International Journal of Engineering Science Invention
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org || PP—119-123

A Review of the Forecasting of Rainfall and Flood Situation

Binaya Kumar Panigrahi[1], Tushar Kumar Nath[2,] , M. R. Senapati[3]
[1]*(Civil Engineering,Utkal University, Odisha, India)*
[2]*(Civil Engineering, IGIT , Sarang, Odisha, India)*
[3](*Computer Science and Engineering, VSSUT , Burla , Odisha , India*)

ABSTRACT: *Forecasts of water inflow into major reservoirs of different rivers are needed for the operational planning over periods ranging from a few hours to several months ahead. Medium-range forecasts of the order of a few days to two weeks have usually been obtained by simple ARMA-type models, which do not utilize information on observed or forecast precipitation, nor stream flow observations from upstream gauging stations. Recently, several different hydrological models have been tested to assess the potential improvements in forecasts that could be obtained by using observed and forecast precipitation as additional inputs. In this paper we have carried out a review of different techniques used for forecasting the water flow into various rivers and fore casting the flood situation.*

KEYWORDS: *Artificial Neural Network(ANN), Particle Swarm Optimization(PSO), ARMA Models***.**

I. INTRODUCTION

River flow forecasts are required to provide basic information for reservoir management in a multipurpose water system optimization framework. An accurate prediction of flow rates in tributary streams is crucial to optimize the management of water resources considering extended time horizons. Moreover, runoff prediction is crucial in protection from water shortage and possible flood damages.

The rainfall-runoff, process represents a complex nonlinear problem and there are several approaches to solve it. Traditionally, hydrological simulation modeling systems are classified into three main groups, namely, empirical black box, lumped conceptual, and distributed physically-based models [3, 2].

Flooding leads to numerous hazards, with consequences including risk to human life, disturbance of transport and communication networks, damage to buildings and infrastructure, and the loss of agricultural crops. Therefore, prevention and protection policies are required that aim to reduce the vulnerability of people and public and private property. Many solutions for flood mitigation and prevention have been suggested however, a vast amount of data and knowledge are required about the causes and influencing factors of floods and their resulting damage. Flood forecasting and prediction capabilities evolved slowly during the 1970s and 1980s. However, recent technological advances have had a major impact on forecasting methodologies. For instance, hydrological models use physical detection systems to forecast flood conditions based on predicted and/or measured parameters [2]. River flow models are used as components in actual flood forecasting schemes, where forecasts are required to issue warnings and to permit the evacuation of populations threatened by rising water levels. The basis of such forecasts is invariably observation and/or predictions of rainfall in the upper catchment area and/or river flows at upstream points along main rivers or tributaries. Forecasts about the discharge are obtained in real-time, by using the model to transform the input functions into a corresponding discharge function time [3].

II. LITERATURE REVIEW

Preliminary concepts and numerous applications of Artificial Neural Networks (ANN) to hydrology are available (ASCE, 2000a,b; Fernando and Jayawardena, 1998). Cheng and Chau (2001), Cheng and Chau (2002) proposed fuzzyiteration methodology and three-person multi-objective conflict decision model respectively for reservoir flood control operation for a case study of Fengman Reservoir, China. Chau et al. (2005) employed the Genetic Algorithm based Artificial Neural Network (ANN-GA) and the Adaptive Network based Fuzzy Inference System (ANFIS), for flood forecasting in a reach of the Yangtze River in China. Similar studies are reported by Cheng et al. (2002, 2008a,b).

Muskingum method is a hydrological flood routing technique (Chow et al., 1988) which was modified by many researchers. In the two parameter Muskingum method, there are number of ways for finding the two parameters, K (travel time) and x (weighing factor for prism and wedge storage of routing reach). These methods were discussed in detail by Singh and McCann (1980) and applied to a set of data to assess their relative efficacy. Gill (1978) proposed segmented curve method, in which least square method was used to find out the parameters of nonlinear form of Muskingum method. Stephenson (1979) demonstrated the way to calculate directly the coefficients of Muskingum method, C0, C1, and C2 using Linear Programming instead of calculating the parameters, K and x.

