SR-7/98-99

APPLICATIONS OF ARTIFICIAL NEURAL NETWORKS IN SURFACE WATER HYDROLOGY

NATIONAL INSTITUTE OF HYDROLOGY
JAL VIGYAN BHAWAN
ROORKEE - 247 667 (UTTARANCHAL)

1998-99

PREFACE

An Artificial Neural Network (ANN) is a computational method inspired by the studies of the brain and nervous system in biological organisms. ANN represent highly idealised mathematical models of our present understanding of such complex systems. One of the characteristics of the neural networks is their ability to learn. A neural network is not programmed like a conventional computer program, but is presented with examples of the patterns, observations and concepts, or any type of data which it is supposed to learn. Through the process of learning (also called training) the neural network organises itself to develop an internal set of features, that it uses to classify information or data. Due to its massively parallel processing architecture the ANN is capable of efficiently handling complex computations, thus making it the most preferred technique today for high speed processing of huge data. These characteristics render ANNs to be very suitable tools for handling various hydrological modelling problems. In surface water hydrology the possible usages of ANNs have only recently begun to be investigated.

This status report reviews some of the important applications of ANNs in surface water hydrology, highlighting their advantages and limitations. The review also covers the basic aspects of ANNs, i.e., various ANN architectures and various ANN learning algorithms.

This status report has been prepared by **Smt. Archana Sarkar,** Scientist `B' under the guidance of Shri R.D. Singh, Scientist `F' and Shri R. Mehrotra, Scientist 'E'.

(K.S. Ramasastri)
Director

CONTENTS

		Page No.
1.0	**INTRODUCTION**	1
1.1	WHAT ARE ARTIFICIAL NEURAL NETWORKS (ANNS)	1
1.2	POTENTIAL OF ARTIFICIAL NEURAL NETWORKS IN SURFACE WATER HYDROLOGY	4
2.0	**ASPECTS OF ANNS AND THEIR CHARACTERISTICS**	7
2.1	THE NEURON	7
2.2	TOPOLOGY	9
2.3	LEARNING ALGORITHMS	14
2.4	STRUCTURE	17
2.5	ARCHITECTURE	20
2.6	PERFORMANCE INDICATORS	20
2.7	LEVELS OF INTERACTION	23
2.8	CLASSIFICATION OF ANNS	27
2.9	VARIOUS ANN NETWORKS	28
2.10	PRACTICAL ISSUES IN USING ANNS FOR ENGINEERING APPLICATIONS	35
3.0	**EXISTING ANNS IN SURFACE WATER HYDROLOGY**	43
3.1	RAINFALL RUNOFF MODELLING	43
3.2	FORECASTING	46
3.3	HYDRAULICS	49
3.4	WATER RESOURCES	50
3.5	ENVIRONMENTAL	52
4.0	**CONCLUSIONS**	55
REFERENCES		57

LIST OF FIGURES

		Page No.
1.	The biological neuron	8
2.	Neuron Schematisation	8
3.	Dot neuron activation function	10
4.	Distance function	10
5.	Probabilistic activation function	10
6.	Delay connections	12
7.	Neuron layers	12
8.	Neuron clusters	12
9.	External feedback connections	13
10.	Internal feedback connections	13
11.	Supervised learning mode	16
12.	Unsupervised learning mode	16
13.	Batch learning mode	16
14.	Input vectors and their dimensions	19
15.	Vertical and horizontal data processing	19
16.	Modularisation of ANN models	22
17.	Hybrid model structure	22
18.	Computational energy surface	22
19.	The neuron	25
20.	Local topology	25
21.	Feedback loops	25
22.	Learning mode	25
23.	Testing the ANN	26
24.	Structure optimisation	26
25.	Global optimisation	26
26.	Classification of ANN	29
27.	Classification by Kosko	30
28.	A back propagation network	30
29.	Adaptive linear element	32
30.	Kohonen network	32
31.	A Hopfield neywork	32
32.	An adaptive resonance theory network	34
33.	A linear vector quantization network	34
34.	An Elman network	34

1.0 INTRODUCTION

The rapid growth of computing power in the last few decennia, has enabled us to develop effective modelling tools. They can perform difficult tasks within a fraction of the original required calculation time. As a result, old solution techniques were enhanced and new, computer based, theories developed, using the power of simple on/off circuits.

One of the most exciting ideas emerging from this vast pool of computer based research, is the thought of emulating the low-level mechanisms of the brain. This biological unit still outperforms any man made tool in terms of recognition, analysis, prediction and especially learning, thus providing puzzled researchers with enough motivation to conduct extensive research into this area of artificial intelligence.

The first research attempts to model the brain capability itself, were made in the 1950's by early cognitive pioneers. Based on the highly interconnected structure of the brain cells, they created the first primitive, so-called, artificial neural networks. However, due to the predominance of serial processing, rule based reasoning and programmed computing, few research efforts into artificial neural networks were made in the following decennia. It was only in the early 1980's, that a new breakthrough in neural network research occurred; stimulating more (theoretical) research into its capabilities world-wide.

Today, Artificial Neural Networks (ANNs) have proven to be very powerful in solving complex mathematical problems, such as speech recognition, successfully establishing themselves among the more traditional approaches (*Goppert and Rosenstiel, 1990*). ANNs characteristically demonstrate that they are fast, robust in noisy environments and very flexible in the range of problems they can solve. This has led to numerous real-world applications, such as image processing, robotics and stock market prediction, and encourages further research into their possible implementation in other related scientific fields.

1.1 WHAT ARE ARTIFICIAL NEURAL NETWORKS

Before giving a detailed description on the elements and characteristics of ANNs in the subsequent chapters, a general introduction on ANNs follows.

Generally speaking, artificial neural networks are computing systems that relate an input vector to an output vector. They are made up of a number of simple, but highly interconnected, signal processing units.

These units or artificial neurons, are the basic elements of any ANN and can be seen as simple I/O devices. Many different neuron types exist, all attempting to represent a certain part of the suspected functions of nerve cells in the brain.

In order to make complicated data processing possible, these neurons are linked together by several connections, so that each neuron will receive (output) signals from a multitude of other neurons. This composed input signal will be processed and the corresponding output will be sent to other neurons; which in their turn will react, continuing the process of action and reaction.

The data passing through the connections from one neuron to another, can be manipulated by weights. These weights indicate the strength or importance of a passing signal. Consequently, when these weights are modified, the data transfers through the network will change and the overall network performance will alter.

These manipulating parameters can all be adjusted and optimised in order to get a specific response from an ANN. This process of adjustment and optimisation is called learning and is defined by the learning algorithm of an ANN. The learning algorithm is a set of optimisation functions which adjust the weights in such a manner, that an input signal is correctly associated with a (desired) output signal. Several learning examples can be presented to the network, each one attributing to the optimisation of the weight distribution. Finally, when an ANN has learned enough examples, it is considered trained.

After the learning cycles, the learning algorithm of a trained network is often deactivated and the weights are frozen. Then, test data is presented to the ANN, which it has never encountered before, enabling a validation of its performance. This is referred to as testing of the ANN. Depending on the outcome, either the ANN has to relearn the examples (with some modifications) or it can be implemented for its designed use.

Summarising, different types and numbers of neurons with different interconnection infrastructures will create different ANN structures. Together with a learning rule and data examples, this will result in different network performances and thus different application possibilities.

However, every Artificial Neural Network will

- process data in parallel through the network (Parallel processing instead of traditional central processing);

- distribute system information throughout the entire network (Distributed memory instead of centralised information storage).

Furthermore, some additional characteristic ANN features can be identified:

- ANNs are made up of interconnected neurons.

- Each neuron will process information based on the signals it received and the local processing function it uses.

- The output signal of one neuron will affect large numbers of other neurons, due to high neuron interconnectivity.

- Learning rules can be applied, to change the characterising parameters of a network, such as the connection weights.

- Through cyclic adaptation of these characterising parameters, the ANN can be manipulated and so learn to produce a certain result.

1.2 POTENTIAL OF ARTIFICIAL NEURAL NETWORKS IN SURFACE WATER HYDROLOGY

Various approaches to the study of hydrologic problems can be grouped under two categories: (i) physical science approach - also referred to as a basic, pure, causal, dynamic or theoretical approach, and (ii) systems approach - also known as an operational, applied, empirical, black box or parametric approach. Hydrologic models are mathematical formulations to simulate natural hydrologic phenomena which are considered as processes or as systems.

Conceptual models are designed to approximate within their structures (in some physically realistic manner) the general internal subprocesses and physical mechanisms which govern the hydrologic cycle. Conceptual models usually incorporate simplified forms of physical laws and are generally non-linear, time invariant, and deterministic, with parameters that are representative of watershed characteristics. Until recently, for practical reasons (data availability , calibration problems, etc.) most conceptual watershed models assumed lumped representations of the parameters. Among the more widely used and reported lumped parameter watershed models are the Sacramento soil moisture accounting (SAC-SMA) model of the U.S. National Weather Service (*Burnash et al., 1973; Brazil and Hudlow, 1980*), HEC-1 Army Corps of Engineers, 1990), and the Stanford watershed model (SWM) (*Crawford and Linsley, 1966*) . While such models ignore the spatially distributed, time varying, and stochastic properties of the Rainfall-Runoff process, they attempt to incorporate realistic representations of the major nonlinearities inherent in the Rainfall-runoff relationships. Conceptual watershed models are generally reported to be reliable in forecasting the most important features of the hydrograph, such as the beginning of the rising limb, the time and the height of the peak, and volume of flow (*Kitanidis and Bras, 1980; Sorooshian, 1983*). However, the implementation and calibration of such a model can typically present various difficulties (*Duan et al., 1992*), requiring sophisticated mathematical tools (*Duan et al., 1993; Sorooshian et al.,1993*) , significant amounts of calibration data (*Yapo et al., 1995*), and some degree of expertise and experience with the model.

While conceptual models are of importance in the understanding of hydrologic processes, there are many practical situations such as streamflow forecasting where the main concern is with making accurate predictions at specific watershed locations. In such a situation, a hydrologist

may prefer not to expend the time and effort required to develop and implement a conceptual model and instead implement a simpler system theoretic model. In the system theoretic approach, difference equation or differential equation models are used to identify a direct mapping between the inputs and outputs without detailed consideration of the internal structure of the physical processes. The linear time series models such as ARMAX (auto regressive moving average with exogenous inputs) models developed by *Box and Jenkins (1976)* have been most commonly used in such situations because they are relatively easy to develop and implement; they have been found to provide satisfactory predictions in many applications (*Bras and Rodrigeuz Iturbe, 1985*). However, such models do not attempt to represent the nonlinear dynamics inherent in the transformation of rainfall to runoff and therefore may not always perform well.

Owing to the difficulties associated with non-linear model structure identification and parameter estimation, very few truly non-linear system theoretic watershed models have been reported (*Ikeda et al., 1976*). In most cases, linearity or piecewise linearity has been assumed (*Natale and Todini, 1976*). The model structural errors that arise from such assumptions can, to some extent, be compensated for by allowing the model parameters to vary with time (*Young and Wallis, 1985*). For example, real-time identification techniques, such as recursive least squares and state space Kalman filtering, have been applied for adaptive estimation of model parameters (*Kitanidis and Bras, 1980*) with generally acceptable results.

Besides the above limitations, the following areas are inadequately covered in existing models:

- Unknown processes: When the underlying physical laws are unknown, it is impossible to make a physically based model of the phenomenon using conventionally applied techniques.

- Incomplete/contradictory data: In all common techniques, pre-processing of data is absolutely necessary: lack of data creates large uncertainties in the model results, as approximations and guesses have to be carried out on the data beforehand. Contradictory data can even prohibit the design of a model or make it useless during validation.

- Simulating human decision paths: Another difficult problem arises when a decision model has to be designed. Usually, one predefines clear rules in a program (if, then else rules) or

tries to implement so-called expert knowledge (creating expert systems). But these models all need some kind of predefinition and often become useless when exposed to an unforeseen and undefined situation.

- Data intensive tasks: Detailed models often require many different parameters, especially when they model dynamic systems. This necessitates data transfer and can lead to excessive processing time. For on-line operation, these slow but detailed models are unacceptable, since the state of the monitored system often changes faster than the simulation frequency of the model.

- Changing environment: To decrease processing time, the number of adjustable parameters is often reduced and in order to retain the authenticity of the model the results are calibrated. This calibration procedure can take a lot of time, fault tolerances decrease and the resulting model will become useless when the simulated process changes. Furthermore, in rapidly changing environments long calibration times are unacceptable.

For the areas mentioned above, conventionally applied modelling techniques can be refined or complemented to achieve improved performance by implementing new or different methods. To this end, several promising modelling techniques can be applied; for instance: machine learning, fuzzy logic, artificial neural networks, or process engineering (based on Fourier series and flow diagrams).

The underlying reason for the Artificial neural Networks to be more appropriate than the other non-physically based modelling techniques, is that the ANN addresses precisely those classic modelling problems encountered in Surface Water Hydrology; namely processing speed (*Goppert and Rosentiel, 1990*), fault tolerance (*Tresp, Aghmad and Neuneier, 1994*), adaptability and learning capabilities (*Dawn, 1994*).

