Application of Artificial Neural Networks in the Field of Geohydrology

by
Gideon Steyl

A dissertation submitted to meet the requirements for the degree of
Magister Scientiae

in the
Institute of Groundwater Studies
Faculty of Natural- and Agricultural Sciences

at the
University of the Free State

Supervisor: Dr. S.R. Dennis

March 2009

Dedication

Here I wish to thank the following people for their support and patience in my latest endeavour to investigate a new field of study.

Prof. A. Roodt (Chemistry) for allowing me to venture into this new field and the understanding that it was a course of action which was required on my part.

Mr. L. Kirsten (Leo), Miss. T.N. Hill (Tania), Mr. T. Muller (Theuns), Miss. M. Steyn (Maryke) and finally Miss. G. Venter (Truidie) for their hard work and diligence in their respective Ph.D. and M.Sc. studies under my guidance at Chemistry. You made my studies easier due to your efforts.

Dr. I. Dennis, Dr. D. Vermeulen and to the “beste professor” Prof. G. Van Tonder thank you for your kindness, including the office space and your helpful contribution to my M.Sc.

Dr. S.R. Dennis for the understanding and patience to stay the course and guiding me through a study which I enjoyed.

My mother for her support during this time, even if the times were hard.

To my lovely wife, Elizabeth Steyl (Liza) and my daughter Anneke for your support and understanding.

Finally, God which gave me talents I do not always understand or fully apply.

"I want to know how God created this world. I am not interested in this or that phenomenon, in the spectrum of this or that element. I want to know His thoughts; the rest are details."

Albert Einstein

Contents

List of Figures

List of Tables

Geohydrological and Related Terms

ABSTRACTION: the removal of water from a resource, e.g. the pumping of groundwater from an aquifer.

ALLUVIAL AQUIFER: an aquifer formed of unconsolidated material deposited by water, typically occurring adjacent to river channels and in buried or palaeochannels.

ALLUVIUM: a general term for unconsolidated deposits of inorganic materials (clay, silt, sand, gravel, boulders) deposited by flowing water.

AQUATIC ECOSYSTEMS: not defined by the National Water Act (Act No. 36 of 1998), but defined elsewhere as the *abiotic* (physical and chemical) and *biotic* components, habitats and ecological processes contained within rivers and their riparian zones and reservoirs, lakes, wetlands and their fringing vegetation.

AQUIFER SYSTEM: a heterogeneous body of intercalated permeable and less permeable material that acts as a water-yielding hydraulic unit of regional extent.

BANK STORAGE: water that percolates laterally from a river in flood into the adjacent geological material, some of which may flow back into the river during low-flow conditions.

BASEFLOW: sustained low flow in a river during dry or fair weather conditions, but not necessarily all contributed by groundwater; includes contributions from delayed interflow and groundwater discharge.

FRACTURED AQUIFER: an aquifer that owes its water-bearing properties to fracturing caused by folding and faulting.

HYDRAULIC CONDUCTIVITY: measure of the ease with which water will pass through earth material; defined as the rate of flow through a cross-section of one square metre under a unit hydraulic gradient at right angles to the direction of flow (in m/d).

HYDROGRAPH: a graphical plot of hydrological measurements over a period of time, e.g. water level, flow, discharge.

HYDROLOGICAL CYCLE: the continuous circulation of water between oceans, the atmosphere and land. The sun is the energy source that raises water by evapotranspiration from the oceans and land into the atmosphere, while the forces of gravity influence the movement of both surface and subsurface water.

INFILTRATION: the downward movement of water from the atmosphere into the ground; not to be confused with *percolation.*

INTERFLOW: the rapid flow of water along essentially unsaturated flow paths, water that infiltrates the subsurface and moves both vertically and laterally before discharging into other water bodies.

PERCOLATION: the process of the downward movement of water in the unsaturated zone under the influence of gravity and hydraulic forces; term used to differentiate from infiltration, which specially refers to the movement of water from the atmosphere into the ground.

WATER TABLE: the upper surface of the saturated zone of an unconfined aquifer at which pore pressure is at atmospheric pressure, the depth to which may fluctuate seasonally.

Acronyms

ANN – An artificial neural network is a mathematical model or computational model based on biological neural networks. It consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation.

MLP – Multilayer perceptrons represents the most prominent class of ANNs in classification, implementing a feedforward, supervised learning paradigm. MLPs consist of several layers of nodes , interconnected through weighted neurons from each preceding layer to the following, without lateral or feedback connections.

MSE – Mean Square Error is the average of the square of the difference between the desired response and the actual system output (the error).

PARMA – Periodic Auto Regressive Moving Average are typically applied to time series data.

PAR – Periodic Auto Regressive process.

RBF – A Radial Basis Function neural network has an input layer, a hidden layer and an output layer. The neurons in the hidden layer contain Gaussian transfer functions whose outputs are inversely proportional to the distance from the centre of the neuron.

SOM – A Self-Organizing Map (SOM) is a type of artificial neural network that is trained using unsupervised learning to produce a low-dimensional, discretized representation of the input space of the training samples, called a map. Self-organizing maps are different than other artificial neural networks in the sense that they use a neighbourhood function to preserve the topological properties of the input space.

RNN – Recurrent Neural Networks are models with bi-directional data flow, which also propagates data from later processing stages to earlier stages.

SRN – A Simple Recurrent Network is a variation on the Multi-Layer Perceptron or Elman network.

ESN – The Echo State Network is a recurrent neural network with a sparsely connected random hidden layer. The weights of output neurons are the only part of the network that can change and be learned. ESN's are good to (re)produce temporal patterns.

MDP – The environment is modelled as a Markov Decision Process with states and actions with a probability distribution consisting of instantaneous cost, observation and transition, while a policy is defined as conditional distribution over actions given the observations.

MC – Markov Chain is a combination of MDP and a policy which defines a conditional distribution over actions giving the observations.

FTDNN – Focused Time-Delay Neural Network is part of a general class of dynamic networks, called focused networks, in which the dynamics appear only at the input layer of a static multilayer feedforward network.

LRNN – The Layer-Recurrent Network is a dynamic network which generalizes the Elman network to have an arbitrary number of layers and to have arbitrary transfer functions in each layer.

GRNN – A Generalized Regression Neural Network is often used for function approximation. It has a radial basis layer and a special linear layer.

PNN – Probabilistic Neural Networks are a kind of radial basis network suitable for classification problems.

RMS – The Root Mean Square is a statistical measure of the magnitude of a varying quantity.

Chapter 1

Introduction

An artificial neural network (ANN) is a mathematical model or computational model based on biological neural networks. It consists of a unified group of artificial neurons that processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structural bias based on information that flows through the network during the training phase. In more practical terms neural networks are non-linear statistical data modelling tools which can be used to model complex relationships between inputs and outputs or find patterns in data.

1.1. History of the neural networks

1.1.1. Connectionism

Neural networks as a concept was initially defined in the late-1800s in order to describe how the human mind functioned. It was first applied to a computational model with Turing's B-type machines and the Perceptron. Nearly a century passed before Friedrich Hayek (1950) conceived the idea that the brain spontaneously orders itself from a decentralized network of simple units (neurons). A decade before Hayek, Donald Hebb[1] made one of the first hypotheses for a mechanism of neural plasticity, *i.e.*, Hebbian learning. The application of Hebbian learning is a typical example of an unsupervised learning rule, in the following years this learning rule will have significant implications on the training of artificial neural networks.

The Perceptron[2] is essentially a linear classifier for classifying data (x) specified by a set of parameters (w weights and b bias) and an output function ($f = w \cdot x + b$).

$$f(x) = \begin{cases} 1, & if\ w \cdot x + b > 0 \\ 0, & else \end{cases}$$

Its parameters are adapted with an ad-hoc rule similar to steepest gradient descent methods. One failure of the Perceptron is that it can only be used to classify a set of data for which different classes are linearly separable in the input space. This short coming is due to the inner product which is a linear operator in the input space. The development of the algorithm initially was met with great enthusiasm because of its apparent relation to biological mechanisms. The later discovery of the linear inadequacy caused such Perceptron type models to be abandoned until the introduction of advanced non-linear models into the field.

The Cognitron[3] (1975) was an early multilayered neural network with an associated training algorithm. The actual structure of the network and the methods used to set the interconnection weights changes from one application to another. This caused each network implementation to have its own advantages and disadvantages. These networks could only propagate information in one direction, or a bounce back and forth algorithm was used until self-activation at a node occurred and the network settled on its final state. The ability for bi-directional flow of inputs between neurons or nodes was produced with the Hopfield's network[4] (1982). This network allowed for the specialization of different node layers for specific purposes and was introduced through the first hybrid networks.

The rise of the personal computer in the 1980s made connectionism popular since it could be implemented in massive parallel distributed processes. The rediscovery of back propagation algorithm was probably the main reason behind the popularisation of neural networks[5]. The original network utilised multiple layers of weight-sum units of the type $f = g(w \cdot x + b)$, where g is a sigmoid function or logistic function such as used in logistic regression. Training of the network was done by a form of steepest descent gradient methods. The inclusion of sigmoid type functions allowed the use of the chain rule of differentiation in deriving the appropriate parameter updates, resulting in an algorithm that seems to back propagate the errors. However, it is essentially a form of gradient descent and not back propagation. Determining the optimal parameters in a model of this type is neither trivial nor linear, and steepest gradient descent methods cannot be relied upon to give the solution without a good starting point. In recent times, networks with the same architecture as the back propagation networks are referred to as Multi-Layer Perceptrons. This name does not impose any limitations on the type of algorithm used for learning.

In Figure 1.1 a graphical time line of the development of neural networks is presented to illustrate the different components combining to give the present day implementation.

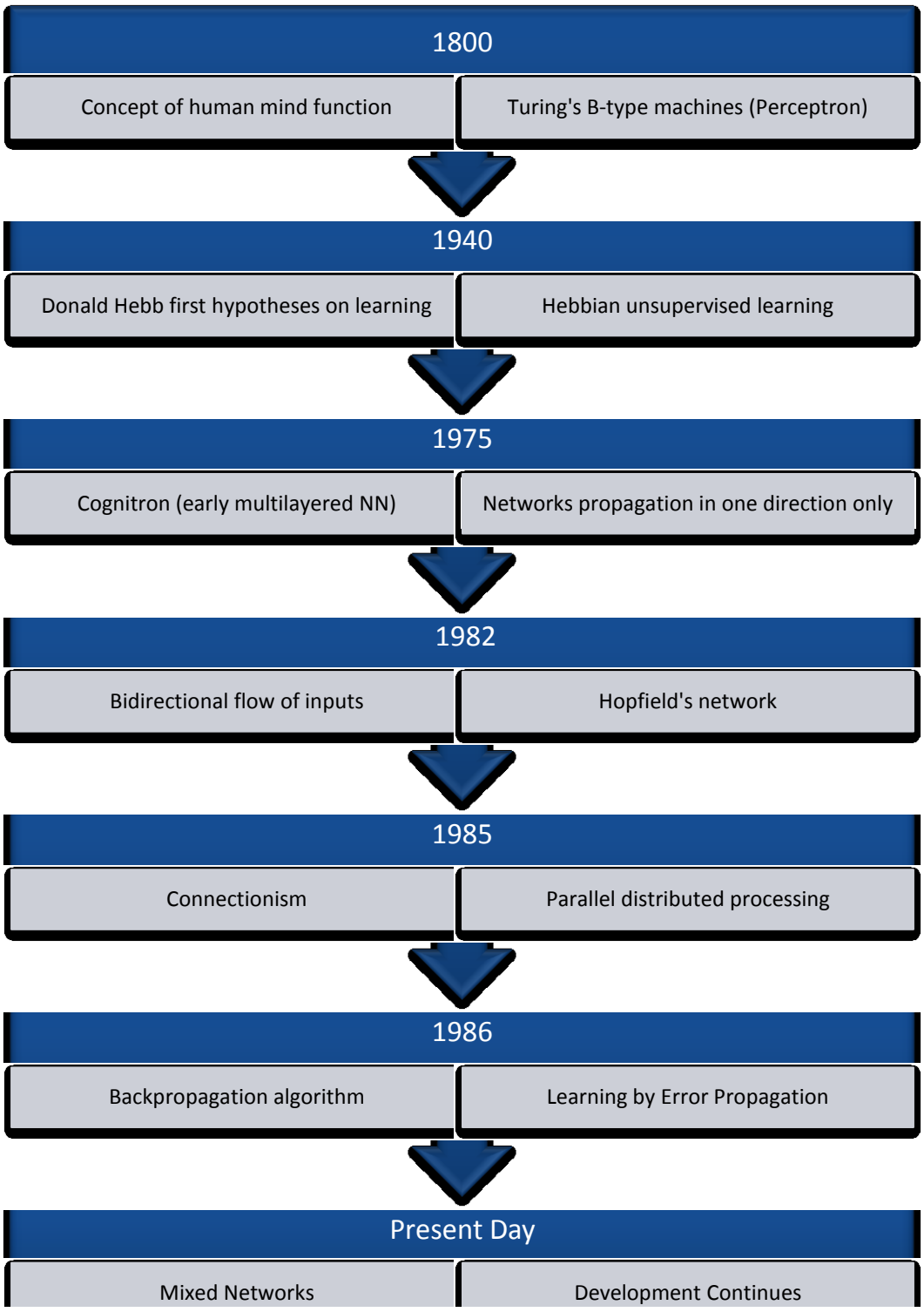


Figure 1.1 A Summary of the History and Development of Artificial Neural Networks to Modern Day Standards.

1.2. Neural Network Applications

The utility of artificial neural networks can be divided into the following three broad categories,

- **Function approximation** or regression analysis, time series prediction and modelling.
- **Classification**, pattern and sequence recognition.
- **Data mining**, filtering, clustering, blind source separation and compression.

Examples of specific application in work related fields are given diagrammatically in Figure 1.2 and discussed in the following paragraphs in this section.

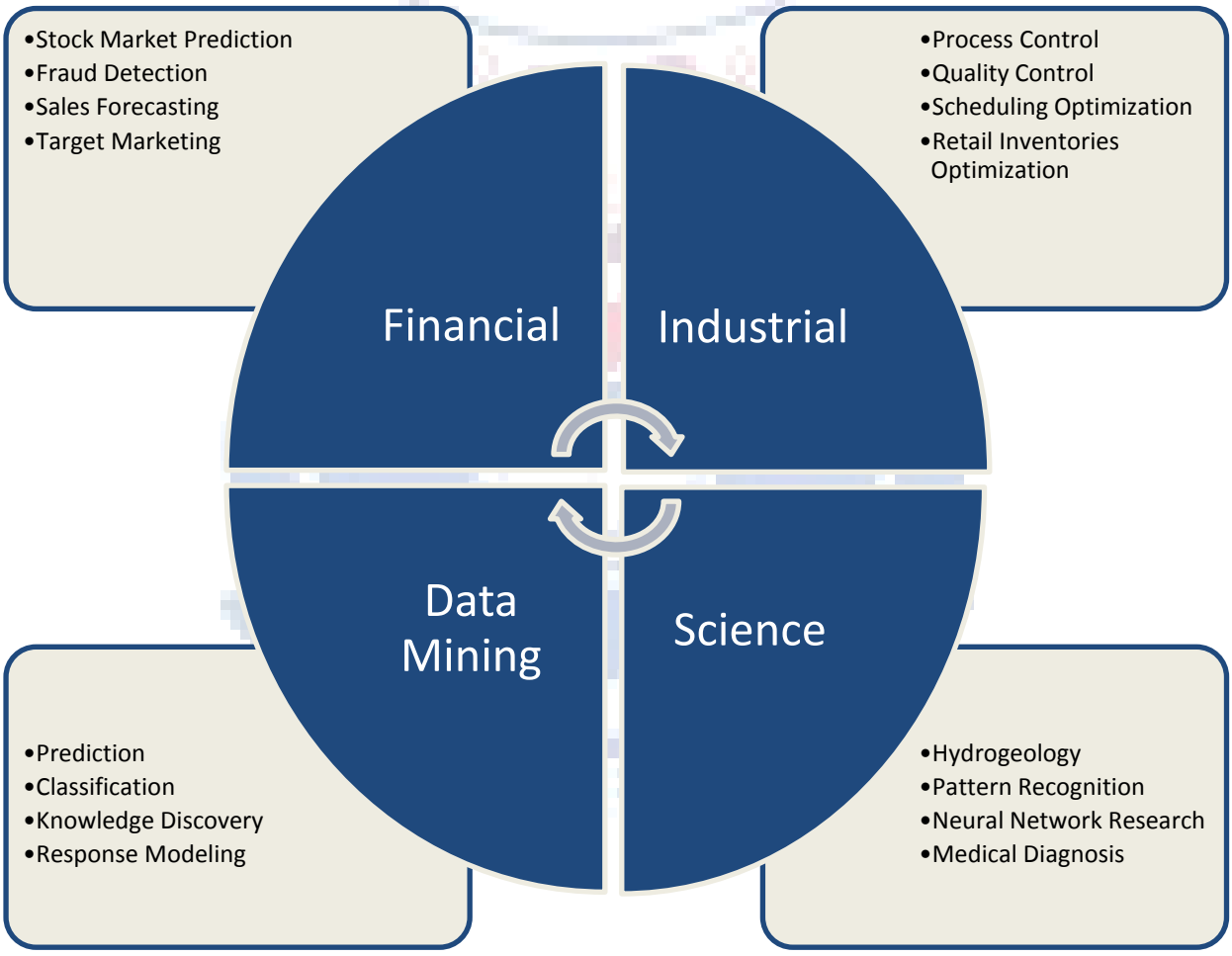


Figure 1.2 A synopses of specific applications of artificial neural networks in present day fields of work.

1.3. Applications

1.3.1. Function approximation

The need for function approximations arises in many branches of applied mathematics and computer science. In general, a function approximation problem asks us to select a function among a well-defined class that closely matches or approximates a target function in a task-specific way.

Function approximation problems can be defined in two major classes.

Firstly to approximate a known set of target functions, such as special functions - *Gamma* or *erfc* function, with a specific class of functions that have desirable properties. Typically, polynomials or rational functions will be employed since these functions are computational inexpensive, continuous, differentiable and have known limit values.

Secondly, the target function (*g*) may be unknown. In these situations, instead of an explicit formula only a set of points of the form (*x*, *g(x)*) is provided. Depending on the structure of the domain and sub-domain of g, several techniques for approximating g may used. Assuming that g is an operation on real numbers, techniques of interpolation, extrapolation, regression analysis and curve fitting can be used. If the sub-domain of *g* is a finite set, then the problem can be reworked to a classification problem and solved.

In artificial neural networks the application of regression and classification analysis has received a unified treatment in statistical learning theory, where it is viewed as a supervised learning problem.

1.3.2. Time series methods

In many fields such as statistics, signal processing and financial market analysis, a time series is a sequence of successive data points over a time period. Time series analysis comprises methods that attempt to understand such time series, often either to understand the underlying context of the data points (origin of phenomena observed or method by which it was generated) or to make forecasts (predicting future values). Time series forecasting is the use of a model to forecast future events based on known past events, this usually results in the use of multiple time series data to forecast a single set of future data points. A standard example in econometrics is the opening price of a share of stock based on its past performance and the financial index for that sector.

In recent work on model-free analyses, wavelet transform based methods (for example locally stationary wavelets and wavelet decomposed neural networks) have gained favour. Multi-scale or multi-resolution techniques decompose a given time series, attempting to illustrate time dependence at multiple scales. Thus, multiple time intervals are analysed for period variation and these variation sequences implemented in a feed-forward prediction method.

1.3.3. Statistical classification

The most widely used classifiers are:

- Artificial Neural Network with Multi-layer Perceptron,
- Support Vector Machines,
- k-Nearest Neighbours,
- Gaussian Mixture Model,
- Gaussian, Naive Bayes,
- Decision Tree and
- RBF classifiers.

1.3.4. Pattern recognition

The classification or description scheme usually uses one of the following approaches; statistical or structural recognition methods. Statistical pattern recognition is based on statistical characterisations of patterns, which is only possible if it is assumed that the patterns are generated by a probabilistic system. Structural pattern recognition is based on the structural interrelationships of features. A wide range of algorithms can be applied for pattern recognition, from very simple Bayesian classifiers to much more powerful neural networks.

An intriguing problem in pattern recognition is the relationship between the problem to be solved or data to be classified and the performance of various pattern recognition algorithms (classifiers). Initially one method might show a preferential classification time but as the data set increases or changes might fail completely or have significant reduction in efficiency. In this regard artificial neural networks have been commonly employed as a robust adaptive method for pattern recognition with the addition of fuzzy logic training sets.

1.3.5. Data processing

The utility of artificial neural network models lies in the fact that they can be used to infer a function from observations. This is particularly useful in applications where the complexity of the data or task makes the design of such a function by hand impractical. The determination of loosely defined relationships increases the data processing methodology of neural networks, creating pathways which might not have been perceived by a human data processor.

1.4. Theoretical properties of Artificial Neural Networks

1.4.1. Computational power

The multi-layer perceptron (MLP) is a universal function approximator, as proven by the Cybenko theorem[6]. The Cybenko theorem is a theorem states that a single hidden layer, feed forward neural network is capable of approximating any continuous, multivariate function to any desired degree of accuracy and that failure to map a function arises from poor choices for parameter values or an insufficient number of hidden neurons.

An investigation by Siegelmann and Sontag[7] has provided a proof that a specific recurrent architecture with rational valued weights (as opposed to the commonly used floating point approximations) has the full power of a Universal Turing Machine[8]. It was shown that the use of irrational values for weights resulted in a machine with trans-Turing power.

The combination of Cybenko's theorem and results from Siegelmann and Sontag significantly reduces the amount of computation resources required to create and apply an artificial neural network to a specified problem.

1.4.2. Capacity

Artificial neural network models have a property called - capacity, thus the capacity for a neural network to model any given function can be quantified using this term. It is directly related to the amount of information that can be stored in the network (layers) and the complexity of the internodes' connections.

1.4.3. Convergence

Convergence of artificial neural networks can be a problematic situation due to the non-linear behaviour of the perceptron and the method of connection in the network itself. Generally, convergence depends on a number of factors. Firstly, the existence of a single global minimum compared to the availability of many local minima which can cause convergence errors. The uniqueness of the solution depends on the cost function and the model architecture. Secondly, the optimization method used might not be guaranteed to converge when far away from a local minimum. This is typically observed with steepest decent methods compared to Newton-Raphson algorithms. Finally, the size of the problem might become unmanageable due to the large amount of data or parameters.

Regarding convergence in general, no theoretical guarantees exist on the convergence of a network and these methods are an unreliable guide to practical applications[9,10,11,12,13].

1.4.4. Generalisation and over-fitting

During the development of artificial neural networks to create a system that can generalise any possible type of example, the problem of overtraining has emerged. This problem arises due to the method in which the neural network is specified and constructed. The network as such has a significantly larger capacity than the parameters given to it, resulting in an over-specified system. In this instance the network requires more data than that which is available and must reiterate to determine its own internal variables such as the weights of the network and the bias of each node.

Two methods are available to avoid this problem of over-fitting. Firstly, the use of cross-validation and similar techniques to check for the presence of overtraining. This allows the network to select the optimal set of parameters to minimise the generalisation error. The second method uses some form of regularisation of input to data. The regularisation method can be assigned to a probabilistic framework - Bayesian. In these systems the regularisation is performed by selecting a significantly larger prior probability set over simpler models. It can also be found in statistical learning theory, where the goal is to minimise the empirical and the structural risk factors. These factors have a close correlation to the error over the training set and the predicted error in the data due to over-fitting.

1.4.5. Confidence analysis of a neural network

Supervised neural networks that use a mean squared error (MSE) performance or cost function can use formal statistical methods to determine the confidence of the trained model. The MSE on a validation set can be used as an estimate of the variance. This value can then be used to calculate the confidence interval of the output of the network, if it is assumed that it follows a normal distribution. The confidence analysis of the network is statistically valid, as long as the output probability distribution stays the same and the network is not modified.

In unsupervised neural networks only inputs exist and no definition is given for the results. In these instances a comparison of the data after training is made to determine the stability of the network.

1.5. Conclusion

A brief summary of the historical context of the development of artificial neural networks have been given. The initial concept of neurons in a mathematical implementation was first introduced by Hayek (1950) and subsequently improvements have been made which allows the use of these networks in complicated real world problems. Artificial neural networks have found applications in almost all settings of work environments, from finance to medicine. Major uses for neural networks are in function approximation, time series analysis, classification, pattern recognition and data processing. Estimation of computational power and capacity of neural networks to solve problems were briefly visited. The problem of convergence and over-fitting was discussed with likely solutions to the problem. Generalisation and confidence values in artificial neural networks were highlighted, with a focus on statistical methods.

1.6. Aims and Objectives

It is clear from the introduction that the application of artificial neural networks (ANNs) to the problem of surface water and groundwater interactions can be estimated. This would only be possible if the appropriate variables are considered in a model system. A further complicating factor on the development of an effective ANN system is the availability of data, due to human error gaps exist in data sets which include all three parameters, considered in this study and subsequently a patching algorithm needs to be developed to complete these data sets.

In this study three different case studies will be presented which might have enough data to successfully use an ANN method to determine surface water and groundwater interactions. Four different artificial neural network architectures will be investigated, in order to determine an optimal neural network configuration for use in hydrological parameter estimation. An initial approach would be used in which only one data point will be predicted in the future, subsequently the model system can be rerun on the new data set to produce further points in the future. It is hoped that this method would also reveal the predictive nature of the artificial neural network used to do the determination. Furthermore, a multiple step prediction will be considered to estimate more than a year into the future with a data set of known variables in this period.

With the above in mind, the following stepwise aims were set for this study.

1. A literature review of surface water and groundwater interactions, which will include possible mechanisms as well as human influences on the system.
2. A review of artificial neural network architectures, mechanism of action and learning paradigms.
3. Methods of patching data using various types of artificial neural networks and its application in surface water and groundwater.
4. Estimation of borehole water levels using a one step ahead or multistep prediction method. Furthermore, a error estimation will be done in order to determine the viability of this method from known data points.

Chapter 2

Surface Water and Groundwater Interaction

2.1. Introduction

The management of water resources in the past were controlled as if surface water and groundwater were two separate entities. The constant development of land and water resources has made it clear that these systems affect each other in resource quantity and quality over an extended time period. In a South African context the development of the country with the increase in industrial and population demand has created a scenario in which water rights are coming more to the fore front. Management systems and reservoir determinations are becoming more frequent, this is in a large part due to the South African government's insistence to determine the amount of water available to the country's citizens. The Department of Water Affairs and Forestry (DWAF) uses surface water models to predict future availability of water in South Africa, one key component is the accurate prediction of groundwater contribution to surface water systems. In this investigation localised interactions between surface water and groundwater systems will be investigated, although it is not envisaged that accurate quaternary scale predictions will be possible.

The building of dams has become quite a costly undertaking and government is slowly starting to take remedial measures against those individuals or companies which poach water from other licensed users. The two main river systems in South Africa, the Vaal River and the Orange River, is gaining more attention from protection agencies[14].

The hydrological cycle describes the continuous movement of water above, on and below the surface of the earth[15]. Water in the atmosphere plays the most significant role in the cycle of the hydrosphere. Precipitation can be nearly ascribed for all the freshwater available in the hydrological cycle. The frequency and quantity varies from region to region and also the form it takes; rain, snow or a combination of both, and can be variable in seasonality. Evapotranspiration in its role returns most of the water back to the atmosphere. Factors that influence evaporation are vegetation coverage and types of plant species present in the area. Accordingly much of the precipitation never reaches the oceans as surface and subsurface runoff before the water is returned to the atmosphere. The water on the surface of the earth can be easily visualised and constitute streams, lakes, dams, wetlands, bays and oceans. Another component of surface water deposits are snow and ice, *i.e.*, polar ice caps which contains mostly freshwater. The water below the surface of the Earth primarily consists of groundwater but it also includes water content in the surface zones (soil).

The interaction of these three zones plays a considerable part in sustaining life on earth. It is in this perspective that the management and understanding of surface water and groundwater interaction will play a vital role in our survival on this planet. Climate change will also affect the availability and distribution of surface water and groundwater aquifers. It is expected that the rainfall pattern in South Africa will decrease, with an increase in bursts of heavy rains in the rainy seasons. The combination of these two factors will reduce the availability of groundwater since South African aquifers will not receive a high enough recharge. In this regard the effective management of groundwater will be critical in ensuring our survival on this content. In the following sections in this chapter different aspects of surface water and groundwater interactions will be discussed. It will focus on situations pertinent to the South African environment.

2.2. Mechanism of groundwater and stream interaction

2.2.1. Streams

Nearly all surface-water features (streams, lakes, reservoirs, wetlands and estuaries) interact with groundwater. This interaction can be illustrated by using a gaining and losing stream definition for a stream flowing through an area, Figure 2.1. This interaction is only possible if the river bed is porous enough for water to flow through this region.

Streams can interact with groundwater in three basic ways; firstly, streams can gain water from the inflow of groundwater through the streambed and is defined as a gaining stream, Figure 2.1A. The discharge of groundwater into a stream channel can only occur if the altitude of the water table in the vicinity of the stream is higher than the altitude of the stream-water surface. This effectively causes a pressure gradient with water moving under natural forces towards the stream bed.

Secondly, a stream can lose water to groundwater by outflow through the streambed (losing stream, Figure 2.1C). Equally for surface water to seep into the saturated zone, the altitude of the water table in the vicinity of the stream must be lower than the altitude of the stream-water surface.

Finally a stream can be both a losing and gaining stream, *i.e.*, a gaining stream in the upper reaches and a losing stream in the lower reaches or *vice a versa* in the stream path area.

The flow of water can also be illustrated by contours of the water-table elevation, indicating a gaining stream by pointing in an upstream direction (Figure 2.1B) or losing stream by pointing in a downstream direction (Figure 2.1C) in the immediate vicinity of the stream.

