN-GA, D-SVR does not depend on pattern identification to carry out prediction. To certain extent, it may be called a parameter model or semi-parameter model which can be uniquely achieved under the SRM principle when the triplet parameters are selected. On the other hand, a fixed prediction result is never expected for ANN model due to the random starting conditions. Moreover, the principle of ERM tends to make ANN and NNM be weak in the aspect of generalization.

The D-SVR model performed a nonlinear approximation for each subset. Obviously, a local nonlinear fitting from D-SVR should be better than an empirically global fitting from LR. Therefore, if over-fitting is carefully avoided, it is inevitable that the D-SVR achieves a better prediction in comparison with LR.

Table 6 should be put here

Fig. 12 should be put here

Conclusions and Recommendations

As one of nonstructural flood protection measures, the future water level at a downstream station was predicted by the water level series at an upstream station. Equipped with LR model as a benchmark and ANN-GA and NNM as counterparts, a novel D-SVR model was established to carry out the forecast using data collected from water level series at the upstream Luo-Shan station and downstream Han-Kou station. ANN-GA and LR models were also used to help determine input variables. A two-step GA algorithm was employed to optimize the triplet parameters (C,ε,σ) for D-SVR model. The validation results revealed that proposed D-SVR model can predict the water level better in comparison with the other models, which may be because it implements a local approximation method and the principle of SRM. However, compared with LR model, NNM and ANN-GA did not exhibit a powerful mapping ability in the present case.

Certainly, the conclusion should not be hastily drawn that NNM and ANN are worse. As a matter of fact, studies in the literature have reported that NNM and ANN are very powerful in terms of nonlinear mapping. Associated with small-size training data, the present case is characterized by a highly linear mapping relation, which restricts the power of NNM and ANN. A complicated mapping between rainfall and runoff may be expected to really expose their capabilities, which will be presented in a future study.

References

Abraharty, J. R., and 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.

ASCE Task Committee on Application of the Artificial Neural Networks in Hydrology. (2000). Artificial neural networks in hydrology I: preliminary concepts. ***J. Hydrol. Engng, ASCE***, 5(2): 115–123.

Bowden, G.J., Dandy, G.C., and Maier, H.R. (2005). Input determination for neural network models in water resources applications: Part 1—background and methodology. ***Journal of Hydrology***, 301, 75–92.

Bray, M. and Han, D. (2004). Identification of support vector machines for runoff modelling. ***J. Hydroinf.***, 6(4), 265–280.

Burges, C.J.C. (1998). A tutorial on support vector machines for pattern recognition. ***Data mining and knowledge Discovery***, 2(2).

Cannon, A. J. and Whitfield, P. H. (2002) Downscaling recent streamflow conditions in British Columbia, Canada, using ensemble neural network models. ***Journal of Hydrology***, 259, 136-151.

Castellano-Me´ndeza, M., Gonza´lez-Manteigaa, W., Febrero-Bande, M., Manuel Prada-Sa´ncheza, J., and Lozano-Caldero´n, R. (2004) Modeling of the monthly and daily behavior of the runoff of the Xallas river using Box–Jenkins and neural networks methods. ***Journal of Hydrology***, 296, 38–58

Chau, K. W., Wu, C. L., and Li, Y. S. (2005). Comparison of Several Flood Forecasting Models in Yangtze River. ***Journal of Hydrologic Engineering***, 10(6), 485-491.

Cheng, C.T., Chau, K.W., Sun, Y.G. and Lin, J.Y. (2005). Long-term prediction of discharges in Manwan Reservoir using artificial neural network models. ***Lecture Notes in Computer Science***, 3498, 1040-1045.

Cheng, C.T., Zhao, M.Y., Chau, K.W. and Wu, X.Y., (2006a). Using genetic algorithm and TOPSIS for Xinanjiang model calibration with a single procedure. ***Journal of Hydrology***, 316(1-4), 129-140.

Cheng, J., Qian, J.S., and Guo, Y.N. (2006b). A Distributed Support Vector Machines Architecture for Chaotic Time Series Prediction. ***Lecture Notes in Computer Science***, 4232, 892-899.

Cherkassky, V. and Ma, Y. (2004) Practical selection of SVM parameters and noise estimation for SVM regression. ***Neural Networks***, 17(1), 113-126.

Dibike, Y. B., Velickov, S., Solomatine, D., and Abbott, M. B. (2001). Model induction with support vector machines: introduction and applications. ***Journal of Computing in Civil Engineering***, 15(3), 208-216.

Elshorbagy, A. and Simonovic, S.P. (2000). Performance evaluation of artificial neural networks for runoff prediction. ***J. Hydrologic Engineering***, 5(4), 424-427.

Galeati, G. (1990). A comparison of parametric and non-parametric methods for runoff forecasting. ***Hydrological Science Journal***, 35(1), 79-84.

García-Pedrajas, N., Ortiz-Boyer, D., and Hervás, C. (2006). An alternative approach for neural network evolution with a genetic algorithm: Crossover by combinatorial optimization. ***Neural Network***, 19, 514-528.

Goswami, M., O'Connor, K.M., Bhattarai, K.P., and Shamseldin, A.Y. (2005). Assessing the performance of eight real-time updating models and procedures for the Brosna River. ***Hydrology and Earth System Sciences***, 9(4), 394-411.