2.0 ASPECTS OF ANNS AND THEIR CHARACTERISTICS

2.1 THE NEURON

Looking at a biological nerve cell, we can roughly distinguish five important features: The dendrites, the soma, the hillock, the axon and the synapse (Fig.1).

The dendrites are hair like extensions of the soma and form input channels. The soma is the body of the cell, where the input signals are added up over time. The soma decides when and how to respond to inputs. The hillock is located at the base of the axon and generates impulses. These impulses go through the axon to the synapse, a region between the axon of one cell and the dendrites of one or more other cells.

The abstract equivalent of the nerve cell, the artificial neuron, is based on these features (Fig.2):

First, there are the weighted input connections to the neuron (dendrites). Then, these input signals are added up and fed into an activation function which determines whether the neuron will react at all (soma); if this is the case, then the signal will pass through a transfer function which determines the strength of the output signal (hillock). Finally the output signal will be sent through all the output connections (synapse) to the other neurons.

Different types of activation functions and transfer functions exist, which combined, generate different types of artificial neurons. These artificial neurons (some times referred to as nodes) can be classified into three group (*Fogelman, 1991*).

(1) **Dot product neurons**: Here the neuron consists of an activation function which is a dot product of inputs and weights, combined with a non linear transfer function (Fig.3), i.e.

$O_j = f(A_j)$

where, $A_j = \sum_{i=1}^{h} W_{ij} O_{i(j-1)}$

Transfer function f() is often sigmoidal but can also be a threshold function or a saturation limited. This type of neuron is often used for associative memory,

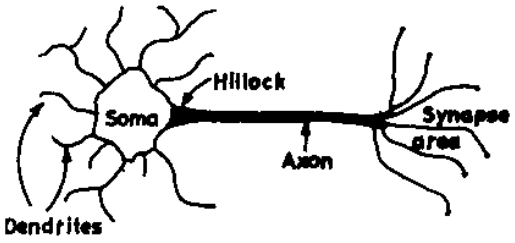


FIG.1. THE BIOLOGICAL NEURAN

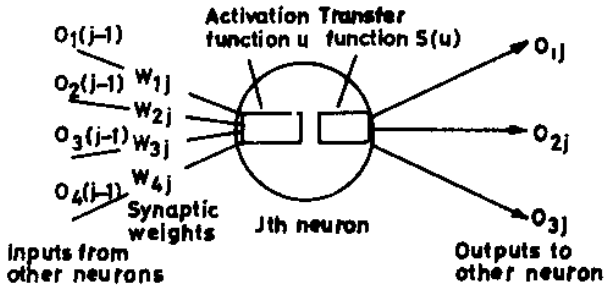


FIG. 2. NEURON SCHEMATISATION

classification and prediction.

(2) **Distance neurons**: The output is the result of a distance function between its weights and input vectors (Fig.4):

$$O_j = \| O_{(i-j)} - W_j \|^2 = \sum_j |O_{i(j-1)} - W_{ij}|^2$$

Where W_{ij} is the weight vector of neuron . Distance neurons are also referred to as Radial Basis Function Neurons and are used for clustering, vector quantization, code book design.

(3) **Probabilistic neurons**: Here, the activation function or level is dependent on some random determined gain (Fig.5). Sometimes even the weights can alter by chance.

2.2 TOPOLOGY

The topology of a network describes the connection infrastructure of an ANN. Connections link the neurons together and transport the data through the network. Different types of connections produce different performance characteristics.

Connections between neurons can be classified as being either inhibitory or excitatory. Inhibitory connections tend to prevent a neuron from reacting (negative term in a sum); while excitatory connections cause firing of the neuron (positive sum). At times, ANNs involve inhibitory connections from one neuron to all the others and this is referred to as lateral inhibition.

Other types of connections are delay connections (Fig.6). They introduce a time lag into the data flow, which can be useful for time related phenomena (*Day and Davenport, 1993*) like the prediction of the flood routing through a sewer system or the control of an overflow weir.

The definition of so called layers and clusters is another frequently applied representation of the topology of an ANN. A layer can be seen as a group of neurons, which share the same input and output connections, but do not interconnect with themselves: Connections occur only between layers and not within a layer (Fig.7). Layers are often classified as being input, output or hidden;

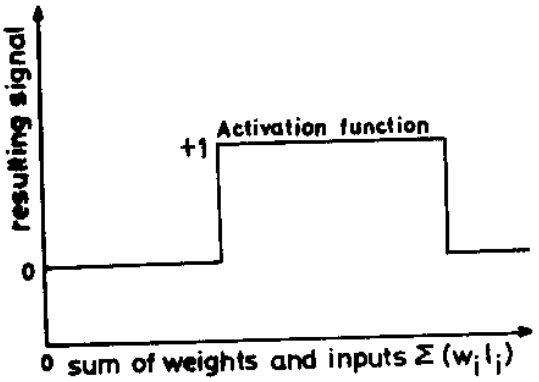


FIG. 3. DOT NEURON ACTIVATION FUNCTION

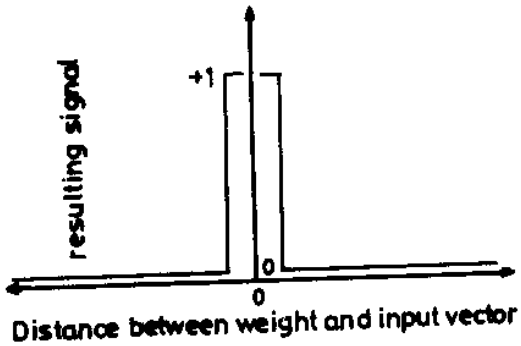


FIG. 4. DISTANCE FUNCTION

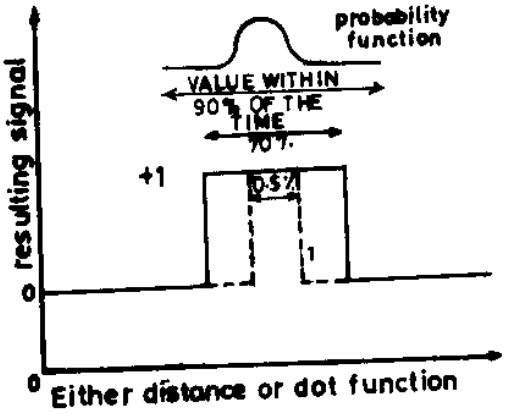


FIG. 5. PROBABILISTIC ACTIVATION FUNCTION

whereby an input layer receives data from the outside world, an output layer returns data to the outside world; and hidden layers perform unknown operations between the input and output layers.

As soon as connections exist within a layer, then reference is made to a cluster of neurons. If within a cluster, lateral inhibition is executed for each individual neuron competition is created (Fig.8). Competition occurs, when all the neurons in a cluster are connected to each other through inhibitory connections (*Rumelhart and Zipser, 1986*). Consequently, neurons in each cluster compete with each other for the right to recognize some feature in the input. The neuron that resembles the input vector the most, wins and yields an output vector, while the other neurons in the cluster are denied any response at all. Eventually, each type of vector presented to the cluster, will cause the response of a different neuron in the cluster. Hence, each cluster in an ANN could classify a certain feature in the input data.

Another important type of connection, which has a large influence on the general behaviour of an ANN, is the feedback connection. A feedback connection directs some or all the data back into the system, thus creating signal loops and cyclic behaviour of the corresponding ANN. According to the literature (*Oppenheim et al, 1983*) a connection is defined as being a feedback connection when the output of a system is used to control or modify the input. Since ANNs consist of numerous I/O neurons, the term feedback will be further clarified in order to avoid confusion.

An external feedback connection directs the current output vector to the current input vector of the ANN; whereby the new vector is re-routed through the neurons (Fig.9). This process is repeated until the output vector shows no significant variations anymore. An internal feedback connection directs the output signal of one neuron to the input of another neuron (Fig.10). Often multiple feedback loops are used between neurons in the same cluster or between different layers.

In general, Feedback networks (also referred to as recurrent networks) are defined as being systems that settle or relax into an output vector. The data will pass through some or all of the neurons more than once. Because the actual state of a network is dependent on its previous states, the same input vector can produce different output vectors. Stability and convergence

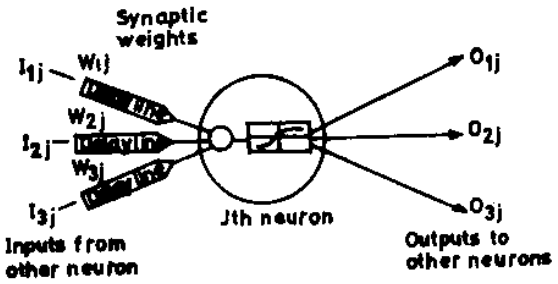


FIG.6. DELAY CONNECTIONS

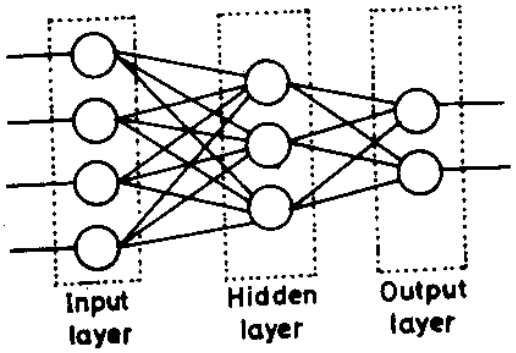


FIG.7. NEURON LAYERS

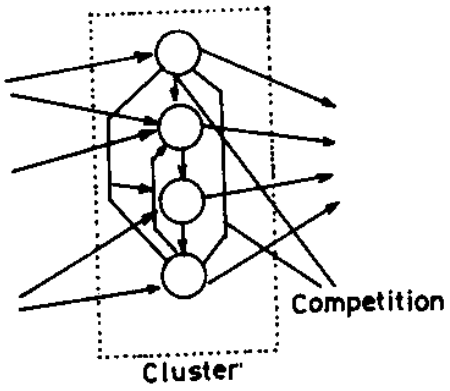


FIG. 8. NEURON CLUSTERS

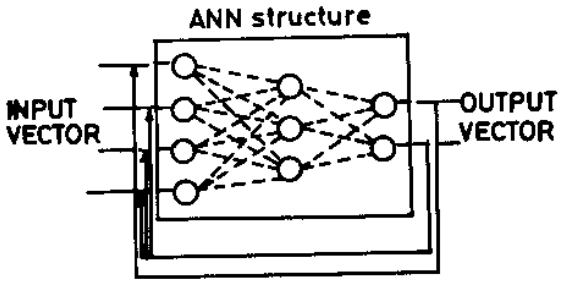


FIG. 9. EXTERNAL FEEDBACK CONNECTIONS

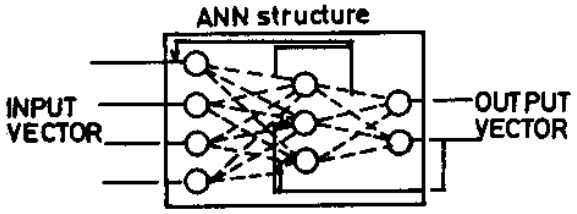


FIG. 10. INTERNAL FEEDBACK CONNECTIONS

characterise the performance of a feedback network. Feedback networks are also referred to as dynamic non-linear systems.

When there is a total lack of feedback connections, one generally speaks of feedforward networks. This means that a given input vector, will always produce one output vector. Once trained (fixed weights), this input vector will always produce the same output vector. Often feedforward networks are referred to as instantaneous static non-linear mapping systems.

2.3 LEARNING ALGORITHMS

The weight distribution in every ANN is unique and will determine the specific response of the network to any given input vector. In order to perform a required process task, these weights must be determined in advance through a learning process. The learning process for ANNs encompasses the adjustment of weights and this process makes use of a learning algorithm and a training set of examples.

The learning process in an ANN can be seen as teaching the network to yield a particular response to a specific input. This often consists of an iterative process; whereby the network tries to match output vectors to desired ones and uses any deviations to adjust some or all of its weights. The rules that determine the magnitude of these adjustments are contained in the learning algorithm.

There are three modes through which the learning process can be carried out: supervised, unsupervised or batch.

In the supervised learning mode, a teacher provides the desired response to the network as soon as an input is applied, thus giving the network an indication how it performs (Fig.11). A child learning the alphabet at school is an example for this type of learning.

In the unsupervised learning mode, the desired response is unknown (Fig.12). Weight adjustments are based on observations of responses to inputs on which there is marginal or no knowledge. Often, this results in self-organisation of neurons, trying to recognise patterns, regularities or separating properties in the given input data. For example, a child learning to ride

a bicycle will do so with minimal help from outside. The child must figure out independently how to find a balance.

In batch learning mode, weights are determined in one go, by using a complete set of I/O vectors (Fig.13). All knowledge must be known a priori and is then implemented instantly in the network. There are no normal incremental learning steps. This method of storing input vectors can be seen as putting data records in a database.