O'Donnell (1985) considered the lateral flow factor in Muskingum two parameter model of single input single output (si-so) nature, which was converted into a three parameter model. The parameters are K, x, a (a shows the fraction of lateral flow in comparison with inflow to the reach). The least square technique is used to find out these parameters in the routing reach automatically. Khan (1993) extended the si-so flood routing model to include lateral flow to form a multi input single output (mi-so) model with lateral flow.

Tung (1985) developed state variable modeling technique for solving the nonlinear form of Muskingum method. The parameters of the model were found out by four methods of curve fitting. Yoon and Padmanabhan (1993) developed software, MUPERS, where both linear and nonlinear relationships were dealt with. Kshirsagar et al. (1995) found parameters by a constrained, nonlinear (successive quadratic) programming. In this work, the Muskingum equation was used for routing the upstream hydrograph and the intermediate un gauged lateral inflow. The lateral inflow was calculated by an impulse response function approach. Mohan (1997) used genetic algorithm for parameter estimation of nonlinear Muskingum method and compared its performance with the approach by Yoon and Padmanabhan (1993).

Samani and Jebelifard (2003) applied multi linear Muskingum method for hydrologic routing through circular conduits. Das (2004) developed a methodology for parameter estimation for the Muskingum model of stream flow routing. Al-Humond and Esen (2006) presented two approximate methods for estimating Muskingum flood routing parameters. Geem (2006) introduced the Broydene Fletchere Goldfarbe Shanno (BFGS) technique, which searches the solution area based on gradients for estimation of Muskingum parameters.

III. ARTIFICIAL NEURAL NETWORK

An alternative approach to flow forecasting has been developed in the recent years, which is based on the ANN [3]. Recent studies have reported that ANN may offer a promising alternative for the hydrological forecasting of stream flow [7]. The ANN is a computer program that is designed to model the human brain and its ability to learn tasks [4]. An ANN differs to other forms of computer intelligence in that it is not rule based, as in an expert system. An ANN is trained to recognize and generalize the relationship between a set of inputs and outputs. Early artificial neural networks were inspired by perceptions of how the human brain operates. In the recent years, ANN technological developments have made it more of an applied mathematical technique with some similarities to the human brain. ANNs retain two characteristics of the brain as primary features: the ability to (1) 'learn' and (2) generalize from limited information [5]. Both biological and artificial neural networks employ massive, interconnected simple processing elements, or neurons. The knowledge stored as the strength of the interconnecting weights (a numeric parameter) in ANNs is modified through a process called learning, using a learning algorithm. This algorithmic function, in conjunction with a learning rule, (i.e., back-propagation) is used to modify the weights in the network in an orderly fashion. Unlike most computer applications, an ANN is not ''programmed,'' rather it is ''taught'' to give an acceptable answer to a particular problem. Input and output values are sent to the ANN, initial weights to the connections in the architecture of the ANN are assigned, and the ANN repeatedly adjusts these interconnecting weights until it successfully produces output values that match the original values. This weighted matrix of interconnections allows the neural network to learn and remember [10]. When using an ANN to solve a problem, the first step is to train the ANN to ''learn'' the relationship between the input and outputs. This action is accomplished by presenting the network with examples of known inputs and outputs, in conjunction with a learning rule. The ANN maps the relationship between the inputs and outputs, and then modifies its internal functions to determine the best relationship that is be represented by the ANN.