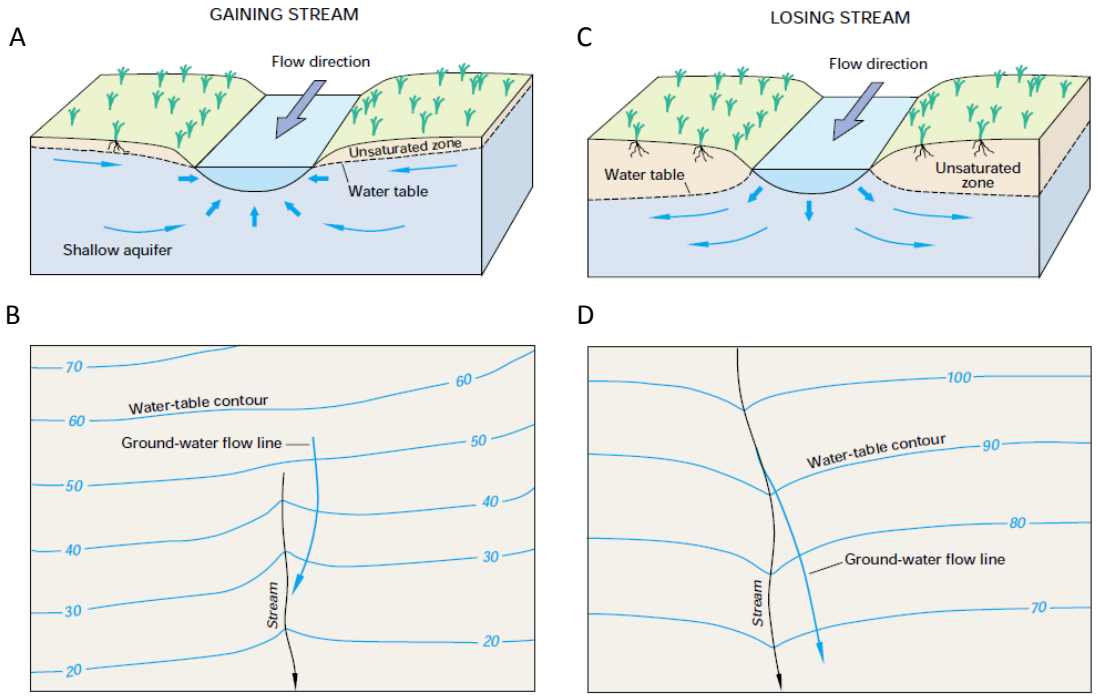


Figure 2.1 A graphical illustration of a gaining and losing stream in an area (above)[15]. Lower part depicts the water table contour in the region of the gaining or losing stream.

Not all streams are in constant interaction with groundwater sources and as such are defined as disconnected streams, Figure 2.2. A bounding layer in the stream bed exists through which water cannot penetrate or which has an extremely low hydraulic conductivity, creating a disconnected stream. Groundwater can only reach these water courses if an alternative surface pathway exists, such as springs, above the stream feeding it through an interflow mechanism.

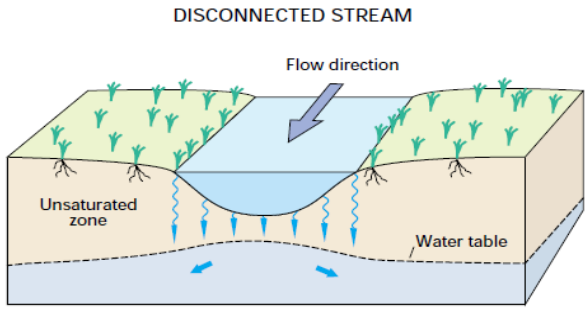


Figure 2.2 Diagrammatic representation of a disconnected stream[15].

2.2.2. Water abstraction and pollution

Surface water features gain water and solutes from groundwater systems and in other areas these surface water features act as sources of groundwater recharge. The chemical content of either the surface water feature or groundwater aquifer can cause significant changes in water quality of the system. A consequence of this interaction is that if excessive amount of water is withdrawn from streams, it can deplete the available groundwater in an area. Conversely, pumping of groundwater can seriously deplete the water levels in streams, lakes or wetlands if a connection exists with the groundwater source. Pollution of surface water can cause degradation of groundwater quality and equally pollution of groundwater can degrade surface water quality. Therefore, effective land and water management requires a clear understanding of the linkages between groundwater and surface water as it applies to any given hydrologic setting.

It should be noted however that the quality of groundwater in a natural system is generally better than the surface water quality, this enables us to analytically quantify the surface water and groundwater interaction, i.e., chloride method and isotope effects.

2.2.3. Bank storage

Another type of interaction between groundwater and streams that can occur during floods is bank storage (Figure 2.3). It takes place in nearly all streams at one time or another, and is a rapid rise in stream stage that causes water to move from the stream into the streambanks. This process usually is caused by storm precipitation, rapid snowmelt or release of water from a reservoir upstream.

As long as the rise in stage does not overflow the streambanks, most of the volume of stream water that enters the streambanks returns to the stream within a few days or weeks. The loss of stream water to bank storage and return of this water to the stream in a period of days or weeks tends to reduce flood peaks and later supplement stream flows.

If the rise in stream stage is sufficient to overflow the banks and flood large areas of the land surface, widespread recharge to the water table can take place throughout the flooded area. In this case, the time it takes for the recharged floodwater to return to the stream by groundwater flow may be weeks, months, or years because the lengths of the groundwater flow paths are much longer than those resulting from local bank storage[15].

Depending on the frequency, magnitude, and intensity of storms and on the related magnitude of increases in stream stage, some streams and adjacent shallow aquifers may be in a continuous readjustment from interactions related to bank storage and overbank flooding.

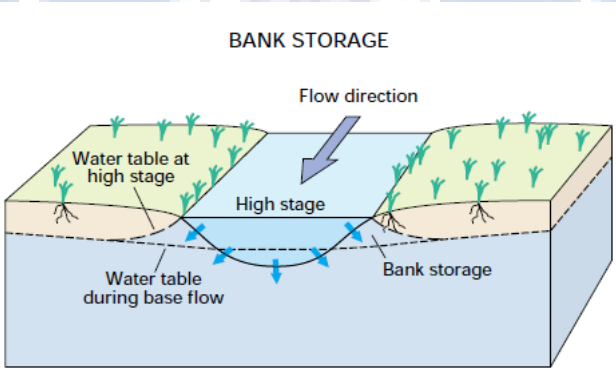


Figure 2.3 Figure showing bank storage during a flood event.

In addition to bank storage, other processes may affect the local exchange of water between

streams and adjacent shallow aquifers. Changes in streamflow between gaining and losing conditions can also be caused by pumping groundwater near streams. Pumping can intercept groundwater that would otherwise have discharged to a gaining stream, or at higher pumping rates it can induce flow from the stream to the aquifer.

2.3. The human influence on groundwater and surface water interaction

Human activities commonly affect the distribution, quantity and chemical quality of water resources. Human activities affect the interaction of groundwater and surface water over a broad range of areas. In the following discussion a survey of human activities that have a direct influence on this interaction will be presented. To provide an indication of the extent to which humans affect the water resources, some of the most relevant activities will be highlighted.

2.3.1. Agricultural Development

Agriculture is seen as the pivotal development of humanity to move from a nomadic lifestyle to a civilized existence. To support agricultural activities significant modifications to the world landscape was required. Tillage of land changes the infiltration and runoff characteristics of the land surface, which in turn affects the recharge to groundwater. The change in landscape features also influenced the delivery of water and sediment to surface-water bodies and evapotranspiration. Agriculturalists are aware of the substantial negative effects of agriculture on water resources and have devised methods to improve some of these effects. An example is the change in tillage practices which have been modified to maximize retention of water in soils and to minimize erosion of soil from the land into surface-water bodies. This was simply done by only tilling the top 5 cm of the soil layer instead of the standard 30 – 50 cm's. Four activities which have an impact on the interaction of groundwater and surface water are irrigation, application of pesticides, herbicides and artificial fertilizers to croplands.

Irrigation systems

Since the time of Samaria and Babylon people have been using surface water irrigation systems to irrigate their crop fields. In modern times surface water irrigation systems symbolize some of the largest integrated engineering works undertaken by humans. In South Africa great water distribution

systems have been installed around our great rivers. Water supply to remote areas has also been introduced to allow farming activities to continue. The Orange River and Caledon River have been specially adapted in parts to allow water to be transported over great distances for agricultural activities. In the current environment many irrigation systems that initially used only surface water now also use groundwater. The pumped ground water commonly is used directly as irrigation water, but in some cases the water is distributed through a system of canals. In arid regions, such as the Free State and Northern Cape, extensive use of groundwater is made to cultivate large areas causing a reduction in water levels in those regions.

Although early irrigation systems made use of surface water, the development of large-scale sprinkler systems or pivots in recent decades has greatly increased the use of groundwater for irrigation. The proliferation of pivots in South Africa is largely due to the following reasons: (1) A reticulation system is not needed, (2) groundwater is more readily available and (3) many types of sprinkler systems can be used on irregular land surfaces. Whether groundwater or surface water was used first to irrigate land, it was not long before water managers recognized that development of either water resource could affect the other.

The influence of chemicals on the quality of water as it moves through cropfields can be significant. Herbicides and pesticides can destroy microbial systems in the soil which might have an effect on the soil to retain moisture. The water lost to evapotranspiration is relatively pure, resulting in the chemicals being left behind to precipitate as salts in the unsaturated zone which can accumulate to dangerous levels. The build-up continues as long as irrigation activities persist, resulting in the total dissolved solids increasing in concentration in the irrigation return flow water. The return water has in some cases a significantly higher concentration than the original irrigation water. In order to prevent excessive build-up of salts in the soil, irrigation water in excess of the needs of the crops is required to dissolve and flush out the salts and transport them to the groundwater system. Once these dissolved solids reach high enough concentrations, the artificial recharge from irrigation return flow can result in degradation of the quality of groundwater.

2.3.2. Urban and Industrial Development

The contamination of surface water features are an expected side effect of urbanization. Typical pollution sources in an urban area are direct discharges of sewage-treatment plants, industrial

facilities and stormwater drains into river systems. These facilities and structures commonly add sufficient loads of a variety of contaminants to streams to strongly affect the quality of the stream for a long distance downstream. Depending on the relative flow magnitudes of the point source and of the stream, discharge from a point source such as a sewage-treatment plant may represent a large percentage of the water in the stream directly downstream from the source. In most of South Africa these contaminants find their way into our streams and rivers, an example is the Vaal River with a large part situated along informal settlements. The contaminants in streams can easily affect groundwater quality if a seepage zone exists which recharges the groundwater, *i.e.*, during excessive groundwater abstraction resulting in boreholes withdrawing water from the streams or after heavy floods caused stream water to become bank storage.

Point sources of contamination to groundwater can include septic tanks, drop latrines, fluid storage tanks, landfills, tailings dams and industrial areas. Three scenarios exist for these systems; the contaminant is totally soluble (solute transport), the contaminant is a sparingly soluble lighter than water non-aqueous phase organic compound (LNAPL), the contaminant is a sparingly soluble denser than water non-aqueous phase organic compound (DNAPL). The rates by which these components move through a groundwater system can be highly variable and the effective total contamination concentration received by the surface water body or person.

If the contaminant is soluble or sparingly soluble in water and reaches the water table, the contaminant will be transported by the slowly moving groundwater. If the source continues to supply the contaminant over a period of time, the distribution of the dissolved contaminant will form a characteristic plume shape, which spreads outwards from the point source. These contaminant plumes commonly discharge into a nearby surface water body or is pumped from the aquifer. If both the concentration of the contaminant and the rate of discharge of plume water are relatively small compared to the volume of the receiving surface water body, the discharging contaminant plume will have only a small effect on the quality of the receiving surface water body.

Furthermore, natural biogeochemical processes may decrease the concentration of the contaminant as it is transported through the groundwater system and the vadose zone. In contrast if the discharge of the contaminant plume is large or has a high concentration of contaminants, it could significantly affect the quality of the receiving surface water body.

2.3.3. Drainage of the Land Surface

In landscapes that are relatively flat and marshy, drainage of the land is a common practice preceding agricultural and urban development. Drainage can be accomplished by constructing open ditches, by burying tile drains beneath the land surface or actively pumping the terrain dry (dewatering). The drainage of lakes and wetlands can change the regional recharge and discharge of groundwater, which can result in significant changes in the biota that are present. Furthermore, these changes can ultimately affect the groundwater contribution to baseflow to streams, which in due course influences the riparian ecosystem. Artificial drainage of areas also alters the water-retention capacity of region as well as increasing the surface runoff rates from land having very low slopes. Urban development creates more efficient runoff systems, resulting in a decrease of recharge rates to the groundwater system. The probability of flooding in this area and lower lying areas are increased, *i.e.*, Cape Flats.

2.3.4. Modifications to River Valleys

Construction of levees

Levees are built along riverbanks to protect adjacent lands from flooding. These structures commonly are very effective in containing smaller magnitude floods that are likely to occur regularly from year to year. Large floods that occur much less frequently, however, sometimes breach the levees, resulting in widespread flooding as observed in New Orleans, America. Flooding of low-lying land is the most visible and extreme example of the interaction of groundwater and surface water.

During flooding, recharge to groundwater is continuous; given sufficient time, the water table may rise to the land surface and completely saturate the shallow aquifer. Under these conditions, an extended period of drainage from the shallow aquifer takes place after the floodwaters recede. The irony of levees as a flood protection mechanism is that if levees fail during a major flood, the area, depth, and duration of flooding in some areas may be greater than if levees were not present.

Construction of reservoirs

The primary purpose of reservoirs is to store water for uses such as public water supply, irrigation, flood attenuation and generation of electric power. Reservoirs also can provide opportunities for recreation and wildlife habitat. Water needs to be stored in reservoirs because stream flow in South

African rivers is highly variable. Stream flow can vary daily in response to individual storms and seasonally in response to variation in weather patterns. In South Africa an extensive network of dams and reservoirs exist to supply potable water to the public.

The effects of reservoirs on the interaction of groundwater and surface water are the greatest near the reservoir and directly downstream from it. Reservoirs can cause a permanent rise in the water table that may extend a considerable distance from the reservoir, because the base level of the stream, to which the groundwater gradients had adjusted, is raised to the higher reservoir levels. Near the dam, reservoirs commonly lose water to shallow ground water, but this water commonly returns to the river as groundwater contribution to baseflow directly downstream from the dam. In addition, reservoirs can cause temporary bank storage at times when reservoir levels are high. In some cases, this temporary storage of surface water in the ground-water system has been found to be a significant factor in reservoir management[15].

Human-controlled reservoir releases and accumulation of water in storage may cause high flows and low flows to differ considerably in magnitude and timing compared to natural flows. As a result, the environmental conditions in river valleys downstream from a dam may be altered as organisms try to adjust to the modified flow conditions.

A second type human built dams that can act as reservoirs are tailings dams of mining companies, the influence these structures have on the surrounding area is slowly coming to the fore in South Africa. The release of these waters is neither intentional nor controlled but in some areas a surface water interaction is observed with groundwater.

At the Institute for Groundwater Studies a conceptual model for the interaction of surface water and groundwater has been proposed. The conceptual model is based on the idea that the most significant interaction between surface water and groundwater occurs at the riparian zone. Considering recharge as an illustrative example, it can be shown that the effective recharge outside of the riparian zone is significantly lower than inside the riparian zone. Typical recharge values for the Karoo area is in the order of 1 – 5 %, while in the riparian zone it can be in the order of 15 % due to the presence of alluvium. Furthermore, the transmissivity of the riparian zone is significantly higher than the transmissivity of the surrounding area. If the conceptual model is deconstructed, it can be divided into three sections, i.e., a general area, a riparian zone and a channel (Figure 2.4). The

channel itself can either have a confining bed with a low or high transmissivity to allow groundwater to play a role in the water level in the river system.

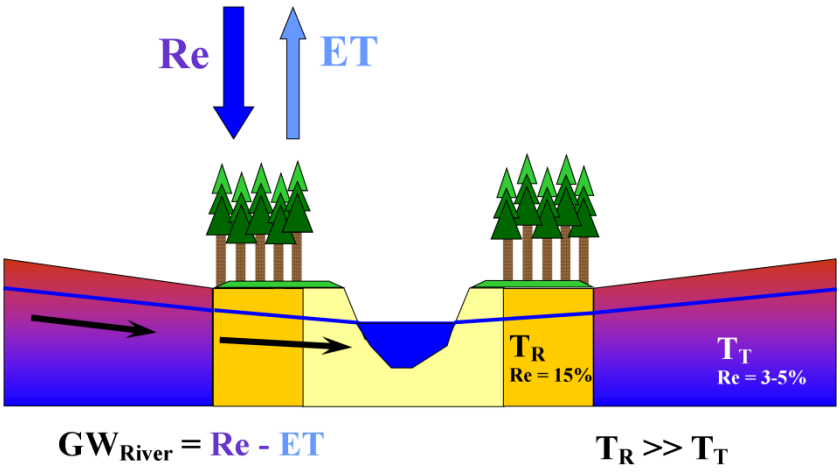


Figure 2.4 A schematic representation of a conceptual model indicating the interaction between surface water and groundwater. The most important parameters are shown as well as their relative orders.

2.3.5. Removal of natural vegetation

Deforestation and invader plant species can cause havoc with a regions surface water and groundwater interactions. The plantation of forests on riverbanks and the black wattle problem can effectively choke streams and lower the water table in the area. The removal of natural vegetation to make land available for agriculture and urban growth, sometimes involves cutting of forests and removal of riparian vegetation and wetlands. Forests and woodland areas have a significant role in the hydrologic regime of watersheds. Deforestation tends to decrease evapotranspiration, increase storm runoff and soil erosion, and decrease infiltration to groundwater and groundwater contribution to baseflow of streams.

Some of the important functions of riparian vegetation and riparian wetlands include preservation of aquatic habitat, protection of the land from erosion, flood mitigation, and maintenance of water quality. Destruction of riparian vegetation and wetlands removes the benefits of erosion control and flood mitigation, while altering aquatic habitat and chemical processes that maintain water quality[15]. A monitoring program from the South African Environmental Observation Network (SAEON) which

has been active from 1938 has shown the following trends in the planting of invasive species (forestry):

1. The onset of stream flow reduction occurred in ca. 5 years.

2. Local stream flow levels are strongly dependant on the plantation age.

3. Peak stream flow reduction occurred at ca. 15 years.

4. Generally, 30 – 40 mm of stream flow reduction occurred per 10 % of catchment area planted (at peak water usage).

If invasive species of trees are cleared from riparian zones an increase of 8000 to 12000 liters per hectare per day in the winter rainfall area is observed, while up to 34000 liters per hectare per day can be gained in summer rainfall areas. This observation makes it critical to quantify the evapotranspiration from the area in the reserve determination.

2.3.6. Quantifying the Impact of Human activities

Reserve determinations usually include a water balance in which the requirements from the environment and the requirements of human activities are calculated. Typically, twenty five liters per day per person is allocated according to law as a basic human need (**BHN**) requirement. This is followed by groundwater usage from agriculture, industrial and domestic use (**USE**). On the environmental side of the equation recharge from rainfall (**Re**), inflow (**IN**) and outflow (**OUT**) of groundwater in the area plays a significant part in the determination of the reserve. Finally, an allocation for the riparian vegetation and biota are included as a fixed requirement. This can in part be allocated to baseflow to rivers or streams (**Baseflow**) and daily evapotranspiration (**ET**). Unfortunately baseflow is mostly over estimated in the determination of the water balance, with the daily evapotranspiration rates being an approximated. All these parameters can be simply visualized from the diagram in Figure 2.5.

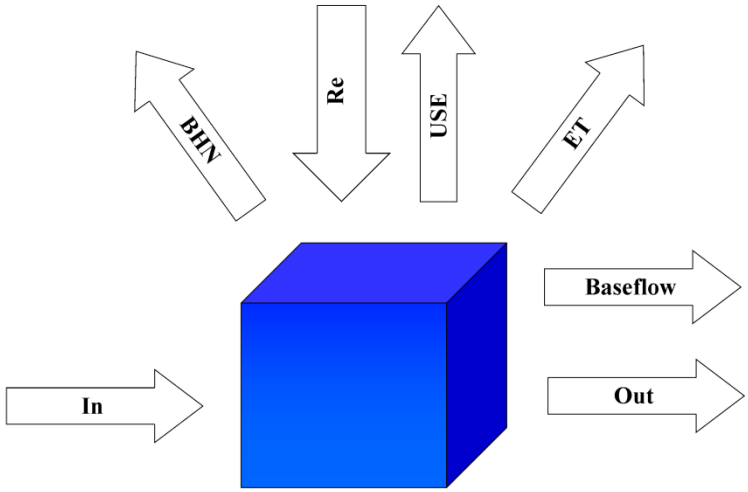


Figure 2.5 A schematic representation of the reserve determination for an area.

Finally, the analytical determination of surface water and groundwater interactions can be determined from methods such as the chloride and isotope ratio method. When using the chloride method, it is generally assumed that the formations in the area does not contain significant evaporate deposits which can alter the result or that the site is located close to the shoreline. In the isotope method for shallow aquifers local enrichment is assumed due to evaporation and precipitation patterns. These are not the only methods available; however, they are commonly employed to estimate recharge or source of groundwater. Further difficulties include the availability of continues data for rainfall, weirs and boreholes close to the area of interest. A significant impediment in quantifying surface water and groundwater interactions is weir design to yield accurate low flow values as well as flood event volumes. These restraints coupled with budget constraints and insufficient human capacity complicates the evaluation of surface water and groundwater interactions.

2.4. Conclusion

The development of an interaction model between surface water and groundwater has been presented and the method by which this forms a component of the total hydrological cycle. A summary of likely mechanisms that affect the groundwater and surface water interaction was discussed. Bank storage on groundwater recharge was briefly summarised. The role that human

activities in agricultural, industrial and urban development's have on water resources has been illustrated. Human intervention in the shape of rivers and flow patterns has been discussed with special reference to reservoir systems and levees. The cumulative effect of deforestation and alien invasive species on groundwater and surface water features have been described as well as the influence it has on the riparian zone and biota of the local region.

In the following chapter the construction and training of neural networks will be presented. This is done to give a more substantial background on these network systems, and clarify the procedures that will be used to apply it to groundwater and surface water interactions.

Chapter 3

Neural Network Design and Programming

3.1. Introduction

In Chapter 1 the general groundwork for the application of neural networks, including a short historical background, was given. In this chapter the development of neural networks in the design and programming phase will be explained. Furthermore, the working methodology of artificial neural networks will be emphasised.

It is generally accepted that daily streamflow and rainfall generation processes are seasonal and nonlinear. These processes have distinct seasonal means and variances which are linked with low or high flow activity time periods[16]. Many models exist which use a divide-and-conquer approach, which fragments the problem into manageable sections and later constructs the solution from the respective segments[17]. Local linear models such as the periodic autoregressive moving average (PARMA) and the abbreviated version periodic autoregressive (PAR) models are widely used to model hydrological time series data[18]. These methods approximate a complex problem locally with linear models.

A second approach exists in which local non-linear models are used to approximate complex problems locally with nonlinear models such as nonlinear regression or artificial neural network (ANN) models. Artificial neural networks are known for having the ability to model nonlinear mechanisms. The use of these models have been increasingly applied to a range of hydrological

problems[19,20]. It has been suggested in the literature that a single ANN cannot predict multiple event scenarios, such as low- and high-runoff events[21]. One of the main reasons given for this inability is that during the different runoff events the underlying mechanism of streamflow generation is quite different.

In the following sections the types of neural networks, training methods and architecture will be discussed. The section will initially start with the building blocks of artificial neural networks and then move on to the more advanced concepts[22].

3.2. The neuron model

3.2.1. A simple neuron

The most fundamental starting point in artificial neural networks is the simple neuron with a single scalar input and no bias (Figure 3.1, left side)[23]. The scalar input *p* is transmitted through a connection that multiplies its strength by the scalar weight *w* to form the product w·p which is again a scalar value. The weighted input *w·p* is the only argument of the transfer function *f*, which produces the scalar output *a*.

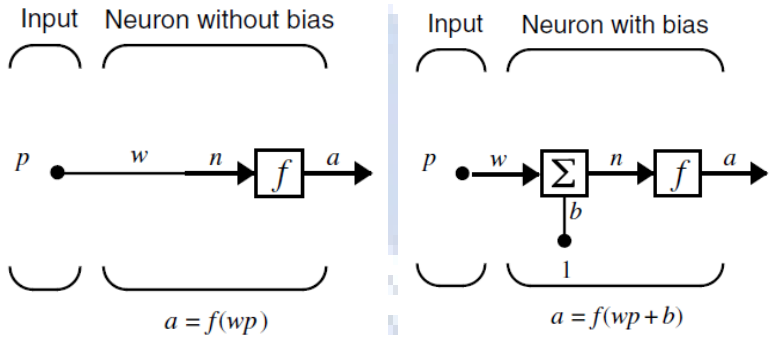


Figure 3.1 A simple neuron with no bias (left side) and a neuron with a bias factor implemented (right side)[22].

The neuron on the right in Figure 3.1 has a scalar bias, *b*. The bias is simply the addition of a value to the product of *w·p*, it acts as *a* if it is shifting the function *f* to the left by an amount *b*. The bias is much like a weight, except that it has a constant input of 1. The transfer function net input *n*, again a scalar, is the sum of the weighted input *w·p* and the bias *b*. This sum is the argument of the transfer function *f*. Here *f* is a transfer function, typically a step function or a sigmoid function, that takes the

argument *n* and produces the output *a*. It should be noted that *w* and *b* are both adjustable scalar parameters of the neuron. However, the constant bias of 1 that drives the bias is an input and must be treated as such when you consider the linear dependence of input vectors. The central idea of neural networks is that such parameters can be adjusted so that the network exhibits some desired or interesting behaviour.

3.2.2. Transfer Functions

A variety of transfer functions exist which can be used to construct an artificial neural network neuron. In many software packages these transfer functions are already built-in the one software package that will be used in the current investigation is Matlab with the Neural Network Toolbox enabled. The following four most commonly used functions are discussed below.

Hard-Limit Transfer Function

The hard-limit transfer function shown in Figure 3.2, limits the output of the neuron to either 0 if the net input argument *n* is less than 0, or 1 if n is greater than or equal to 0. This function is used to create neurons that make classification decisions.

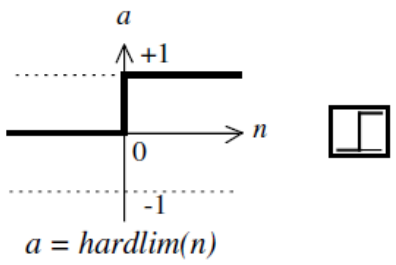


Figure 3.2 The hardlimit transfer function[22].

Linear Transfer Function

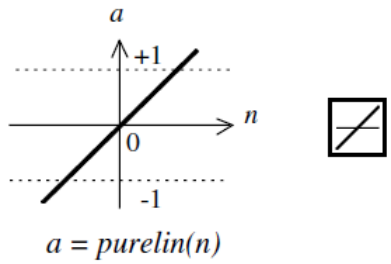


Figure 3.3 The linear transfer function[23].

Neurons of this type are used as linear approximators in linear filters (Figure 3.3). The linear transfer function calculates the neuron's output by simply returning the value passed to it. This neuron can be trained to learn an affine function of its inputs, or to find a linear approximation to a nonlinear function. A linear network cannot be made to perform a nonlinear computation.

Log-sigmoid transfer function

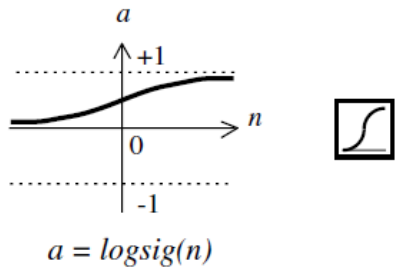


Figure 3.4 Log-simoig transfer function[23].

The sigmoid transfer function (Figure 3.4) takes the input received from the neuron and reduces the value so that the output ranges from 0 to 1. These input values can have any value between plus and minus infinity. This transfer function is commonly used in backpropagation networks[24], in part because it is differentiable.

Tan-sigmoid transfer function

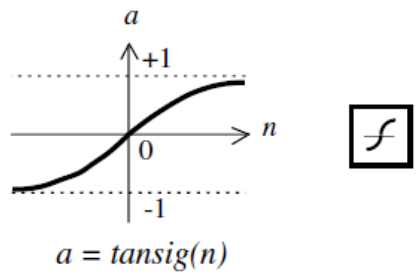


Figure 3.5 Tan-sigmoid transfer function[22].

The sigmoid transfer function (Figure 3.5) takes the input and reduces the output into the range -1 to 1. The input value, similar to the logsig type values, can have any value between plus and minus infinity. This transfer function is commonly used in backpropagation networks, in part because it is differentiable.

3.2.3. A neuron with a vector input

A neuron with a single R-element input vector is shown below in Figure 3.6. The individual element inputs p_1, p_2,... p_R are multiplied by the weights $w_{1,1}$, $w_{1,2}$, ... $w_{1,R}$ and the weighted values are fed to the summation junction. The sum is simply $W \cdot p$, the dot product of the single row matrix W and the vector p.

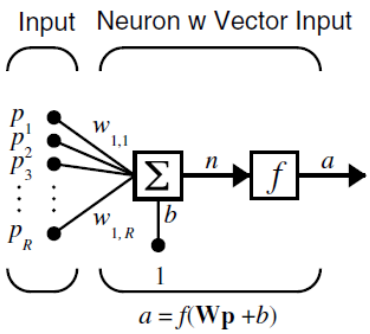


Figure 3.6 A neuron with a vector input and a single value output[23].

The neuron has a bias b, which is summed with the weighted inputs to form the net input n. This sum, n, is the argument of the transfer function f.

$$n = w_{1,1}p_1 + w_{1,2}p_2 + ... + w_{1,R}p_R + b$$

Abbreviated Notation

At this stage a reduction in the detail in Figure 3.6 might be required, since it is only a single vector input. Once the networks with many neurons or multiple layers of many neurons are used, the amount of detail represented in Figure 3.6 might become confusing or lost. It is in this regard that an abbreviated notation for an individual neuron exists, which retains the visual elegance of the simple neuron. This notation is shown in Figure 3.7.

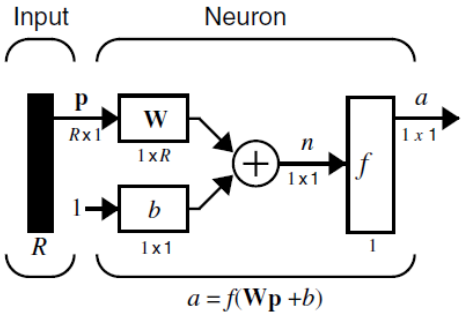


Figure 3.7 Abbreviated notation for a neural network[23].

In Figure 3.7 the input vector *p* is represented by the solid dark vertical bar at the left. The dimensions of *p* are shown below the symbol *p* in the figure as *Rx1*. The input vector *p* is a vector of R input elements. These inputs are multiplied (dot product) with the single-row R-column matrix *W*. A constant 1 enters the neuron as an input, which is multiplied by a scalar bias *b*. Thus the net input to the transfer function *f* is *n*, the sum of the bias *b* and the dot product $W \cdot p$. This sum is then passed to the transfer function *f*, which operates on it to give the neuron's output *a*. In this instance the output is a single scalar value. The inclusion of more than one neuron in the network will result in a vector being returned as the result. In all instances the graphical representation stays the same.