Learning algorithms themselves are often based on error minimisation. Examples are the least mean square (LMS) learning rule or error gradient descent; but numerous other, more refined routines exist, all of which try to optimise some kind of learning signal (learning rate, maximum likelihood value, cross entropy) and so improve network performance. The resulting modifications made to the weights, are then either based on an award/punishment rule (dot product neuron) or chance (probabilistic neurons).

After numerous training cycles, once the ANN has learned the examples with considerable accuracy, test data is presented to the ANN, which it has never encountered before. The resulting outputs are validated and the network performance is tested using multiple criteria such as generalisation ability, robustness, stability, convergence and plasticity. It is only after these results are proven satisfactory, that the ANN is implemented. If test results or performances are unsatisfactory, the network is often retrained using other learning examples, set in a different order, or using more training cycles, etc. Often, for instance, the number of nodes is changed to improve learning; however certain drawbacks to this practice exist.

- When enough nodes are available, the ANN can reproduce any desired response because it stores the information instead of learning the mechanics of the cause/effect relationship of the data. This is called overlifting of the data and as a consequence the ANN will have a poor generalisation ability.

- Too few nodes, insufficient data or incorrect data can lead to underfitting of data, again resulting in bad generalisation abilities.

Special algorithms exist which not only change the weights during learning, but also change the

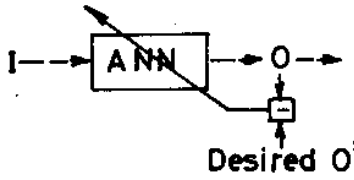


FIG.11. SUPERVISED LEARNING MODE

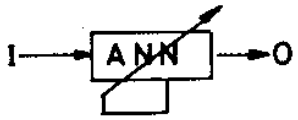


FIG 12 UNSUPERVISED LEARNING MODE

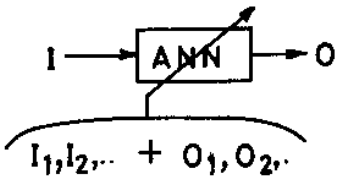


FIG.13. BATCH LEARNING MODE

topology and architecture of the ANN, as done on various levels of interaction. Such an algorithm could, for example, determine weights, network structure and even decide on which training and test examples to use; for instance, a situation could be thought of, where the ANN is confronted with hundreds of rainfall events and the next rain must be predicted.

Finally, some algorithms are not restricted to training use only. On line learning is a powerful characteristic that enables an ANN to adapt temporarily or permanently to changing conditions. Self organising maps, for example, can continue learning with each new input vector they receive. An ANN could be created, for example, which simulates flow through a sewer pipe and adapts its parameters when the resistance of a sewer pipe becomes higher; independent of any intervention from an external source.

2.4 STRUCTURE

In order for an ANN to learn a certain response, it must be provided with numerous examples. The data contained in these examples is crucial for the performance to the network. Incorrect input data will certainly result in slow learning, unstable or unreliable networks. Therefore, the training and test examples should be chosen with care and (pre-) processed accordingly.

The term input vector will be used to refer to the input data needed for one training, test or on-line example (Fig14). Consecutively output vector refers to the final calculated result of the ANN. Each input/output vector has a certain dimension R, representing different features of the data, e.g. 1(1) represents the catchment size, 1(4) is the type of vegetation, etc. When Q different vectors are presented to the ANN randomly or in ordered fashion, we can define a matrix P, which defines a set of Q I/O vectors of dimension R (P = Q x R). In the training of an ANN, which should, for example, determine the runoff coefficient of an urban catchment area, the input matrix P could consist of 100 catchments (Q=100), each representing specified catchment characteristics. The output vectors would then consist of single values, representing the runoff coefficients.

The most important pre-processing actions performed on input data or learning examples, is normalisation, filtering and scaling.

Data is normalised, when outliers are present in the data-series. It reduces the influence of these outliers and assumes that while the overall magnitude of each signal may vary, the relation between each feature may not.

Data is filtered, when unwanted high or low frequency signals perturb the main signal (e.g., high frequent waterlevel fluctuations due to wind-waves). However, ANNs are known to act as filters themselves, making pre-filtering of data only necessary in extreme cases.

Scaling of data is performed to increase training speed only. It is common practice to scale different data series to a uniform range [0 1]. For example, when the degree of pollution for a surface water sample is determined by using the concentrations of nitrate and benzene; the smaller amounts of benzene will give a better indication of the degree of pollution than the larger nitrate concentrations. If data is not scaled, the learning procedure will initially be dominated by the (in absolute terms) larger nitrate values instead of the smaller benzene values.

Vertical processing (Fig.15) is carried out to remove the influence of differences between the varying dynamic ranges for each feature in the input series. This ensures that features will not dominate due to their range. It is performed on one feature in an I/O vector and covers the whole series (i.e. 1xQ matrix).

Horizontal processing (Fig.15) is commonly applied when the influence between varying I/O vectors should be avoided. Each I/O vector (i.e. Rx1 matrix) can be represented in a R-dimensional vector space. By doing this, the direction of the I/O vector is maintained, while its magnitude is scaled to a uniform value (eg. One). For example, in determining the runoff coefficient, each catchment (input vector) is treated with equal importance.

With regard to both normalisation and scaling, it should be taken into account, that it is possible that two initially independent vectors are no longer distinguishable afterwards. For instance two vectors, such as (3,3) and (6,6) both become a (1,1) vector.

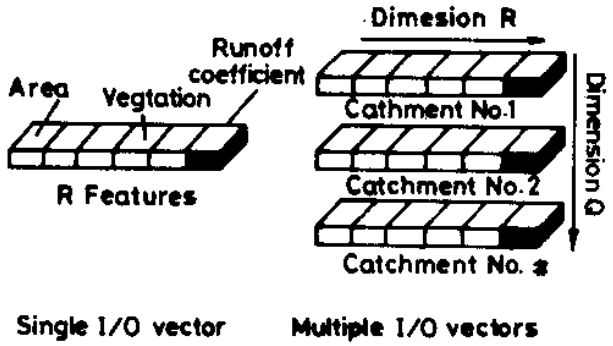


FIG.14. INPUT VECTORS AND THEIR DIMENSIONS

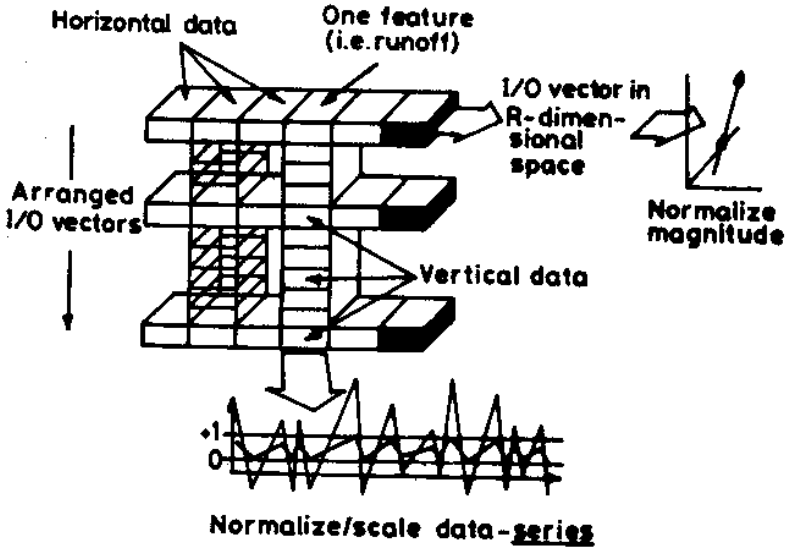


FIG.15. VECTICAL AND HORIZONTAL DATA PROCESSING

2.5 ARCHITECTURE

The architecture of an ANN describes the layout of its structure. It defines the number and size of the implemented clusters and layers, as well as the topology used to connect these groups of neurons with each other. The ANN architecture itself is often changed when learning algorithms and I/O data modifications fail to improve the ANN performance. Simple modification can be made by reducing or increasing the number of nodes or deleting neuron interconnections. Different techniques exist for determining an optimal ANN architecture for a given I/O data problem (*Refenes and Vithlani, 1991*) . In addition to implementation of improved neuron functions, learning algorithms or determination of the optimal number of nodes, modularisation can be applied (Fig.16). Modularisation is implemented when one specific type of ANNs for different tasks within a system makes best use of the specialised capabilities of each of the independent ANN modules (*Nadi, 1991*). For example, the vast amount of real time flow data in a complex sewer system, could be compressed to several abstract parameters first (using a self organising ANN), before it is fed into another ANN which simulates the actual outflow of the sewer system.

Considered at a more strategic level, the boundaries of ANN research can be surpassed. Though the combination of classical methods with ANNs, hybrid models are created (Fig.17). A problem can be decomposed into several modules where initialisation, training and calibration are done separately; after this, a global optimisation routine could link the modules and produce an even more optimal result. It is known, for example, that adaptive PID controllers are very effective for local control purposes (gate control); ANNs could simulate their performance, but they would never improve on it. Consequently, it is far more efficient to develop an ANN for a data processing task and then combine it with the PID controllers.

2.6 PERFORMANCE INDICATORS

During learning or network optimisation, it is often necessary to monitor the effects of a certain intervention or system alteration. Numerous performance indicators exist to quantify progression or drawbacks of certain methods:

- Generalisation is the ability of the ANN to formulate an answer to a problem it has never

seen before (predict a future flow rate using on-line data).

- Fault tolerance is the ability to keep processing, albeit with reduced accuracy and/or speed, even though data is missing or neurons have been disabled/destroyed (A waterlevel meter fails).

- Dynamic stability is the ability of a network to remain within its functional boundaries and reach a stable state (Find the best control strategy out of 10 predetermined cases).

- Convergence speed is the rate at which the network state changes as it moves to a stable state.

- The states of an ANN can be mathematically expressed in a function, representing a 3 dimensional surface of the computational energy of the ANN (Energy surface, Fig.18). Computational energy describes the stable states or solutions of an ANN and the paths leading to them. These stable states are represented as valleys (energy minima) in the 3 dimensional surface, also called basins of attraction. By changing the weights, this energy surface is changed and the valleys get larger and deeper, increasing the convergence speed of an ANN.

- Adaptability is the ability of an ANN to modify its response to changing conditions. Four characteristics govern this ability: learning, self organisation, generalisation and training.

- Plasticity is the ability of a group of neurons to adapt their functions to different needs over time (Simulate treatment plant inflow during day, night, dry or wet periods).

- Reliability is the ability to produce the same result, when the same input vector is repeatedly presented to a network. Reliability is mostly used to describe the performance of feedback networks, since feedforward networks always produce the same result.

- Robustness is the ability to produce the same result, even though input data is noisy, contains data gaps and contradictory data.

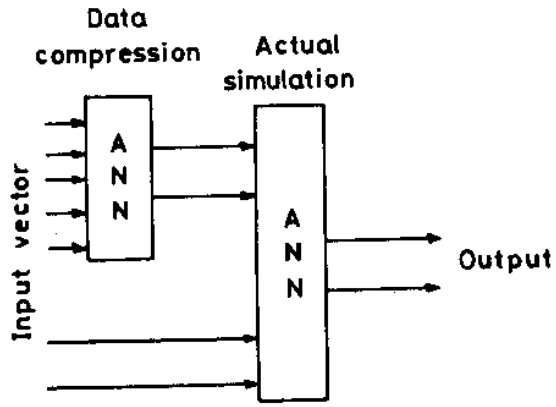


FIG. 16. MODULARISATION OF ANN MODELS

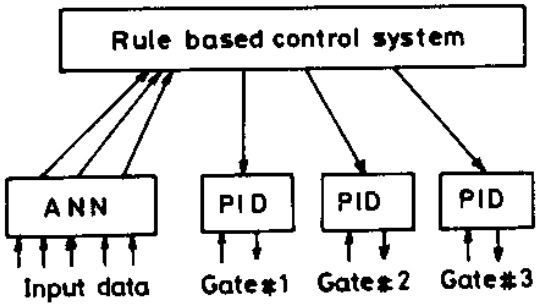


FIG. 17. HYBRID MODEL STRUCTURE

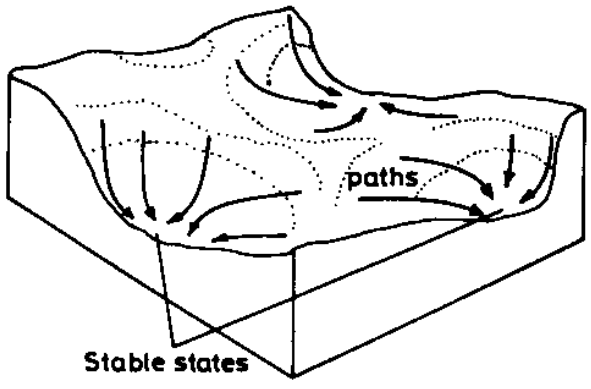


FIG. 18. COMPUTATIONAL ENERGY SURFACE

- Sensitivity is closely related to robustness, it shows the extent to which a network response will change, due to variations in the features of the input vector (*Fu and Chen, 1993*).

Three other less obvious indicators that can also be used are:

- Memory requirements: Some ANNs need less (hardware) memory to perform their task, than others. This can be decisive for data base related problems that use pattern storage for instance (Hopfield network).