Gunn, S.R. (1998). Support vector machines for classification and regression. ***Image, Speech and Intelligent Systems Tech. Rep.***, University of Southampton, U.K.

Hsu, C. W., Chang, C.C. and Lin, C. J. (2003). A practical guide to support vector classification. Available at http://www.csie.ntu.edu.tw/cjlin/papers/guide/guide.pdf. [Accessed on 20/6/07].

Hsu, K.L., Gupta, H.V., and Sorooshian, S. (1995). Artificial neural network modeling of the rainfall-runoff process. ***Water Resources Research***, 31(10), 2517-2530.

Huang W.R., Xu B., and Hilton, A.(2004). Forecasting Flows in Apalachicola River Using Neural Networks. ***Hydrological Processes***, 18, 2545-2564.

Jain, S. K., Das, A., and Drivastava, D. K. (1999). Application of ANN for reservoir inflow prediction and operation. ***J. Water. Resour. Plann. Manage***., 125(5), 263–271.

Kachroo, R.K., and Natale, L.(1992) Non-liner modelling of the rainfall-runoff transformation. ***Journal of Hydrology***, 135, 241-369.

Karlsson, M., and Yakowitz, S. (1987). Nearest-neighbour methods for non parametric rainfall runoff forecasting. ***Wat. Resour. Res.,*** 23(7), 1330-1308.

Kecman, V. (2001). Learning and soft computing: support vector machines, neural networks, and fuzzy logic models. **MIT press**, Cambridge, Massachusetts.

Kişi, O. (2003). River flow modeling using artificial neural networks. ***Journal of Hydrologic Engineering***, 9(1), 60-63.

Legates, D. R., and McCabe, Jr, G. J. (1999). Evaluating the use of goodness-of-fit measures in hydrologic and hydroclimatic model validation, ***Water Resour. Res***., 35(1), 233– 241.

Li, Y., and Gu, R.R. (2003). Modeling Flow and Sediment Transport in a River System Using an Artificial Neural Network. ***Environmental Management***, 31(1), 122-134.

Lin, J.Y., Cheng, C.T. and Chau, K.W. (2006) Using support vector machines for long-term discharge prediction. ***Hydrological Sciences–Journal,*** 51(4), 599-611.

Liong, S.Y., Lim, W.H., Paudyal, G.N.(2000). River Stage Forecasting in Bangladesh: Neural Network Approach. ***Journal of Computing in Civil Engineering***, ASCE 14(1), 1-8.

Liong, S.Y., and Sivapragasam, C. (2002). Flood stage forecasting with support vector machines***. Journal of American Water Resour***. 38(1):173 -186.

Luchetta, A., and Manetti, S. (2003). A real time hydrological forecasting system using a fuzzy clustering approach. ***Computers & Geosciences***, 29(9), 1111-1117.

Maier, H. R., and Dandy, G. C. (2000). Neural networks for the prediction and forecasting of water resources variables: a review of modeling issues and applications. ***Environmental Modeling and Software***, 15(1): 101-123.

Muttil, N. and K.W., Chau.(2006). Neural network and genetic programming for modelling coastal algal blooms. ***Int. J. Environment and Pollution***, Vol. 28, Nos. 3/4, pp.223–238.

Nash, J. E. and Sutcliffe, J. V. (1970), River flow forecasting through conceptual models part I — A discussion of principles, ***Journal of Hydrology***, 10 (3), 282–290.

Nord, L.I., and Jacobsson, S.P. (1998). A novel method for examination of the variable contribution to computational neural network models. ***Chemometrics and Intelligent Laboratory Systems***, 44(1), 153–160.

Prochazka, A. (1997). Neural networks and seasonal time-series prediction. ***Artificial Neural Networks****, Fifth International Conference on* (Conf. Publ. No. 440), Vol., Iss., pp:36-41.

Qin, G.H., Ding, J., and Liu, G.D. (2002) River flow prediction using artif icial neural networks: self-adaptive error back-propagation algorithm. ***Advances in Water Science*** (in Chinese), 13 (1), 37-41.

Raman, H., and Sunilkumar, N. (1995). Multivariate modeling of water resources time series using artificial neural networks. ***Hydrological Sciences Journal***, 40(2):145-163.

Salas, J.D., Markus, M., and Tokar, A.S. (2000). Stream flow forecasting based on artificial neural networks. In ***artificial neural networks in hydrology***, (eds.) Govindaraju, R.S., and Rao, A.R. Water Science and Technology Library.

Shamseldin, A.Y., and O'Connor, K.M. (1996). A nearest neighbour linear perturbation model for river flow forecasting. ***Journal of Hydrology***, 179, 353-375.

Sheta, A.F., and El-Sherif, M.S. (1999). Optimal prediction of the Nile River flow using neural networks. ***Neural Networks*, 1999**. IJCNN '99. International Joint Conference on, Vol.5,3438-3441.

Sivapragasam, C., Liong, S.Y. and Pasha, M.F.K. (2001) Rainfall and runoff forecasting with SSA-SVM approach. *J. Hydroinf.*, 3(7), 141–152.

Sivapragasam, C., and Liong, S.Y. (2004). Identifying optimal training data set – a new approach. In: Liong, S.Y., Phoon, K.K., Babovic, V. (Eds.), ***Proceedings of the Sixth International Conference on Hydroinformatics***, Singapore, 21–24 June 2004. World Scientific Publishing Co., Singapore.