A layer of a network was defined in Figure 3.7. A layer includes the combination of the weights, the multiplication and summing operation, the bias *b*, and the transfer function *f*. The array of inputs, vector *p*, is not included in or called a layer. Each time this abbreviated network notation is used, the sizes of the matrices are shown just below their matrix variable names. This notation allows the user to understand the architectures and follow the matrix mathematics associated with them.

In the transfer function f many different types of functions can be placed, Figure 3.8. When a specific transfer function is to be used in a figure (*i.e.* logsig), the symbol for that transfer function replaces the f shown Figure 3.8.

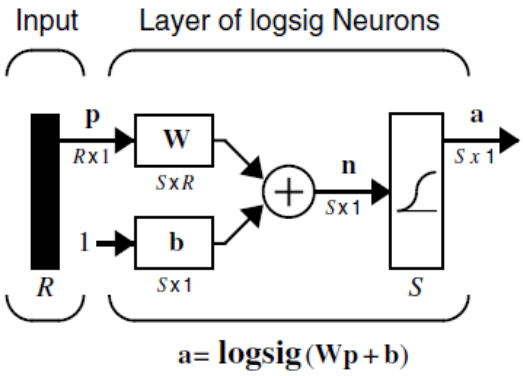


Figure 3.8 A graphical illustration of the transfer function in a vectorised input example[22].

3.3. Network Architectures

Two or more of the neurons shown in earlier sections can be combined in a single layer. A network is the cluster of such a set of layers into one unit. The amount of layers does not matter, only that they are combined to form a single network. In this section different types of network architecture will be presented, as these units will be used to model groundwater and surface water interactions. First consider a single layer of neurons.

3.3.1. A Layer of Neurons

A one-layer network with R input elements and S neurons are shown in Figure 3.9. In this network, each element of the input vector *p* is connected to each neuron input through the weight matrix *W*. The i^{th} neuron has a summation that gathers its weighted inputs and bias to form its own scalar output $n_{(i)}$. The various $n_{(i)}$ taken together form an S-element net input vector *n*. Finally, the neuron layer outputs form a column vector *a*. The expression for a is the transfer function of $W \cdot p + b$ or mathematically $a = f(W \cdot p + b)$.

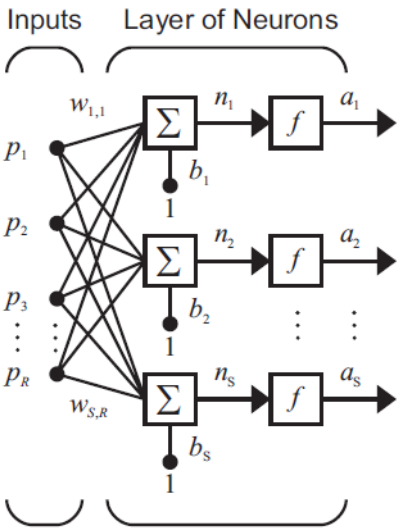


Figure 3.9 Single layer neuron network[22].

It should also be highlighted that the number of inputs to a layer is usually different from the number of neurons, *i.e.*, R is not necessarily equal to S. A layer is not constrained to have the number of its inputs equal to the number of its neurons.

A single (composite) layer of neurons can be created containing different transfer functions, by simply placing the two networks in parallel. Both these networks would have the same inputs and each network would in turn create some of the outputs.

The input vector elements entered into the network through the weight matrix *W* would be

$$W = \begin{bmatrix} w_{1,1} & \cdots & w_{1,R} \\ \vdots & \ddots & \vdots \\ w_{S,1} & \cdots & w_{S,R} \end{bmatrix}$$

In the matrix *W*, the row indices indicate the destination neuron of the weight, and the column indices indicate the source of the input for that weight.

The S neuron R input one-layer network also can be drawn in abbreviated notation (Figure 3.10). The input vector *p* has an (1,R)-dimension, *W* (SxR)-matrix, and *a* and *b* are (1,S)-dimension vectors.

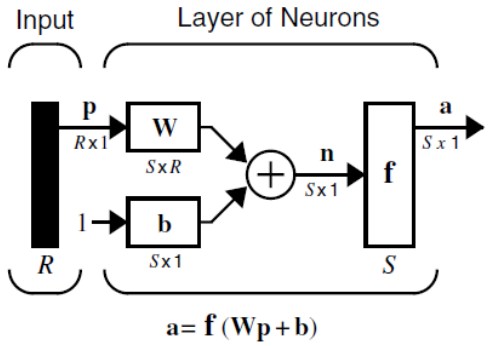


Figure 3.10 Abreviated form for a S neuron and R input one layer network[22].

3.3.2. Inputs and Layers

In order to simplify and clarify the notation used in artificial neural networks, a distinction needs to be made between weight matrices that are connected to inputs and weight matrices that are connected between layers. The weight matrices must also identify the source and destination from which it is derived. Weight matrices that are connected to inputs are termed input weights and weight matrices resulting from layer outputs, layer weights. Furthermore superscripts will be used to identify the source (second index) and the destination (first index) for these various weights and other elements in the network.

In Figure 3.11 the one-layer multiple input network is shown in the abbreviated form with superscripts indicating the relevant source and destination of layer components. The weight matrix connected to the input vector p is labelled as an input weight matrix ($IW^{1,1}$) having a source 1 (second index) and a destination 1 (first index). Elements of layer 1, such as its bias, net input, and output have a superscript 1 to say that they are associated with the first layer. Multiple Layers of Neurons uses layer weight (*LW*) matrices as well as input weight (*IW*) matrices.

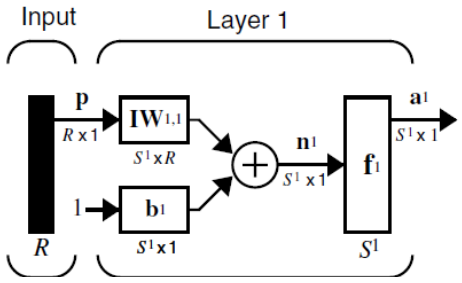


Figure 3.11 Illustration of superscript definition in determining the origin and destination of elements in a neural network[23].

3.3.3. Multiple Layers of Neurons

A network can have several layers that can be connected in a multitude of configurations. Each layer has a weight matrix *W*, a bias vector *b*, and an output vector *a*. To distinguish between the weight matrices, output vectors, etc., for each of these layers in the figures, the number of the layer is appended as a superscript to the variable of interest. This scheme is presented in Figure 3.12 to illustrate the layer number method as well as the cumulative layers have on the complexity of the neural network.

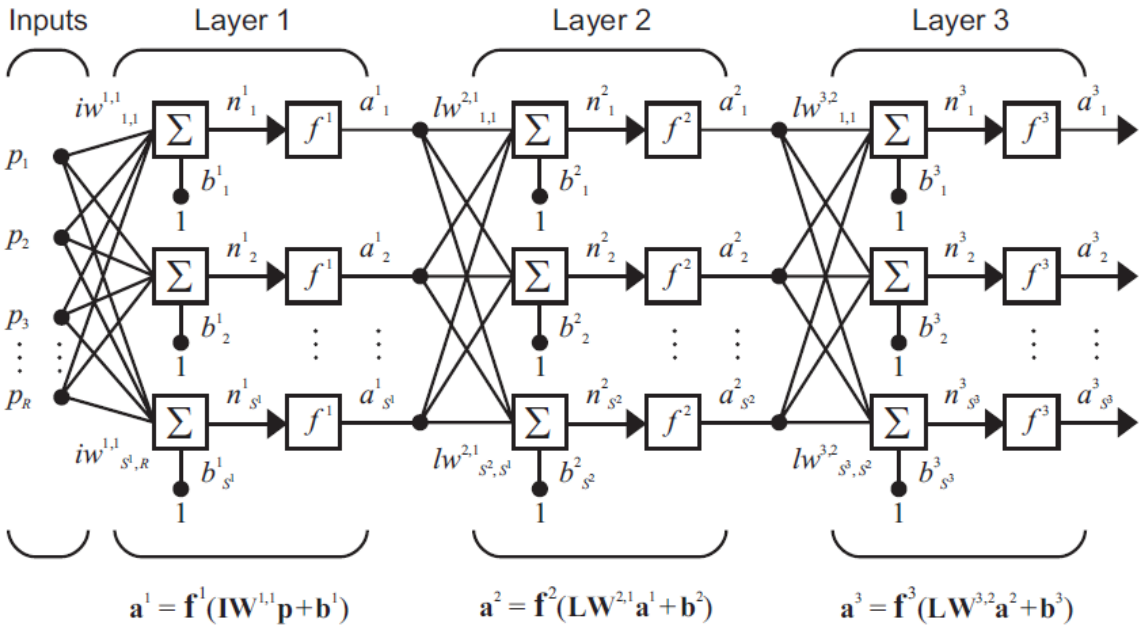


Figure 3.12 Multi-layer network, with numbered layer segments[23].

The network in Figure 3.12 has R_1 inputs, S_1 neurons in the first layer, S_2 neurons in the second layer, etc. It is common for different layers to have different numbers of neurons. A constant input 1 is fed to the bias for each neuron.

Note that the outputs of each intermediate layer are the inputs to the following layer. Thus layer 2 can be analyzed as a one-layer network with S_1 inputs, S_2 neurons, and an S_2xS_1 weight matrix W_2. The input to layer 2 is a_1; the output is a_2. Now that all the vectors and matrices of layer 2 have been identified, it can be treated as a single-layer network on its own. This approach can be taken with any layer of the network.

The layers of a multilayer network play different roles. A layer that produces the network output is called an output layer. All other layers are called hidden layers. The three-layer network shown earlier has one output layer (layer 3) and two hidden layers (layer 1 and layer 2).

The same three-layer network can also be drawn using abbreviated notation (Figure 3.13).

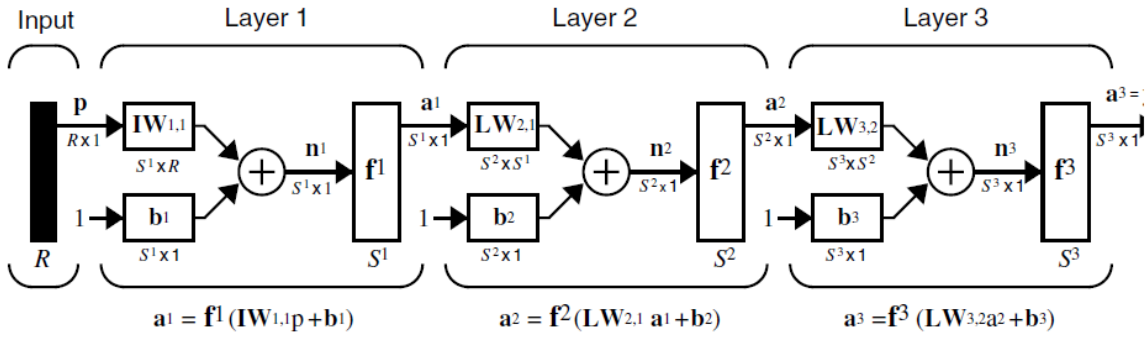


Figure 3.13 Three layer network of Figure 3.12 in the abbreviated format[23].

3.3.4. Pre- and post-processing

Neural networks require that the information passed to it must be in a useable format; this usually involves pre-processing to transform the input data to a form that is easier or more efficient for a network. Typically, during this pre-processing step the data will be transformed to fall into the interval [-1, 1] or [0, 1], depending on the transfer function used. This can speed up learning for many networks.

Similarly, network outputs can also have associated processing functions. The post-processing functions transform the network output in the reverse-processes to produce output data with the same characteristics as the original user-provided targets.

3.4. Types of neural networks

In this section a brief review of neural networks will be given. The most notable network architectures will be discussed in more detail, since these methodologies will be used in the application sections.

3.4.1. Feedforward neural network

The feedforward neural network was the first and arguably simplest type of artificial neural network devised. In this network the information moves in only one direction, forward, from the input nodes

through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network (Figure 3.14).

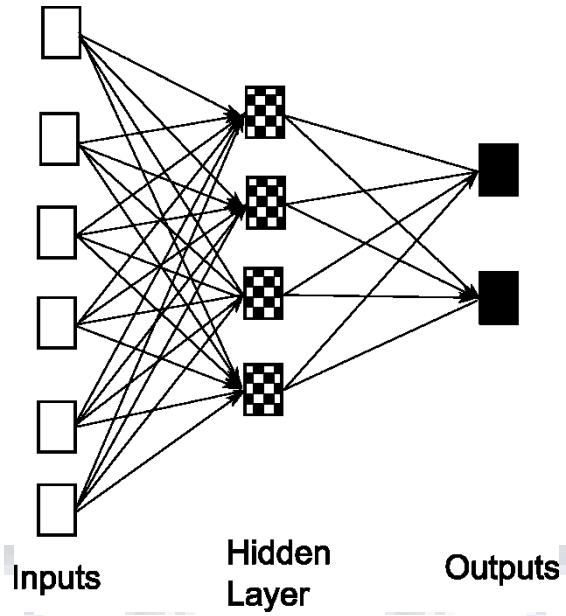


Figure 3.14 Diagramatic representation of a feedforward network.

3.4.2. Radial basis function (RBF) network

Radial Basis Functions (RBF) are powerful techniques for interpolation in multidimensional space. A RBF is a function which has a distance criterion build into the function. Radial basis functions have been applied as a substitute for sigmoidal functions in a hidden layer section in multi-layer perceptrons (MLP). The RBF network has two layers of processing; firstly, input is mapped onto each RBF in the 'hidden' layer. In regression problems the output layer is then a linear combination of hidden layer values representing mean predicted output. The interpretation of this output layer value is the same as a regression model in statistics.

RBF networks have the advantage of not suffering from local minima in the same way as Multi-Layer Perceptrons. This is because the only parameters that are adjusted in the learning process are the linear mapping from hidden layer to output layer. Linearity ensures that the error surface is quadratic and therefore has a single minimum.

The drawback of RBF networks is that it requires a good coverage of the input space by radial basis functions. The RBF centres are determined with reference to the distribution of the input data, but does not account for the distribution of the prediction task.

3.4.3. Kohonen self-organizing network

The self-organizing map (SOM) invented by Teuvo Kohonen[25] performs a form of unsupervised learning. A set of artificial neurons learn to map points in an input space to coordinates in an output space. The input space can have different dimensions and topology from the output space and the SOM will attempt to preserve these.

A self-organizing feature map network identifies a winning neuron using the same procedure as employed by a competitive layer. However, instead of updating only the winning neuron, all the neurons within a certain area of the winning neuron are updated using the Kohonen rule.

3.4.4. Recurrent network

Recurrent neural networks (RNNs) are models with bi-directional data flow; this is in contrast to the feedforward networks with a unidirectional data flow path.

Simple recurrent network - Elman

A simple recurrent network (SRN) is a variation on the Multi-Layer Perceptron, sometimes called an Elman network due to its invention by Jeff Elman[26]. A three-layer network is used, with the addition of a set of context units in the input layer. There are connections from the middle (hidden) layer to these context units fixed with a weight of one. At each time step, the input is propagated in a standard feed-forward fashion, and then a learning rule (usually back-propagation) is applied. The fixed back connections result in the context units always maintaining a copy of the previous values of the hidden units (since they propagate over the connections before the learning rule is applied). Thus the network can maintain a sort of state, allowing it to perform such tasks as sequence-prediction that are beyond the power of a standard Multi-Layer Perceptron.

In a fully recurrent network, every neuron receives inputs from every other neuron in the network. These networks are not arranged in layers. Usually only a subset of the neurons receive external

inputs in addition to the inputs from all the other neurons, and another disjunctive subset of neurons report their output externally as well as sending it to all the neurons. These distinctive inputs and outputs perform the function of the input and output layers of a feed-forward or simple recurrent network, and also join all the other neurons in the recurrent processing.

Hopfield network

The Hopfield network is a recurrent neural network in which all connections are symmetric. Invented by John Hopfield in 1982, this network guarantees that its dynamics will converge. If the connections are trained using Hebbian learning then the Hopfield network can perform as robust content-addressable (or associative) memory, resistant to connection alteration.

Echo state network

The echo state network (ESN) is a recurrent neural network with a sparsely connected random hidden layer. The weights of output neurons are the only part of the network that can change and be learned. ESN are good to (re)produce temporal patterns.

Long short term memory network

The Long short term memory is an artificial neural net structure that unlike traditional RNNs doesn't have the problem of vanishing gradients. It can therefore use long delays and can handle signals that have a mix of low and high frequency components.

Stochastic neural networks

A stochastic neural network differs from a typical neural network because it introduces random variations into the network. In a probabilistic view of neural networks, such random variations can be viewed as a form of statistical sampling, such as Monte Carlo sampling.

Boltzmann machine

The Boltzmann machine can be thought of as a noisy Hopfield network. Invented by Geoff Hinton and Terry Sejnowski in 1985, the Boltzmann machine is important because it is one of the first neural networks to demonstrate learning of latent variables (hidden units). Boltzmann machine learning was at first slow to simulate, but the contrastive divergence algorithm of Geoff Hinton (circa 2000) allows models such as Boltzmann machines and products of experts to be trained much faster.

3.5. Employing artificial neural networks

Perhaps the greatest advantage of ANNs is their ability to be used as an arbitrary function approximation mechanism which 'learns' from observed data. This has been discussed in great detail in Chapter 1. It should be noted that although ANNs have great advantages the application of neural networks are not so straightforward and a relatively good understanding of the underlying theory is required. The following three factors should be considered before using a neural network to investigate a phenomenon. An initial step is the choice of model, is there enough data available to model the application. Overly complex models tend to lead to problems with learning. Secondly the choice of learning algorithm for the neural network model plays a significant role in the success of the application. Finally, if the model, cost function and learning algorithm are selected appropriately the resulting ANN can be extremely robust.

The method by which an artificial neural network learns or is trained can significantly affect the application and robustness of the network. In the following section the three methods of learning will be presented as it is implemented in the Matlab software code.

3.5.1. Learning paradigms

Three major learning paradigms exist with each one corresponding to a particular abstract learning task. These are supervised learning, unsupervised learning and reinforcement learning. Usually any given type of network architecture can be employed in any of those tasks.

Supervised learning

In supervised learning, a known set of example pairs are given to the neural network and the aim is to find a function in the allowed class of functions that matches the examples as closely as possible. Thus, the model network must infer the mapping implied by the data; the cost function is used to relate the mismatch between mapping and the data. This implies implicitly that a prior knowledge of the domain area should be known.

Tasks that fall within the domain of supervised learning are pattern recognition and regression. The supervised learning method is also applied to sequential data. Learning of the network provides a continuous feedback on the quality of solutions obtained.

Unsupervised learning

In unsupervised learning we are given a set of data and the cost function to be minimized. The cost function is dependent on the task related to the mode and *a priori* assumptions.

In unsupervised learning, the weights and biases are modified in response to the network inputs only. There are no target outputs available. Most of these algorithms perform clustering operations. The input patterns are classified into a finite number of classes. This is especially useful in such applications as vector quantization.

Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering.

Reinforcement learning

In reinforcement learning, the data is usually not given but generated by an instrument interacting with the environment. At each point in time, the instrument performs an action and the environment generates an observation with an instantaneous cost calculated by some mechanism. The goal is to minimize some measure of a long-term cost. The environment's mechanism and long-term cost for each policy must generally be estimated.

The environment is modelled as a Markov decision process (MDP) with states S and actions with the following probability distributions: the instantaneous cost distribution $P(c_t \mid s_t)$, the observation distribution $P(x_t \mid s_t)$ and the transition $P(s_{t+1} \mid s_t, a_t)$, while a policy is defined as conditional distribution over actions given the observations. Taken together, the two define a Markov chain (MC). The aim is to discover the policy that minimizes the cost, *i.e.*, the MC for which the cost is minimal[27].

ANNs are frequently used in reinforcement learning as part of the overall algorithm.

Tasks that fall within the paradigm of reinforcement learning are control problems, games and other sequential decision making tasks.

3.5.2. Learning algorithms

The training of a neural network model in essence means the selection of one model from a set of allowed models that minimises the cost criterion. There are numerous algorithms available for training neural network models; most of them can be viewed as a straightforward application of optimization theory (Steepest Descent, Newton-Raphson) and statistical estimation (Bayesian). A large section of the algorithms used in training artificial neural networks, employ in some form or another a gradient descent method. This is done by simply taking the derivative of the cost function with respect to the network parameters and then changing those parameters in a gradient-related direction. Evolutionary methods, simulated annealing, and expectation-maximization and non-parametric methods are among other commonly used methods for training neural networks.

It should be stressed that learning algorithms can be extremely specific to the model that is being used, and it is best practice to use a set of training methods to calculate an optimal network parameterisation value.

3.6. A Preliminary Investigation into Neural Network Capabilities

The application of artificial neural networks to model different water resource variables is becoming more essential to assist in management and planning strategies. A preliminary study of these methods will be reported in this section, with special focus on four known neural network architectures:

Focused Time-Delay Neural Network (FTDNN), Layer-Recurrent Neural Network (LRNN), Generalized Regression Neural Networks (GRNN) and Probabilistic Neural Networks (PNN).

Each of these networks has been reported previously, the application of these networks have been shown to work to a reasonable extent in other fields such as water table fluctuation prediction and recharge events[28,29]. Methods employed by Coulibaly *et al.*[28] were of special interest since it gave a clear description of the methods used. A data set of *ca.* four years of rainfall, water levels and flow gage measurements were obtained from the USGS. The state of Nebraska had an installed flow gage in the North Platte River on the border with Wyoming. The rainfall and water level data was obtained from the same region, with a four year monitoring program on the Broadwater site. The respective data is represented in Figure 3.15 - Figure 3.17.

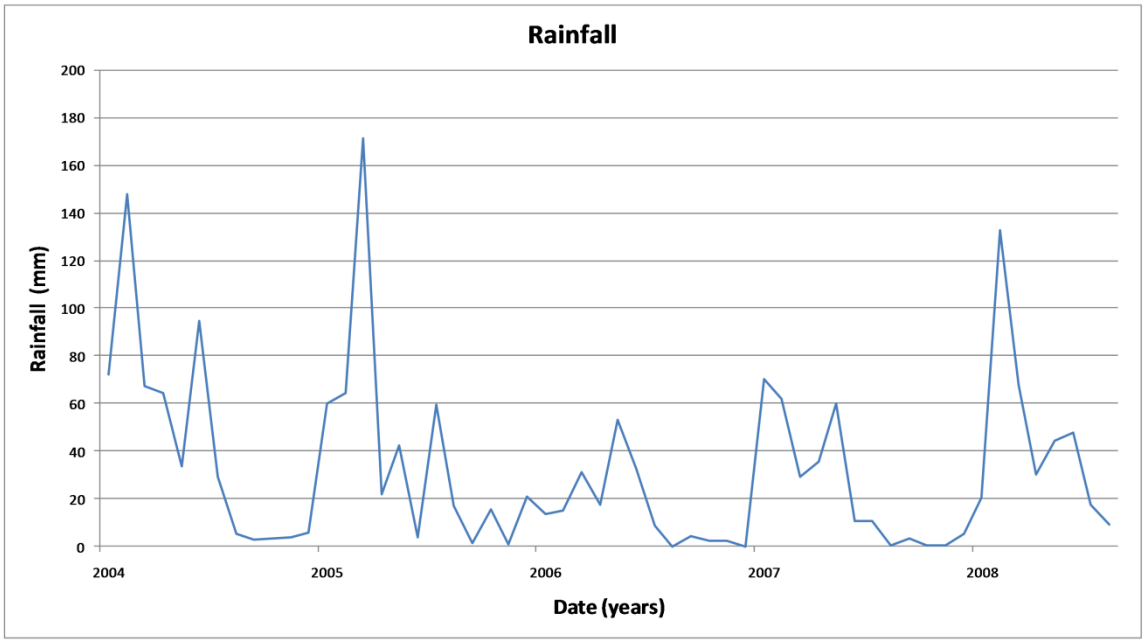


Figure 3.15 Rainfall data for the region of Broadwaters in Nebraska, USA.

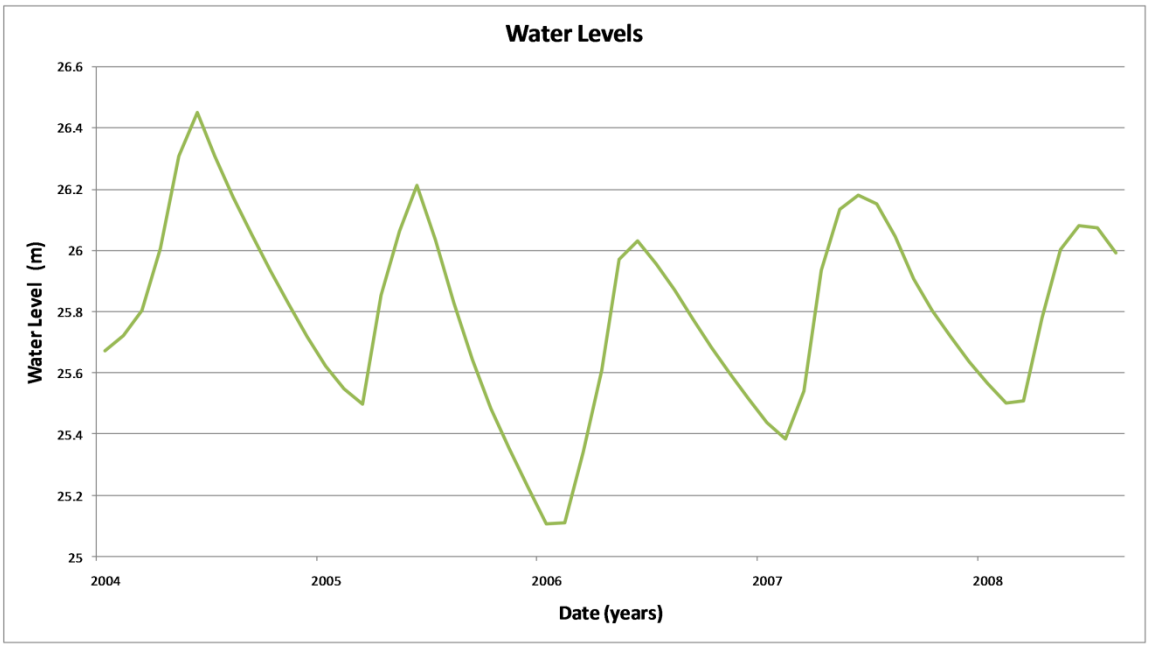


Figure 3.16 Water level data for the region of Broadwaters in Nebraska, USA.

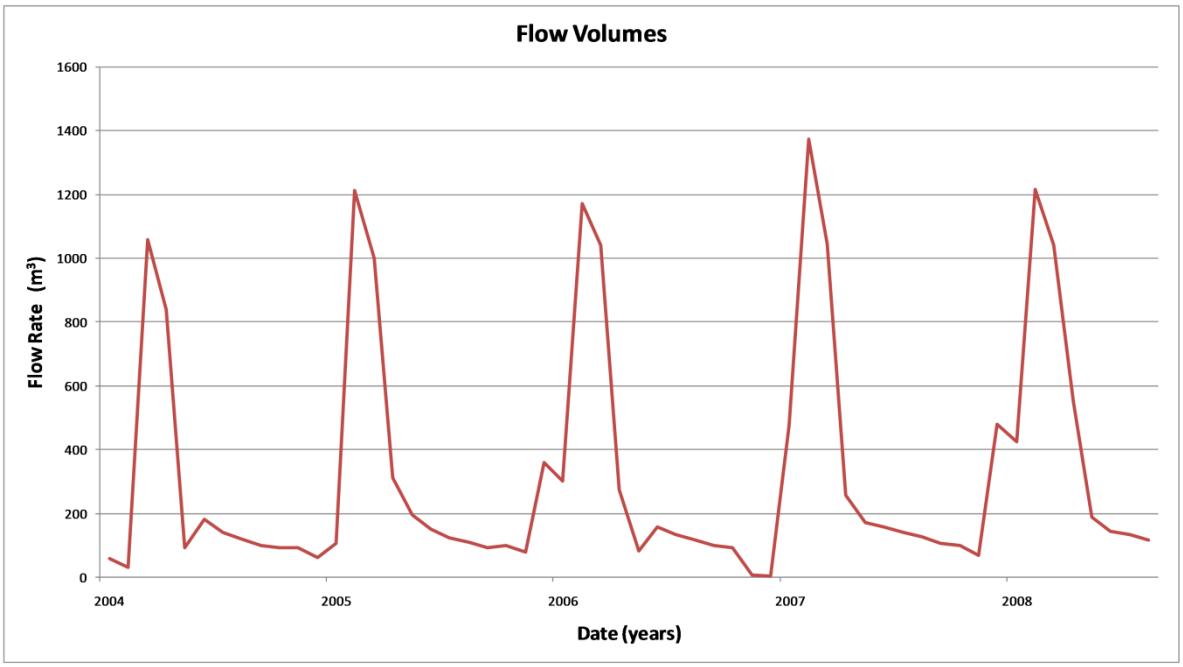


Figure 3.17 Flow volume data from the North Platte River on the Nebraska-Wyoming boarder, USA.

Four neural networks were constructed each using architecture related to its specific application, Focused Time-Delay Neural Network (FTDNN), Layer-Recurrent Neural Network (LRNN), Generalized Regression Neural Networks (GRNN) and Probabilistic Neural Networks (PNN). As an illustration the training and prediction method of the FTDNN system will be shown (Figure 3.18, Figure 3.19) and the remaining network approximations will be tabulated in Table 3.1. Training and validation tests were performed on the input data to show the correlation between simulated and actual data.