- Amount of training data needed: for some problems, lack of data poses a constant hindrance to efficient modelling. Thus an ANN requiring less training data to learn a specific response, is better suited for these kinds of problems .

- Learning speed: The speed at which new data is learned, can be crucial for on-line applications in fast changing environments. This performance indicator is often used to evaluate new learning algorithms.

With these performance indicators, it is possible to make an evaluation table, showing the different ANN types and their relative performances.

2.7 LEVELS OF INTERACTION

The highly interconnected architecture of an ANN causes data to flow back and forth through the network. Numerous data interaction mechanisms modify this information continuously, enabling complex data processing functions. In order to gain better insight into this process of change and alteration and thus to be able to intervene and optimise network performance, these mechanisms are classified into levels of interaction. The different types of data interactions are shown below, starting from the smallest element and ending at global problem optimisation.

***1st level**: the basic neuron element (Fig.19)*

Here the activation and transfer function can be chosen. The main neuron types are the probabilistic, distance, and the dot product neuron. Numerous transfer functions exist, with the

sigmoid and threshold function being the most common.

***2nd level**: the local topology of each neuron (Fig.20)*

How is the neuron locally connected (inter-layer, cluster) ? What is the number and type of the I/0 connections (inhibitory, excitatory or delayed)?

***3rd level**: feedback loops (Fig.21)*

The ANN can become feedback (i.e. recurrent) by using data flows which go back into the system. Feedback links can be internal (between neurons) or external (between complete I/O vectors).

***4th level**: learning (Fig.22)*

Learning can be supervised, unsupervised or batch processed. Which training examples are used, what features are considered important? Which learning algorithms are used and what are the learning rate indicators? After the training examples are learned, the network is considered trained.

***5th level**: testing (Fig.23)*

After learning the weights are frozen and the ANN is exposed to new test data. Which test examples are used and what are the performance indicators?

***6th level**: Network optimisation (Fig.24)*

With the test results, the specific ANN type can be optimised further. Which rules or algorithms are used to optimise topology and architecture?

***7th level**: Global optimisation (Fig.25)*

Apart from optimising a single ANN type, the problem can be sub-divided into different

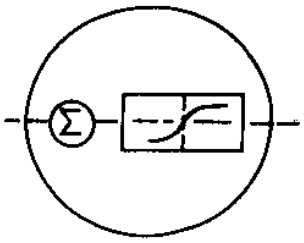


FIG.19. THE NEURON

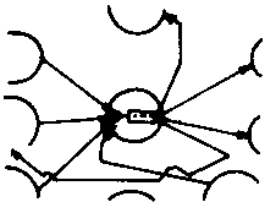


FIG.20. LOCAL TOPOLOGY

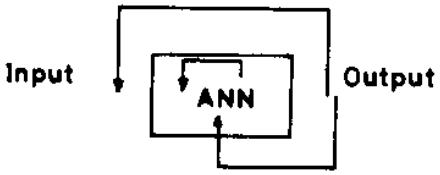


FIG.21. FEEDBACK LOOPS

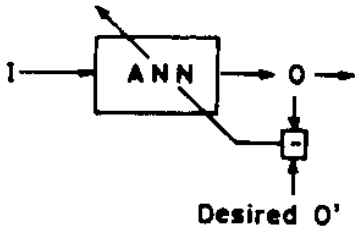


FIG. 22. LEARNING MODE

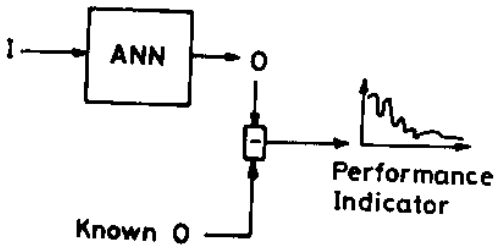


FIG. 23. TESTING THE ANN

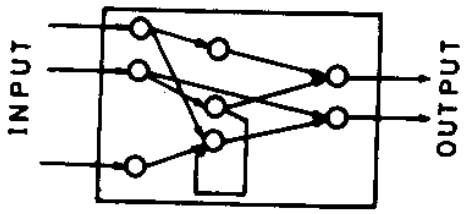


FIG. 24. STRUCTURE OPTIMIZATION

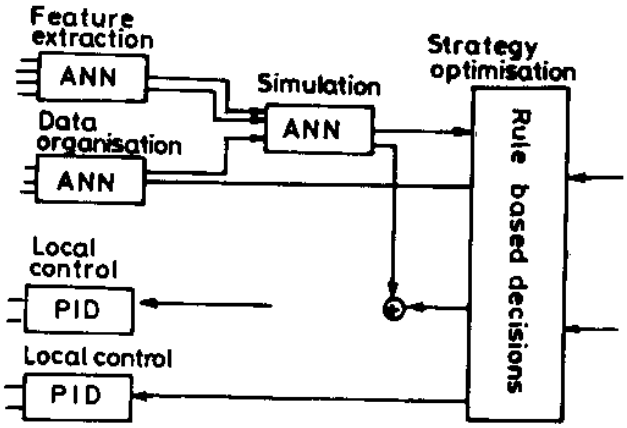


FIG. 25. GLOBAL OPTIMIZATION

modules, implementing modularisation or hybridisation. What are the global optimisation rules.

***Special level:** Adaptability*

A special case of interaction is formed by adaptive processes. Adaptability is a very confusing term as it can be applied to every level of the previous list of interactions. A normal definition of adaptability is given earlier : Adaptability is the ability to modify a response to changing conditions. However, all ANNs are equivalent to this description and a better definition is necessary (especially for the term conditions).

It should be noted that adaptation could be applied before and after network implementation (i.e. adaptation during training or adaptation during operation).

In this report, the term adaptable ANNs is only used to refer to already trained networks, which modify their weights, topology or architecture during on-line operation. Thus, adaptability is the ability of an ANN to modify itself after it has been implemented (i.e. under operational conditions).

2.8 CLASSIFICATION OF ANNS

The diversity of the characterising elements of ANNs already indicates that there are numerous ways to classify artificial neural networks. For instance, classification is possible in terms of the neuron-types used; or in terms of topology, architecture or learning algorithms implemented. The classification presented here will only focus on the used learning and recall mode of the ANN.

The recall mode specifies how an ANN responds to new inputs. Either it uses feedback links or not, i.e., input data is modified by output responses. When feedback loops are absent, the network is classified as being feedforward. A feedforward network is transparent, fast and reacts only to its present input; thus, it is independent of previous network states that and can be seen as having no memory. If feedback loops exist, then the network is classified as being a feedback network (Also known as a recurrent network). A feedback network uses some time to converge into a stable solution. This dynamic system is dependent on its previous network states and therefore can be said to have memory.

The learning mode specifies whether an ANN uses supervised, unsupervised or batch learning.

During supervised learning, the desired responses to the network are taught via an external source. Unsupervised learning is used when the desired responses are not known; hence the network must determine independently how to use the input data. Batch learning is a special process, whereby all the information is known a priori and the weights are fixed in advance. Many ANN memories use this last procedure (Associative memories).

All important ANNs can be categorised using the above criteria in a classification tree (see Fig.26). This same classification can be visualised in quadrants (*Kosko, 1990b*; Fig.27), where the upper left corner area depicts the most transparent but care intensive networks; and where the lower right corner area depicts unclear networks which determine independently how to classify or associate received data.

The same classification tree can also be applied solely to adaptable ANNs; since each normal ANN type could theoretically be modified for adaptive operation. As stated earlier, this report refers only to adaptability after the learning phase, i.e. during operation. Real Time Recurrent Learning (RTRL) is an example of an adaptive feeback, supervised back propagation network.

2.9 VARIOUS ANN NETWORKS

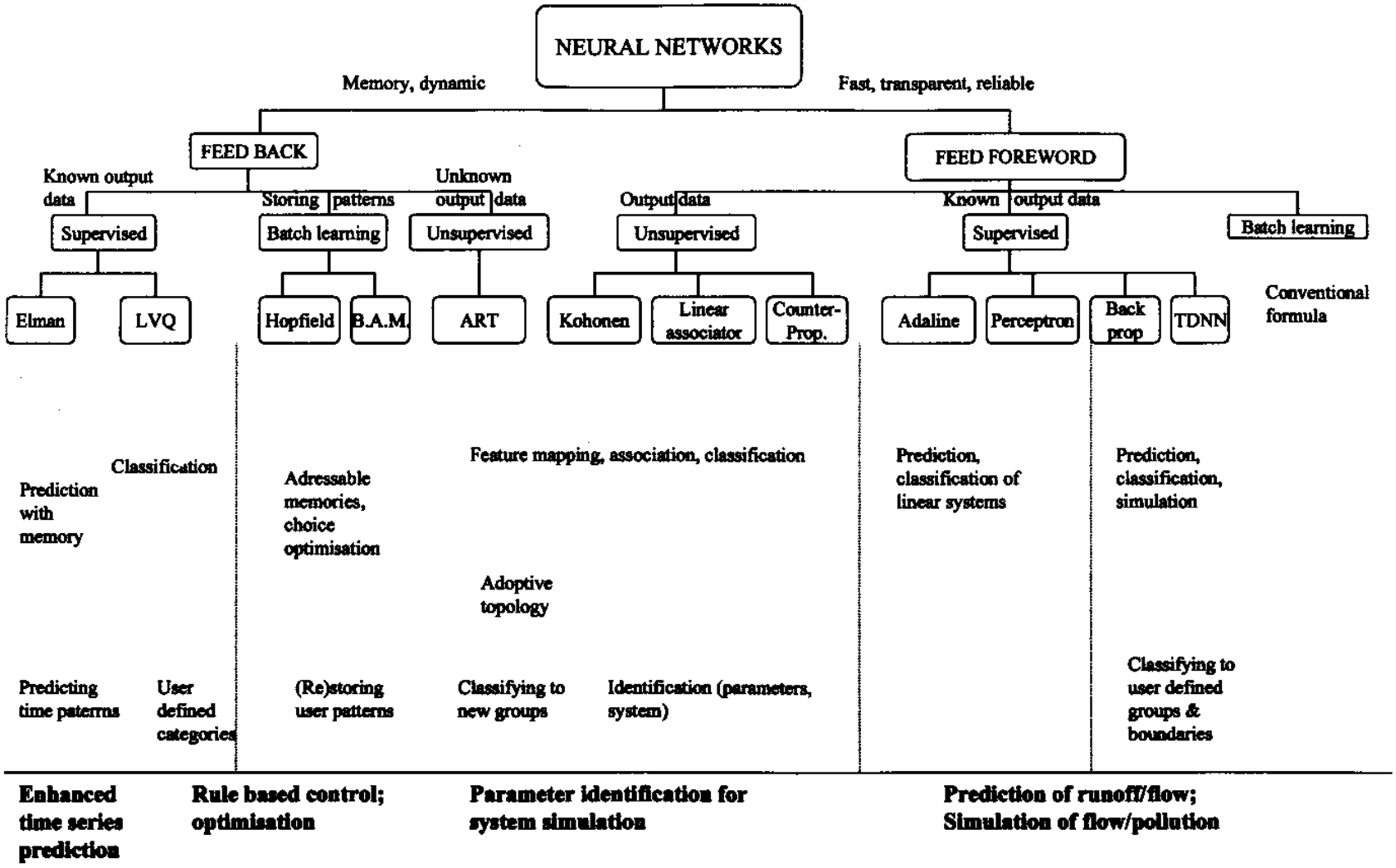
Some of the most important networks are explained below:

2.9.1 Back Propagation Network (BP)

Back Propagation network (BP) are the most widely used ANNs. The name comes from the fact that an error term is back propagated through the network during learning and used to change the weights (Fig.28). However, no feedback links are actually incorporated and there are many other ANNs which also back propagate error terms; so this (historical) name can be confusing.

Normal BP networks have simple supervised feedforward structures and often consist of an input and an output layer with one or more hidden layers in between. They are fast, relatively simple to train and the most easy to understand. Theoretically any recurrent ANN can be simulated by

Classification of Artificial Neural Networks



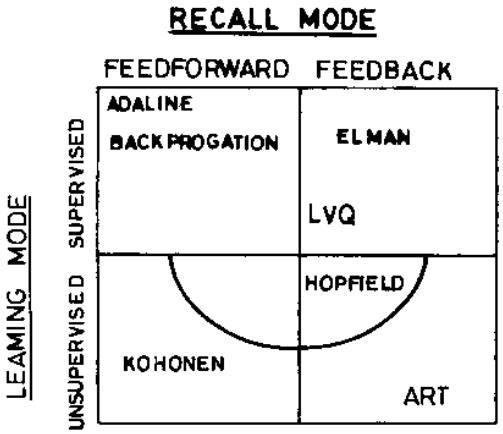


FIG.27. CLASSIFICATION BY KOSKO

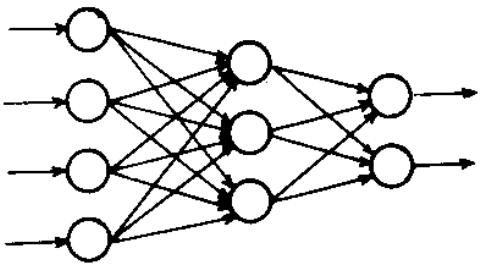


FIG.28. A BACK PROPAGATION NETWORK

a back propagation algorithm (Such as a Fourier series approximation). As BP networks suffer from learning deficiencies, like slow learning and convergence to local minima, numerous enhancements have been proposed (*Haario and Jokinen, 1991, Sato, 1991, Yu et al., 1993*). Nevertheless, BP networks are used in 80% of today's applications and excellent in the areas of prediction and simulation.