Sivapragasam, C., and Liong, S. Y. (2005). Flow categorization model for improving forecasting. ***Nordic Hydrology***,36 (1), 37–48.

Thirumalaiah, K, and Deo, M.C. (1998). River stage forecasting using artificial neural networks. ***Journal of Hydrologic Engineering***, 3 (1), 26 -32.

Thirumalaiah, K., and Deo, M. C. (2000). Hydrological forecasting using neural networks. ***Journal of Hydrologic Engineering***, Vol. 5, No. 2, 180-189.

Vapnik, V. (1995). *The nature of statistical learning theory*, Springer, New York.

Vapnik, V. (1998). ***Statistical learning theory***, Wiley, New York.

Vapnik, V.N. (1999). An overview of statistical learning theory. **IEEE Transactions on Neural Networks** 10 (5), 988–999.

Wang, W., Van Gelder, P.H.A.J.M., Vrijling, J.K., and Ma, J. (2006) Forecasting Daily Streamflow Using Hybrid ANN Models. ***Journal of Hydrology***, 324, 383-399.

Yakowitz, S. (1987). Nearest-neighbour methods for time series analysis. ***Journal of time series analysis***, 8(2), 235-247.

Yu, P.S., Chen, S.T., and Chang I.F. (2006). Support vector regression for real-time flood stage forecasting. ***Journal of hydrology***, 328,704-716.

Yu, X.Y., Liong, S.Y., and Babovic, V. (2004). EC-SVM approach for real-time hydrologic forecasting. ***Journal of Hydroinformatics***, 6(3), 209-233.

Table 1. Statistical Parameters for Training, Testing, and Validation Sets

Model variables and data sets	Statistical parameters				
	Mean	Standard deviation	Minimum	Maximum	Range
X_{t-4} (m)					
Training set	23.46	3.71	17.37	30.96	13.59
Testing set	23.46	3.71	17.35	30.93	13.58
Validation set	23.46	3.71	17.37	31.04	13.67
X_{t-2} (m)					
Training set	23.46	3.71	17.35	31.04	13.69
Testing set	23.46	3.71	17.39	30.96	13.57
Validation set	23.47	3.71	17.37	30.80	13.43
X_t (m)					
Training set	23.47	3.71	17.37	30.96	13.59
Testing set	23.46	3.71	17.35	30.93	13.58
Validation set	23.46	3.71	17.37	31.04	13.67
Y_{t+1} (m)					
Training set	18.64	3.75	12.20	25.71	13.51
Testing set	18.64	3.75	12.26	25.70	13.44
Validation set	18.64	3.75	12.21	25.69	13.48

Table 2 Akaike's Information Criterion (AIC) for Models

Model	RMSE_trn (m)	RMSE_tst (m)	RMSE_vali (m)	AIC
LR(X_6,X_8,X_{10})	0.2396	0.240	0.237	1.630
LR(X_6,X_8,X_{10},X_9)	0.2395	0.242	0.238	1.632
LR(X_6,X_8,X_{10},X_3)	0.2394	0.240	0.237	1.634
ANN-GA(X_6,X_8,X_{10})	0.213*	0.223*	0.237*	1.601*
ANN-GA (X_6,X_8,X_{10},X_9)	0.210*	0.229*	0.245*	1.778*
ANN-GA (X_6,X_8,X_{10},X_3)	0.210*	0.229*	0.242*	1.778*

*Average over ten time tests

Table 3 RMSE of Train and Test Set with Changing Hidden Nodes (M) and Epochs

M	2		3		4		5		6		7	
Epoch	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
1000	0.2075	0.2326	0.2035	0.2399	0.1992	0.2775	0.1958	***0.2364***	0.1952	***0.2788***	0.4705	0.4533
3000	0.2075	***0.2325***	0.2080	0.9157	0.1980	0.2545	0.1937	0.7127	0.1857	0.3041	0.1855	***0.3210***
5000	0.6113	0.5673	0.2035	0.2399	0.1991	0.2573	0.1936	0.2873	0.1891	0.4448	0.1866	0.3679
7000	0.2148	0.2362	0.2032	***0.2285***	0.2008	***0.2554***	0.1869	0.2598	0.1905	0.4036	0.1902	0.4881
9000	0.2075	0.2325	0.2033	0.3080	0.1991	0.2769	0.1964	0.3792	0.1913	0.3381	0.1861	0.3975
11000	0.2075	0.2325	0.2021	0.2387	0.2002	0.3402	0.1942	0.5286	0.1869	0.2877	0.1831	0.3218
M	8		9		10		11		12		13	
1000	0.1871	0.3467	0.1760	1.4003	0.1746	0.4095	0.1711	0.4803	0.1664	0.6671	0.1644	0.4986
3000	0.1783	***0.3004***	0.1701	***0.3191***	0.1682	0.6884	0.1666	0.6667	0.1574	***0.3723***	0.1564	0.5020
5000	0.1877	2.5142	0.1795	0.4923	0.1761	0.9704	0.1728	***0.3499***	0.1576	1.1967	1.0416	0.9948
7000	0.1823	1.0629	0.1769	0.5054	0.1776	***0.3178***	0.8488	0.8334	0.1596	0.3657	0.1540	0.5319
9000	0.1797	0.4496	0.1806	0.5123	0.1702	0.8181	0.1662	0.4935	0.1676	2.2748	0.1579	***0.3514***
11000	0.4303	0.4220	0.1785	0.6683	0.1695	0.5109	0.1688	0.7671	0.1649	0.5135	0.1577	0.3803
M	14		15		16		17		18		19	
1000	0.1571	0.4847	1.5085	1.4068	0.1520	0.7772	0.1465	0.8242	0.1477	0.8621	0.1411	1.0257
3000	0.1505	0.7819	0.1486	0.5772	0.1544	0.9932	0.1508	0.5340	0.1324	0.7250	0.1327	0.7653
5000	0.1587	0.9445	0.1519	***0.3877***	0.1464	1.2091	0.1413	0.5150	0.1403	0.5975	0.1340	***0.6271***
7000	0.1600	***0.4541***	0.1459	0.6086	0.1461	***0.7584***	1.0348	0.9819	0.1316	***0.5012***	0.1387	0.7769
9000	0.1580	0.6617	0.1486	0.6439	0.1436	1.0023	0.1400	1.7086	0.1332	1.0227	0.1417	0.9096
11000	0.1552	0.5632	0.1465	1.2319	0.1467	0.7592	0.4745	***0.4632***	0.1362	0.8169	0.1341	2.5940