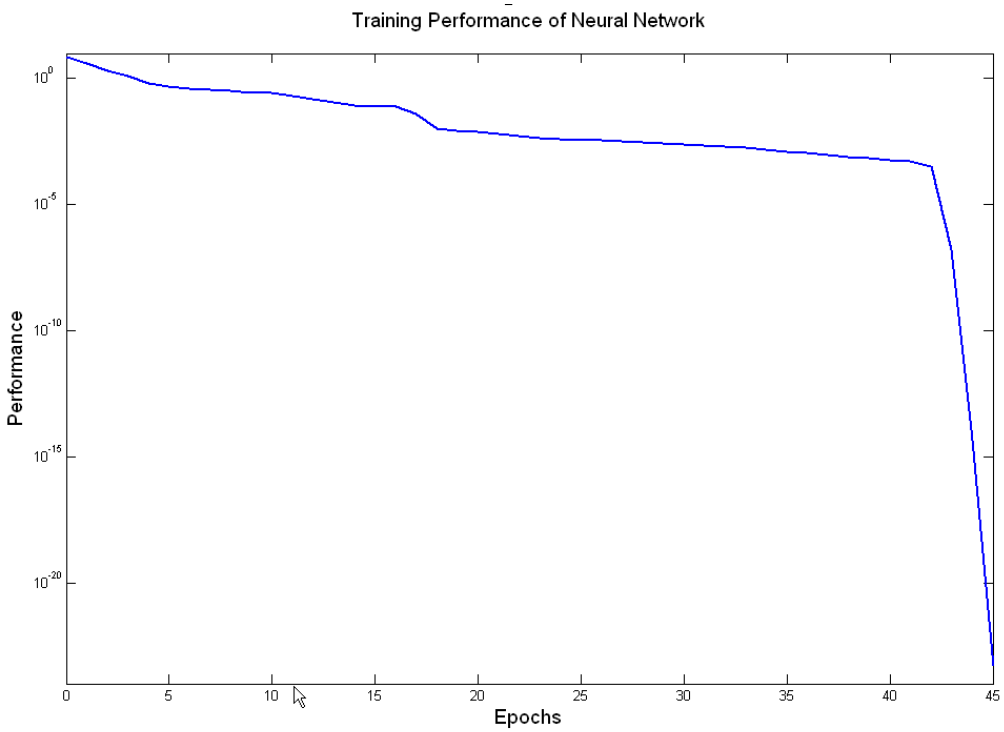


Figure 3.18 Training time *vs.* performance of the neural network on the data set. A delay of 8 months was used to train the neural network.

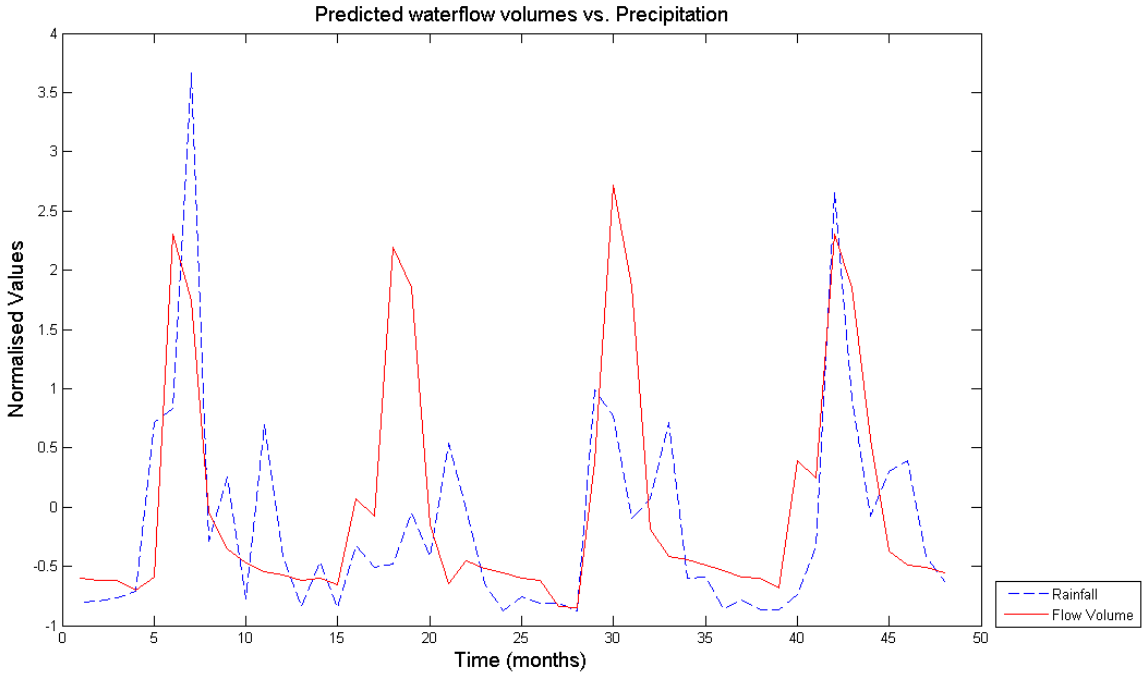


Figure 3.19 Precipitation values and predicted water flow rate changes in the Nebraska region.

The data was simulated by the focussed delay network to a high degree of accuracy, and it should be noted that the data was smoothed by the USGS before release to the public. In this regard, however, the model does approximate the river flow volumes and water levels to the rainfall data given. In Table 3.1 the most important values are reported, with each of the different neural network architectures performing to a different degree of accuracy (root mean square values). A further drawback of the system is the required data to be used to estimate the behaviour. The lack of an accurate and reasonable data set was a significant drawback, with the Nebraska set representing the best case scenario. Both the GRNN and PNN systems suffer to accurately simulate the data; however this will be investigated in the following sections. The memory order *p* is an indication of the amount of memory used in calculating and training of the neural network.

Table 3.1 A summary of preliminary values for the respective neural networks used to approximate precipitation interaction with water levels and flow rates.

Model	Memory Order p	Memory Structure	RMS fit Flow	RMS fit WL
FTDNN	P = 2 (static)	Time delay	0.000018	0.00022
LRNN	P ≥ 1 (adaptive)	Recurrent connection	0.000060	0.00096
GRNN	P = 1 (static)	Moving Window	0.025800	0.46100
PNN	P = 1 (static)	Moving Window	0.066200	0.92500

FTDNN = Focused Time-Delay Neural Network, LRNN = Layer-Recurrent Neural Network, GRNN = Generalized Regression Neural Networks and PNN = Probabilistic Neural Networks.

A further objective was to evaluate the correlation of the precipitation data with the flow volumes and the water level fluctuations. In order to determine this interaction cross correlation diagrams were prepared in Matlab, see Figure 3.20 and Figure 3.21. The delay of rainfall to water level recharge in this region was *ca.* 8 months while flow volume changes were affected in less than 2 months. The correlation of the data gives an estimate of the cyclic nature of the observations, which might assist certain neural networks in predicting the correct value.

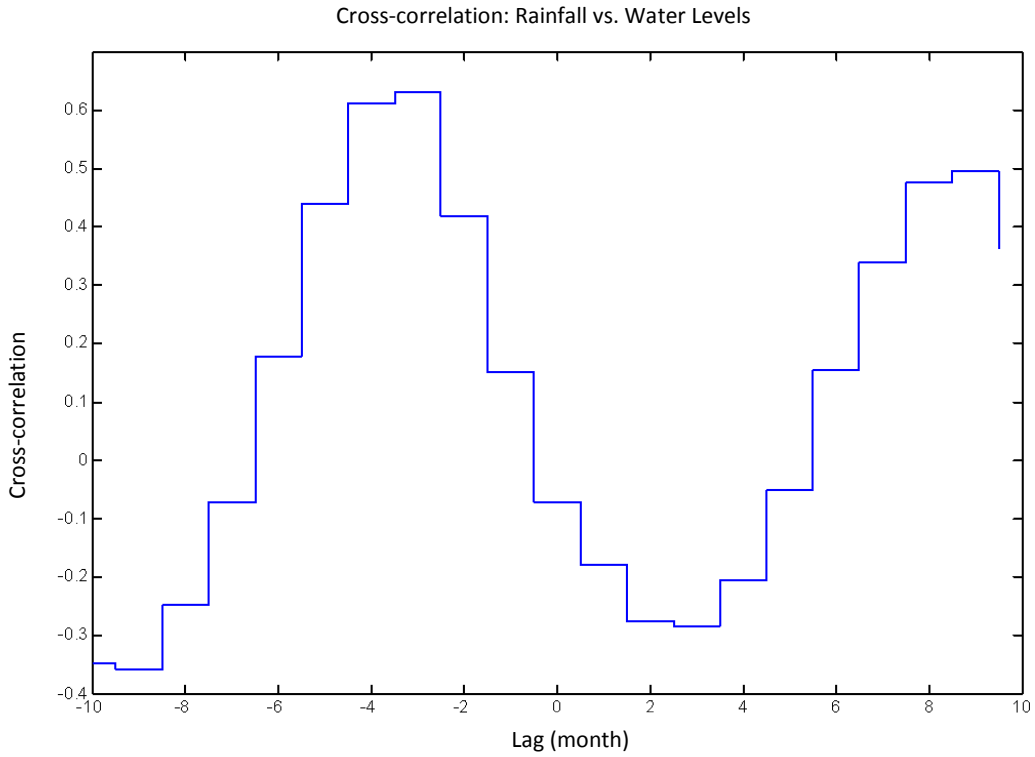


Figure 3.20 Cross correlation of rainfall data with water levels in the Broadwater system.

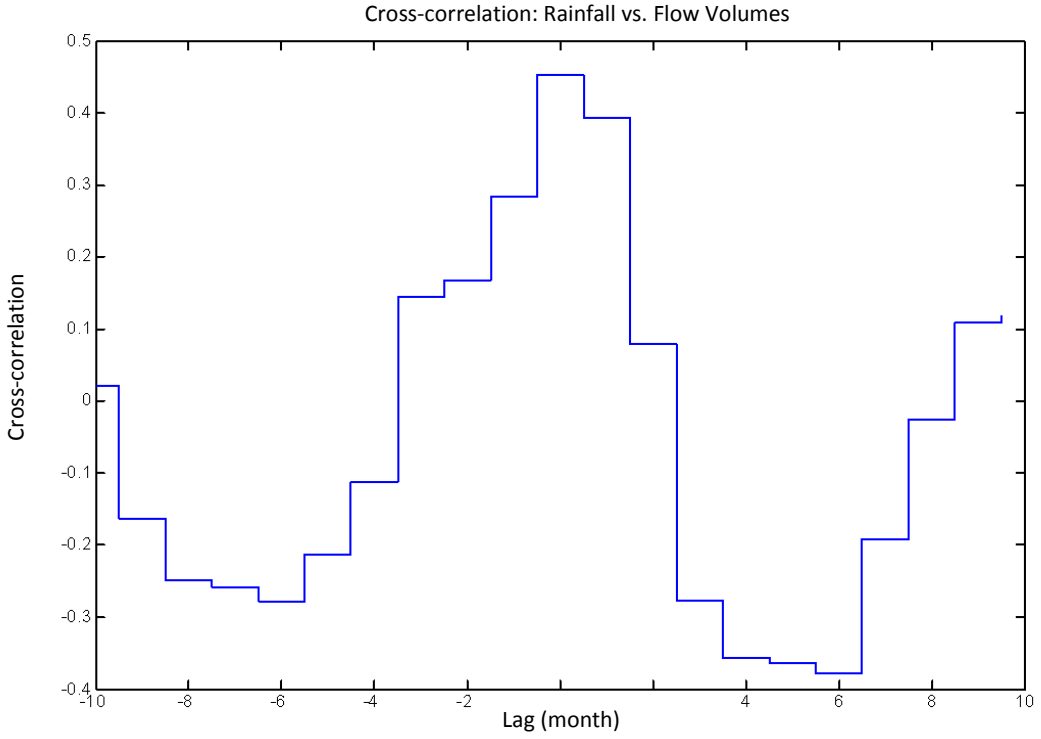


Figure 3.21 Cross correlation of rainfall data with flow volumes in the North Platte River.

3.7. Conclusion

In this chapter the design and construction of neural networks were discussed, ranging from the simple neuron to training methods. Transfer functions which enable the neuron to make a decision have been presented. Four popular transfer functions (hardlim, purelin, logsig and tansig) have been highlighted. The abbreviated notation for artificial neural networks was shown and the simplification it brings about in understanding the architecture of neural networks. The types of neural networks were briefly discussed with each one's associated use in an application.

A preliminary investigation into the behaviour of four neural network architectures were reported, and will be further studied to determine their efficacy in modelling the surface water and groundwater interaction in following chapters. In the following chapter patching of time series data will be presented using standard feedforward networks.

Chapter 4

Patching Algorithms and Artificial Neural Networks

4.1. Introduction

In this section the background of the preceding chapters will be combined to model variable time series data for natural events such as rainfall and water levels. The first step in this modelling approach is the evaluation of the data and the use of more classical methods. The data that will be used in this section is the monthly rainfall of Bloemfontein over a period of *ca.* 90 years from 1911 to 2003. The raw data is plotted in Figure 4.1 to show the amount of variance observed.

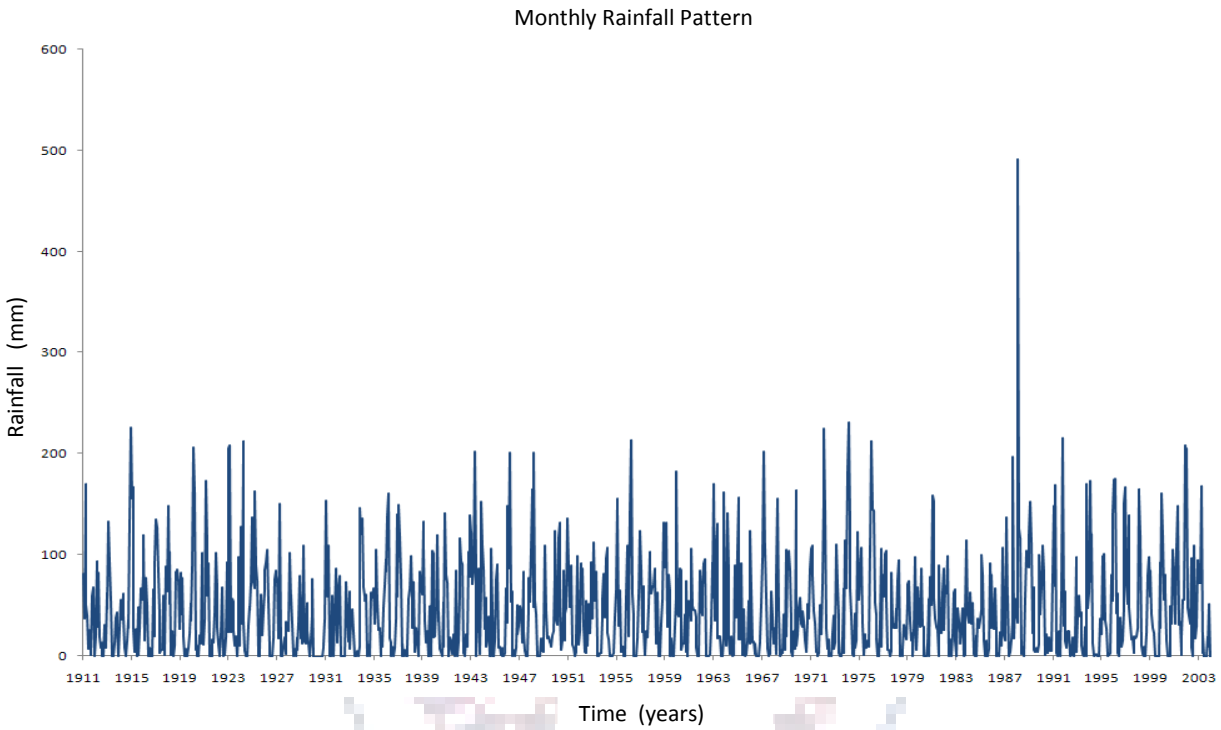


Figure 4.1 Monthly rainfall data plotted for the time period from 1911 till 2003.

In order to develop a baseline for the current investigation a statistical method will also be evaluated to determine the application of neural networks in hydrology. Furthermore, different scenarios will be constructed to verify the efficiency of the network such that the network structure can simulate a specific pattern. These methods will assist in creating a method by which data patching could be attempted.

4.2. Classical statistical methods

A commercial software package was used to predict the seasonal changes observed in the rainfall data of Figure 4.1. The software package used was Crystal Ball Predictor[30] (CB Predictor) from Oracle.

The time series data for the whole time period was used to determine the predicting capacity of the software. A seasonality of 12 months was used with the Holt-Winter's additive method. The time period from 1961 till 2002 was used to perform the analysis.

There is an advantage to using the Holt double exponential smoothing over Brown's double exponential smoothing. The most significant difference is that the single exponential smoothing

equation is applied for a second instance, while Holt's method uses a different smoothing constant (β) for the second operation.

The formulas for single and Holt's double exponential smoothing methods:

$$S[1](t) = \alpha \times y(t-1) + (1-\alpha) \times S[1](t-1)$$

$$S[2](t) = \beta \times \big(S[1](t) - S[1](t-1)\big) + (1-\beta) \times S[2](t-1)$$

where *S[2](t)* represents the double exponential smoothing or smoothed estimate for time period *t*. *S[1](t)* is the single exponential smoothing or smoothed estimate for time period *t, y(t)* is the historical data value at time *t*. The constants *α* and *β* are the smoothing constants with values between 0 and 1.

Other statistical methods were also used however; in each run the Holt-Winter's additive method gave consistently the best results. The methods included the following: double exponential smoothing, double moving average, Holt-Winter's additive, Holt-Winter's multiplicative, seasonal additive, seasonal multiplicative, single exponential smoothing, single moving average.

The lowest root mean square error observed was 46.14 and mean absolute deviation values are in the order of 31.43 units (Figure 4.2).

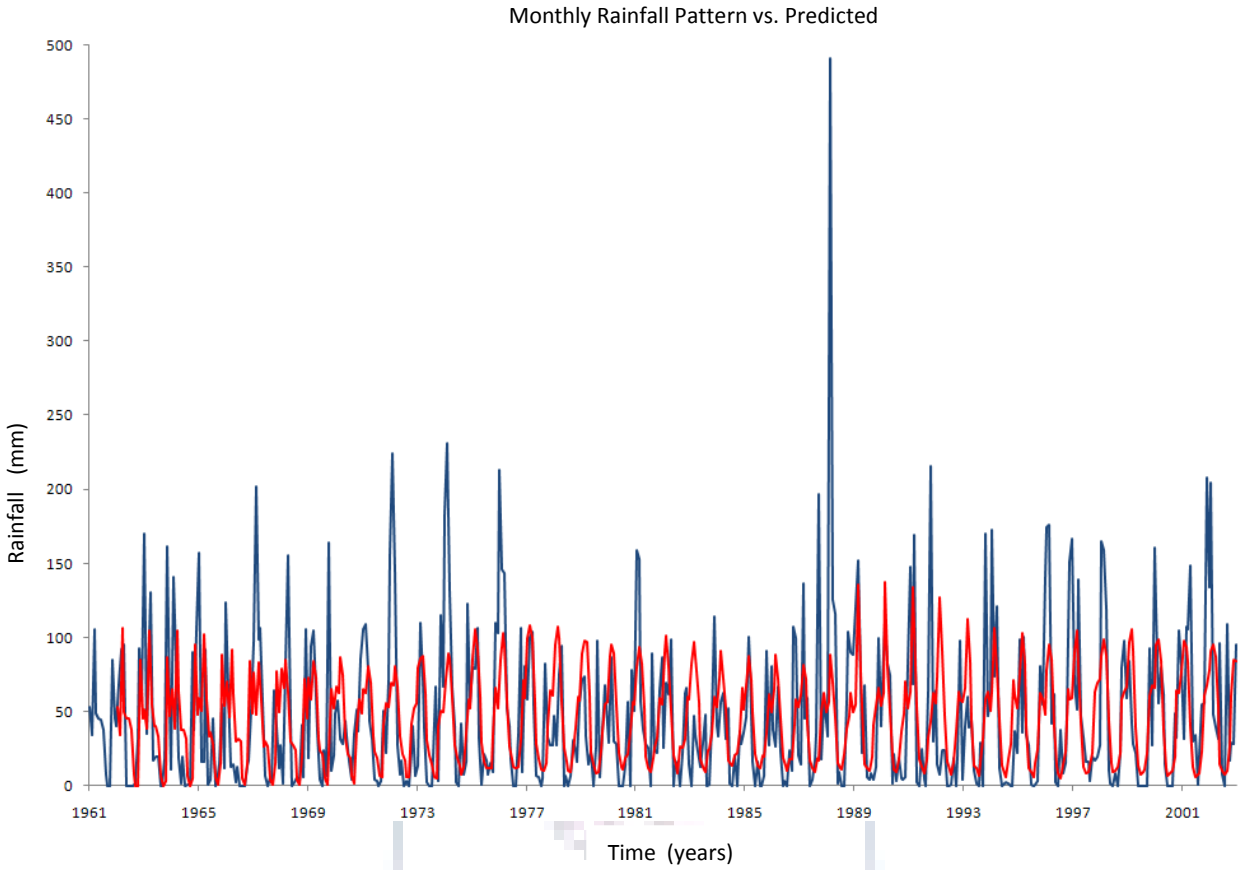


Figure 4.2 Best prediction value for the time series data using statistical methods.

Using the seasonal additive method better results could be obtained, however negative rainfall values were observed (Figure 4.3). Once the values are scaled back to their original component length the observed negative rainfall values are in the order of *ca.* 20 to 80 mm per month. This method might be useful if these values could be restricted to the baseline.

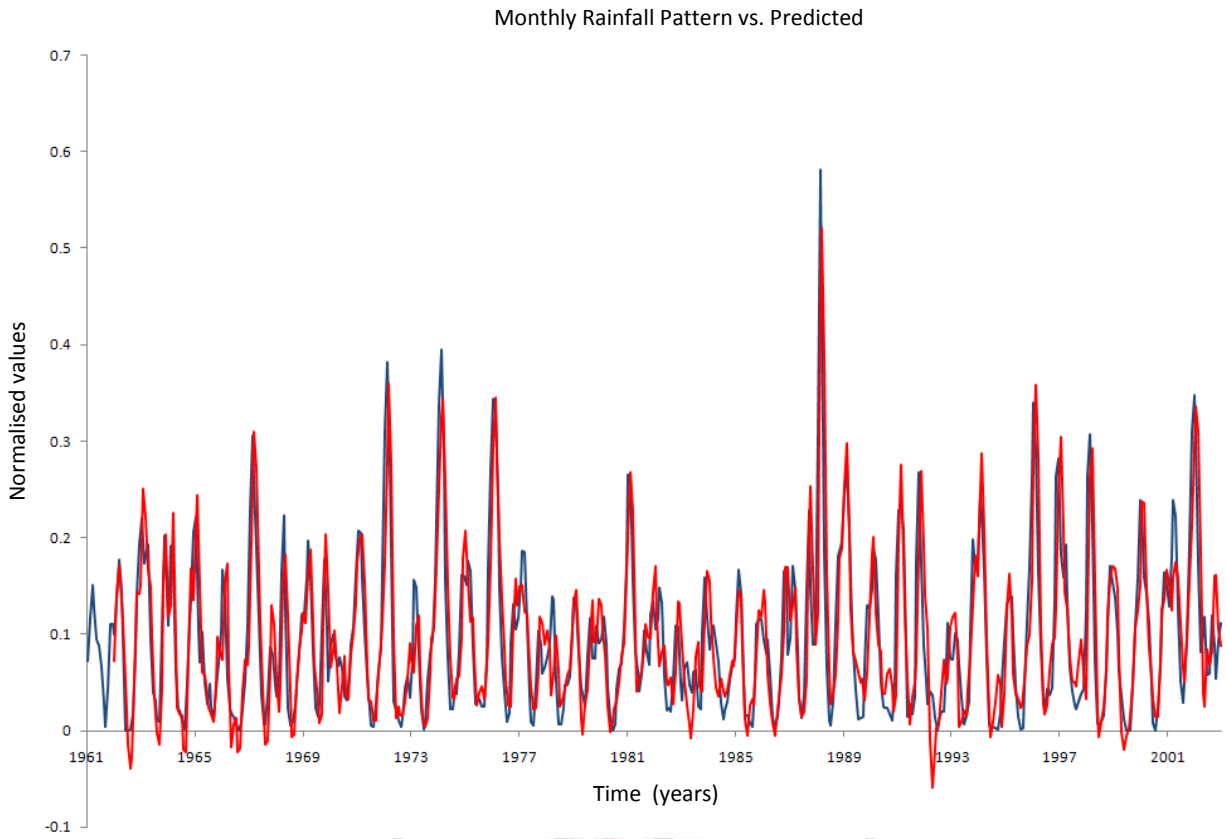


Figure 4.3 Seasonal aditive prediction method.

The results shown thus far are the best case scenarios. It is clear that the in Figure 4.2 that a seasonal flux could be observed, while in Figure 4.3 the data was fitted more accurately but showed significant areas of negative peaking.

In the next section the effect of neural networks on the time series will be reported. Similar time periods will be used to verify if these networks can outperform statistical methods.

4.3. Initial Neural Network Design

It is clear from the previous chapter that the design of neural network has a few stages through which it has to go before it can be implemented. One of the first steps is the determination of the seasonality of the data to be fitted. A range of seasonal cycles were evaluated, ranging from 1 year to 21 years (Figures 4.4 – 4.15).

In a similar method to the statistical approach used in the previous section, neural networks can also be trained to increase its efficiency. Once an optimal training stage was reached, the model was used to simulate the extended data set. In the following sections a graphical representation of each of the individual scenarios is given. The initial network configuration used was a 3 point moving average smoothing of the data points. A three layer neural network, with each layer containing 100 neurons. In the following simulations an epoch will be defined as a single training event time unit.

4.3.1. Scenario 1: A seasonal cycle of 1 year.

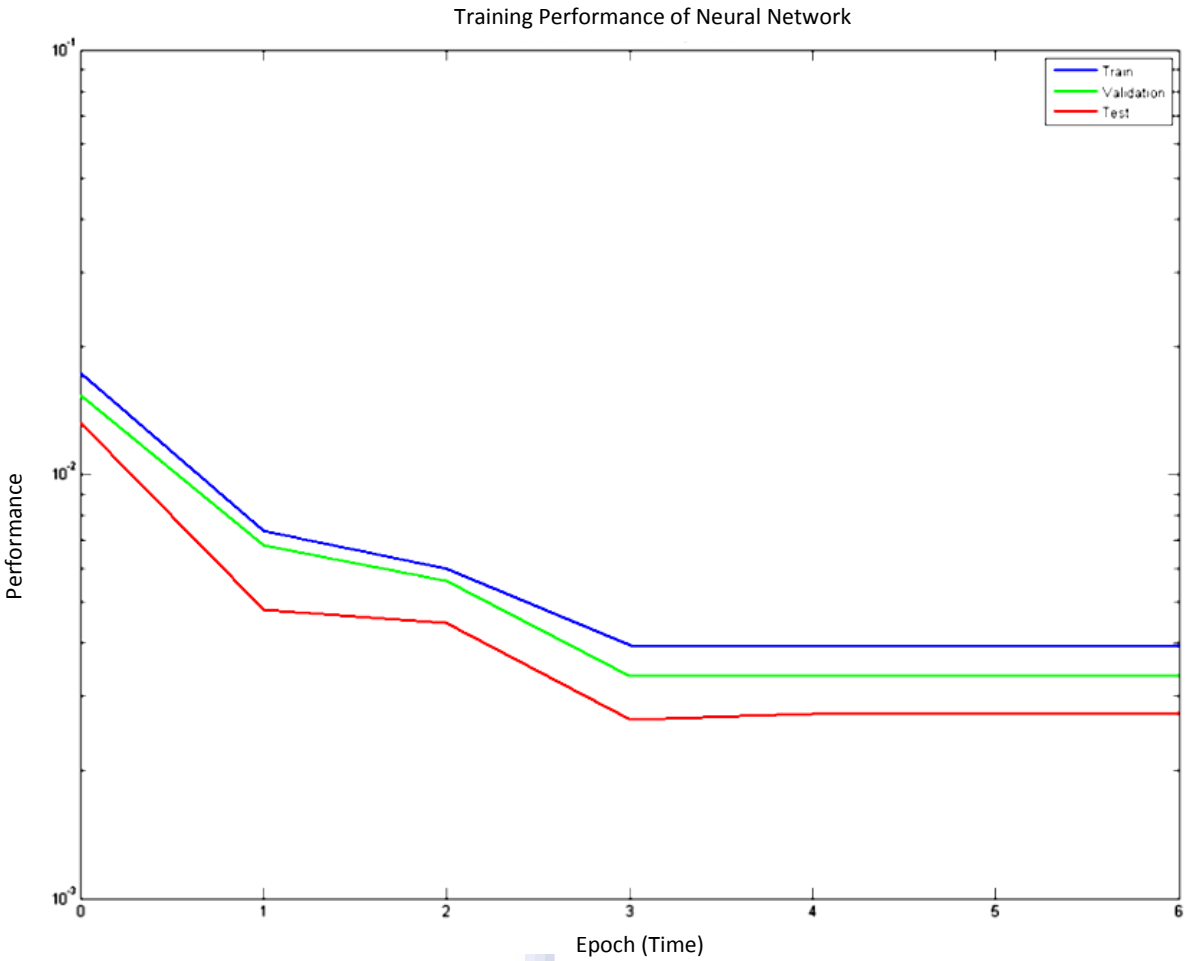


Figure 4.4 Training time *vs.* performance of the neural network on the data set. A seasonal cycle of one year was used.