2.9.2 Adaline Network

Adaline networks were one of the earliest ANNs. They consist of single neuron elements employing only linear functions and a simple Least Mean Square (LMS) learning rule (Fig.29). This makes them suited for simple classifications and restricted non-linear system simulation.

2.9.3 Kohonen Network

The Kohonen network (*Kohonen, 1988*) was one of the earliest unsupervised feedfoward networks; being able to self organise its neuron weights (Fig.30). The network maps input data into a 2 dimensional grid of neurons with a special distance learning algorithm. The result is a surface area that shows peaks at different areas for different input vectors. This network was originally designed for speech mapping and recognition.

2.9.4 Hopfield Network

The Hopfield network is an example of a batch learning feedback network. A given set of known vectors can be stored in the network by using a special formula to determine the weights. After that, any input vector will slowly converge to the nearest stored pattern (Fig.31). These types of networks are referred to as associative memories; and relate an input to some stored patter; numerous variations have been invented (BAM, CAM). Hopfield networks are used for database managing, image restoration and other addressable memory problems.

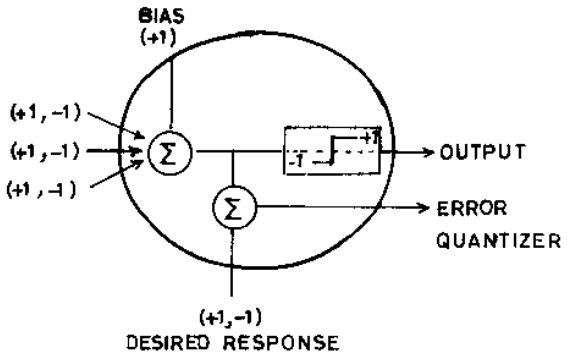
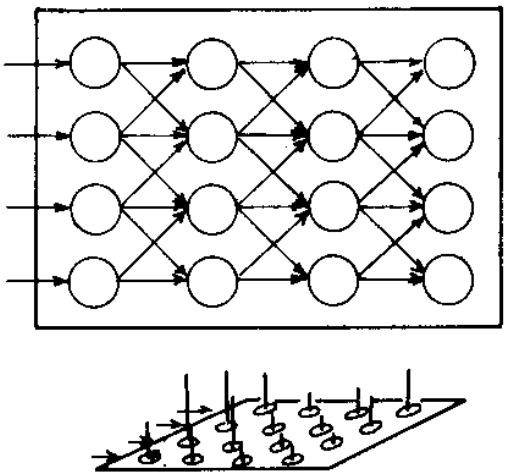


FIG.29. A DAPTIVE LINEAR ELEMENT



FEATURE MAP

FIG. 30. KOHONEN NETWORK

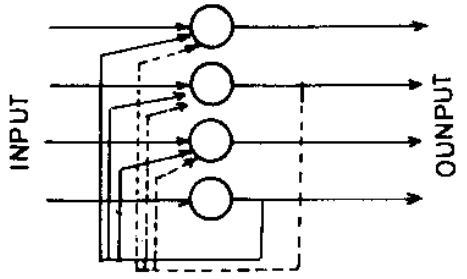


FIG. 31. A HOPFIELD NETWORK

2.9.5 Adaptive Resonance Theory Network (ART)

ART or adaptive resonance theory network is an unsupervised feedback network, which has a complicated, changing structure (*Rusell, 1991*). It uses competitive neurons in self organising and self stabilising clusters, which classify input vectors to self defined groups (Fig.32). When an input vector is discriminating enough, the network is able to define a new classification group and thus store a new pattern.

2.9.6 Linear Vector Quantization Network (LVQ)

Linear Vector Quantization (LVQ) network is a supervised feedback network. It is a method whereby supervision and competitive layers are combined. The competitive layer finds subclasses among then input data; and these are then classified into user defined target classes (Fig.33). In contrast to back propagation classification networks, LVQ can also classify non-linearly separable sets of vectors.

2.9.7 Elman Network

Elman are two hidden layer back propagation networks, with the addition of a feedback connection from the output of the first hidden layer to its input (Fig. 34). This feedback path allows Elman networks to learn to recognise and generate temporal patterns as well as spatial patterns (*Elman, 1990*).

An evaluation table can be made for consecutively different ANN types and their relative performances. The indicators used are described earlier and give here an indication of the expected behaviour of the ANNs. This table offers a fast reference in the process of determining a suitable ANN.

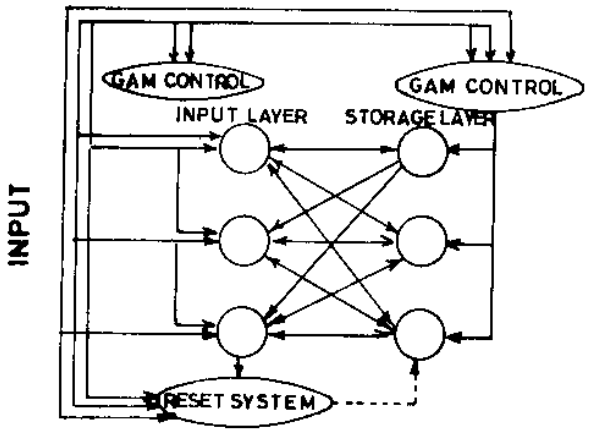


FIG. 32. AN ADAPTIVE RESONANCE THEORY NETWORK

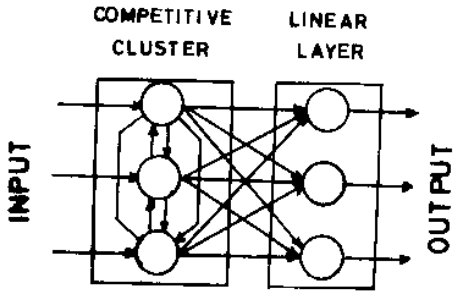


FIG. 33. A LINEAR VECTOR QUANTIZATION NETWORK

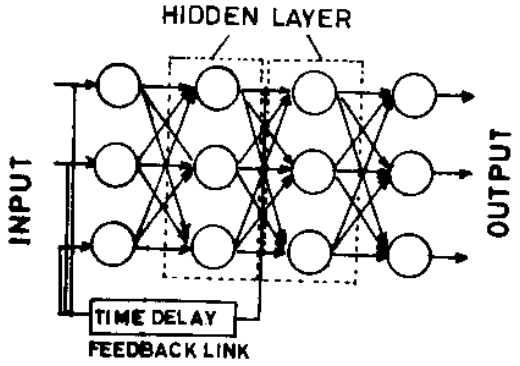


FIG. 34. AN ELMAN NETWORK

Table 1: Evaluation of different types of ANNs

	BP	Adaline	Kohonen	ART	Hopfield	LVQ	Elman
Generalisation	++	+	(-)	(+)	(--)	+++	+++
Fault tolerance	+	-	(+)	++	(-)	(+)	(+)
Dynamic stability	+++	+++	+	++	--	+	++
Convergence speed	+++	+++	-	-	---	-	+
Adaptability	---	---	+	+++	---	+	--
Plasticity	---	---	++	+++	---	++	---
Reliability	+++	+++	-	(-)	--	+	++
Robustness	+	-	++	(+++)	+++	+++	++
Required memory	++	+++	-	-	---	-	+
Required data (- a lot)	---	--	---	+	+	--	---
Learning speed (- slow)	--	--	+	+	+++	+	-

(+ means good performance; brackets indicate uncertainty)

2.10 PRACTICAL ISSUES OF USING ANN FOR ENGINEERING APPLICATIONS

Where processes to be modelled are complex enough to be described mathematically, neural networks are considered to outperform the conventional, deterministic models most of the time. However, one should be aware of the applicability of neural networks to a specific problem and the basic conditions for getting the best performance out of it. In many cases neural networks for research are used 'blindly' by choosing all the possible input variables and without considering much of the possibilities to maximize the performance.

This section provides some practical information of taking the maximum advantage of the artificial neural network models. Mainly the section is based on Swingler (1996).

2.10.1 *Analysing the problem*

In general, neural networks are suitable for problems where the underlying process is not known in detail and the solution can be learned form the input-output data set. Nevertheless, the following points have to be stressed:

1) It has to be made sure that the problem is difficult to be solved by conventional method and neural network can be used as a good alternative.

2) If there are logical non-chaotic relationships or structural properties that similar initial configurations indicate mapping to the similar solutions, one can expect a generalization by neural network. It simply means, the same input should always result in the same output.

3) If the data set to train the network is impossible to be represented or coded numerically, the problem cannot be solved by a neural network approach.

4) Non-linearity and the change of variables in time are possible to be dealt with neural networks.

2.10.2 *Data preparation and analysis*

This is one of the most important stages of neural network application because the accuracy of solution for most of the networks depend on the quality and quantity of training data set. Although neural networks can accept a wide range of inputs, they work with data of certain format encoded numerically. There are two main issues in data preparation:

- The number of variables to be used, which determines the dimensionality

- Explicitness or data resolution and in what extent and amount the data has to be presented to the network

To avoid of analysis of large amount of data, a sample data set may be used by choosing it randomly from the complete database. For input and as well for output variables the data must

be analysed and prepared with the following sequence, which is sometimes called as data pre-processing:

1) Determining the data type (discrete or continuous)

2) Data generation: The data to train the network can be generated by measurement, by simulation of relevant models or by derivation of virtual examples by introducing noise into the existing data set. Also it is good if the data set evenly covers the input data subspace. In other words the data has to be normally distributed.

3) Calculations of simple statistics such as mean, standard deviation for continuous data and the number of different events for discrete data.

4) Removal of outliers: Outliers mean the data points lay outside of two standard deviations from the mean. Two standard deviations cover 95% of normally distributed data. If such data example exist, those are preferably to be removed, unless those are significantly important for the given problem. For some of the dynamic systems (chaotic) those outliers are important.

5) Quality and quantity check: What amount of data has to be collected is mainly decided by the network size (number of variables), required data resolution etc. Concerning the network size it is advisable to collect training data set of equal number to *(1/target error) x number of weights*. Also as a quality check statistical tests can be carried out in order to make sure the corresponding data set contains a required information.

6) Dimensionality reduction: Large number of input variables increases the training time considerably. It is advisable to reduce the number of input variables, which are the most important and best representing the output variable while maintaining the correct level of network complexity. The covariance or correlation between the variables can help to decide which variable is the most useful.

7) Data scaling has to be done when data set has too different order of magnitudes. It is also advisable to have all the input data within the same range of scaling.

8) Data encoding has to be done in the end of data preprocessing in case of necessity. Categorical data must always be encoded.

2.10.3 *Model selection and building*

Because of its accuracy and fault tolerance capability error backpropagation network is the mostly used type of neural network. However, there are different types of learning algorithms that are quite suitable for specific problems. By using the classification tree and the characteristics of the problem a specific ANN type can be chosen. For example, the simulation should be fast for on line operation (feedforward), there is enough data available on the process (supervised) and much of the data is time related; thus a Time Delay Neural Network (TDNN) network could be used.

In case of too many input and output variables, the training of the network become computationally demanding. Therefore, one way to solve this sort of problem is to divide the problem into several small sub-problems that can be solved separately by the network. There is no specific rule for building a network, however, some practical hints on this aspect is listed below.

2.10.3.1 *Network structure approximation*

Multi-layer Perceptron

- The training examples should be at least equal to 1/e , where e is a target value for error
- The maximum number of hidden units should be guided by the formula $h \pounds 2i+1$, where h and i are number of hidden and input units respectively
- Number of weights can be related to the number of training patterns $w=i\log_2 p$, where p and w are number of training patterns (exemplars) and number of weights respectively
- For feature extraction, number of hidden nodes should be less than the number of input

variables

- For classification, the number of hidden units is increased with the number of expected classes

- Number of hidden layers should be as less as possible and usually one or two layers are used in most of the published applications. It was shown that any function could be approximated by at most 4 hidden layers (Swingler, 1996).

- It is suggested that the activation function in the specific neuron has to be chosen as non-linear for non-linear process model. In case of more than one hidden layers, the activation function for one of the layers has to be linear, in order to discard the linear components that may be existing in non-liner models.