The inner workings and processing of an ANN are often thought of as a ''black box'' with inputs and outputs. One use-ful analogy that helps to understand the mechanism occurring inside the black box is to consider the neural network as a super-form of multiple regressions. Like linear regression, which finds the relationship that {y} = f{x}, the neural network finds some function f{x} when trained. However, the neural network is not limited to linear functions. It finds its own best function to the best of its ability, given the complexity used in the network, and without the constraint of linearity (Hewitson and Crane [5]). The most common type of artificial neural network consists of three groups, or layers, of units: (1) a layer of ''input'' units are connected to (2) a layer of ''hidden'' units, which are connected to (3) a layer of ''output'' units (Fig. 1)The activity of the input units represents the raw information that is fed into the network. The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input units and the hidden units. The behavior of the output units depends on the activity of the hidden units and the weights between the hidden units and output units [12].

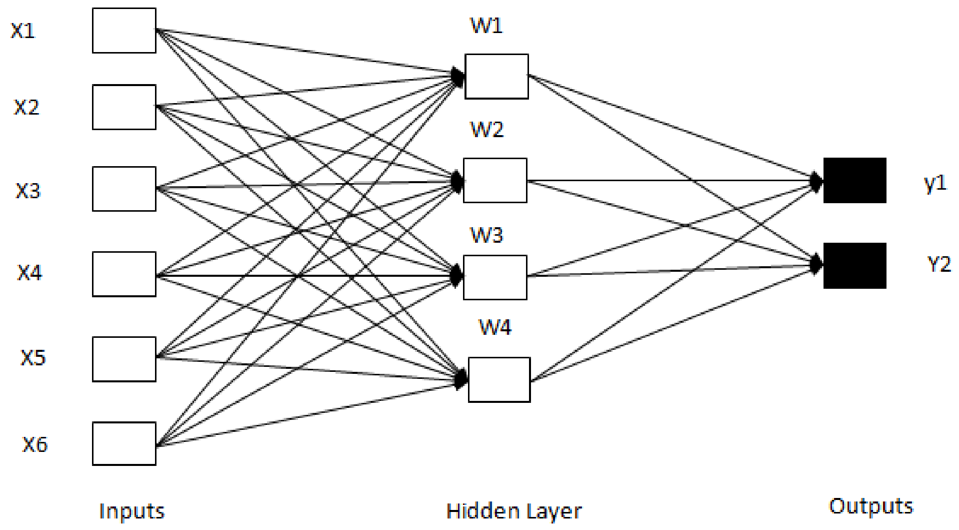

Figure 1: Simple feed forward network. http://dx.doi.org/10.1016/j.aej.2014.06.010

IV. PSO ALGORITHM

The principle of PSO algorithm is founded on the assumption that potential solutions will be flown through hyperspace with acceleration towards more optimum solutions. It is a populated search method for optimization of non-linear functions resembling the movement of organisms in a bird flock or fish school. Candidate solutions to the problem are termed particles or individuals. Instead of employing genetic operators, the evolution of generations of a population of these individuals in such a system is by cooperation and competition among the individuals themselves. In essence, each particle adjusts its flying based on the flying experiences of both itself and its companions. During the process, it keeps track of its coordinates in hyperspace which are associated with its previous best fitness solution, and also of its counterpart corresponding to the overall best value acquired thus far by any other particle in the population. In the algorithm, vectors are taken as representation of particles since most optimization problems are convenient for such variable presentations. The population is responding to the quality factors of the previous best individual values and the previous best group values. The allocation of responses between the individual and group values ensures a diversity of response. Its major advantages are the relatively simple and computationally inexpensive coding and its adaptability corresponding to the change of the best group value. The stochastic PSO algorithm has been found to be able to find the global optimum with a large probability and high convergence rate (Clerc and Kennedy, 2002).
Hence, it is adopted to train the multi-layer perceptrons, within which matrices learning problems are dealt with. Adaptation to network training A three-layered perceptron is chosen for this application case. Here, W[1] and W[2] represent the connection weight matrix between the input layer and the hidden layer, and that between the hidden layer and the output layer, respectively. When a PSO is employed to train the multi-layer perceptrons, the ith particle is denoted by