Note: values in Train and Test columns correspond to their RMSE

Table 4 Characteristics of Subsets Partitioned by FCM

Subset n	Clustering center				Training & testing data number
	X(t-4)	X(t-2)	X(t)	Y(t+1)	
1	27.9	27.9	27.9	23.1	304
2	24.9	24.9	24.9	20.1	258
3	26.7	26.6	26.6	21.8	252
4	23.4	23.4	23.4	18.7	307
5	18.5	18.5	18.5	13.5	284
6	21.7	21.6	21.6	16.9	240
7	29.5	29.5	29.5	24.5	143
8	19.8	19.8	19.8	14.9	384
Sum					2172

Table 5 Calibration Results of Triplet Parameters (C, ε, σ) in D-SVR

Model		Triplet Parameters			RMSE_trn	Percentage of SVs
		C	ε	σ	(m)	(%)
D-SVR	Submodel1	242.83	0.0004	0.689	0.210	68.5
	Submodel2	144.03	0.0065	0.422		
	Submodel3	80.53	0.0031	3.297		
	Submodel4	724.31	0.0033	0.477		
	Submodel5	239.29	0.0062	0.066		
	Submodel6	873.04	0.0231	0.527		
	Submodel7	894.02	0.0006	0.596		
	Submodel8	137.51	0.0035	0.922		
Conventional SVR		649.36	0.0049	0.515	0.213	68.6

Table 6 Performances for Different Models

Model	RMSE_trn (m)	RMSE_tst (m)	RMSE_vali (m)	E_vali	Training time (s)
LR	0.234	0.240	0.237	0.9960	-
NNM	-	0.234	0.242	0.9961	10
ANN-GA	0.213	0.223	0.237	0.9960	53
Conventional SVR	0.213	0.216	0.236	0.9960	153532
D-SVR	0.210	0.209	0.211	0.9968	7110

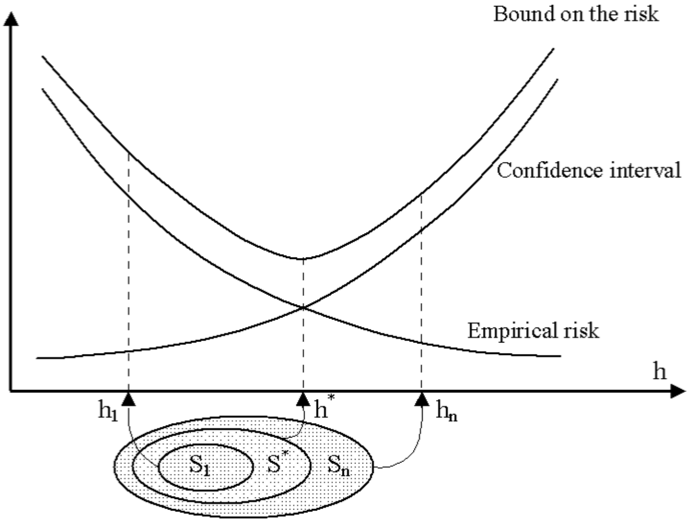


Fig. 1 Bound on Actual Risk Is Sum of Empirical Risk and Confidence Interval (adapted from Vapnik, *1998*)