Self Organising Feature Maps

SOFM consists of the input layer and the output map. Concerning the size of the network following rule is mainly suggested:

$2N_{class} < N_{units} << N_{pattern}$

N_{class} - the number of expected class or cluster (user must have some primary expectation)

N_{units} - Number of processing units in the output layer

$N_{pattern}$ - Number of input pattern

2.10.4 *Training and testing the network*

Training is the learning process of neural networks. Training stage can be started when the network is designed, data sets are collected and encoded. After the training is fulfilled, the testing phase starts. Various phases are:

- Defining the topology: Training the network has to be started by defining the topology of the neural network. The best topology is found by adjusting the parameters by trial and error, therefore it is better to start with a small network which learns fast and is easy to change the parameters. Initial weights are also defined by trial and error method. When the appropriate network topology is defined, it is possible to speed up or slow down the process by changing the learning rate and make fine-tuning.

- Stopping criterion: There is no specific rule signifying when to stop the training and the stopping criteria are different for different type of the network. In general, the stopping criterion can be the minimum value for a learning rate or if applicable, it is also advisable to use target value of training error and also an option of cross validation, which allows you to know whether the error in verification phase starts to increase.

- Testing: Once the network is considered to be sufficiently trained, the network needs to be tested under realistic circumstances. However testing is not necessarily applicable for every type of neural networks. The final integration or implementation of the neural network has to be delayed till sufficient confidence is achieved, so that the network can work by all means and without damaging the system in case of its failure.

2.10.5 *Output and Error Analysis*

Errors do not always mean the network parameters are chosen wrong. If the network is built and organised systematically then the reason for large error can be found by changing few parameters by small amount between the two configurations. But if the problem cannot be found, the reason is not in the initial configurations. Sometimes errors or unsuccessful results can not be simply termed as errors as they might be caused by uncertainty.

Error criteria can be a maximum net output error, which is the difference between the net output and the target output. Average error and moreover, the total distribution of error are good performance criteria. If network makes error in some cases and not in others, it means the data balance is proper. In order to prevent the accumulation of rounding error, training the network

with too large or too small numbers should be avoided.

For noisy and overlapping classification problems it is impossible to get zero error level. Function mapping is more difficult than the classification problem and the different parts of the input space can give a different degree of error. Certain parts of a data set can represent non-linearity function that can be learned easily. Also there can be more variance in training data set than in the other parts. In short, the inadequate generalization usually is caused by too complex non-linearity of the function, too high variance in data set.

There are different ways to evaluate and analyse the network output error. One easy way to analyze the output error is to calculate simple statistics of the network output such as the correlation coefficient between the net output and the target output. One more advanced method to evaluate and minimize the error is finding the structure of error, which could determine the areas with different level of predictability. Even the structure can be the function describing the distribution of error in the input space. This goal can be achieved by training the second neural network to learn and predict the error of the original model after its being trained at certain level (for details see Swingler, 1996).

2.10.6 *Implementation of a neural network based project*

This is the step to build the real product from the neural network prototype. Implementing the neural network is the part of the software we need. It includes special requirements such as time or space restrictions, porting the neural network solution to an application environment and interface development etc. Most of the time the neural network project can be easier than to the rule-based approach as the domain specific knowledge is not much required for neural networks. In terms of risk involved in neural network project, the main risk would be the non-presence of the information necessary for the problem in the data set available.

There are three general steps in implementing a neural network based project, each of which consists of small substeps:

- Project planning stage (task definition, feasibility study, input/output specification, defining

data requirement, data coding)

- Network development stage (data collection and validation, data encoding/recoding, network design and training, network testing and error analysis, implementation)

- Documentation stage (defining data source and conditions, defining the coding method, architecture, parameter setting, the number of training epochs, defining the conditions, reporting the final results).

3.0 EXISTING NEURAL NETWORKS IN SURFACE WATER HYDROLOGY

3.1 Rainfall Runoff Modelling

French et al. (1992) have developed a neural network to forecast rainfall intensity fields in space and time; it is a three layer learning network with input, hidden, and output layers. Training is conducted using back propagation where the input and output rainfall fields are presented to the neural network as a series of learning sets. After training is complete, the neural network is used to forecast rainfall intensity fields with a lead time of 1 hour using only the current field as input. Rainfall fields are generated using a space time mathematical rainfall simulation model, and forecasted fields are compared with the perfectly known model produced fields. Results indicate that a neural network is capable of learning the complex relationship describing the space-time evolution of rainfall such as that inherent in a complex rainfall simulation model. One hour ahead forecasts is produced, and comparisons with true mean areal intensities and percent area coverage indicate that in most cases the method performs well when applied to the events used in training. The neural network is used to forecast a series of events not included in the training data and is shown to perform well when a relatively large number of hidden nodes are utilised. performance of the neural network is compared with two other methods of short-term forecasting and persistence.

Hsu et al (1995) have presented a new procedure (entitled linear least square simplex, or LLSSIM) for identifying the structure and parameters of three layer feed forward ANN models and demonstrates the potential of such models for simulating the non-linear hydrologic behaviour of watersheds. The non-linear ANN model approach is shown to provide a better representation of the rainfall-runoff relationship of the medium size Leaf River basin near Colins, Mississippi, than the linear ARMAX (autoregressive moving average with exogenous inputs) time series approach or the conceptual SAC-AMA (Sacramento soil moisture accounting) model. Because the ANN approach presented here does not provide models that have physically realistic components and parameters, it is by no means a substitute for conceptual watershed modelling. However, the ANN approach does provide a viable and effective alternative to the ARMAX time series

approach for developing input output simulation and forecasting models in situations that do not require modelling of the internal structure of the watershed.

To obtain riverflow data, a neural network is developed and applied to rainfall-runoff transformation by *Lorrai and Sechi (1995)*. The NN has been built considering a hidden two layer net and the sigmoidal has been used as a response function. Training is conducted using a back-propagation learning rule. In the input layer, both areal and point data values may be considered. The capability to provide a suitable forecast of river runoff has been examined for the Araxist watershed in Sardinia. Experiments have been made dividing the total extension of observed data into three ten-year periods, assuming each as a training set, learning the NN and simulating the other two decades over the same period. The obtained model efficiency confirms the capability of this approach to be a useful tool in the evaluation of rainfall-runoff transformations.

Spatially distributed rainfall patterns can be detected using a variety of remote-sensing techniques ranging from weather radar to various satellite-based sensors. Conversion of the remote-sensed signal into rainfall rates, and hence into runoff for a given river basin, is a complex and difficult process using traditional approaches. Neural-network models hold the possibility of circumventing these difficulties by training the network to map rainfall patterns into various measures of runoff that may be of interest. To investigate the potential of this approach, *Smith and Eli (1995)* have used a very simple 5 x 5 grid cell synthetic watershed used to generate runoff from stochastically generated rainfall patterns. A back propagation neural network is trained to predict the peak discharge and the time of peak resulting from a single rainfall pattern. Additionally, the neural network is trained to map a time, series of three rainfall patterns into a continuum of discharges over future time by using a discrete Fourier series fit to the runoff hydrograph.
A critical problem in the estimation of rainfall rate (RR) from satellite infrared (IR) imagery is that the non-linear relationship between the IR brightness temperature (T) and RR varies regionally. To provide accurate estimation of RR, a model must be able to detect such variations and adjust its behaviour accordingly *Hsu et al (1996)* have used a kind of artificial neural network (ANN), called a Modified Counter Propagation Network (MCPN) to model the complex non-linear IR-RR relationship. The ability of this model to adapt to regional variations is illustrated using data from Japan and Florida.

A series of numerical experiments by *Minns & Hall (1996)* in which flow data were generated from synthetic storm sequences routed through a conceptual hydrological model consisting of a single non-linear reservoir, have demonstrated the closeness of fit that can be achieved to such data sets using Artificial Neural Networks (ANNs). The application of different standardisation factors to both training and verification sequences has underlined the importance of such factors to network performance. Trials with both one and two hidden layers in the ANN have shown that, although improved performances are achieved with the extra hidden layer, the additional computational effort does not appear justified for data sets exhibiting the degree of non-linear behaviour typical of rainfall and flow sequences from many catchment areas.

The lumped daily rainfall runoff process for the Leaf River Basin in Mississippi was modelled by *Hsu et al (1997 a,b)* using two different Artificial Neural Network (ANN) model structures. Their results indicate that both structures, the popular Three Layer Feedforward neural Network (TLFNN) and the Recurrent Neural Network (RNN), perform well. However, the TLFNN requires trial and error testing to identify the appropriate number of time delayed input variables to the model. Further, it is not suitable for distributed watershed modelling, i.e., when distributed precipitation information (multiple gages or radar images) is available. The RNN structure provides a representation of the dynamic internal feedbacks loops in the system, thereby eliminating the need for lagged inputs and resulting in a reduction in the number of network weights (and hence training time). The RNN models are far more suitable for distributed watershed modelling.

Jain and Chalisgaokar (1997) have applied an ANN model to rainfall runoff simulation of an Indian catchment. Hourly rainfall, discharge and potential evaporation data were used. The results show acceptable match between the observed and computed discharges.

Dawson and Wilby (1998) have applied Artificial Neural Networks to flow forecasting in two flood prone UK catchments using realtime hydrometric data. The results have shown that given relatively brief calibration data sets it was possible to construct robust models of 15 min flows with six hour lead times for the Rivers Amber and Mole. Comparisons were made between the performance of the ANN and those of conventional flood forecasting systems. The results obtained for validation forecasts were of comparable quality to those obtained from operational

systems for the River Amber. The ability of the ANN to cope with missing data and to learn from the event currently being forecast in real time makes it an appealing alternative to conventional lumped or semi distributed flood forecasting models. However, further research is required to determine the optimum ANN training period for a given catchment, season and hydrological contexts.

3.2 FORECASTING

The various techniques currently used to retrieve snow parameters from microwave measurements are usually not robust enough to give consistent results from very different snow conditions. A new technique has been developed by *Chang and Tsang (1992)* where the inversion of snow water equivalent (SWE) from passive microwave remote sensing measurements is accomplished by using a neural network trained with a dense media multiple scattering models. Brightness temperatures from 19 GHz vertical and horizontal polarisation, 22 GHz vertical polarisation and 37 GHz vertical and horizontal polarisation which are available from the Special Sensor Microwave Imager sensors, are used as input to the neural network. Different combinations of three input parameters are used: the mean grain size of ice crystals in the snowpack assuming a Rayleigh size distribution, snow density, and snow depth. A model atmosphere is then superimposed onto the calculated emerging microwave radiations from the snow surface. Backpropagation multiple layered neural networks were used to estimate the SWE of a snowpack using simulated data and the technique of explicit inversion. When using the entire simulated data set (720 cases), results from using neural networks seemed to reproduce the simulated data set better than when using the multiple regression method. By reducing the size of the training set, the accuracy of the SWE retrievals using the neural networks decreased rapidly. However, reducing the size of the training data set did not affect the results of the regression method. The percentage error for estimated SWE varied from 9% to 57% for different snow conditions.

Alberotanza and Pavanati (1992) have reported the use of an adaptive resonance neural networks scheme to analyse multispectral remotely sensed images on a lagoon environment and coastal waters. An attempt was made to develop a substantially unsupervised classifier, obtaining an algorithm capable of independently organising the learning stage on the basis of the same data under analysis. Information is requested from the external operator to limit the number of classes

to be recognised. It is not necessary to use models for samples of spectral signature. The experience acquired can be transferred to successive recognitions. The method makes use of the separation of colour information from that of the intensity of the analysed radiation and derives from radiometric corrections and calibrations.

The surface water hydrographs of rivers exhibit large variation due to many natural phenomena. One of the most commonly used approaches for interpolating and extending streamflow records is to fit observed data with an analytic power model. However, such analytic models may not adequately represent the flow process, because they are based on many simplifying assumptions about the natural phenomena that influence the river flow. *Karunanithi et al (1994)* have demonstrated how a neural network can be used as an adaptive model synthesiser as well as a predictor. Issues such as selecting an appropriate neural network architecture and a correct training algorithm as well as presenting data to neural networks are addressed using a constructure algorithm called the cascade correlation algorithm. The neural network approach is applied to the flow prediction of the Huron River at the Dexter sampling station, near Ann Arbor, Mich. Empirical comparisons are performed between the predictive capability of the neural network models and the most commonly used analytic non-linear power model in terms of accuracy and convenience of use. The preliminary results are quite encouraging. An analysis performed on the structure of the networks developed by the cascade correlation algorithm shows that the neural networks are capable of adapting their complexity to match changes in the flow history and that the models developed by the neural network approach are more complex than the power model.

Navone and Ceccatto (1994) have applied the neural network approach in predicting the Indian monsoon rainfall. These computational structures are used as a non-linear method to correlate preseason predictors to rainfall data, and as an algorithm for reconstruction of the rainfall time-series intrinsic dynamics. A bomined approach is developed which captures the information built into both the stochastic approach based on suitable predictors and the deterministic dynamical model of the time series. The hierarchical network so obtained has forecasting capabilities remarkably improved with respect to conventional methods.