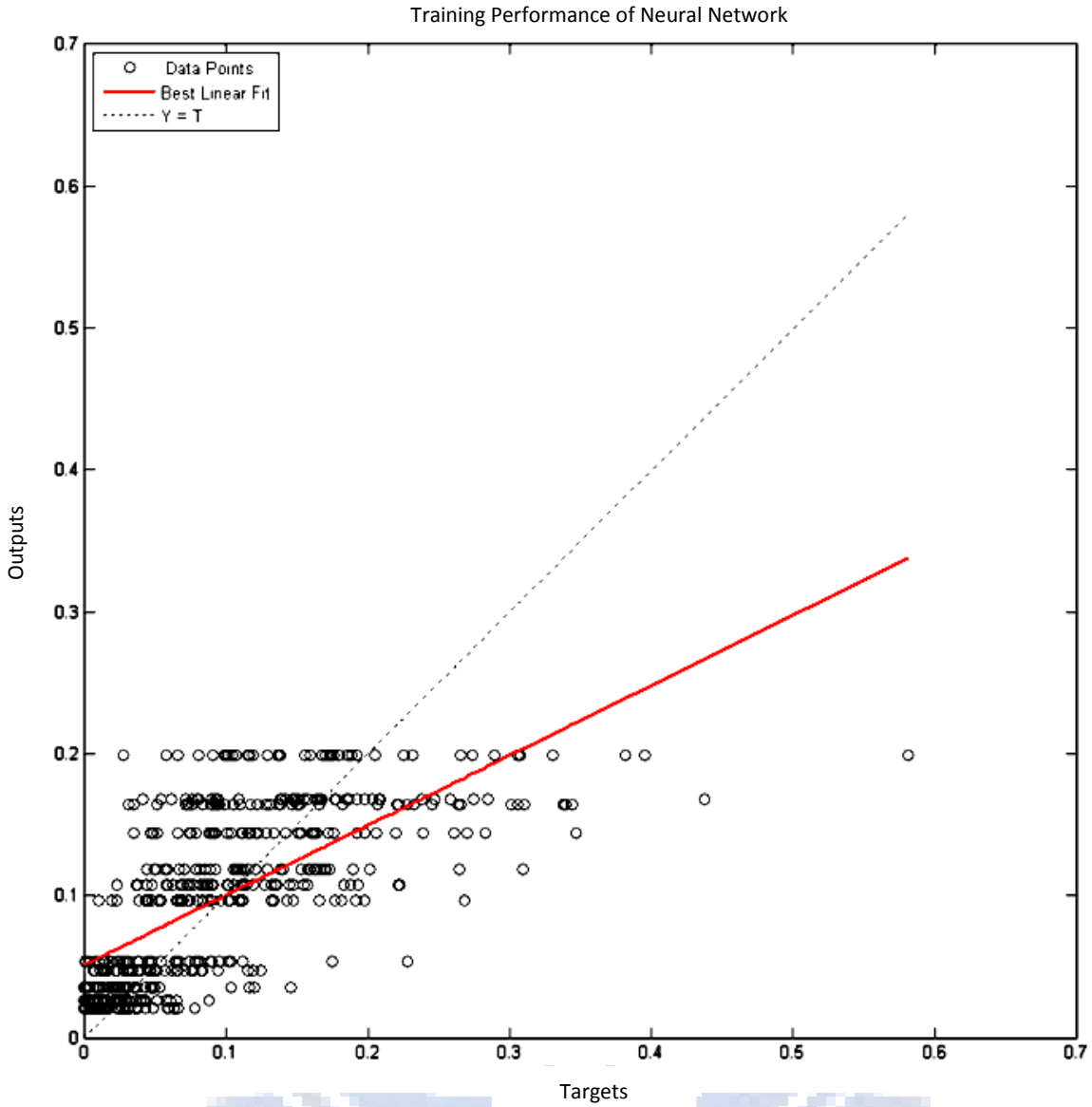


Figure 4.5 Linear regression of simulated and data points for the time series. Red line indicates the best fit linear regression and dotted black line a 1:1 representation.

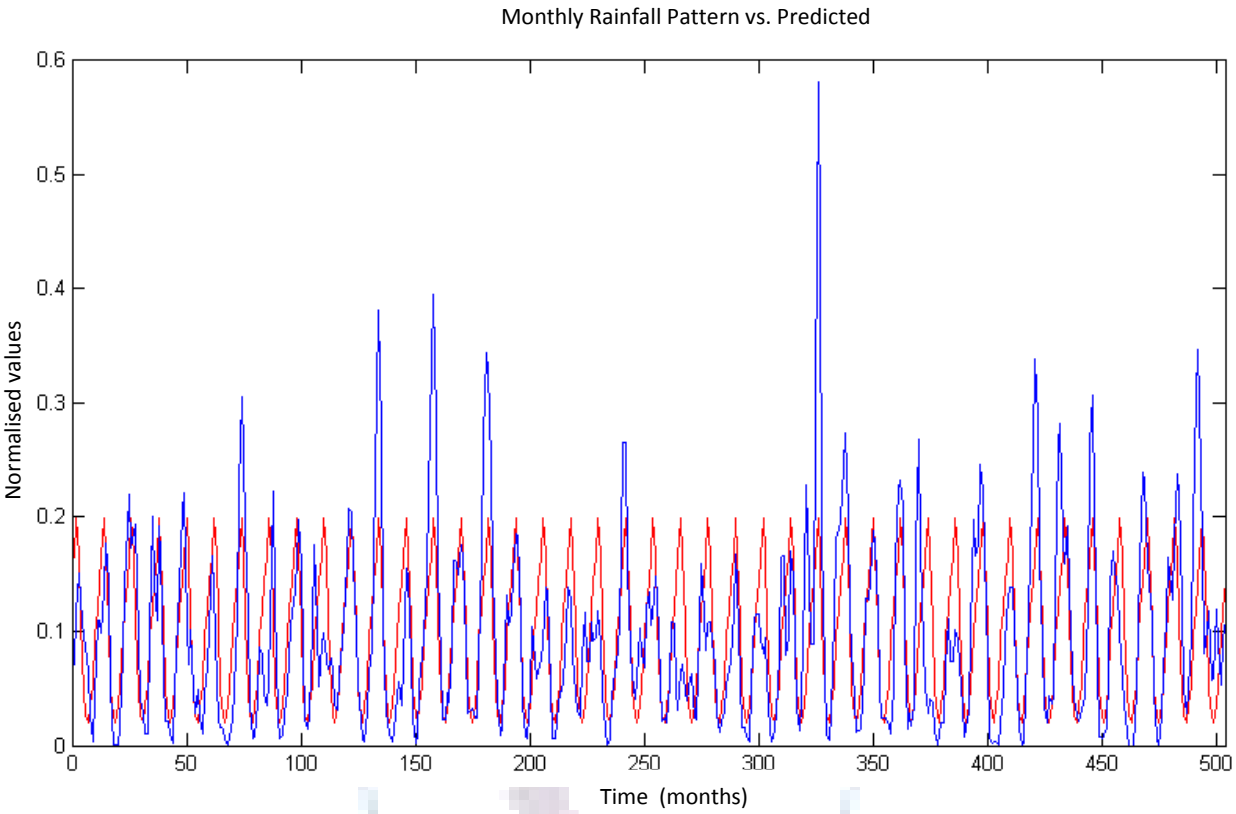


Figure 4.6 Neural Network estimation of rainfall in the Bloemfontein area using a one year cycle. Blue and red graphs indicate actual and simulated rainfall data, respectively.

4.3.2. Scenario 2: A seasonal cycle of 6 years.

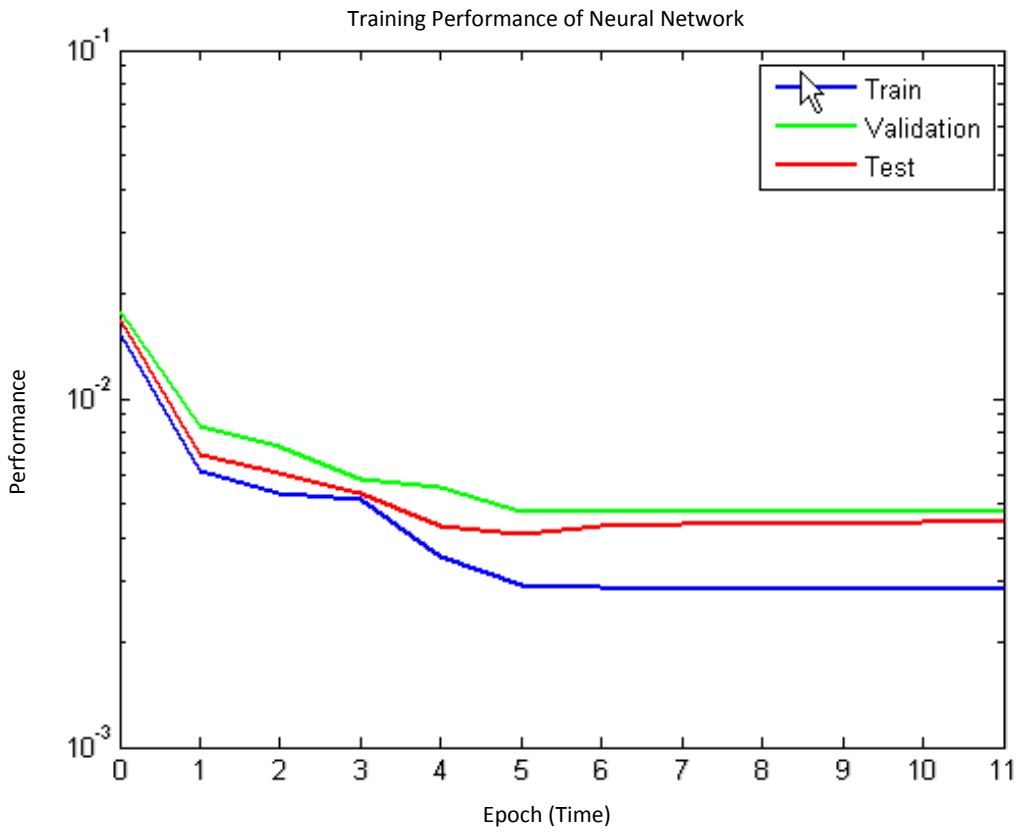


Figure 4.7 Training time *vs.* performance of the neural network on the data set. A seasonal cycle of six years was used.

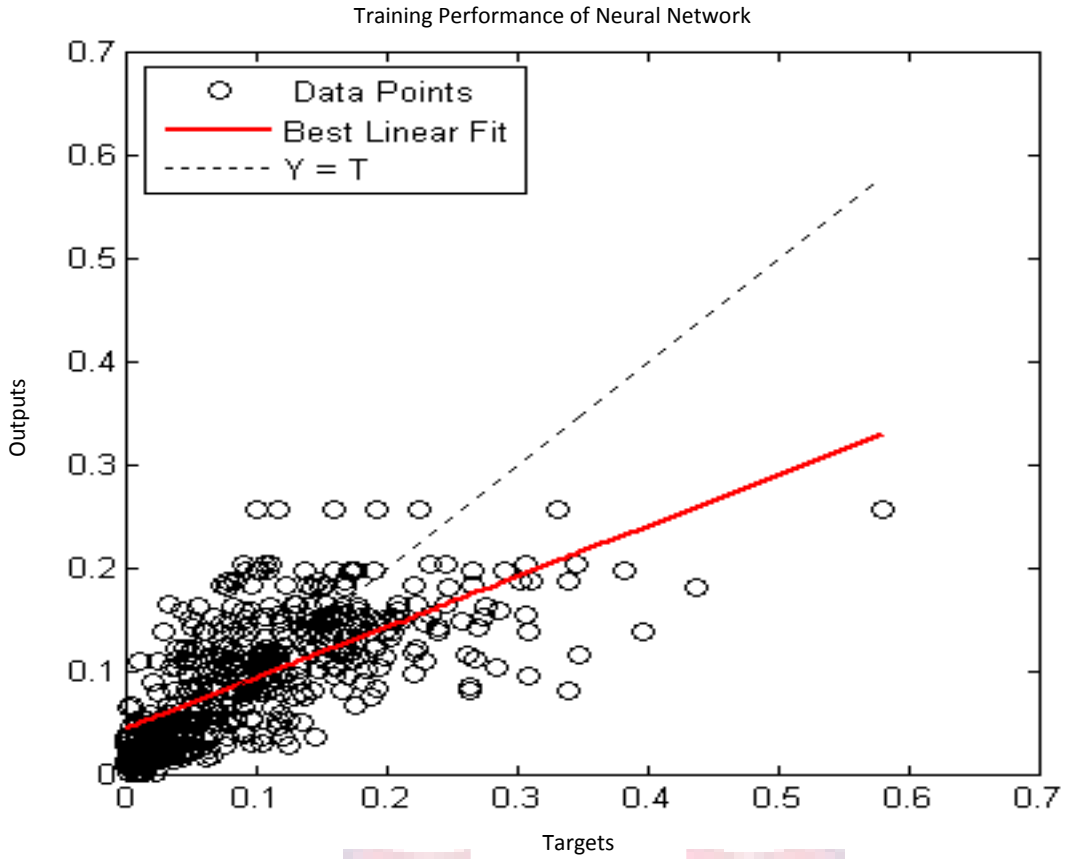


Figure 4.8 Neural Network training, validation and testing values (top). Linear regression fit (red line) of data points and deviation from 1:1 correlation (dotted line).

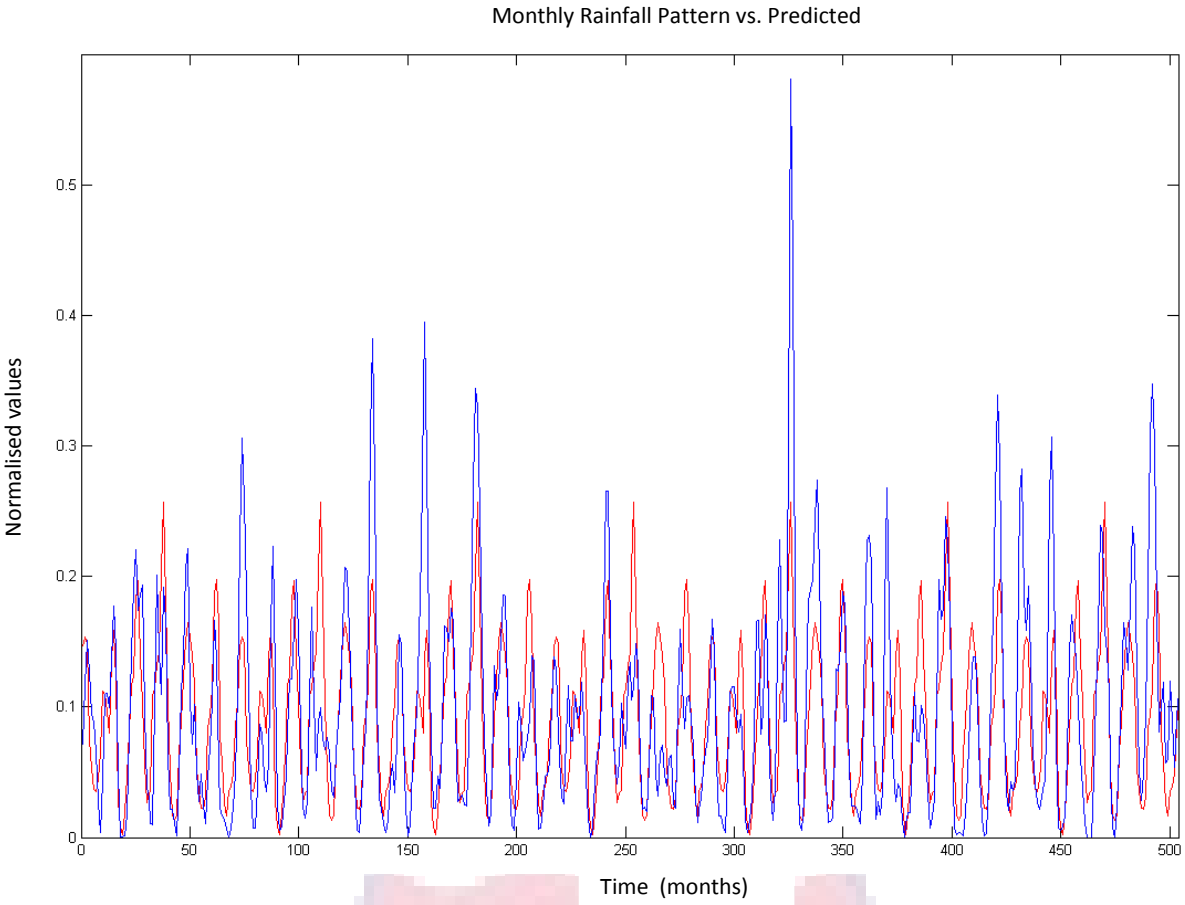


Figure 4.9 Neural Network estimation of rainfall in the Bloemfontein area using a six year cycle. Blue and red graphs indicate actual and simulated rainfall data.

4.3.3. Scenario 3: A seasonal cycle of 14 years.

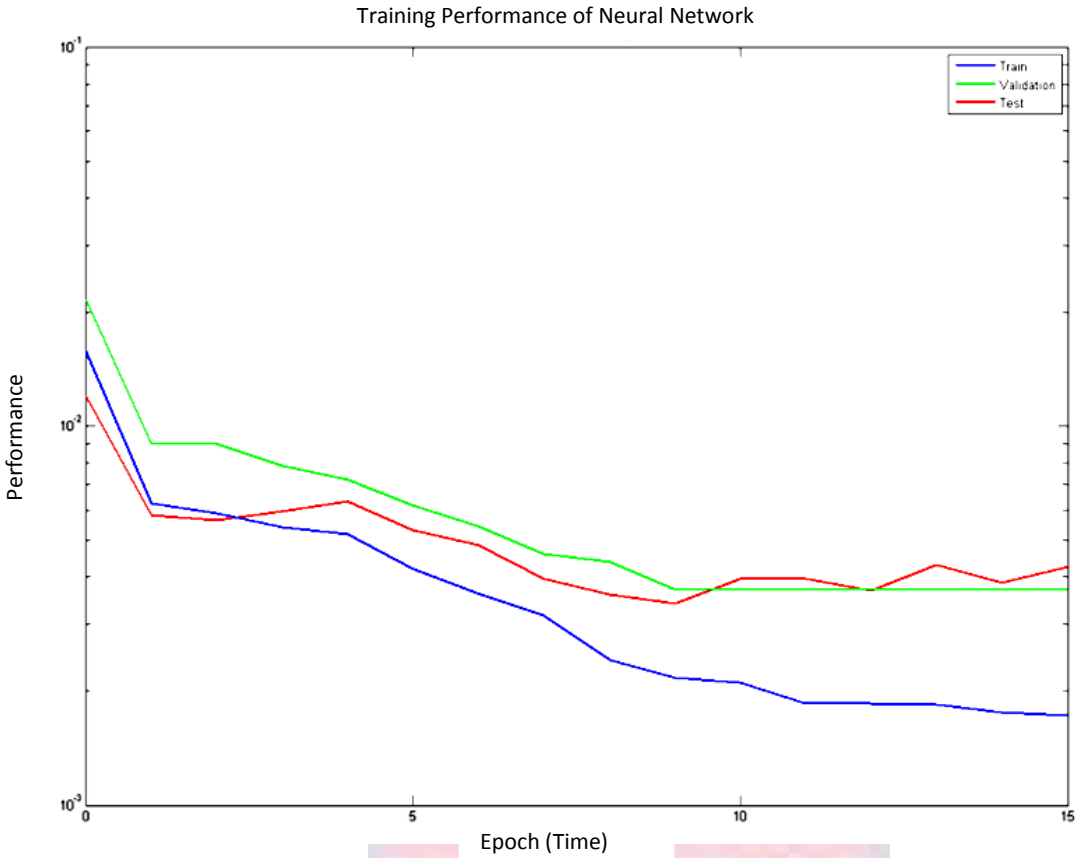


Figure 4.10 Training time *vs.* performance of the neural network on the data set. A seasonal cycle of fourteen years was used.

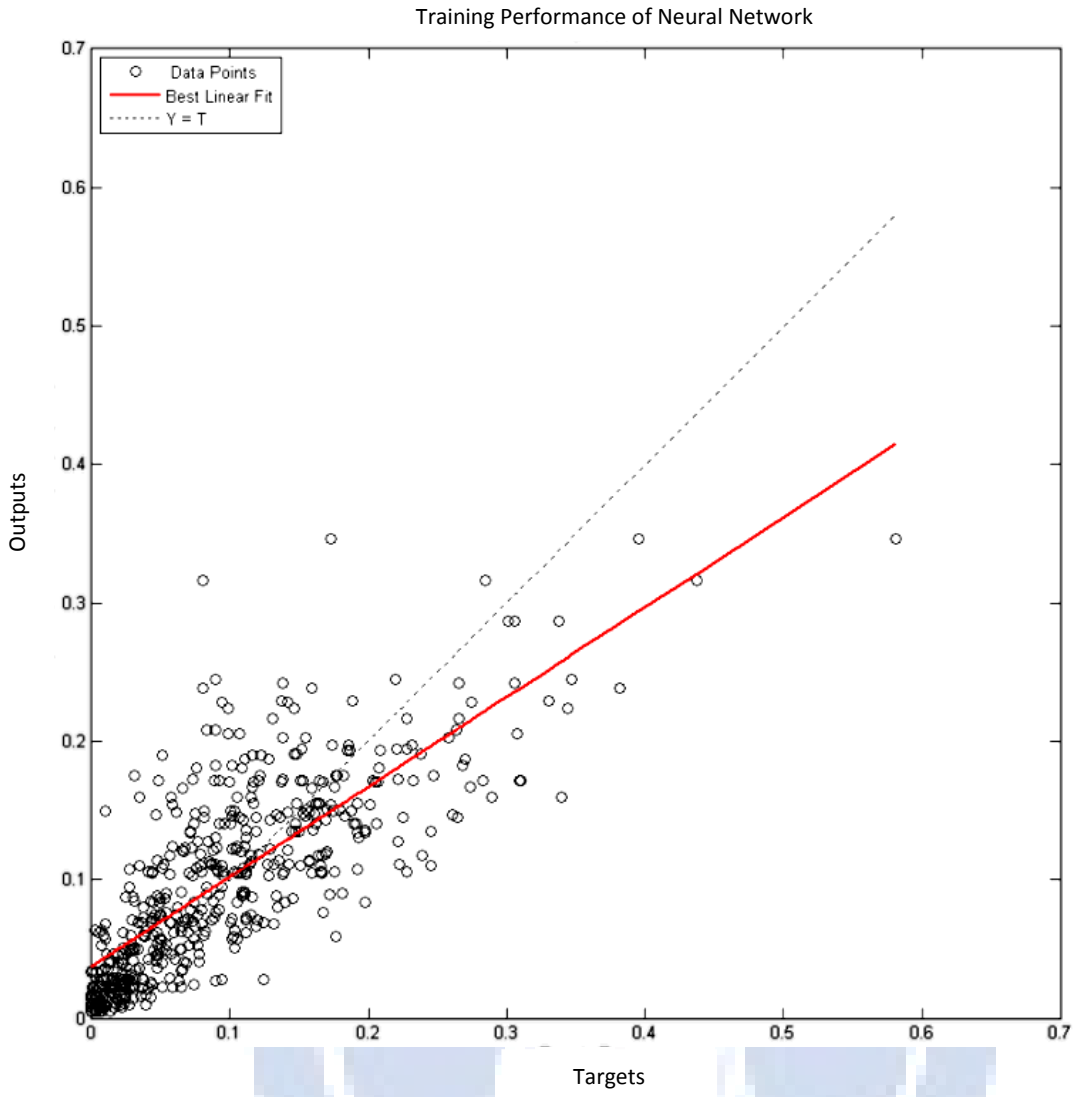


Figure 4.11 Neural Network training, validation and testing values (top). Linear regression fit (red line) of data points and deviation from 1:1 correlation (dotted line).

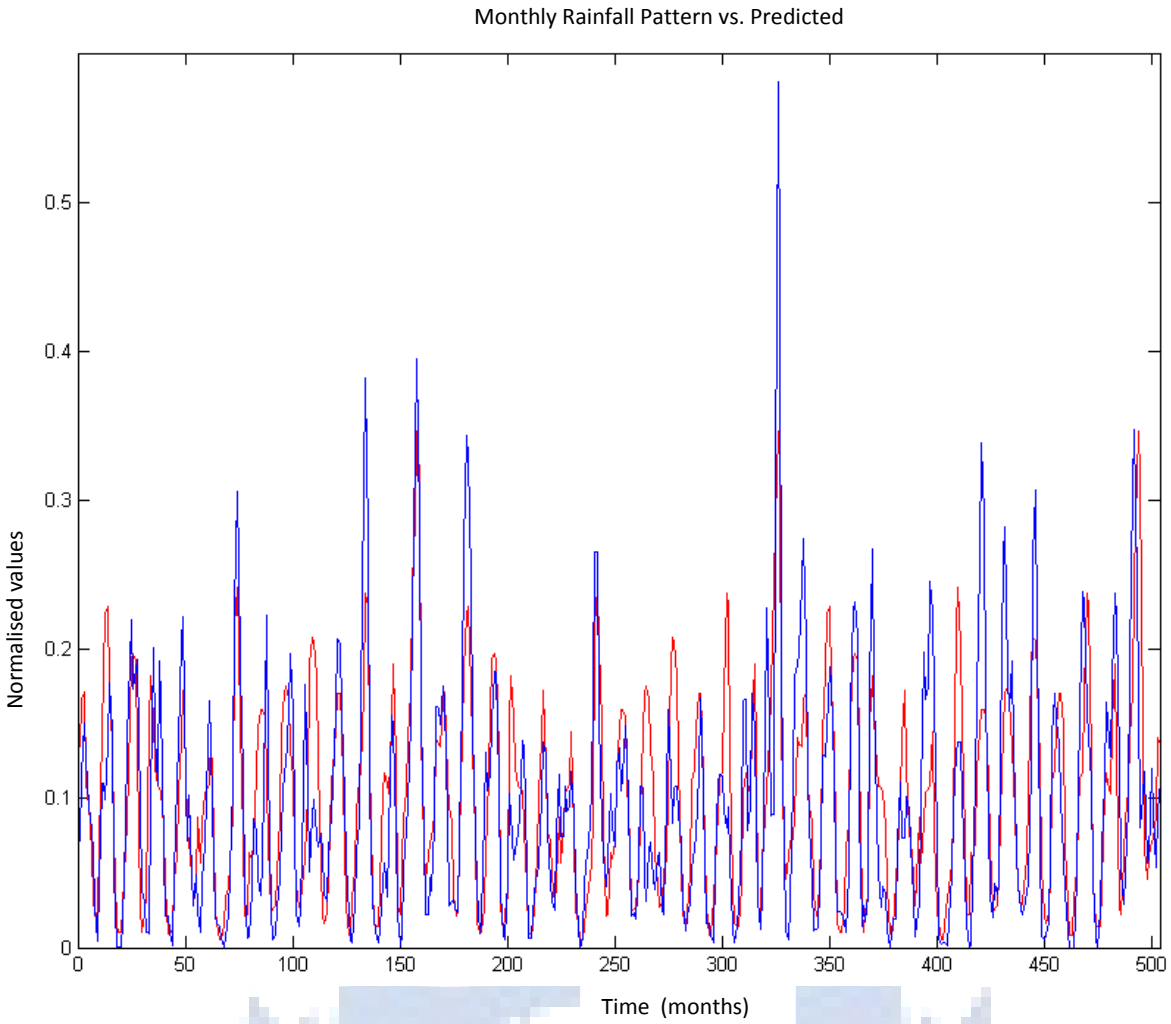


Figure 4.12 Neural Network estimation of rainfall in the Bloemfontein area using a fourteen year cycle. Blue and red graphs indicate actual and simulated rainfall data.

4.3.4. Scenario 4: A seasonal cycle of 21 years.

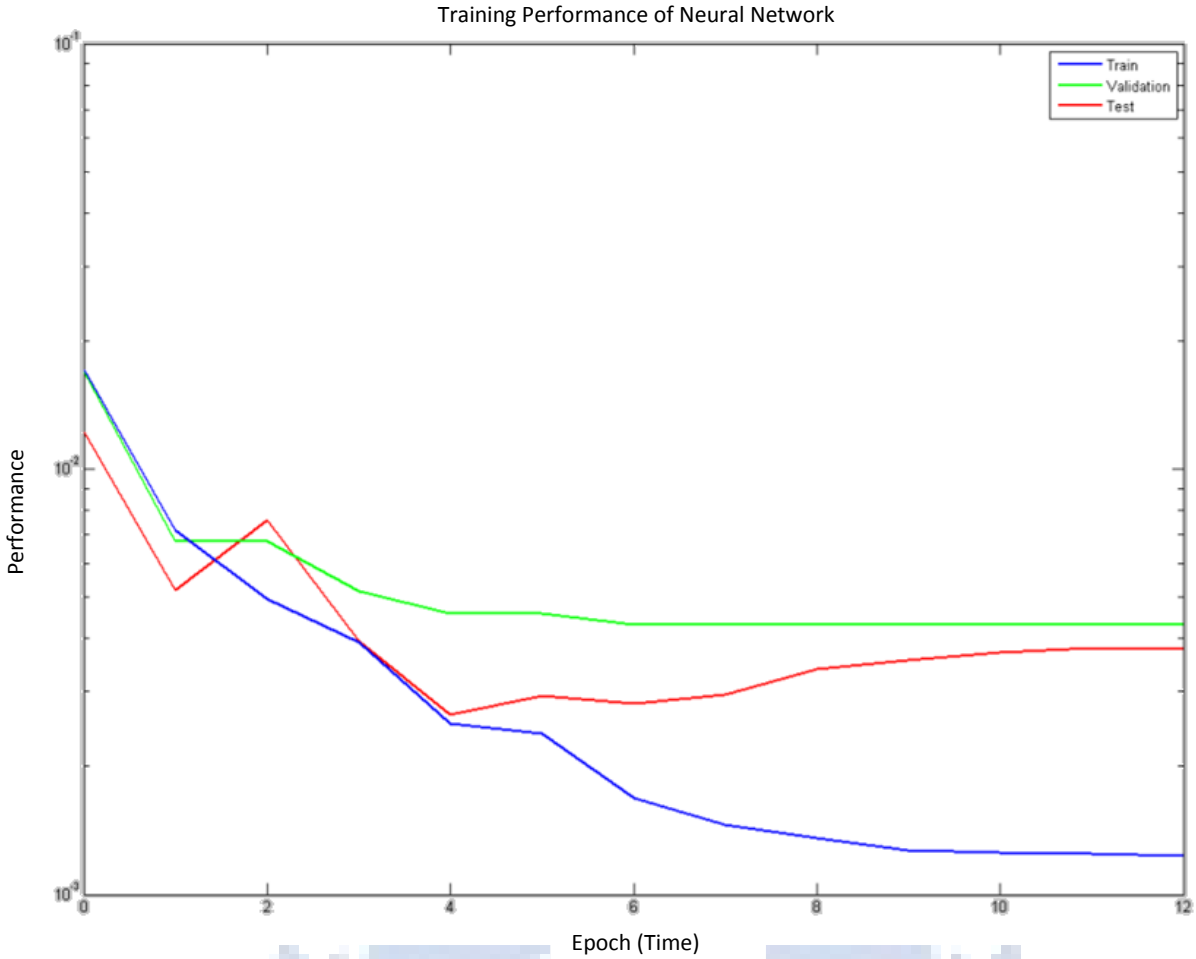


Figure 4.13 Training time *vs.* performance of the neural network on the data set. A seasonal cycle of twenty one years was used.