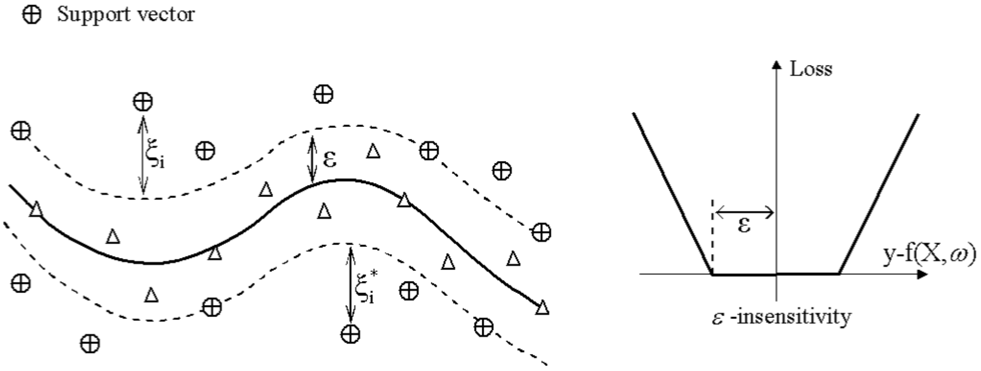


Fig. 2 Nonlinear SVR with Vapnik's ε-insensitive loss function

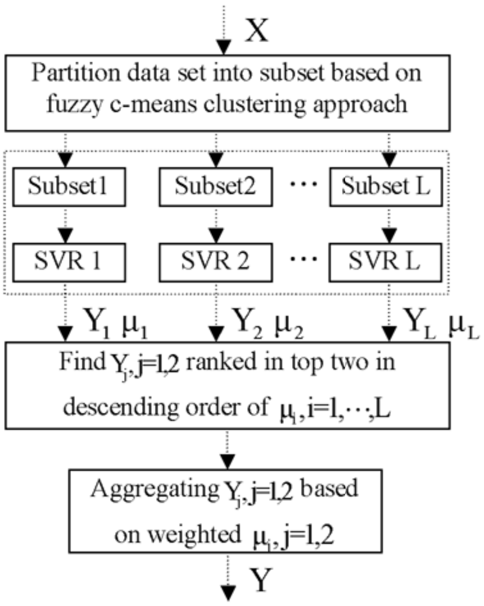

X
Partition data set into subset based on fuzzy c-means clustering approach
Subset1
Subset2
···
Subset L
SVR 1
SVR 2
···
SVR L
Y_1 μ_1
Y_2 μ_2
Y_L μ_L
Find Y_j, j=1,2 ranked in top two in descending order of μ_i, i=1,···,L
Aggregating Y_j, j=1,2 based on weighted μ_j, j=1,2
Y

Fig. 3 Configuration of D-SVR

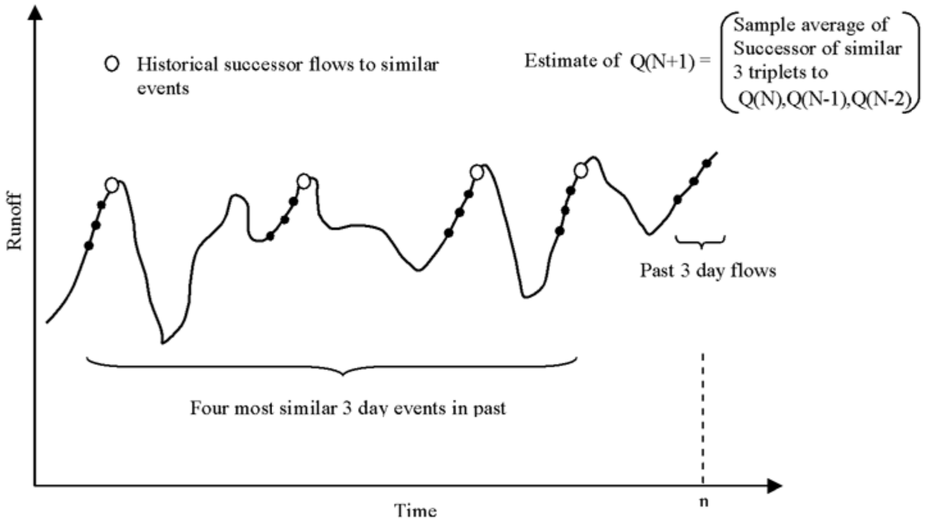

Historical successor flows to similar events
Estimate of Q(N+1) =
Sample average of Successor of similar 3 triplets to Q(N),Q(N-1),Q(N-2)
Runoff
Past 3 day flows
Four most similar 3 day events in past
Time
n

Fig.4 NN rule for runoff case (adapted from *Karlsson and Yakowitz, 1987*)