$$W_i = \{ W_i^{[1]}, W_i^{[2]} \} \tag{1}$$

The position representing the previous best fitness value of any particle is recorded and denoted by

$$p_i = \{ p_i^{[1]}, p_i^{[2]} \} \tag{2}$$

If, among all the particles in the current population, the index of the best particle is represented by the symbol b, then the best matrix is denoted by

$$p_i = \{ p_b^{[1]}, p_b^{[2]} \} \tag{3}$$

The velocity of particle i is denoted by

$$V_i = \{ V_i^{[1]}, V_i^{[2]} \} \tag{4}$$

If m and n represent the index of matrix row and column, respectively, the manipulation of the particles are as follows

$$V_i^{``[j]}(m, n) = V_i^{[j]}(m, n) + \{ r\alpha [P_i^{[j]}(m, n) - W_i^{[j]}(m, n)] + s\beta[P_b^{[j]}(m, n) - W_i^{[j]}(m, n)] \} / t \qquad (5)$$

And

$$W_i^{``[j]} = W_i^{[j]} + V_i^{[j]} t$$

where j = 1, 2; m = 1,. . . ,M_j; n = 1,. . . ,N_j; M_j and N_j are the row and column sizes of the matrices W, P, and V; r and s are positive constants; α and β are random numbers in the range from 0 to 1; t is the time step between observations and is often taken as unity; V`` and W`` represent the new values. Eq. (5) is employed to compute the new velocity of the particle based on its previous velocity and the distances of its current position from the best experiences both in its own and as a group. In the context of the social behavior, the cognition part, i.e., the second element on the right hand side of Eq. (5), represents the private thinking of the particle itself whilst the social part, i.e., the third element on the right hand side of Eq. (5), denotes the collaboration among the particles as a group. Eq. (6) then determines the new position according to the new velocity.

The fitness of the ith particle is expressed in term of an output mean squared error of the neural networks as follows

$$f(W_i) = \frac{1}{s}\sum_{k=1}^{s}[\sum_{t=1}^{0}\{t_{kl} - p_{kl}(W_i)\}^2] \qquad (6)$$

where f is the fitness value, t_{kl} is the target output; p_{kl} is the predicted output based on Wi; S is the number of training set samples; and, O is the number of output neurons.

V. ARMA MODELS

Most of the time-series techniques traditionally used for modeling water resources series fall within the framework of the ARMA class of linear stochastic processes. They are usually denoted as ARMA (p,q) models, where p and q are the auto-regressive and moving-average orders, respectively (Box and Jenkins, 1976; Brockwell and Davis, 1987; Bras and Rodriguez-Iturbe, 1994). They describe each observation of the time series as a weighted sum of p previous data, and the current as well as q previous values of a white noise process

$$x_t = \phi_1 (x_{t-1} - \mu_x) + \phi_2 (x_{t-2} - \mu_x) + \cdots + \phi_p (x_{t-p} - \mu_x) + \eta_t + \phi_{1}\eta_{t-1} + \phi_{2}\eta_{t-2} + \cdots + \phi_{q}\eta_{t-q} + \mu_x \qquad (15)$$

where x_t is the investigated time series; ηt , a white noise, i.e. a non-correlated, zero-mean random variable that is also not correlated with the past values of x_t; $\phi_1;\ldots;\phi_p$ and $\phi_1;\ldots;\phi_p$, the auto-regressive and moving-average parameters, respectively; and μ_x; the mean of the time series. Parameter estimation for ARMA models can be performed in several ways. We applied here an approximation in the spectral domain of the Gaussian maximum likelihood function, which was first proposed by Whittle (1953) for short-memory models.