Using Advanced Very High Resolution Radiometer data, *Bankert (1994)* has classified 16 pixel x 16 pixel sample area into one of ten output classes using a Probabilistic Neural Network

(PNN). The ten classes are cirrus, cirrocumulus, cirrostratus, altostratus, nimbostratus, stratocumulus, stratus, cumulus, cumulonimbus and clear. Over 200 features drawn from spectral, textural and input physical measures are computed from the pixel data for each sample area. The input parameters presented to the neural network are a subset of these features selected by a routine that indicates the discriminatory potential of each feature. The training and testing input data used by the PNN are obtained from 95 expertly labelled images taken from seven maritime regions; these images provide 1633 sample areas. Theoretical accuracy of the PNN classifier is determined using two methods. In the hold-one-cut method, the network is trained on all data samples minus one and is tested on the remaining sample. Using this technique, 79.8% of the samples are classified correctly. A bootstrap method of 100 randomly determined sample sets produce an average overall accuracy of 77.1%, with a standard deviation of 1.4%. In a more general classification using five classes (low clouds, altostratus, high clouds, precipitating clouds and clear), 91.2% of the samples are accurately classified. A two-layer, four-network system that determines the general classification of a sample followed by a specific classification in another network is proposed. Testing of this system produces mixed results compared to the single ten-class PNN.

Zhang and Scofield (1994) have presented an Artificial Neural Network (ANN) technique for heavy convective rainfall estimation and cloud merger recognition from satellite data. An Artificial Neural network expert system for Satellite-derived Estimation of Rainfall (ANSER) has been developed in the NOAA/NESDIS Satellite Applications Laboratory. Using artificial neural network group techniques, the following can be achieved: automatic recognition of cloud mergers, computation of rainfall amounts that will be ten times faster and average errors of the rainfall estimates for the total precipitation event that will be reduced to less than 10 per cent.

Allen and Marshall (1994) have applied neural networks and discriminant analysis to forecasting 24 hour rainfall for the city of Melbourne. Several different approaches were tested with each of these techniques, using a training data set of 1997 cases and an independent test data set of 665 cases. Performance comparisons indicated that both neural network and discriminant analysis methods offered improvements over the operational model output statistics (MOS) method and operational forecasts. As a result of these tests a discriminant analysis method has been incorporated into an expert system to assist forecasters making rainfall forecasts. This system is undergoing an operational trial in the Victorian Regional Forecast Centre.

Diaz and Alfonso (1995) have compared two methods for fog forecasting in Cuba. Use of the classical Fisher discriminant function, and the LVQ algorithm developed by the Laboratory of Computer and Information Sciences, Helsinki University of Technology. The relative number of correct forecasts is over 70% for both, which can be considered a good performance. When the learning sample is large enough and nearby equi-probabalistic, the LVQ algorithm provides a greater number of correct forecasts than those obtained via the Fisher discriminant function. However, the results attained via the LVQ algorithm are not steady when the learning sample is far from being equi-probabalistic, because the number of fog cases is much reduced. Until larger samples are available for some regions, it will be necessary to use both methods for fog forecasting in Cuba.

Methods to continuously forecast water levels at a site along a river are generally model based. Physical processes influencing occurrence of a river stage are, however, highly complex and uncertain, which makes it difficult to capture them in some form of deterministic or statistical model. Neural networks provide model free solutions and hence can be expected to be appropriate in these conditions. Built in dynamism in forecasting, data error tolerance, and lack of requirements of any exogenous input are additional attractive features of neural networks. *Thirumalaiah and Deo (1998)* have used ANN approach in real time forecasting of water levels at a given site continuously throughout the year based on the same levels at some upstream gauging station and/or using the stage time history recorded at the same site. The network is trained by using three algorithms, namely, error back propagation, cascade correlation, and conjugate gradient. The training results are compared with each other. The network is verified with untrained data.

3.3 HYDRAULICS

Dartus et al (1993) have used a neural net to study the propagation of a flood wave in an open channel. The aim is to show that this kind of tool is accurate enough to be used in real time management of sewers systems. Ability of such a neural network to answer correctly is highlightened with an extensive learning base and with a reduced one.

Mase et al (1995) have examined the applicability of a neural network to analyse model test data of the stability of rubble-mound breakwaters. Seven parameters concerning the stability of rock

slopes are used: the stability number, the damage level, the number of attacking waves, the surf-similarity parameter, the permeability parameter, the dimensionless water depth in front of the structure, and the spectral shape parameter. The damage levels predicted by the neural network, calibrated by using a part of *Van der Meer's 1988* experimental data, agree satisfactorily well with the measured damage levels of another part of the data source by *Van der Meer 1988* and by *Smith et al, 1992* data.

A feed forward back propagation type neural network was used by *Grubert (1995)* to predict the flow conditions when interfacial mixing in stratified estuaries commences. This was achieved by training the network to extrapolate data from laboratory experiments performed over many years by several researchers. Before this training could be carried out, however, many decisions concerning the size of the network required and its training parameters had to be made. These decisions were made on the basis of successfully training a similar stratified flow condition, that of thermal wedges downstream of a power plant's outlet, where the theoretical solution is known. Finally, these results were compared with an approximate stability equation utilizating results from inviscid flow theory, rough turbulent flow theory, and laboratory experiments on interfacial friction. Although the agreement was not exact it was close enough to predict what the stability conditions in real estuaries should be. This prediction was verified with the only prototype data available, that from three fjords, which agreed with both the neural network and theoretical results.

3.4 WATER RESOURCES

The design, analysis and management of the water resources systems, involves modelling and prediction of the behaviour of complex systems. The Artificial Neural Networks (ANN) can be used in a large variety of problems. e.g. mapping, dynamic process modelling, optimisation, image processing, data analysis, forecasting, simulation, function approximation etc. Due to the distributed nature of ANNs, destruction of a few nodes or presence of some inconsistent data does not adversely affect the performance of ANNs.

The artificial neural network approach described by *Raman and Kumar (1995)* for the synthesis of reservoir inflow series differs from the traditional approaches in synthetic hydrology in the sense that it belongs to a class of data-driven approaches as opposed to traditional model driven

approaches. Most of the time series modelling procedures fall within the framework of multivariate autoregressive moving average (ARMA) models, modelling procedures suggest a four-stage iterative process, namely, model selection, model order identification, parameter estimation and diagnostic checks. Although a number of statistical tools are already available to follow such a modelling process, it is not an easy task, especially if higher order vector ARMA models are used. This paper investigates the use of artificial neural networks in the field of synthetic inflow generation. The various steps involved in the development of a neural network and a multivariate autoregressive model for synthesis is presented. The application of both types of model for synthesising monthly inflow records for two reservoir sites is explained. The performance of the neural network is compared with the statistical method of synthetic inflow generation.

Reservoir operating policies are derived to improve the operation and efficient management of available water for the Aliyar Dam in Tamil Nadu, India, using a dynamic programming (DP) model, a stochastic dynamic programming (SDP) model, and a standard operating policy (SOP). *Raman & Chandramouli (1996)* have derived a general operating policy for reservoirs using neural networks. The objective function for this case study was to minimise the squared deficit of the release from the irrigation demand. From the DP algorithm, general operating policies were derived using a neural network procedure (DPN model), and using a multiple linear regression procedure (DPR model). The DP functional equation is solved for 20 years of fortnightly historic data. The field irrigation demand was computed for this study by the study by the modified Penman method with daily meteorological data. The performance of the DPR, DPN, SDP and SOP models were compared for three years of historic data using the proposed objective function. The neural network procedure based on the dynamic programming algorithm provided better performance than the other models.

Crespo and Mora (1993) have proposed an artificial neural network model to derive streamflow from precipitation data. It is tested with actual data coming from a nearby river, referred to a basin area of 356 km super (2) and a time period of 11 years. A feedforward multilayer perception with linear output has been built to deal with this problem. The dynamics are caught by the filter structure of the input layer. A special study on crossing properties, based on training sample selection, is made to measure the performance of the network for drought analysis. Sample selection leads to increased accuracy within the sample range and degraded performance

for points that are clearly out. Predicted number of droughts, average drought length and deficit are compared with the actual data. The results show that very simple neural network models can give fine results.

Three layer back-propagation networks have been used by *Bischof (1992)* for the classification of Landsat TM data of the surroundings of Vienna on a pixel by pixel basis. The aim of the classification with the neural network was to distinguish between four categories: built-up land, agricultural land, forest, and water. The resulting thematic map was compared to the Gaussian classification. In order to achieve approximate Gaussian distributions, the categories had to be further subdivided into 12 subcategories. Two thematic maps were prepared by visual classification of the Landsat image, using additional data from aerial photographs, maps and field work. These maps were considered to represent the true classification. A small set of samples was selected, training of the neural network and half were used for testing and classification of the result. After 50 epochs of learning the neural network made fewer errors than maximum likelihood classification. A special neural network was used for post-classification smoothing.

3.5 ENVIRONMENTAL

Ruck et al (1993) have presented a method of predicting benthic community structure from environmental variables that uses artificial neural networks. The input variables represent geophysical, limnological and sedimentological characteristics of sites in Canadian waters of the Laurentian Great lakes. A single output from the network predicts the number of individuals of a given taxon to be found in a 5.5 cm by 10 cm deep core sample of lake sediment taken at the site in question. Networks have been trained for four texa: Oligochaeta, Porifera, Chironomidae and Pelecypoda. Three input vector sets were compared: the 28 dimension raw data set, a subset of 9 variables and a 7 dimension eigen vector set. Performance tests were carried out using a 1-fold cross validation technique, which maximises data utility while maintaining independence between the training and test sites. It was concluded that artificial neural networks have potential for use in biological monitoring systems.

Models for the prediction of conductance in nonbrine water samples through the measurement of ionic concentrations and other parameters are compared by *Hughes et al (1994)*. Such

predictions are often used for quality assurance purposes by comparing them with actual measurements to determine whether gross analysis errors have been made. A currently recommended method for making such predictions is a semiempirical relation that adapts the Debye-Hueckel-Onsagar equation to mixed electrolyte systems by incorporating modified definitions of ionic charge and concentration. The limitations of this model are examined, and extensions to it are considered. Other predictive methods, including muiltiple linear regression (MLP), principal components regression (PCR), partial least-squares (PLS) regression, continuum regression (CR), and neural networks (NN), are also considered. Models employ the concentrations of 1-0 ions as well as, in some cases, additional water quality measurements. Best results were obtained with an extended form of the Debye-Hueckel-Onsagar equation and an optimised MLR model. POCR, PLS, CR, and NN did not offer significant advantages.

French and Recknagal (1994) have developed a neural network model for predicting algal blooms. The neural netowrk consists of a 3 layer structure with input, hidden, and output layers. Training is conducted using back-propagation where the data are presented as a series of learning sets such that the inputs are observable water quality parameters and outputs are the biomass quantities of specific algal groups. Training is conducted using three years of daily values of water quality parameters and validation is performed using two years of independent daily values to predict the magnitude and timing of blooms of 7 different algae groups with a lead time of 1 day using only the current day water quality parameters. The water quality data represent physical and limnological characteristics of a drinking water reservoir in Germany. Results indicate that the neural network model is capable of learning the complex relationships describing the seasonal succession of phytoplankton in freshwaters.

The economic development activities of an increasing world population threaten the assimilative capacity of our environment and have stimulated interest in the concept of environmental carrying capacity. While the pace of land transformations has encouraged the refinement of information technologies such as satellite remote sensing to provide a synoptic view of earth-system processes, the volume of information these systems generate and the high level of expertise required to translate these data retard effective and timely land management decision making. *Lein (1995)* has introduced a methodology that employs an artificial neural network trained to recognise categories of population support capacity from satellite data acquired from the NOAA-AVHRR. The network, functioning as an 'intelligent' mapping tool, achieved a

classification accuracy of 77.5 per cent for the study site and points to the potential role a model of this type may play in land degradation monitoring.

Maier and Dandy (1996) have presented the use of artificial neural networks (ANNs) as a viable means of forecasting water quality parameters. A case study is presented in which ANN methods are used to forecast salinity in the River Murray at Murray Bridge (South Australia) 14 days in advance. It is estimated that high salinity levels in the Murray cause $US 22 million damage per year to water users in Adelaide. Previous studies have shown that the average salinity of the water supplied to Adelaide could be reduced by about 10% if pumping from the Murray were to be scheduled in an optimal manner. This requires forecasts of salinity several weeks in advance. The results obtained were most promising. The average absolute percentage errors of the independent 14-day forecasts for four different years of data varied from 5.3% to 7%. The average absolute percentage error obtained as part of a real time forecasting simulation for 1991 was 6.5%.

4.0 CONCLUSIONS

The research work reported so far indicates that ANNs are ideal for providing timely and cost effective solutions to a variety of field scale problems. Due to the ease of application and simple formulation, this technique has already become a prospective research area with great potential. The applications of ANNs in the field of surface water hydrology have been to several diverse nature of problems and the results in each case have been very encouraging.

There are certain issues that need to be resolved before employing neural networks. The main issue is the choice of the network. As mentioned earlier, a wide variety of network architecture and paradigms exist. In all these cases it is very important to specify the form in which data is presented to the network and the network size. Besides this, network training times may be long which precludes trying a number of configurations and several presentation strategies. Careful preprocessing of the input data can dramatically reduce the number of input nodes, and so reduce the network size and training times. Conversely, poor preprocessing can remove information required by the network to converge and so increase the training overhead. Increasing the number of input nodes may make the network learning easier, but would also increase the number of interconnecting weights which would take their own time to set.