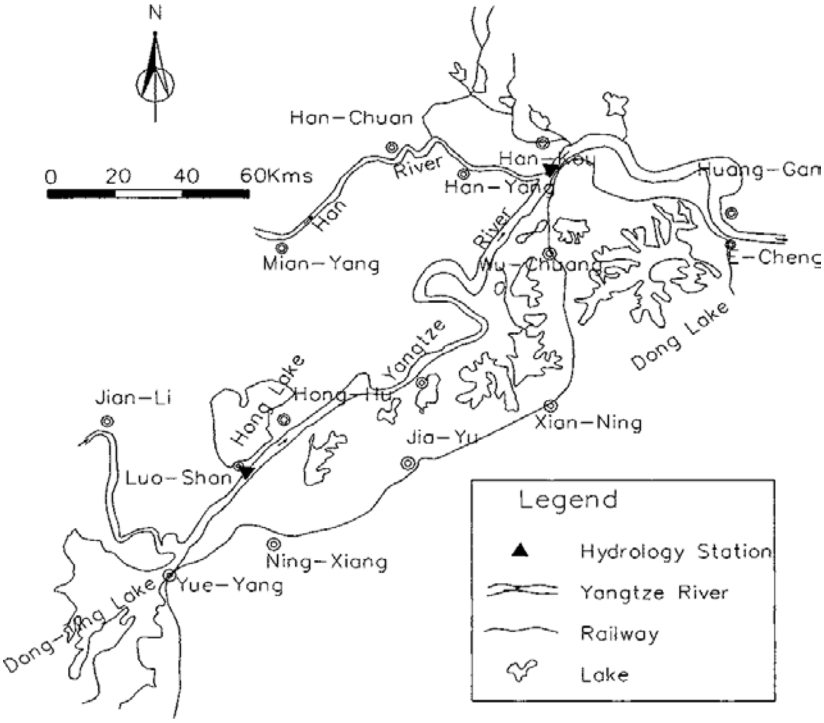


Fig. 5 Study Area

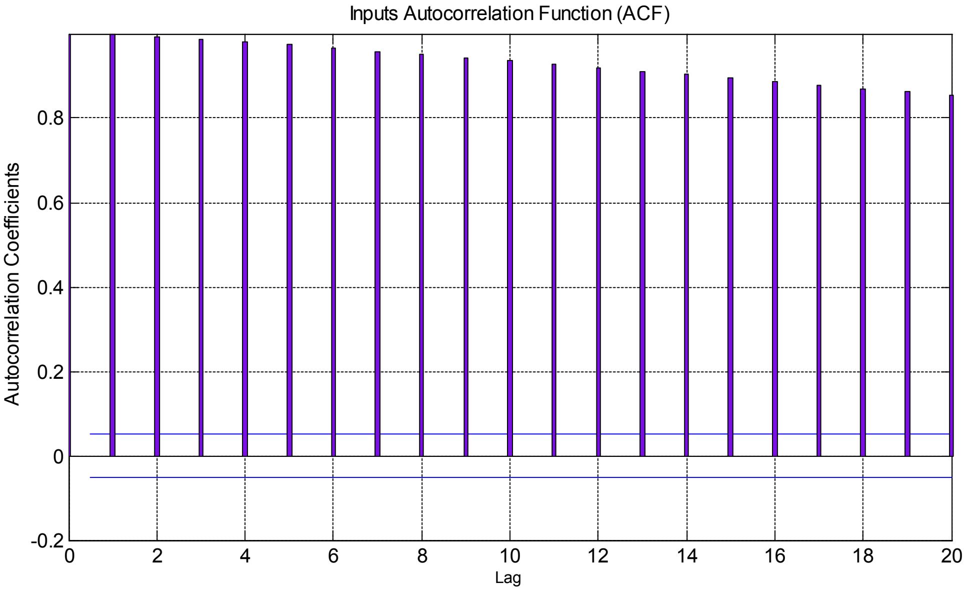
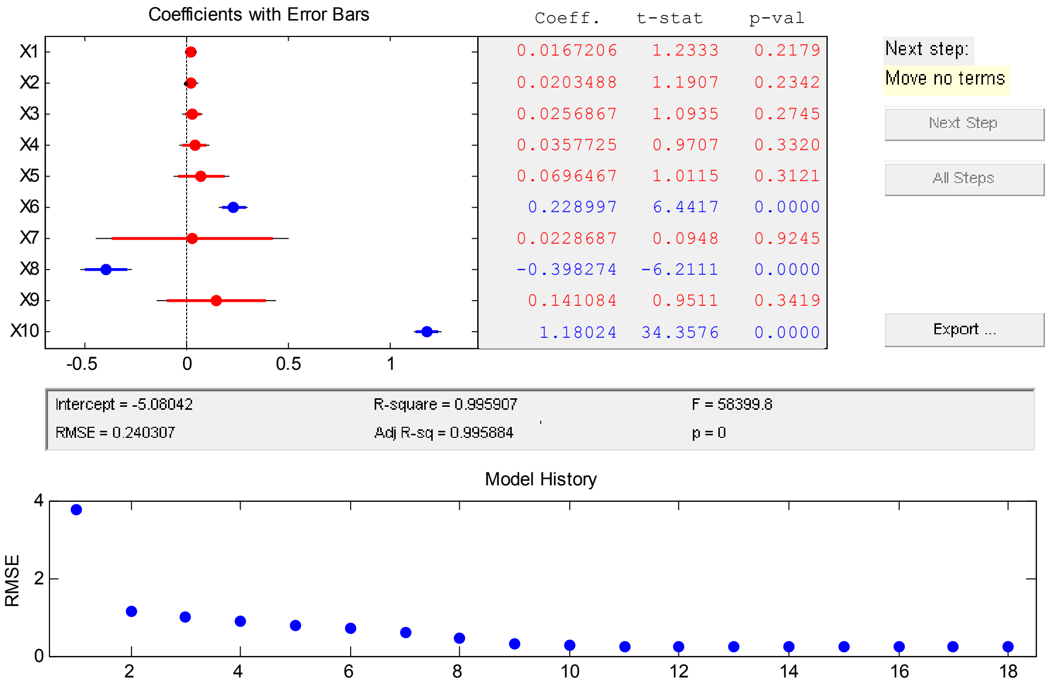


Fig. 6 Auto-correlation of Water Levels at Lou-Shan Station



	Coeff.	t-stat	p-val
X1	0.0167206	1.2333	0.2179
X2	0.0203488	1.1907	0.2342
X3	0.0256867	1.0935	0.2745
X4	0.0357725	0.9707	0.3320
X5	0.0696467	1.0115	0.3121
X6	0.228997	6.4417	0.0000
X7	0.0228687	0.0948	0.9245
X8	-0.398274	-6.2111	0.0000
X9	0.141084	0.9511	0.3419
X10	1.18024	34.3576	0.0000

```
Intercept = -5.08042          R-square = 0.995907          F = 58399.8
RMSE = 0.240307               Adj R-sq = 0.995884          p = 0
```