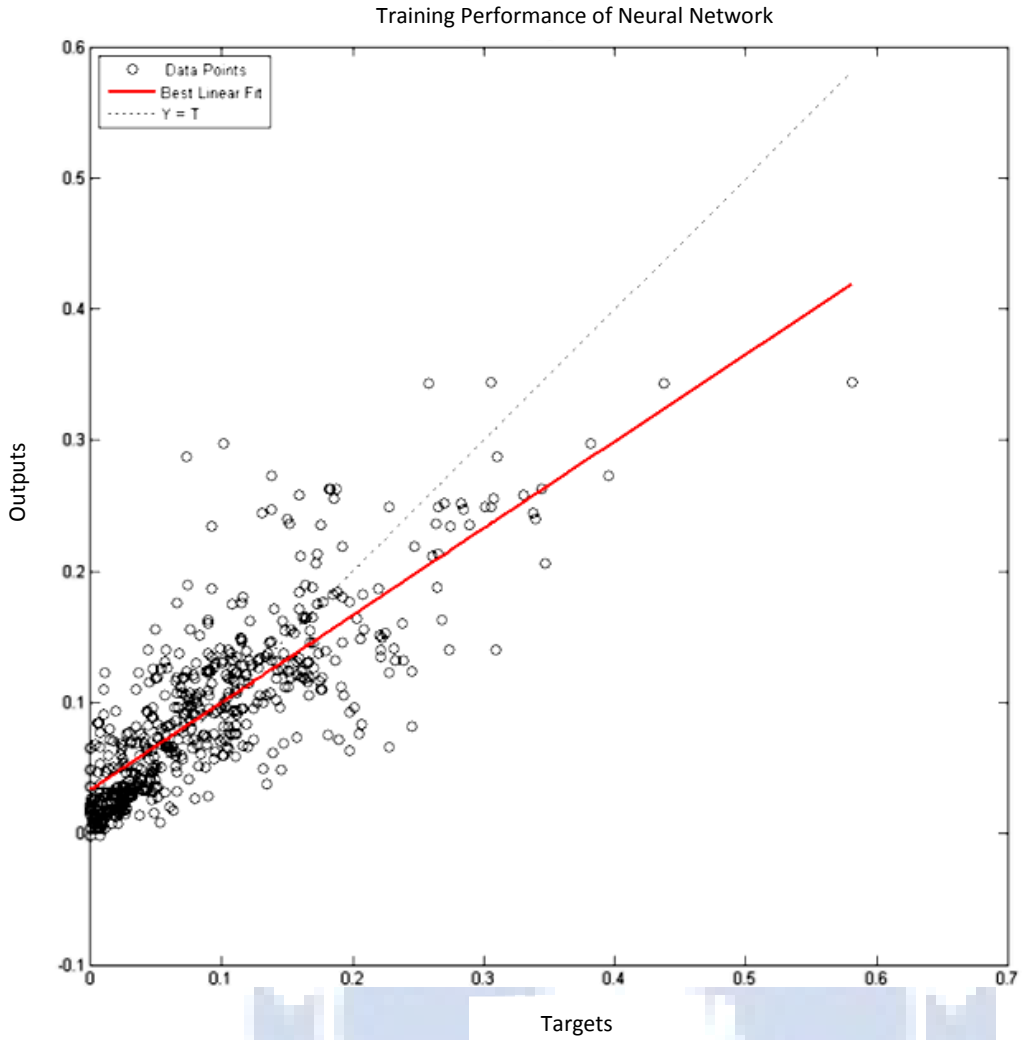


Figure 4.14 Neural Network training, validation and testing values (top). Linear regression fit (red line) of data points and deviation from 1:1 correlation (dotted line).

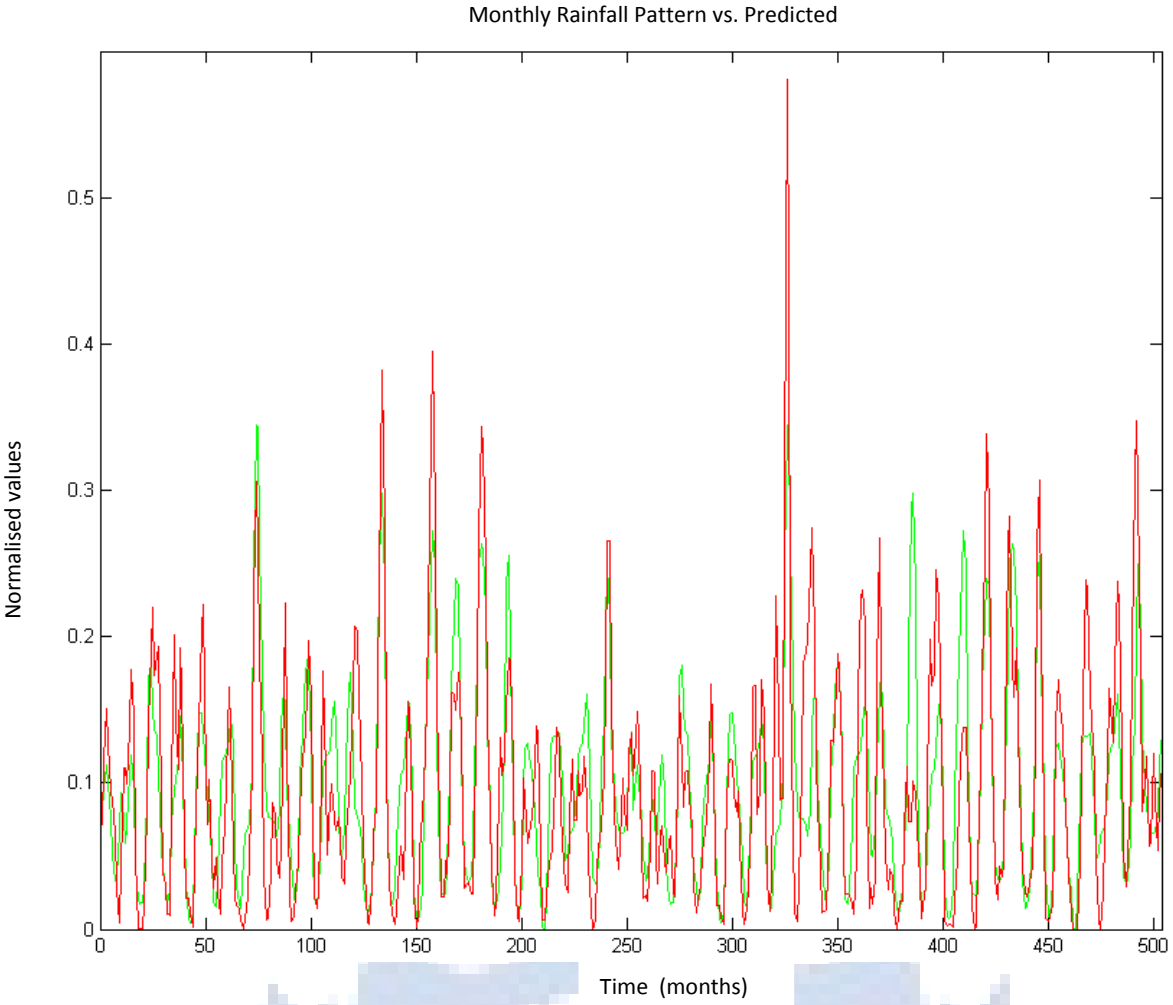


Figure 4.15 Neural Network estimation of rainfall in the Bloemfontein area using a twenty one year cycle. Red and green graphs indicate actual and simulated rainfall data.

Summary of Scenario Data

The optimal period to fit the time series data with the selected neural network is 14 years, see Table 4.1 (Linear regression percentage), this value represents ca. 25 % of the total data. This observation is in close correlation with the time period selection method of Wang et al.[16]. The performance of each of the neural network's used slowly increased as the time period was extended. Although the linear regression percentage increased with the amount of time series data, the maximum performance to data was reached at *ca.* 14 years (see Table 4.1). During the 21 year simulation an increase in accuracy of prediction (measured vs. calculated compared to the 14 year cycle) was

observed in the order of only 2.5 %, while a similar change in period from 6 years to 14 years resulted in a total increase of 8 %.

Table 4.1 A summary of the most important training values for neural network including variable period times.

Period (years)	Linear regression percentage	Performance value
1	68.66	0.00393
6	69.51	0.00284
14	77.59	0.00173
21	80.05	0.00123

4.3.5. Number of Neurons per Layer

The next challenge due to size constraints of the eventual model was to determine the total amount of neurons required per layer. A cycle period for the data was set to 14 years, since it was a good trade off in accuracy *vs.* total data required (Figure 4-16 – Figure 4.18).

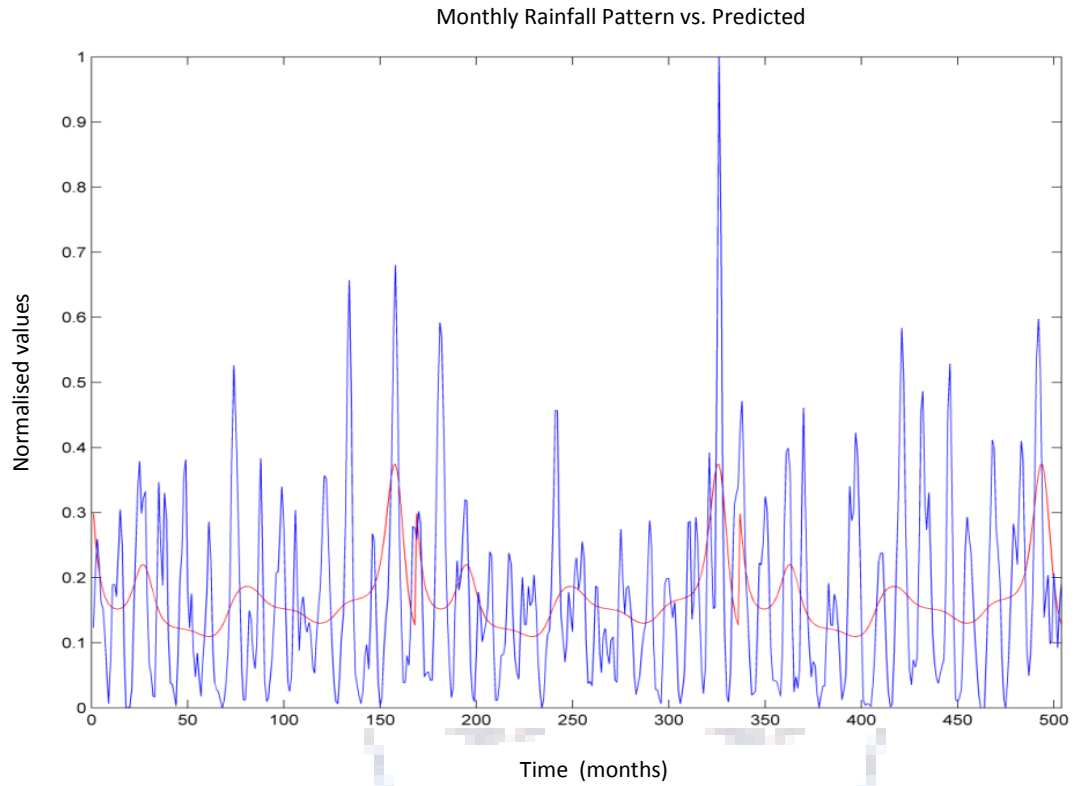


Figure 4.16 A total of 10 neurons were used per layer in the neural network to simulate the rainfall data.

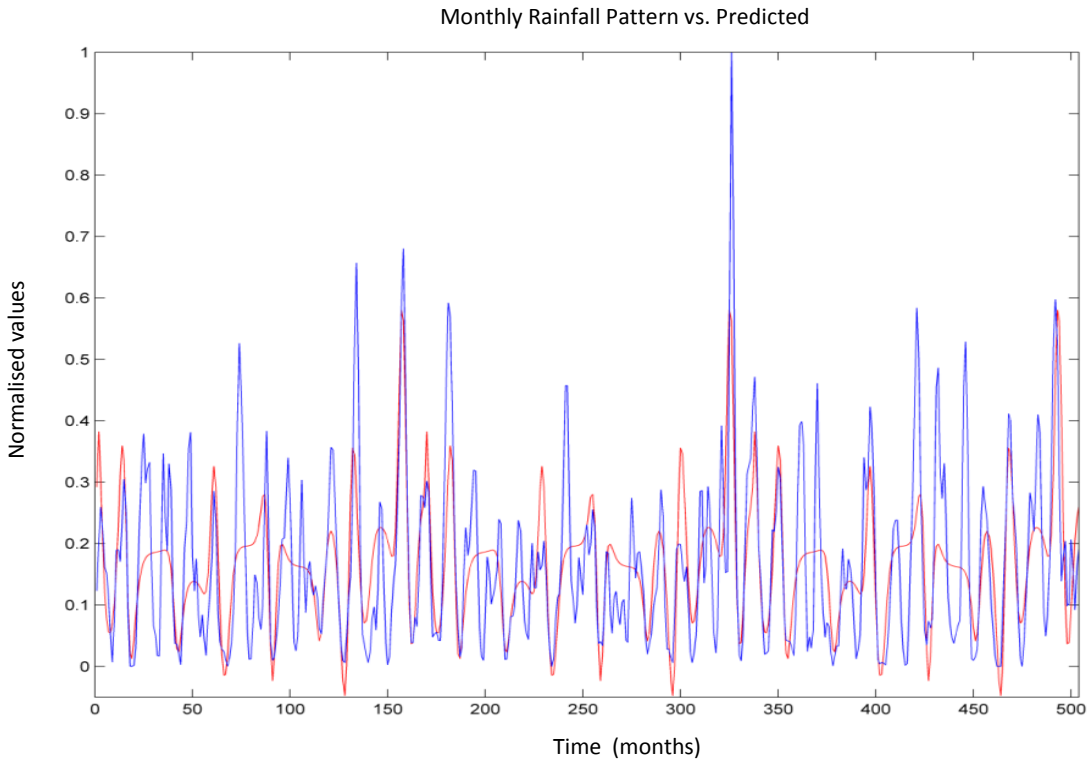


Figure 4.17 A total of 20 neurons were used per layer in the neural network to simulate the rainfall data.

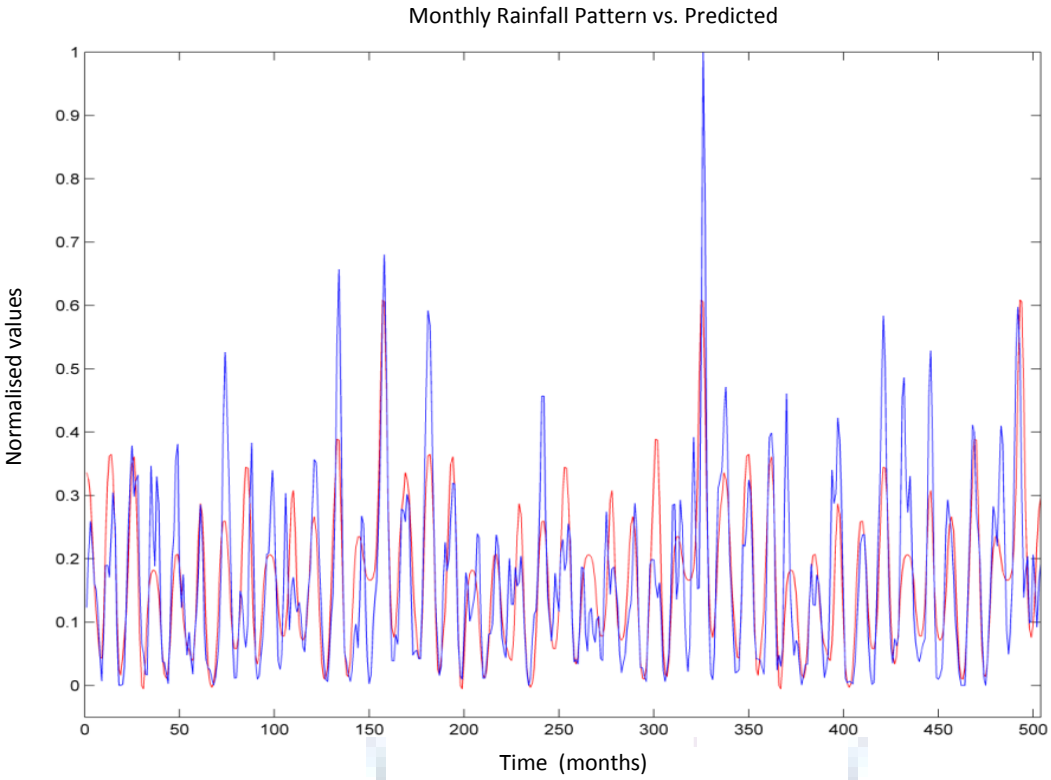


Figure 4.18 A total of 35 neurons were used per layer in the neural network to simulate the rainfall data.

It was expected that a steady increase in neurons per layer would increase the resolution of the fitting pattern of the neural network data to the actual observed values. At 35 neurons per layer the data had a good enough fit, with an increase of neurons above this level having no real effect on the prediction anymore, see Table 4.2. However the resolution of each peak increased as the number of neurons increased. In order to get the best fit and resolution a range of 50 to a 150 were found to be optimal per layer.

Table 4.2 Variable neurons per layer effect on fitting of Rainfall data

Neurons	Linear regression percentage	Performance value
10	34.11	0.01796
20	67.44	0.01094
35	76.50	0.00701
50	77.96	0.00660
100	77.99	0.00581
150	77.78	0.00508
200	77.24	0.00683

4.3.6. Number of Layers in the Neural Network

The influence of the number of layers was also investigated. Interestingly, two and three layer neural networks performed the best; while 4 and more layers gave no results. These results also indicate that the 3 layer network is only marginally better than the 2 layer network. This allows for neural networks to be constructed for either scenario, although 2 layer networks have less resolution than 3 layer neural networks. Table 4.3 presents the results of the number of layers.

Table 4.3 Variable number of layers, with a total of 100 neurons per layer.

Number of Layers	Linear regression percentage	Performance value
2	76.89	0.00491
3	77.99	0.00581
4	No Fit	No Fit

4.3.7. The effect of transfer function on data estimation

A two and three layer network was constructed with different transfer functions contained within it to determine the stability of the neural network. Simulations were run using 10 random neural network starting points; the results are given below in Table 4.4.

Table 4.4 Simulation runs for a 100 neuron neural network. Average values reported for each data set.

Trial Run	2 Layer	2 Layer	3 Layer	3 Layer
Transfer function	Logsig	Tansig	Logsig	Tansig
1	75	76	76	76
2	77	76	77	78
3	74	75	76	76
4	70	69	77	12
5	75	69	79	76
6	71	74	78	76
7	76	73	67	77
8	75	70	76	74
9	76	65	78	76
10	76	73	78	77
Average Fit (%)	75	72	76	70

Both tansig and logsig transfer function simulated the data to nearly the same level. However, the stability of these networks does differ. The difference can be observed from the average fit value in Table 4.4, in which the three layer logsig neural network performed the best. In this instance the average value corresponds to the stability or ability of the neural network to give a constant result.

A further observation is that once the fitting of the data is inspected, negative fitting values were found for the two layer network (Figure 4.19). This clearly indicates that although two layer

networks might be better on average at predicting time series data, that this method might share similar drawback as the statistical methods.

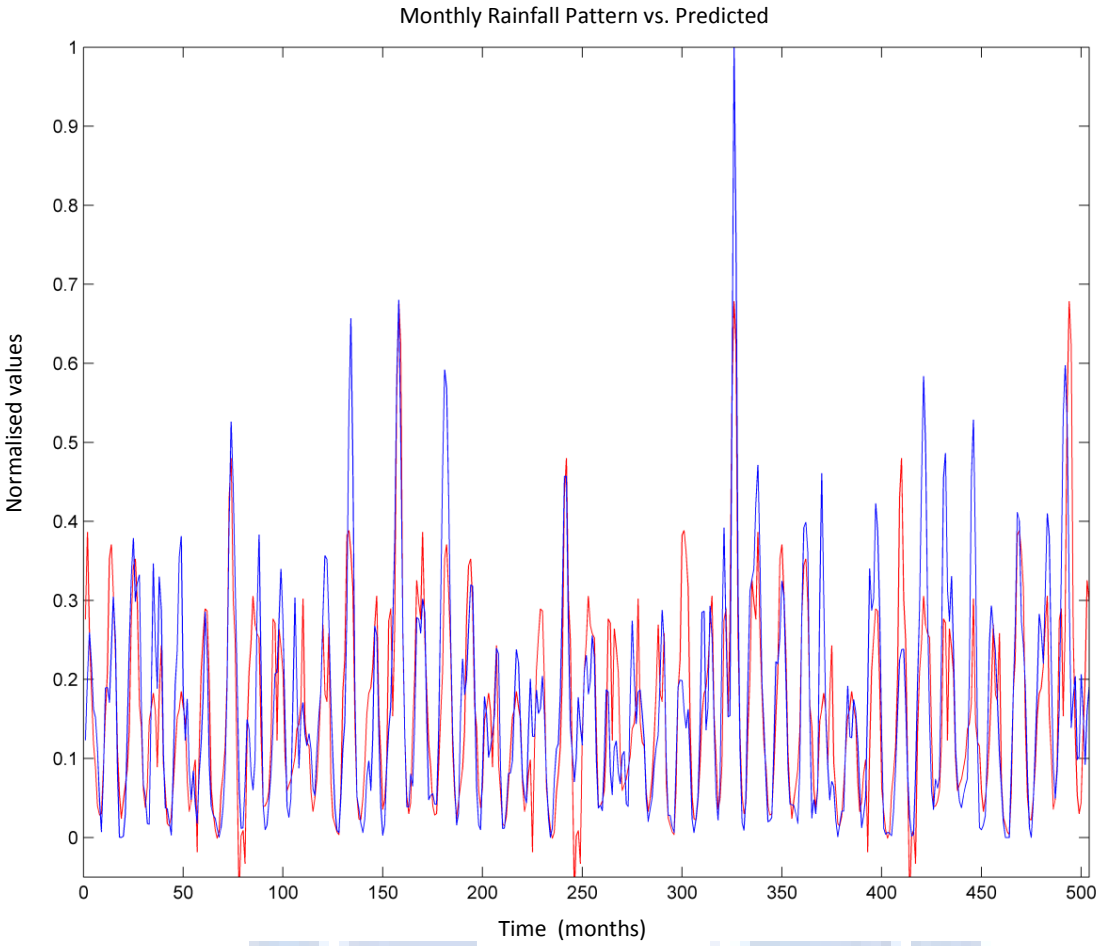


Figure 4.19 A two layer neural network from the trial run simulations of Table 4.4. Blue line indicates actual data while red line indicates simulated results.

4.3.8. The effect of different number of neurons per layer

In order to establish if the neural network was significantly affect by the change in number of neurons per level a statistical data set of possible combinations were simulated. The data set contained 100 variations and that were repetitively simulated over a subset of 100 entries. Ironically, from the two and three layer neurons per network, the two layer networks outperformed the three layer neural networks. These top 17 values are tabulated in Table 4.5. The average performance, linear regression, median and standard deviation were calculated. The best performing neural

networks were homogeneous in neuron content, with the 100 neuron neural network showing a slight preference for modelling the rainfall data.

Table 4.5 Training results for two layer neural networks. Transfer functions used are tansig, logsig ad purelin.

Entry	Layers	Neurons per Layer			Transfer function		Ave Perf	Ave R	Med R	Std R
		1	2	3	1	2				
17	2	100	100	100	Logsig	Purelin	0.007	76.8	76.6	1.2
25	2	50	50	50	Logsig	Purelin	0.007	76.3	76.3	1
26	2	50	50	50	Tansig	Purelin	0.007	75.5	75.5	1.4
49	2	100	100	50	Logsig	Purelin	0.006	75.5	75.3	1.5
18	2	100	100	100	Tansig	Purelin	0.006	74.7	75.8	3.6
50	2	100	100	50	Tansig	Purelin	0.006	73.6	74.2	2.5
9	2	150	150	150	Logsig	Purelin	0.006	72.8	74.1	3.9
1	2	200	200	200	Logsig	Purelin	0.006	71.7	72.2	2.3
10	2	150	150	150	Tansig	Purelin	0.006	64.7	65.2	5.2
2	2	200	200	200	Tansig	Purelin	0.005	61.2	60	5.8
113	2	50	100	20	Logsig	Purelin	0.013	56.7	57	1.5
58	2	50	100	50	Tansig	Purelin	0.012	56.5	56.8	1.8
121	2	50	100	10	Logsig	Purelin	0.012	56.5	57.1	2.1
105	2	50	100	30	Logsig	Purelin	0.012	56.4	57.7	3.8
106	2	50	100	30	Tansig	Purelin	0.012	56.1	57.4	4
114	2	50	100	20	Tansig	Purelin	0.012	56.1	56	1.7
97	2	50	100	40	Logsig	Purelin	0.013	55.9	57.4	3.7

4.4. Conclusion

The determination of artificial neural network configuration has been attempted in this chapter. The effect of number of neurons, layers and transfer functions have been reported. In order to simulate data a comparison with statistical methods had been made and fitting of these values were far inferior to that observed by neural networks. Using artificial neural networks to patch rainfall data does seem to be an option, with simulated data having an average confidence value of 77 %.

Chapter 5

Surface water and Groundwater modelling with Neural Networks

5.1. Introduction

In the following section the background of Chapter 3 will be used to model variable time series data, i.e., rainfall, flow volumes in rivers and water levels in boreholes. An attempt will be made to use surface water parameters (rainfall and flow volumes) to predict groundwater behaviour (water levels). During the past ten years a significant amount of work has been done on using artificial neural networks to predict water table fluctuations[16,31,32,33]. Each of these network applications were specifically tailored to a problem or adapted to give a favourable result, which is only possible with artificial neural networks.

One particular report from Coulibaly et al.[31] was of specific interest, since the authors used rainfall and temperature data to predict water table fluctuations. However, five input variables were required and seven years of monthly training data was used to prepare the artificial neural network for prediction. In the current context of South Africa obtaining more than seven years of continuous data for a borehole site is nearly impossible, this is mostly due to the prohibitive cost of monitoring programs. It is the aim of this section to determine if fewer data points can be used to effectively predict water levels.

Three case studies were chosen for this investigation. Firstly, an idealised model was constructed using the River package in ModFlow. Secondly, a site was identified in the Dwars River system.

Finally, data was obtained from the Vaal River region to determine if long distance (> 1 km) interactions between surface water and groundwater could be modelled.

5.2. Case Study 1 Idealised model system

In order to simplify the model, a 100 m x 100 m area was selected with a grid size of 1 m x 1 m. The single model layer was 20 m thick and was run under confined conditions. The transmissivity and storage coefficients were set to be user specified. An observation borehole was placed 30 m from the lower border and 50 m from the sides, this was done to monitor water level deviation at each of the different stress periods, see Figure 5.1. A 20 m wide and 100 m long river was passed through the area which was located 10 m from the top section with a depth of 5 m. Due to the River Package setup in ModFlow, multiple monthly stress periods were specified over a five year time span.

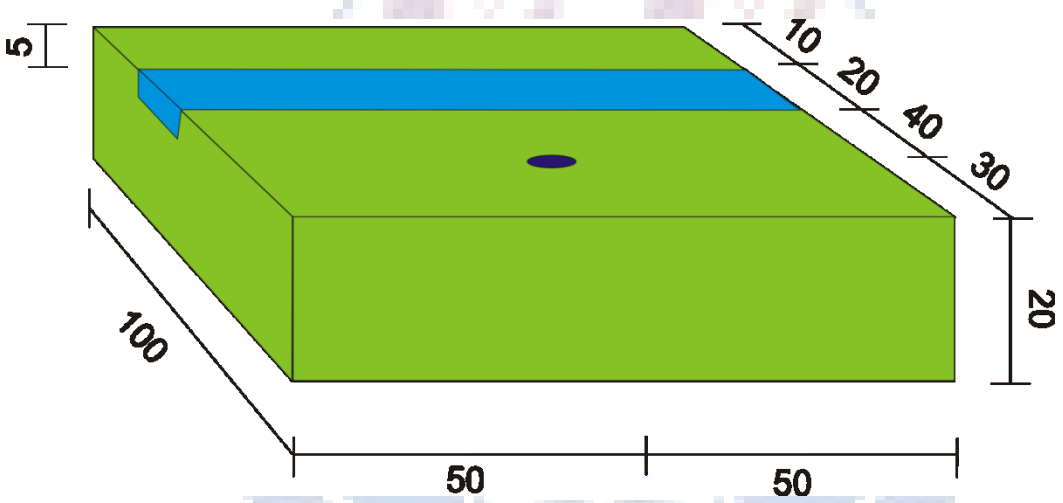


Figure 5.1 Diagrammatic representation of model area with river zone.

The initial hydraulic head for the total area excluding the river was set to 2.5 m below ground level surface. Horizontal hydraulic conductivity was set to 0.001 m/d and a transmissivity value of 0.01 m^2/d was used. In order to observe changes in the monitoring borehole, the storage coefficient was set 0.001. As noted previously multiple stress periods (60) were used to adjust the head in the river, with a hydraulic conductivity of the riverbed set to 10 m^2/d and elevation of riverbed 5 m below the ground level surface.

Considering the North Platte River system, previously reported in Section 3.6 and Figure 3.20, a nearly linear correlation exists between rainfall values and flow volumes in a terrestrial system. If the

current model is to be a representation of a natural system, artificial rainfall values were added. The values as noted previously should follow the global pattern simulated in the stress periods. Seasonal rainfall data was generated and flow volumes in the river was adjusted to match. The result of the simulation is shown in Figure 5.2.