VI. CONCLUSION

Thus, in our case, the K-NN algorithm looks through all consecutive d-dimensional vectors in the entire historical rainfall depths database and locates K of these d-ples, which are closest to the vector of d most recent rainfalls. The prediction of the next rainfall is then taken to be the average of the rainfall subsequent to these K historical nearest neighbors. It may be noticed that the K-NN approach does not require the selection of a class of models and the estimation of the model parameters, so that the identification of a specific form of the input/output relationship is not needed.

REFERENCES

[1]. R. Baratti,B. Cannas,A. Fanni,M. Pintus,G.M. Sechi,River flow forecast for reservoir management , through neural networks, www.elsevier.com/locate/neucom doi:10.1016/S0925-2312(03)00387-4

[2]. Sulafa Hag Elsafi, Artificial Neural Networks (ANNs) for flood forecasting at Dongola Station in the river Nile,Sudan, Sudan,doi.org/10.1016/j.aej.2014.06.010

[3]. Nile Basin Capacity Building Network (NBCBN). Flood and Drought, Forecasting and Early Warning Program, 2005.

[4]. Moore, R.J., Jones, D.A., Black, K.B., Austin, R.M.,Carrington, D.S., Tinnion, M.,Akhondi, A., 1994. RFFS and HYRAD: Integrated System for Rainfall and River Flow Forecasting in Real-Time and their Application in Yorkshire.BHS Ossasional paper No. 4, 12.
[5]. G.E.P Box, G.M. Jenkins, Time Series Analysis: Forecasting control, Holden –Day, Oakland, California, 1976.
[6]. S.J. Yakowitz, Markov flow models and the flood warning problem, Water Res Res 21 (1985) 81–88.
[7]. D. Nagesh Kumar , Falguni Baliarsingh, K. Srinivasa Raju , Extended Muskingum method for flood routing, doi:10.1016/j.jher.2010.08.003
[8]. Govindoraju, R.S., Rao, A.R., Artificial neural networks in hydrology. Netherlands, 2000.
[9]. Ozgur. Kisi, A combined generalized regression neural network wavelet model for monthly stream flow prediction, KSCE J.Civil Eng. 15 (8) (2011) 1469–1479.
[10]. Haykin, S., Neural networks. http://www.cul.salk.edu/.tewon/ICA/teaching-KAIST/references.htmc/1994.
[11]. Anderson, Dave, McNeil, George. Artificial neural networks technology. Data and analysis Centre for Software, Rome,August 1992 <http://www.dtic.mil>.
[12]. G.E.P Box, G.M. Jenkins, Time Series Analysis: Forecasting control, Holden – Day, Oakland, California, 1976.
[13]. B.C Hewiston, Crane, Precipitation Controls in SouthernMexico, in Neural Nets, Kluwer Academic Publisher, 1994.
[14]. Santosh. Patil, Sharda. Patil and Shriniwas. Valunjkar , Study of Different Rainfall-Runoff Forecasting Algorithms for Better Water Consumption, International Conference on Computational Techniques and Artificial Intelligence (ICCTAI'2012) Penang, Malaysia
[15]. MariuszZiejewski ,Goettler,H.J., Comparative analysis of the exhaust emissions for vegetable oil based alternative fuels Society of Automotive Engineers (1992) Paper No:920195.
[16]. Gaurav Srivastava , Sudhindra N. Panda , Pratap Mondal , Junguo Liu , Forecasting of rainfall using ocean-atmospheric indices with a fuzzy neural technique, doi:10.1016/j.jhydrol.2010.10.025.
[17]. K. W Chau, Particle swarm optimization training algorithm for ANNs in stage prediction of Shing Mun River, doi:10.1016/j.jhydrol.2006.02.025.
[18]. Fangqiong Luo and Jiansheng Wu2010 " Rainfall Forecasting Using Projection Pursuit Regression and Neural Networks"IEEE2010 Third International Joint Conference on Computational Science and Optimization pp 488 to 491.
[19]. E. Toth*, A. Brath, A. Montanari , Comparison of short-term rainfall prediction models for real-time flood forecasting.