Fig. 7 Stepwise Linear Model Process

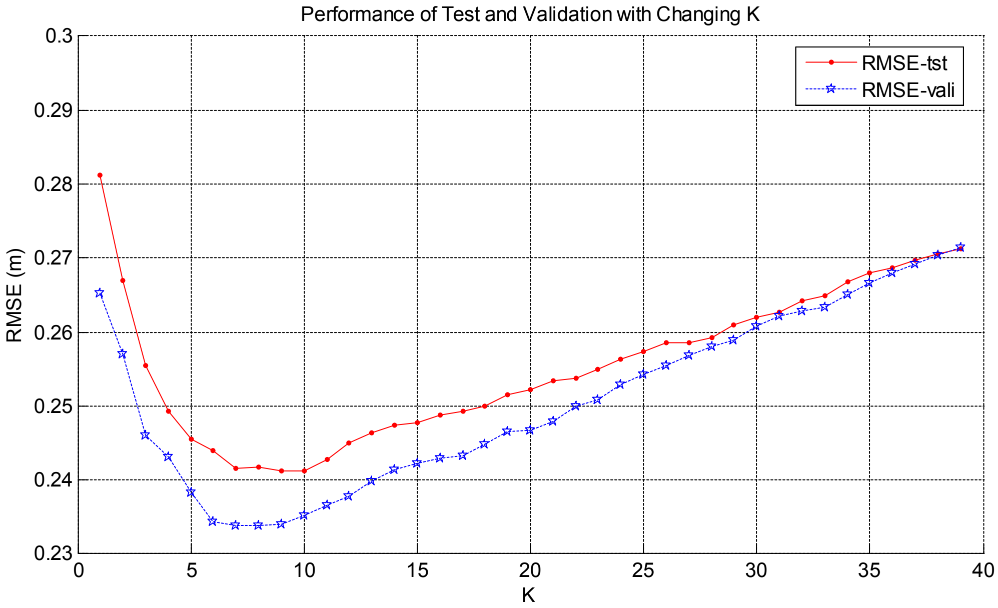


Fig. 8 Determining Optimal *k* for NNM

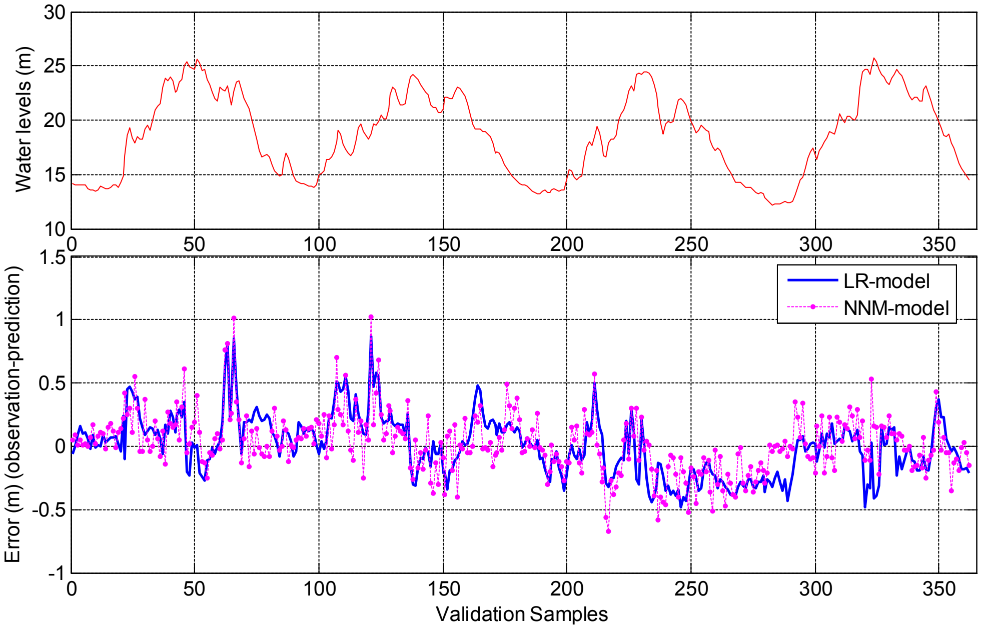


Fig. 9 Comparison of absolute errors between LR and NNM

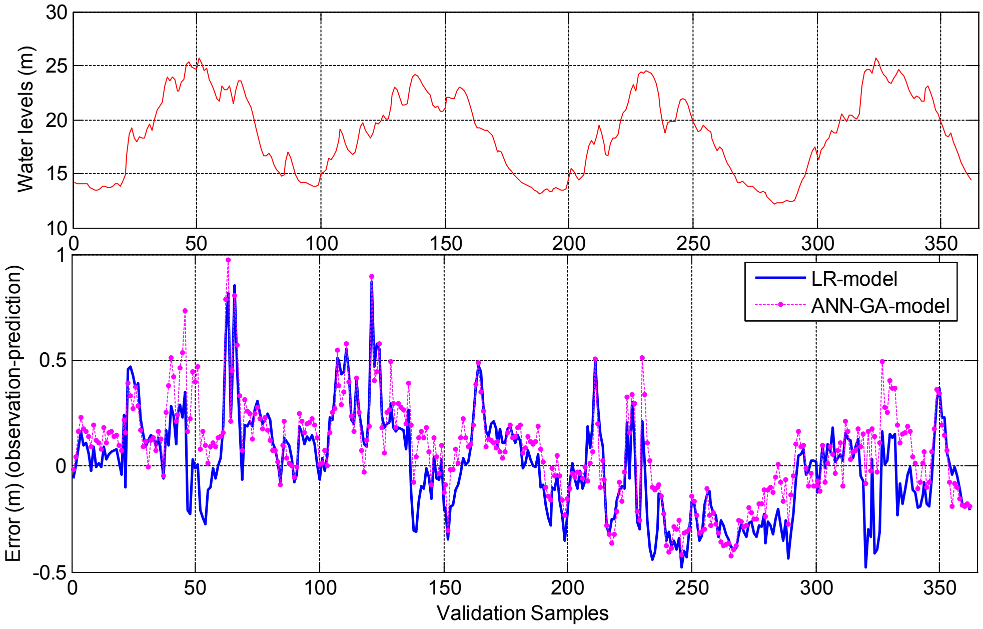