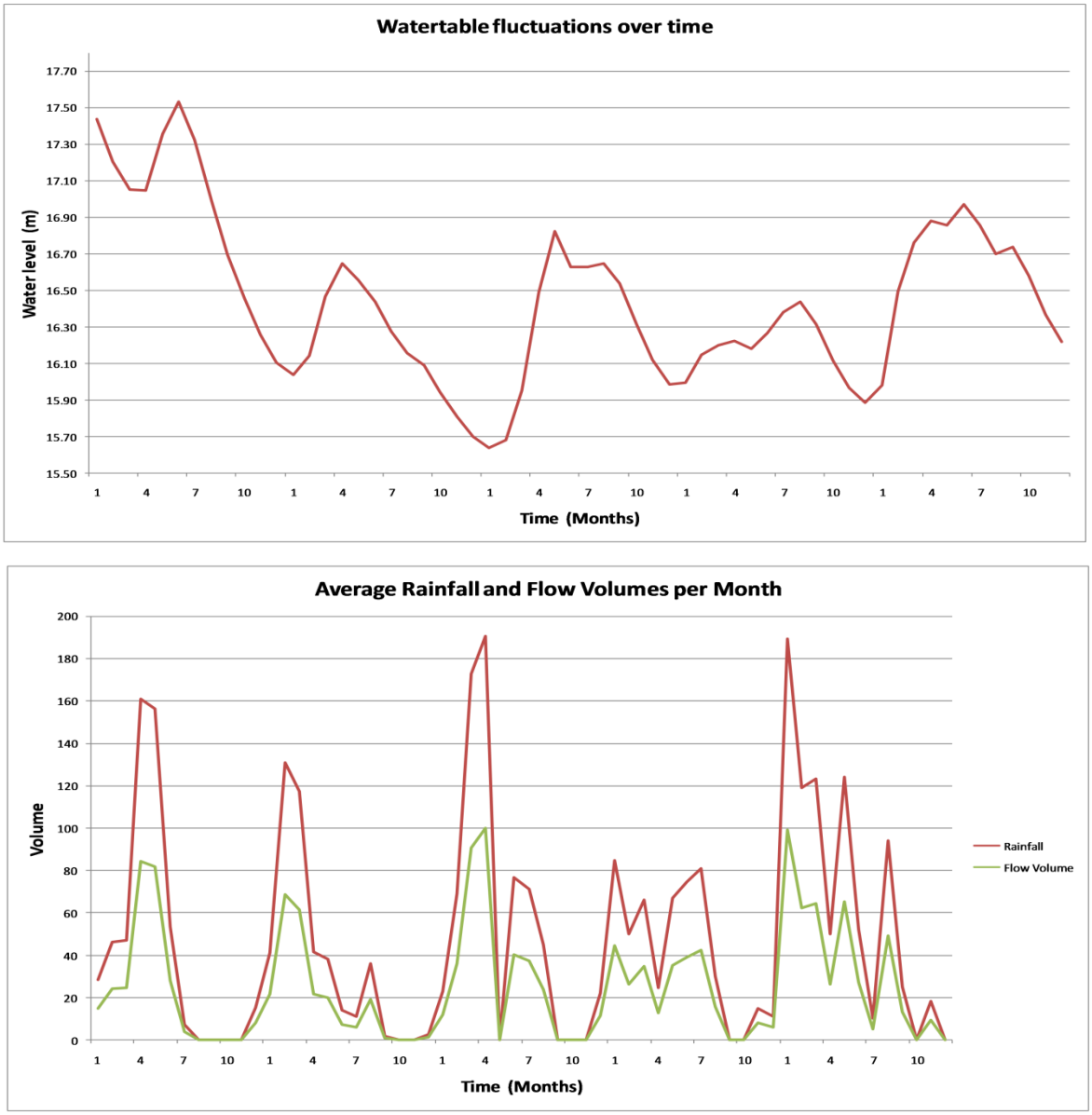


Figure 5.2 Idealised model data from ModFlow (River Package). Top section indicates average water table fluctuations over a 5 year time period in m, while lower section shows average rainfall per month (red) and flow volumes discharged from the river (green) in mm and m^3 per month respectively.

The data generated was used in four different artificial neural network architectures to determine if the groundwater interaction with the surface water contribution could be predicted. A Focused

Time-Delay Network, Layer-Recurrent Network, Radial Basis Network and Probabilistic Neural Network were tested to verify its respective efficiency to predict “one step ahead” values for the borehole from surface water data.

5.2.1. Focused Time-Delay Neural Network

To effectively monitor the performance of the focused time-delay neural network, the number of neurons per layer was adjusted from five to ten. A delay of two and six months was used on the rainfall and flow volume data to assist in predicting the water level in the borehole. The results of these variations in network configuration are reported in Figure 5.3 - Figure 5.5. During the training phase fifty nine months of data was used and only the last point in the five year cycle was predicted, i.e., one step-ahead.

The time delay effect in the neural network can play a significant role in the predictive capabilities of the system. In Figure 5.3 a six month time delay was used, which resulted in a reasonable fit to the data with only five neurons per layer. The remedial mean square error value for the training set and predicted data is less than thirteen percent. In contrast if a two month delay is used, the data correlation visibly worsens with a remedial mean square error value exceeding thirty six percent (Figure 5.4). Furthermore, if the number of neurons is increased to ten and a six month delay period is used the fit of experimental to observed data is nearly one hundred percent (Figure 5.5).

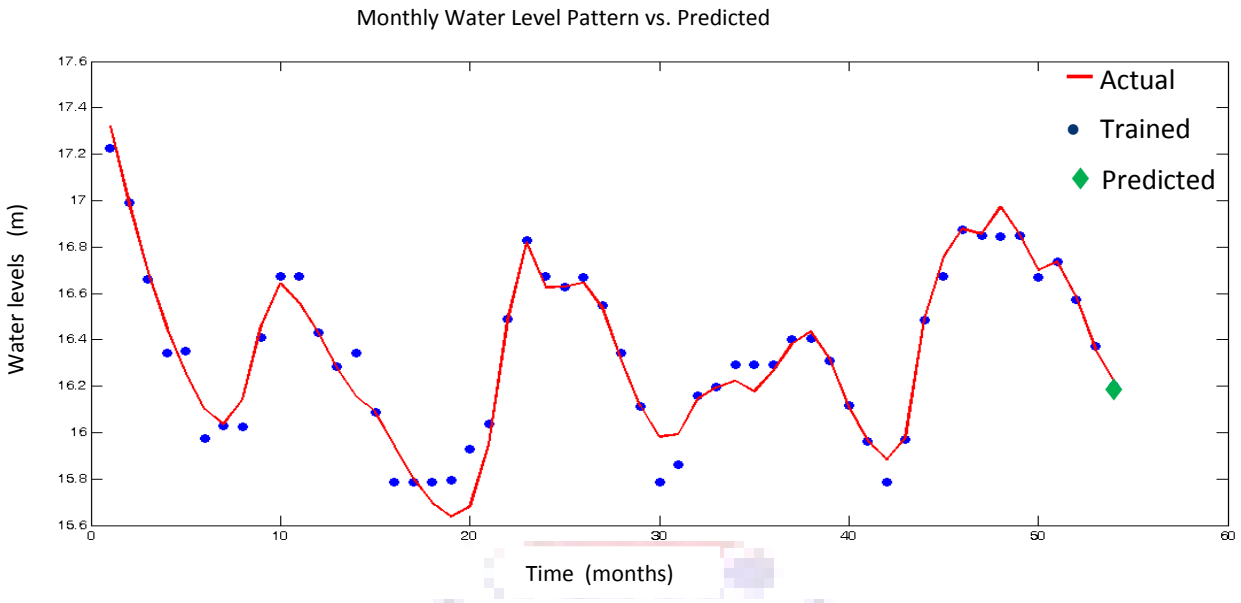


Figure 5.3 Predicted Water levels for the model system using Rainfall and Flow volume in river. Focused time-delay neural network contained two layers with 5 neurons per layer and a time delay of 6 months. Data estimation error less than 12.7 %.

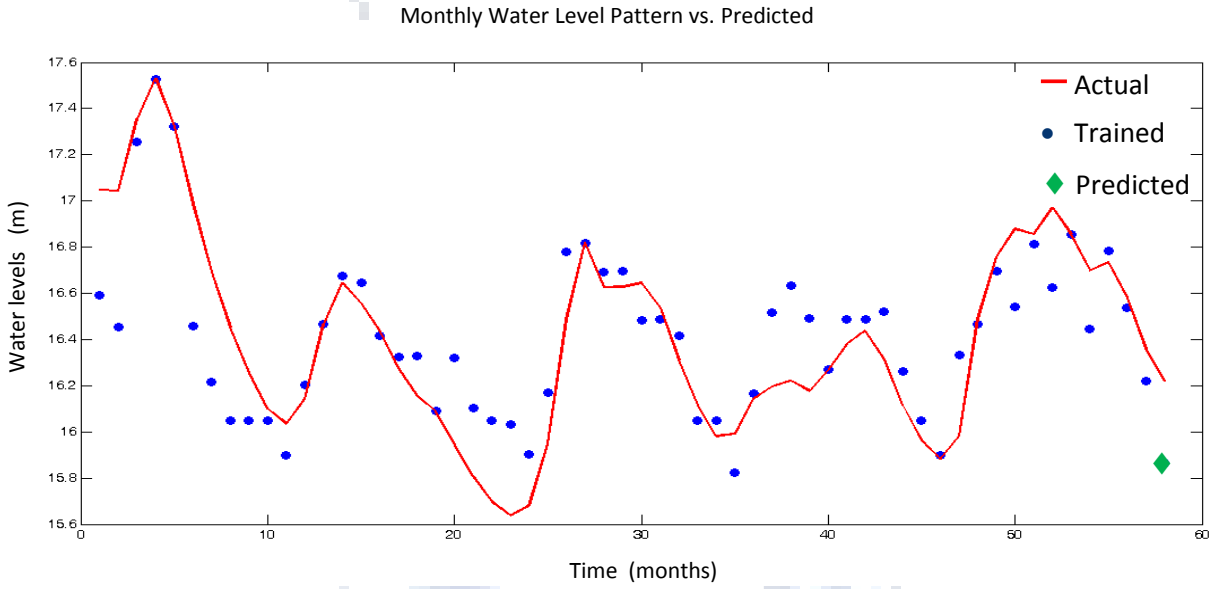


Figure 5.4 Predicted Water levels for the model system using Rainfall and Flow volume in river. Focused time-delay neural network contained two layers with 5 neurons per layer and a time delay of 2 months. Data estimation error less than 36.7 %.

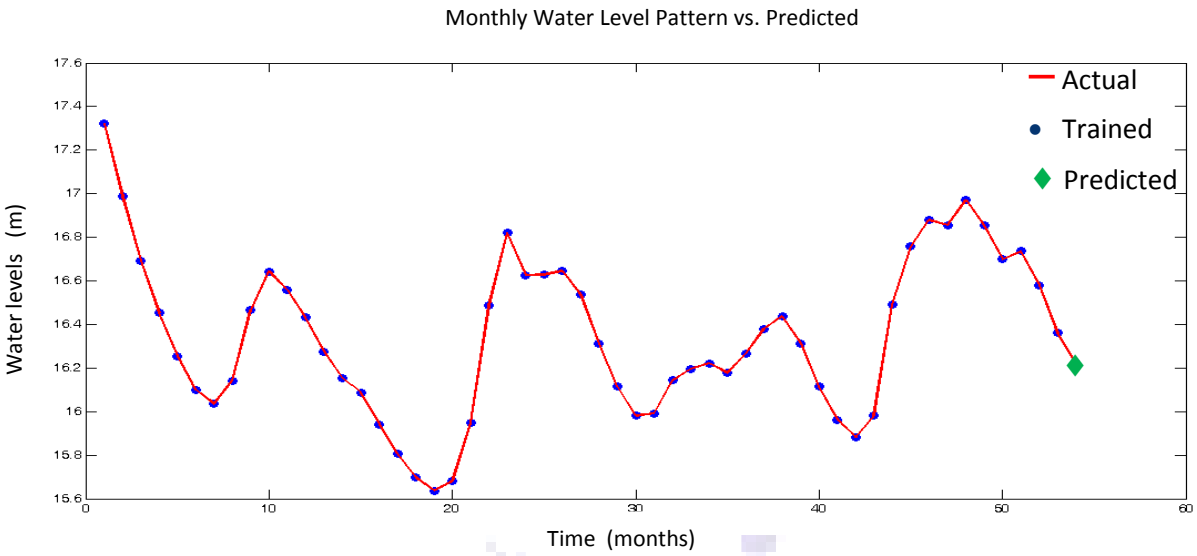


Figure 5.5 Predicted Water levels for the model system using Rainfall and Flow volume in river. Focused time-delay neural network contained two layers with 10 neurons per layer and a time delay of 6 months. Data estimation error less than 0.0001 %.

The predictive power of the focused time-delay neural network was estimated by running a sample set containing one thousand independently trained neural networks and observing the predicted value in the final step compared to the actual value. Using five neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.6. Only a few outliers are observed as found in the histogram plot, Figure 5.7, indicating that the neural network reliably predicts the observed data. The data has a maximum and minimum of 18.84 m and 3.00 m, respectively. The average value for the data set is 15.98 m with a standard deviation of 0.58. The observed (16.22 m) value is well within the two standard deviation value from the average which indicates that the focused time-delay neural network with five neurons effectively predicts the trend.

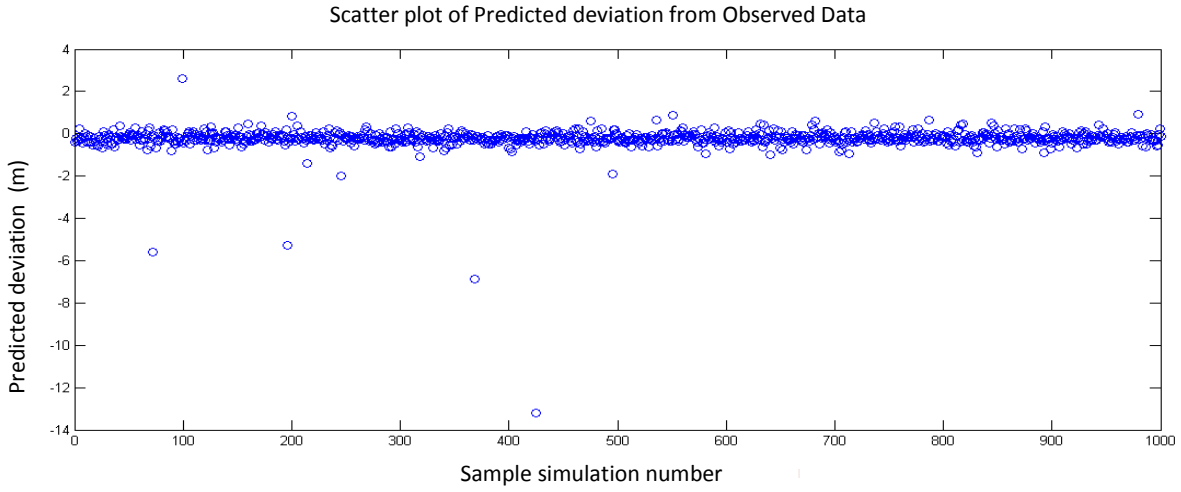


Figure 5.6 Neural network containing 5 neurons. Predicted deviation from observed data point from 1000 sample runs.

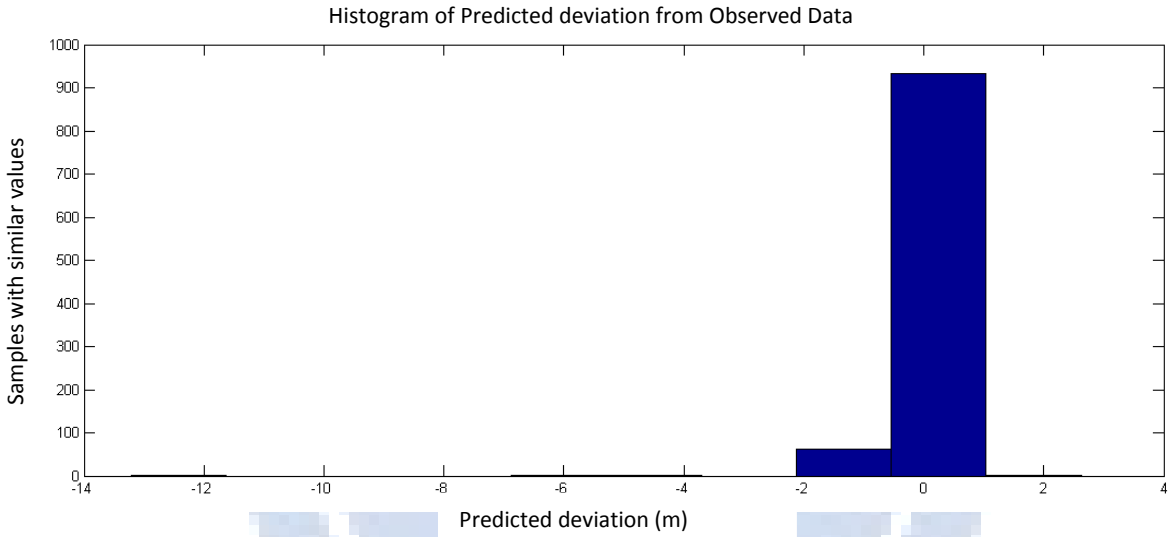


Figure 5.7 Histogram showing the number of results deviating from the observed value.

Using ten neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.8. Only a few outliers are observed as found in the histogram plot, Figure 5.9, indicating that the neural network reliably predicts the observed value. The data has a maximum and minimum of 19.59 m and 8.88 m respectively. The average value for the data set is 15.71 m with a standard deviation of 0.83. The observed (16.22 m) value is well within the two standard deviation value from the average which indicates that the focused time-delay neural network with ten neurons effectively predicts the trend. Additionally, the spread of the solutions are significantly reduced compared to the five neuron network.

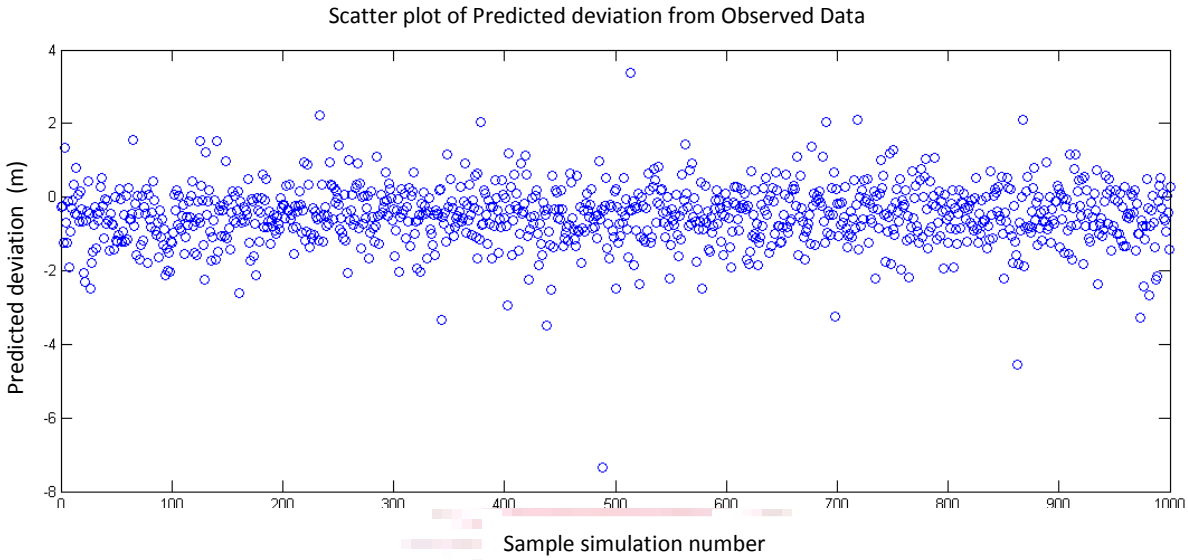


Figure 5.8 Neural network containing 10 neurons. Predicted deviation from observed data point from 1000 sample runs.

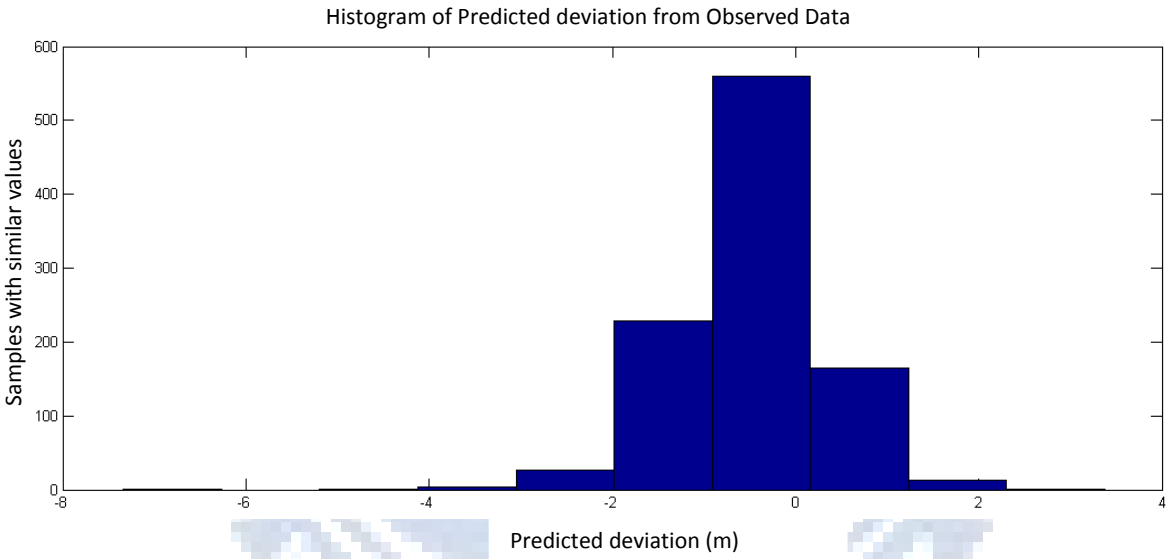


Figure 5.9 Histogram showing the number of results deviating from the observed value.

5.2.2. Layer-Recurrent Neural Network

To investigate layer-recurrent networks, the number of neurons per layer was adjusted from five to ten and multiple simulations were run to determine its efficacy. During the training phase fifty nine months of data was used and only the last point in the five year cycle was predicted, i.e., one step-ahead. The results of these simulations are shown in Figure 5.11 – Figure 5.15.

The predictive power of the layer-recurrent neural network was estimated by running a sample set containing one thousand independently trained neural networks and by observing the predicted value in the final step compared to the actual value. Using five neurons per layer, the sample set was calculated with an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.11. Only a few outliers are observed as found in the histogram plot, Figure 5.12, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.10. The data has a maximum and minimum of 17.07 m and 15.89 m respectively. The average value for the data set is 16.22 m with a standard deviation of 0.06, the observed value (16.22 m) matches exactly the average predicted value.

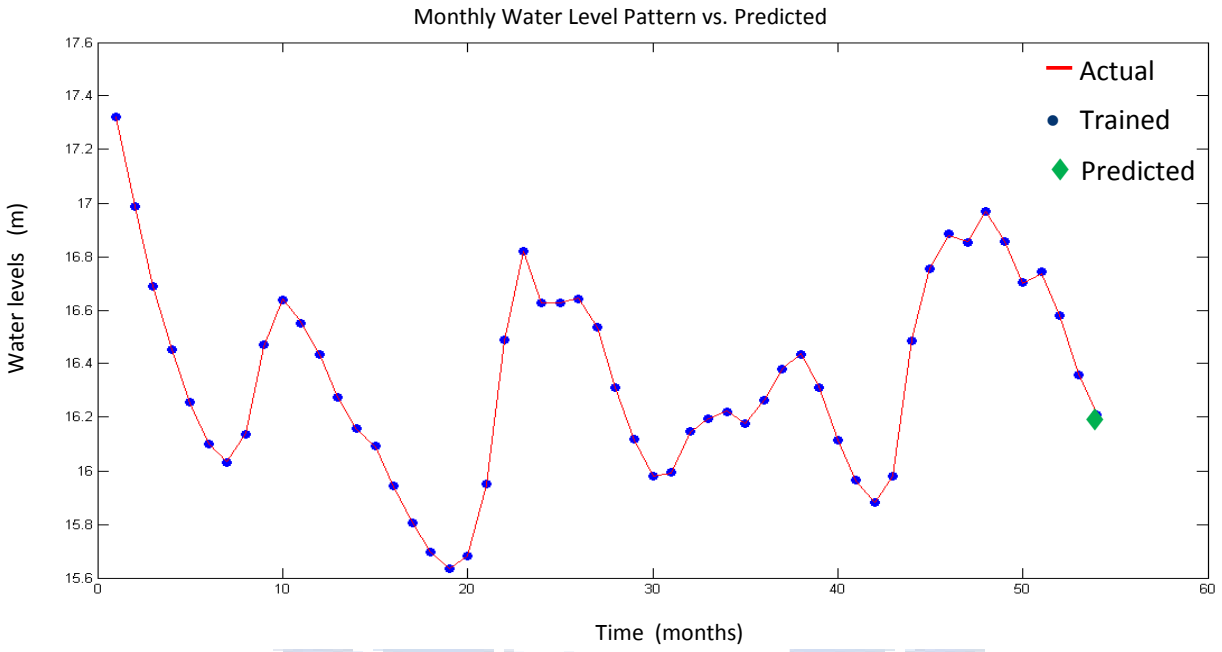


Figure 5.10 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 5 neurons per layer. Data estimation error less than 0.0001 %.

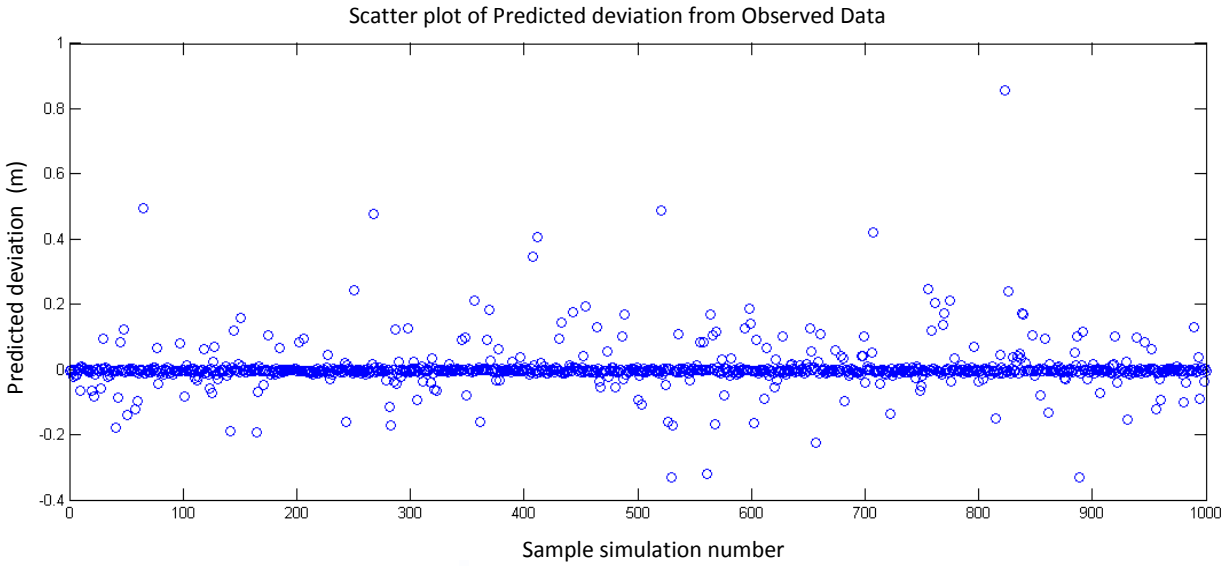


Figure 5.11 Neural network containing 5 neurons. Predicted deviation from observed data point from 1000 sample runs.

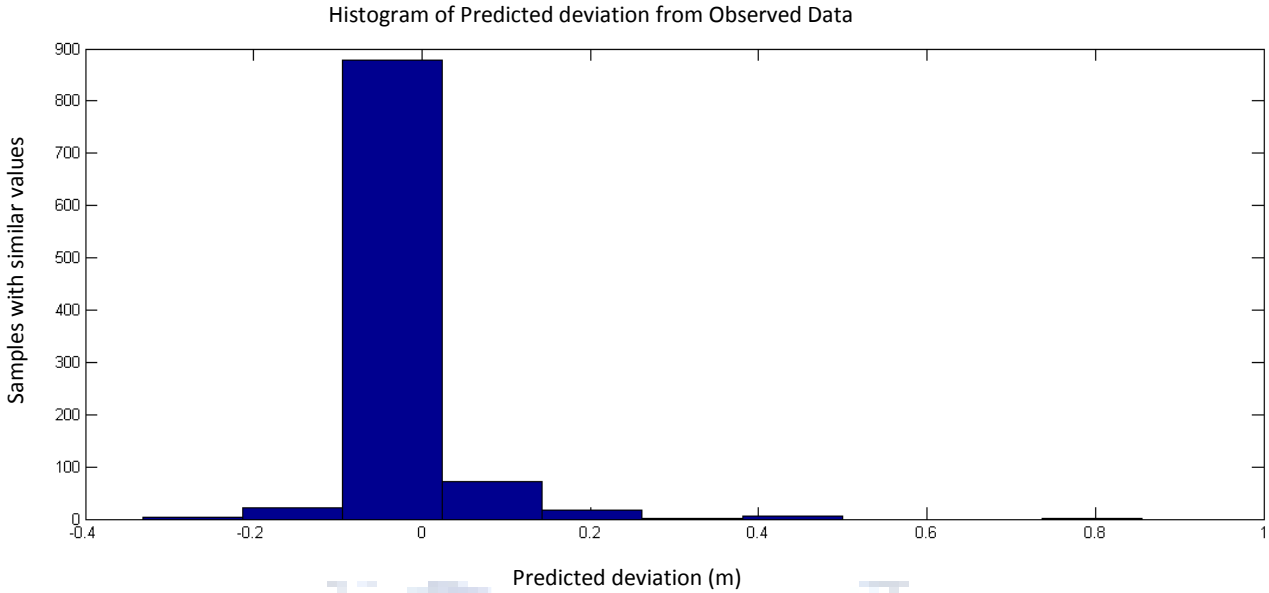


Figure 5.12 Histogram showing the number of results deviating from the observed value.

Using ten neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.14. Only a few outliers are observed as found in the histogram plot, Figure 5.15, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.13. The data has a maximum and minimum of 16.84 m and 14.76 m respectively.

The average value for the data set is 16.22 m with a standard deviation of 0.07; the observed (16.22 m) value is exactly the same. However, the spread of solutions are significantly increased compared to the five neuron network from 1.4 to 2.5 m.

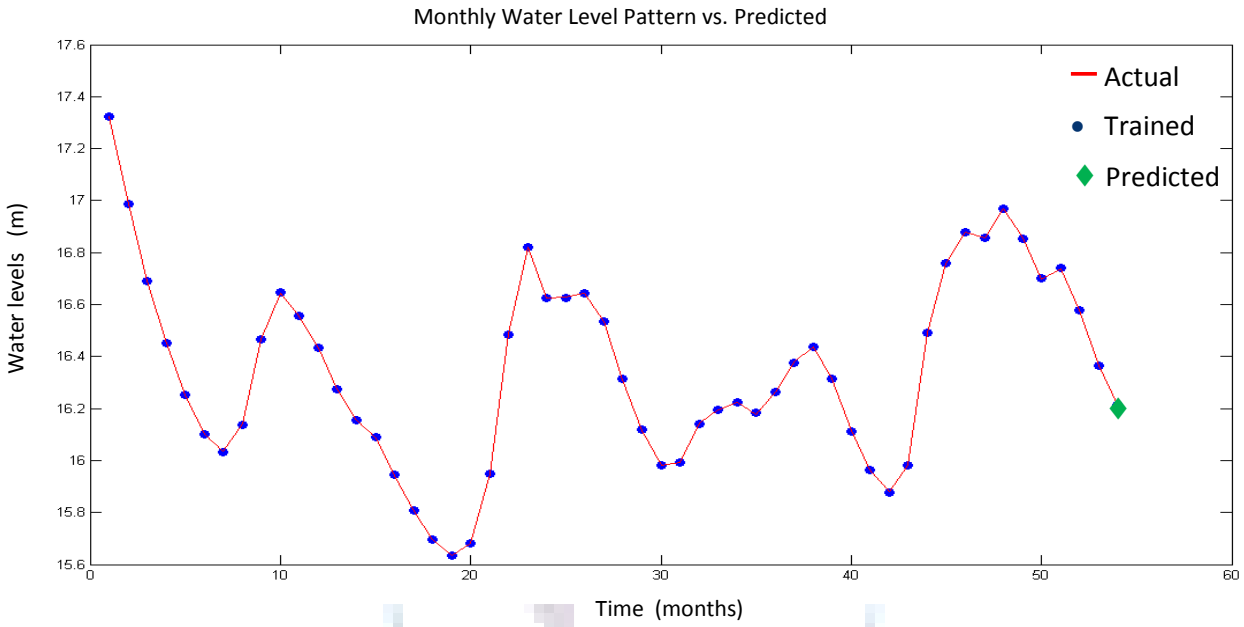


Figure 5.13 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 10 neurons per layer. Data estimation error less than 0.0001 %.