An optimum training pattern or data is thought to be the one containing many more examples than the number of network weights, therefore coding strategies must be such that the number of weights is optimum and the results effective. In addition to defining the input and output layers, the number and size of the hidden layers must also be specified. Too few hidden nodes would inhibit learning, while too many may result in overtraining. In the latter condition, the network's accuracy performance improves but the network's capability to predict patterns it has not seen, actually declines. The number of hidden layer nodes most suited to a problem may be obtained using a number of algorithms.

Besides the suitable choice and optimal design of the neural network, there are issues to their application which should not be ignored. First, ANNs can only generate meaningful results/predictions over the problem dimensions defined by the training patterns. If the scope of the problem changes, then the training patterns must be recreated or at least augmented with new examples. Secondly, if ANNs are not trained to high levels of predictive accuracy, as measured

by their performance on test examples, errors may occur in predictions and during sensitivity analysis.

The possible applications of ANNs in the field of surface water hydrology are numerous. For example, they can be used for rainfall runoff modelling; calibration of rainfall runoff models; precipitation estimation from remotely sensed information; estimation of rainfall rate from satellite infrared imagery; retrieval of snow parameters from microwave measurements; flood forecasting; drought estimation; multivariate modelling of water resources time series; modelling of water retention curves; deriving general operating policy for reservoirs; prediction of water quality parameters.

The ANN technique is still under rigorous research and there is a good scope in the yet-to-be discovered potential of ANNs trying to simulate the power of the human brain. Despite the limitations, the noted advantages of ANNs are significant and promising for field scale applications in surface water hydrology.

REFERENCES

Alberotanza, L. and Pavanati, M., 1992. Remote sensed spectral signatures classification and recognition by neural networks: An approach to unsupervised techniques. Technical Rep., Consiglio nazionade delle Ricerche, Venice, Italy.

Allen, G. and Marshall, J.F., 1994. An evaluation of neural networks and discriminant analysis methods for application in operational rain forecasting. Australian Meteorological Magazine, Vol.3, No.1, pp 17-28.

Bankert, R.L., 1994. Cloud classification of AVHRR imagery in maritime regions using a probabilistic neural network. Jour. Of Applied Meteorology, vol.33, No.8, pp 909-918.

Bischof, H., Schneider, W. and Pinz, A., 1992. Multispectral classification of Landsat images using neural networks. IEEE Transactions on Geoscience and Remote Sensing IGRSDZ, Vol. 30, No.3, p 482-490.

Box, G.E. and Jenkins, G.M., 1976. Time sries analysis: Forecasting and control. Holden-Day, Oakland, Calif.

Bras, R.L. and Rodriguez-Iturbe, I., 1985. Random functions and hydrology, Addison-Wesley, Reading, Mass.

Brazil, L.E. and Hudlow, M.D., 1980. Calibration procedures used with the Natinal Weather Service Forecast System. IFAC Symp. On water and Related Land Resource Systems, Int. Fed. of Autom, Control, Cleveland, Ohio.

Burnash, R.J.E., Ferral, R.L. and McGuire, R.A., 1973. A generalized streamflow simulation system. Rep. 220, Jt. Fed.-State River Forecast. Cent., Sacramento. Calif.

Chang, A.T.C. and L. Tsang, 1992. A neural network approach to inversion of snow water equivalent from passive microwave measurements. Nordic Hydrology, Vol.23, No.3, pp. 173-182.

Crawford, N.H. and Linsley, R.K., 1966. Digital simulation in hydrology: stanford watershed model IV. Tech. Rep. 39, Dep. Of Civil Engg., Stanford Univ., Stanford, Calif.

Crespo, J.L. and E. Mora, 1993. Drought estimation with neural networks. Adv. Engg. Software, Vol.18, No.3, pp. 167-170.

Diaz, N. and Alfonso, A.P., 1995. Fog forecasting in Cuba. Neural networks versus discriminant analysis. Jour. of Meteorological Applicatons, Vol.2, No.1, pp 31-34.

Dartus, D., Courivaud, J.M. and Dedecker, L., 1993. Use of neural net for the study of a flood wave propagation in an open channel. Jour. Of Hydraulic Research, Vol.31, No.2, pp 161-170.

Dawn, T., 1994. Neural computing makes it mark in science. Scientific Computation, Sept., pp 25-30.

Dawson, C.W. and Wilby, R., 1998. An artikficial neural network approach to rainfall-runoff modelling. Hydrological Sciences, Vol.43, No.1.

Day, S.P. and Davenport, M.R., 1993. Continuous time temporal back-propagation with adaptable time delays. IEEE Transactions on Neural Networks. Vol.4, No.2, pp 348-354.

Duan, Q., Gupta, V.K. and Sorooshian, S., 1993. A shuffled complex evolution approach for effective and efficient global optimization. Jour. Optim. Theory Appl., Vol.73, No.3, pp 501-521.

Elman, J.L., 1990. Finding structure in time. Cognitive Science, Vol.14, pp 179-211.

French, M.N., Krajewski, W.F., and Cuykendall, R.R.,1992. Rainfall forecasting in space and time using a neural network. Journal of Hydrology, Vol.137, pp. 1-31.

French, M. and Recknagel, F., 1994. Modelling of algal blooms in fresh waters using artificial neural networks. Proc. of 5th Int. Conf. on Development and Aplication of Computer Techniques

tro Environmental Studies, san Francisco, CA (USA), 16-18 Nov.

Goppert, J. and Rosenstiel, W., 1990. The use of neural networks in on-line analysis. Received via internet.

Grubert, J.p., 1995. Application of neutral networks in stratified flow stability analysis. Jour. of Hydraulic Engg. Vol. 121, No.7, pp 523-531.

Haario, H. and Jokinen, P., 1991. Increasing the learning speed of back-propagation algorithm by linearisation. Artificial Neural Networks, T. Kohonen et al., pp 629-634.

Hsu, K., Gupta, H.V. and Sorooshian, S., 1995. Artificial neural network modelling of the rainfall runoff process. Jour. of Water Resourc. Res., Vol.31, No.10, pp 2517-2530.

Hsu, K., Gupta, H.V., Sorooshian S., and Gao X., 1996. An artificial neural network model for rainfall estimation from satellite infrared. Proc. Third International Workshop on Application of Remote Sensing in Hydrology, Oct. 16-18, 1996, NASA Goddard Space Flight Center, Greenbelt, maryland.

Hsu, K., Gao, X., Sorooshian S., and Gupta H.V., 1997. Precipitation estimation from remotely sensed information using artificial neural networks. Journal of Applied Meteorology, Feb 1997.

Hsu, K., Gupta, H.V. and Sorooshian, S., 1997. Application of recurrent neural networks to rainfall-runoff modeling. Proc. ASCE Conf. on Water Resources Planning and Management Division, April 7-10, Houston, Texas

Hughes, S.G., taylor, E.L., Wentzell, P.D., McCurdy, R.F. and Boss, R.K., 1994. Models for conductance measurements in quality assurance of water analysis. Jour. of Analytical Chemistry, Vol. 66, No. 6, pp 830-835.

Ikeda, S., Ochiai, S.M. and Saearagi, Y., 1976. Sequential GMDH algorithm and its application to river flow prediction. IEEE Trans. Syst. Man Cybern., Vol. 6, No.7, pp 473-479.

Jain, S.K. and Chalisgaonkar, D., 1997. Application of artificial neural networks in rainfall-runoff modelling. Proceedings of the Int. Symp. on Emerging Trends in Hydrology, Sept. 25-27, 1997, Dept. of Hydrology, UOR, Roorkee, India.

Karunanithi, N., Grenney, W.J., and Bovee, K., et.al, 1994. Neural networks for river flow prediction. Journal of Computing in Civil Engineering, Vol.8, No.2.

Kitanidis, P.K. and Bras, R.L., 1980. Adaptive filtering through detection of isolated transient errors in rainfall-runoff models. Water Resources Res., Vol.16, No.4, pp 740-748.

Kohonen, T., 1988. Self organization and associative memory. 2nd ed., Springer Verlag, New York.

Kosko, B., 1990b. Unsupervised learning in noise. IEEE Transactions on Neural Networks, Vol.1, No.11.

Lein, J.K., 1995. Mapping environmental carrying capacity using an artificial neural network: A first experiment. Jour. of Land Degradation Rehabilitation, Vol. 6, No.1, pp 17-28.

Lippmann, R.P., 1987. Introduction to computing with neural nets. IEEE Magazine on Acoustics, Speech and Signal Processing, pp 4-22.

Lorrai, M. and G.M. Sechi, 1995. Neural nets for modeling rainfall-runoff transformations. Water Resources Management, Vol.9, pp. 299-313.

Maier, H.R. and G.C. Dandy, 1996. Use of artificial neural networks for prediction of water quality parameters. Water Resources Research, Vol. 32, No.4, pp. 1013-1022.

Mase, H., Sakamoto, M. and Sakai, T., 1995. Neural network for stability analysis of rubble mound break waters. Jour. of Waterway, port, Coastal and Ocean Enggg., Vol. 121, No.6, pp 294-299.

Minns A.W. and Hall, M.J., 1996. Artificial neural networks as rainfall-runoff models. Journal

of Hydrological Sciences, Vol.41 No.3.

Nadi, F., 1991. Topological design of modular neural networks. Artificial Neural Networks, T. Kohonen et al., pp 213-217.

Natale, L. and Todini, E., 1976. A stable estimator for linear models, 1, theoretical development and Monte Carlo experiments. Water Resources Res., Vol. 12, No.4, pp 664-671.

Natale, L. and Todini, E., 1976. A stable estimator for linear models, 2, real world hydrologic applications. Water Resources Res., Vol. 12, No.4, pp 672-676.

Navone, H.D. and Ceccatto, H.A., 1994. Predicting Indian monsoon rainfall : A neural network approach. Jour. of Climate Dynamics, Vol. 10, No.6-7, pp 305-312.

Oppenheim Alan V. Et al., 1983. Signals and systems. Prentice-Hall Editions, p 685.

Raman H. and Sunilkumar, N., 1995. Multivariate modelling of water resources time series using artificial neural networks. Journal of Hydrological Sciences, Vol.40, No.2.

Raman H. and Chandramouli, V., 1996. Deriving a general operating policy for reservoirs using neural network. Journal of Water Resources Planning and Management Vol.122, No.5.

Refenes, A.N. and Vithlani, S., 1991. Constructive learning by specialisation. Artificial Neural Networks, T. Kohonen et al., pp 923-929.

Rumelhart, D.E. and Zipser, D., 1986. Feature discovery by competetive learning. Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Vol.1, pp 318-364.

Ruck, B.M., Walley, W.J., Reynoldson, T.B. and day, K.E., 1993. A neural network predictor of benthic community structure in the Canadian waters of the Laurentian Great lakes. In: Modelling, measuring and prediction. Water Pollution II, Computational Mechanics, Boston, M.A., pp 287-294.

Rusell, I.F., 1991. Self organising competitive learning and adaptive resonance networks. Int. Jour. of Neural Networks, Vol.2, No.2/3/4.

Sato, A., 1991. An analytical study of the momentum term in a back-propagation algorithm. Artificial Neural Networks, T. Kohonen et al., pp 617-622.

Smith, J. and Eli, R.N., 1995. Neural network models of rainfall-runoff process. Journal of Water Resources Planning and Management, Vol.121, No.6.

Sorooshian, S., 1983. Surface water hydrology : On-line estimation, Rev. Geophys., Vol.21, No.3, pp 706-721.

Sorooshian, S., Duan, Q. and Gupta, V.K., 1993. Calibration of rainfall-runoff models: Application of global optimization to the Sacramento soil moisture accounting model. Water Resources Res., Vol. 29, No.4, pp 1185-1194.

Swingler K, 1996. Applying Neural Networks, A Practical Guide. Academic Press Ltd., London

Thirumalaiah, K. and Deo, M.C., 1998. River state forecasting using artificial neural networks. Journal of Hydrologic Engineering, Vol.3 No.1.

Tresp, V., Ahmad, S. and Neuneier, R. Training neural networks with deficient data. Received from internet.

Yapo, P., Gupta, V.K. and Sorooshian, S., 1995. Calibration of conceptual raifall-runoff models: Sensitivity to calibration data, Jour. Hydrol.

Young, P. and wallis, S., 1985. Recursive estimation: A unified approach to the identification, estimation and forecasting of hydrological systems. Appl. Math. Comput., Vol.17, pp 299-334.

Yu, X., Loh, N.K. and Miller, W.C., 1993. A new acceleration method for the back-propagation algorithm. IEEE Transactions on Neural Networks, PP 1157-1161.

Zhang, M. and Scofield, R.A., 1994. Artificial neural network techniques for estimating heavy convective rainfall and recognising cloud merges from satellite data. Int. Jour. of Remote Sensing, Vol.15, No.16, pp 3241-3261.

STUDY GROUP

DIRECTOR:	**Dr. K.S. RAMASASTRI**
DIVISIONAL HEAD:	**R. MEHROTRA** **Sc. EI**
SCIENTIST:	**ARCHANA SARKAR** **Sc. B**