Fig. 10 Comparison of absolute errors between LR and ANN-GA

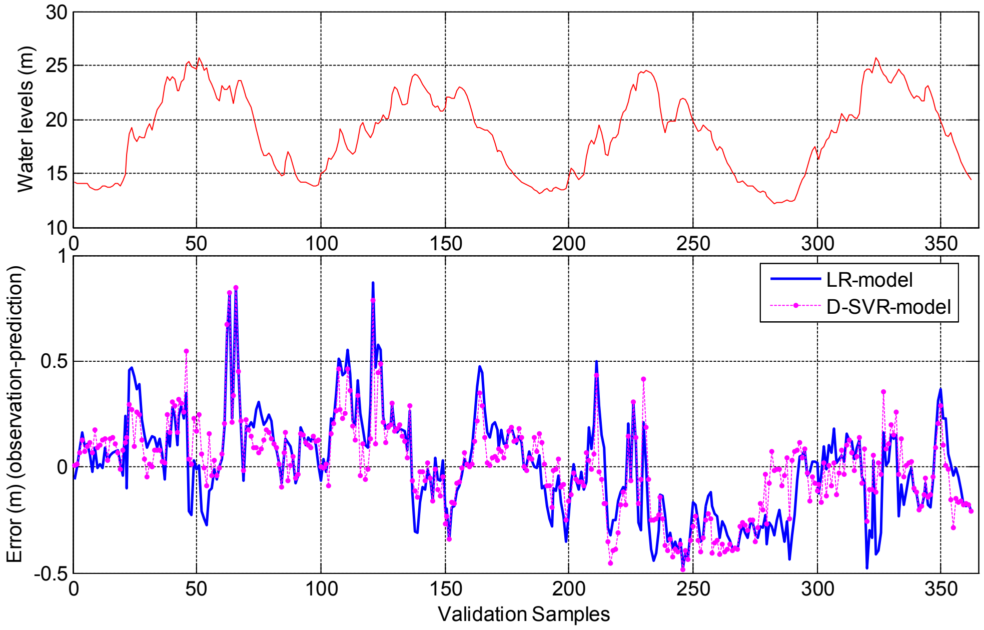


Fig. 11 Comparison of absolute errors between LR and D-SVR

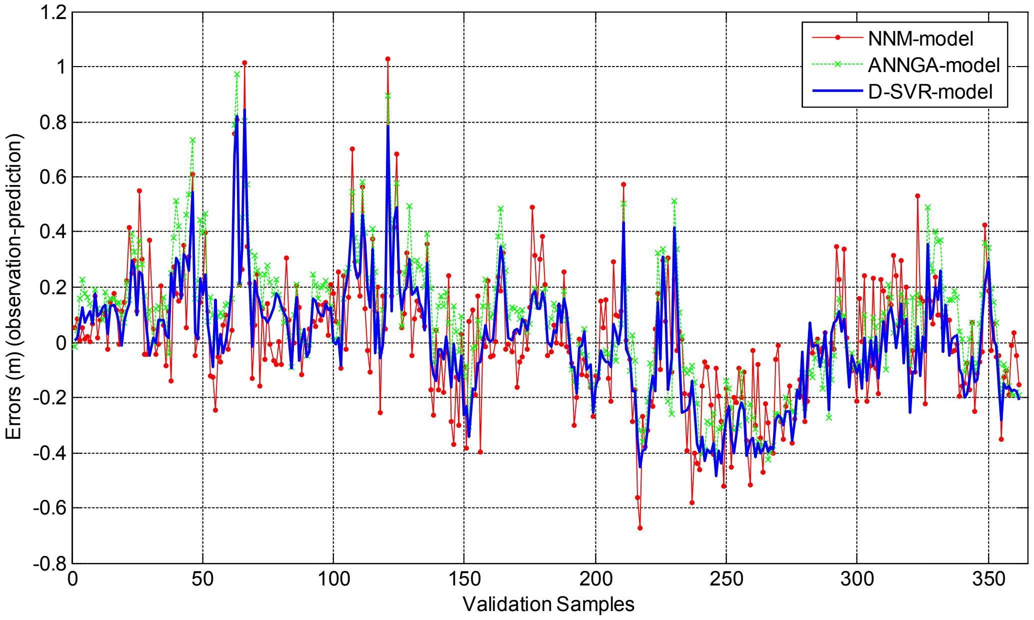


Fig. 12 Comparison of absolute errors among NNM, ANN-GA, and D-SVR