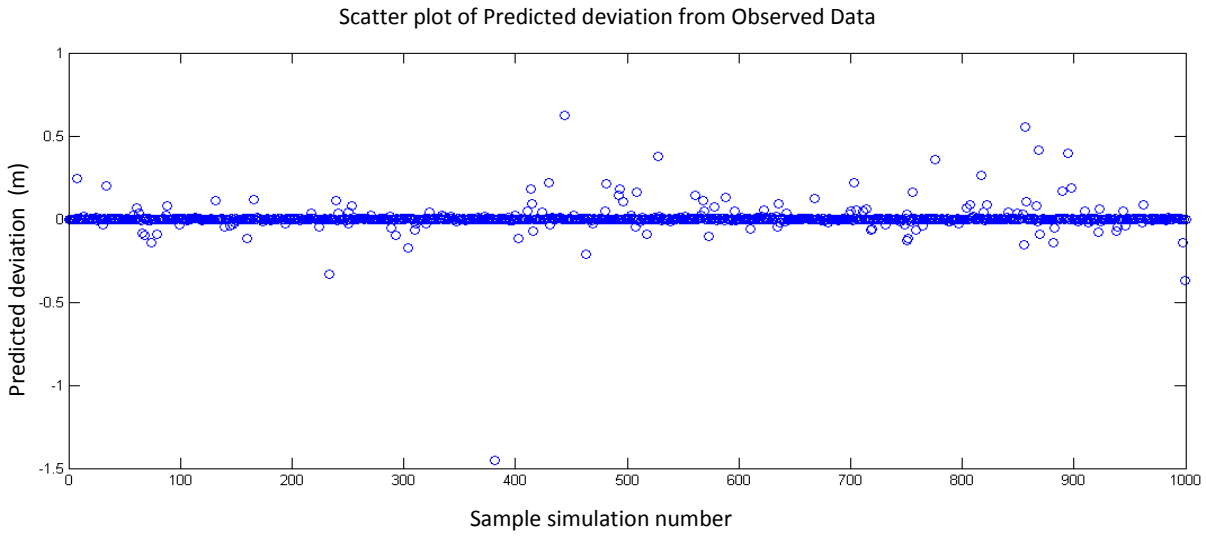


Figure 5.14 Neural network containing 10 neurons. Predicted deviation from observed data point from 1000 sample runs.

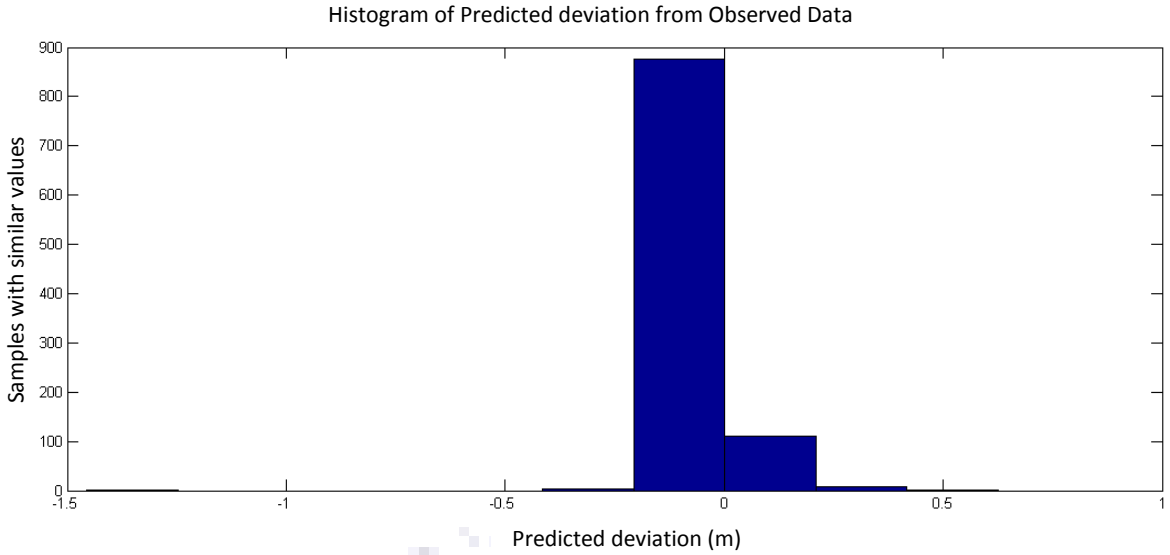


Figure 5.15 Histogram showing the number of results deviating from the observed value.

5.2.3. Radial Basis Neural Network

Radial basis networks were investigated using the standard sampling technique. However, the spread of the radial basis function was adjusted from 0.25 to 1.25. This determines the width of an area in the input space to which each neuron will respond. During the training phase fifty nine months of data was used and only the last point in the five year cycle was predicted, i.e., one step-ahead. The results of these simulations are shown in Figure 5.17 – Figure 5.18.

The predictive power of the radial basis neural network was estimated by running a sample set containing one thousand independently trained neural networks and by observing the predicted value in the final step compared to the actual value. Using an adjustable radial basis function, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.17. Only a few outliers are observed as found in the histogram plot, Figure 5.18, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.16. The data has a maximum and minimum of 16.38 m and 16.37 m respectively. The average value for the data set is 16.38 m with a standard deviation of 0.001. The observed (16.22 m) value is not within the two standard deviation value from the average which indicates that the radial basis neural network with adjusted radial basis function does not effectively predict the trend. Secondly, from Figure 5.17, it is

clear that the radial extent must be further adjusted to obtain a more optimal simulation of the data.

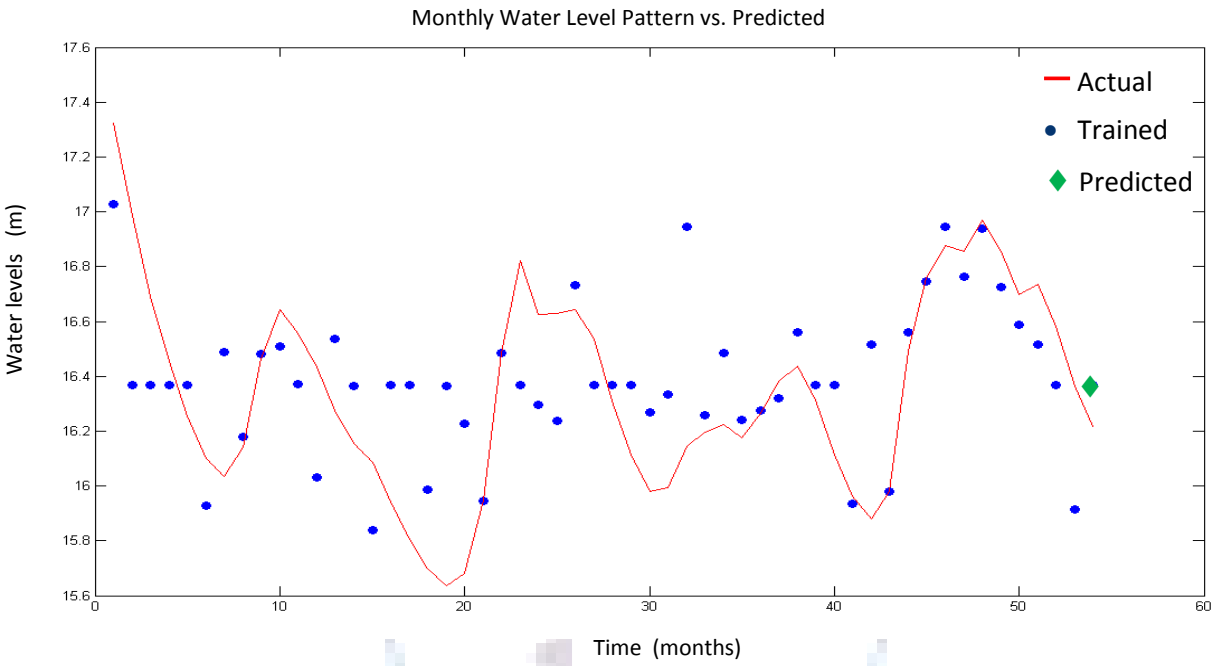


Figure 5.16 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Radial basis extent was chosen as 0.651. Data estimation error less than 45 %.

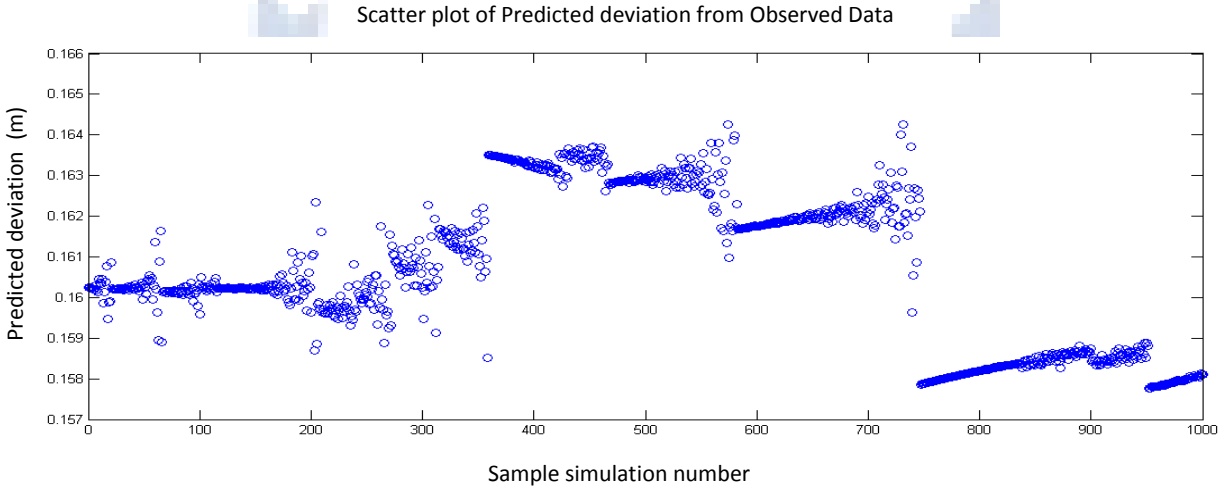


Figure 5.17 Neural network containing radial basis functions adjusted incrementally from 0.25 to 1.25. Predicted deviation from observed data point from 1000 sample runs.

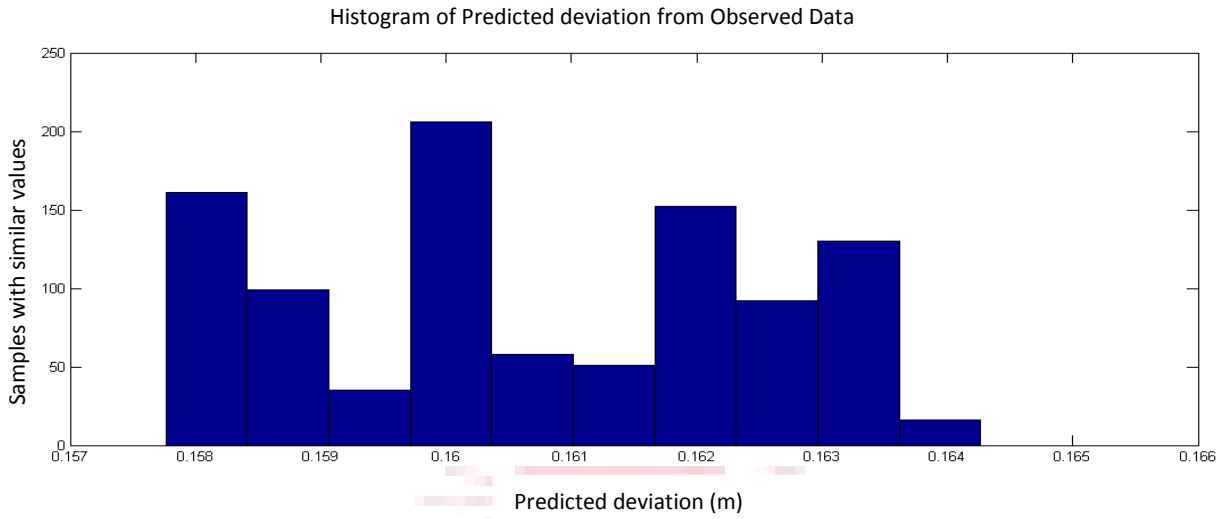


Figure 5.18 Histogram showing the number of results deviating from the observed value.

5.2.4. Probabilistic Neural Network

Probabilistic neural networks can be used for classification problems. When an input is presented, the first layer computes distances from the input vector to the training input vectors and produces a vector whose elements indicate how close the input is to a training input. In these networks only a single output value is presented. The simulation of the data by this network is presented in Figure 5.19. The error in the simulation using the probabilistic network does not follow the data with a significant degree of accuracy, resulting in a highly variable fit. Considering only the last step the estimation of the actual value (16.22 m) is 15.81 m which has a consistent underestimation of 0.41 m; similar behaviour has been observed in statistical models.

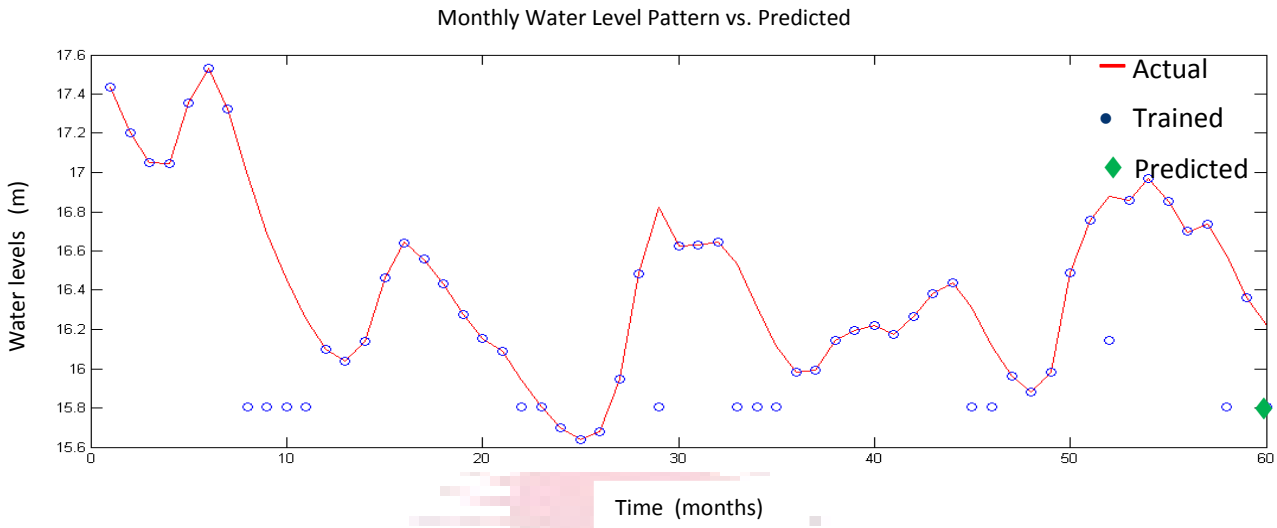


Figure 5.19 Predicted Water levels for the model system using Rainfall and Flow volume in river. Actual data presented as a red line while blue dots indicate estimation values.

5.3. Case Study 2 Dwars River system

In this case study a borehole was identified next to the Dwars River system, which had a constant change in water flow patterns. The borehole coordinates are shown in Figure 5.20.

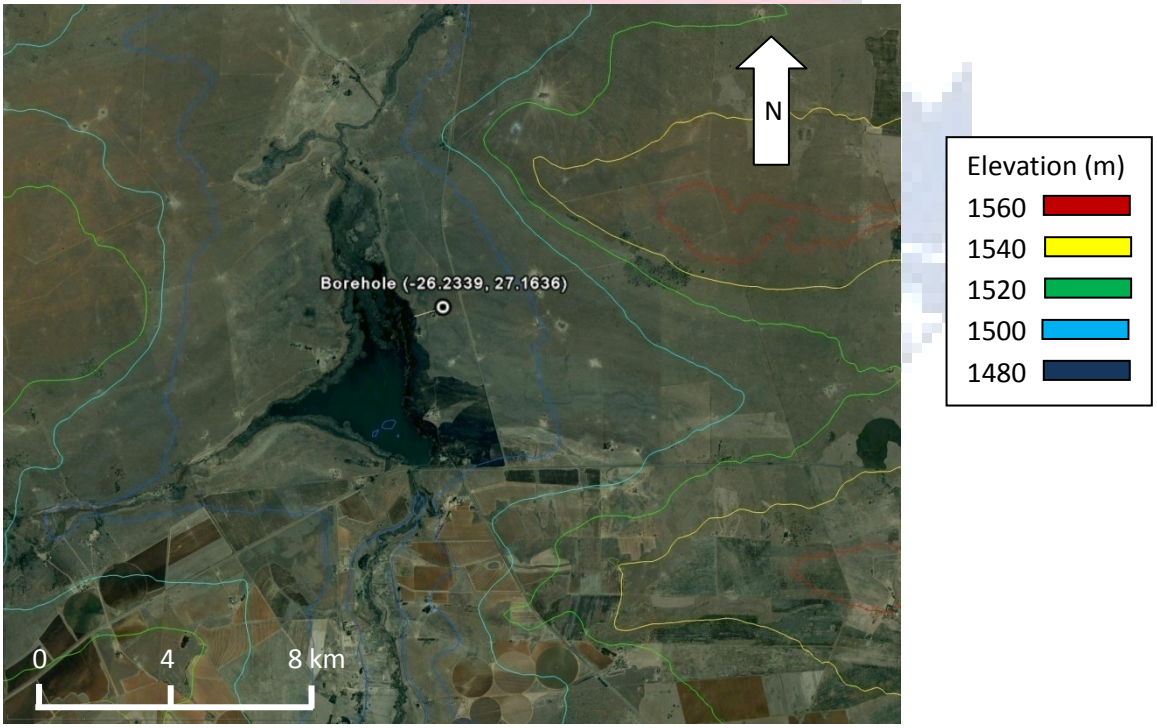


Figure 5.20 Aerial photo with contour data of the study area in the Dwars River system. Borehole is ca. 300 m from the rivers edge as indicated by the white line.

Water levels, flow volumes and rainfall data were collected from the relevant government organisations; this data is represented in Figure 5.21.

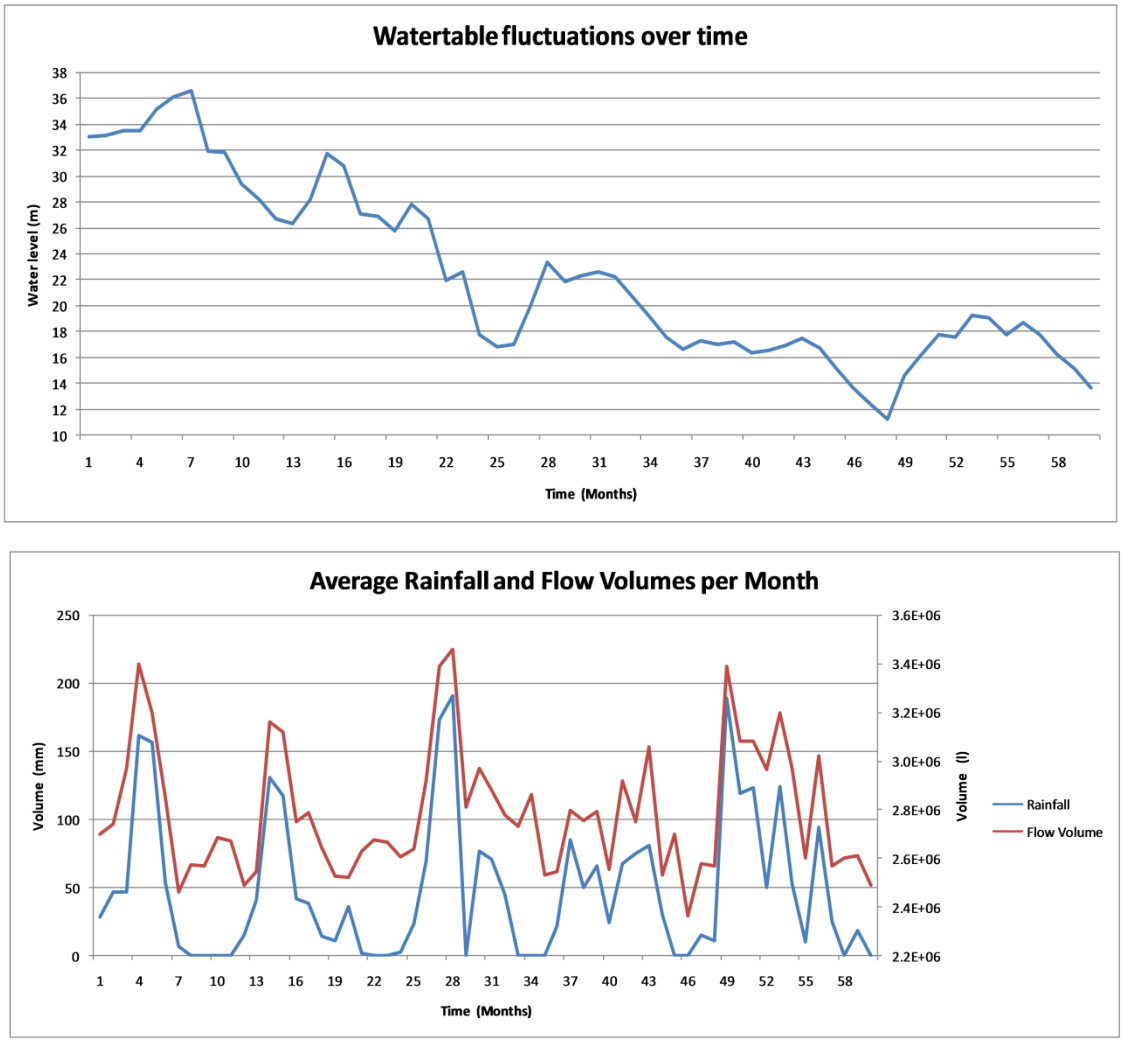


Figure 5.21 Dwars River system data. Top section indicates average water table fluctuations over a 5 year time period in m, while lower section shows average rainfall per month (blue) and flow volumes discharged from the river (red) in mm and m^3 per month respectively.

The data obtained was used in all four of the previously identified artificial neural network architectures to determine if the groundwater interaction with the surface water contribution could be simulated. It should be noted that in this case study only five years of data was available. In the ideal case study it was shown that five and ten neurons per layer were the most effective. In the current investigation only these two possibilities will be examined.

5.3.1. Focused Time-Delay Neural Network

The predictive power of the focused time-delay neural network was estimated by running a sample set containing one thousand independently trained neural networks and observing the predicted value in the final step compared to the actual value.

Using five neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.23. Only a few outliers are observed as found in the histogram plot, Figure 5.24, indicating that the neural network reliably predicts the observed value within error. An example of one of these neural network simulations are shown in Figure 5.22. The data has a maximum and minimum of 24.68 m and 10.94 m, respectively. The average value for the data set is 14.37 m with a standard deviation of 2.05. The observed (13.57 m) value is well within the two standard deviation value from the average. However, the standard deviation of the predicted data set is still reasonably large.

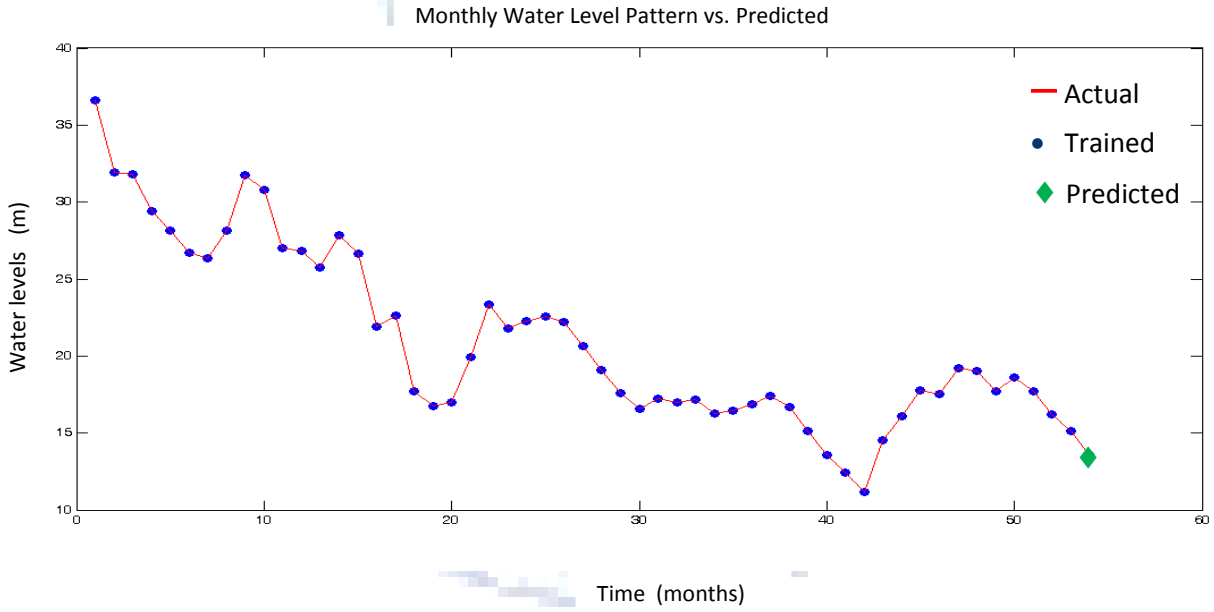


Figure 5.22 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Focused delay neural network contained two layers with 5 neurons per layer. Data estimation error less than 0.0001 %.

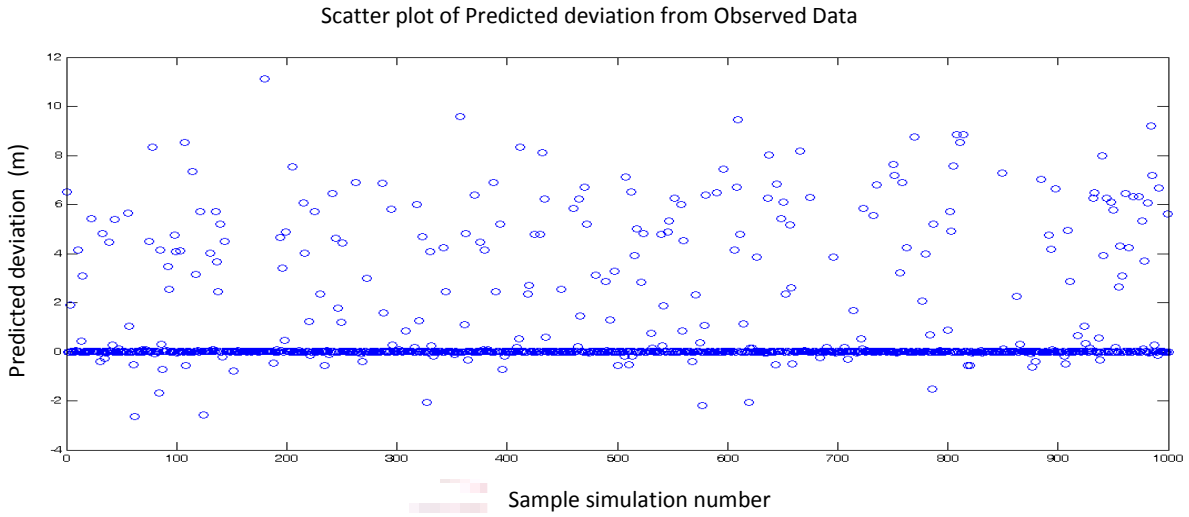


Figure 5.23 Neural network containing 5 neurons. Predicted deviation from observed data point from 1000 sample runs.

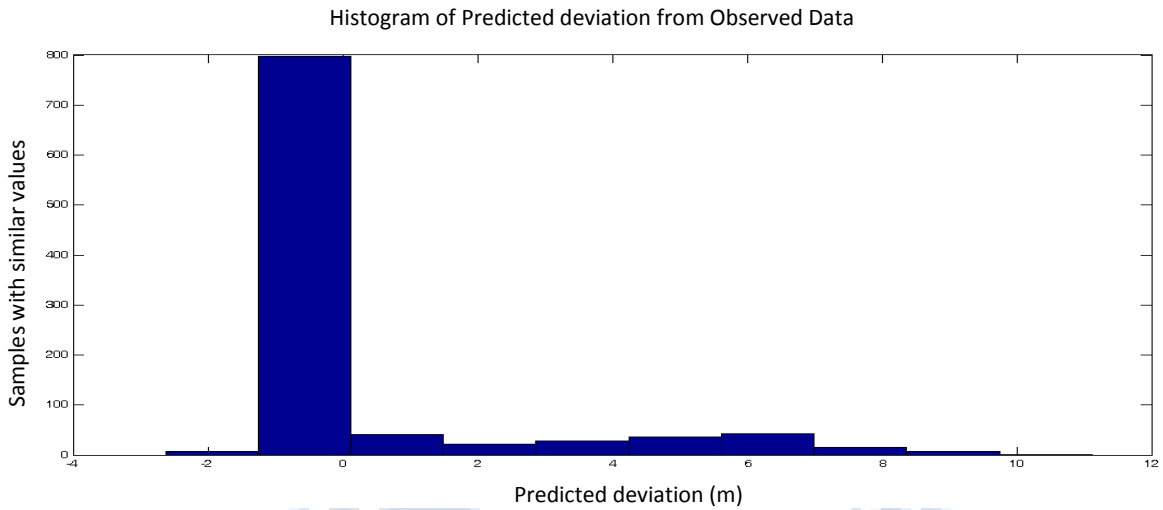


Figure 5.24 Histogram showing the number of results deviating from the observed value.

Using ten neurons per layer, the predicted values for the water level increases in accuracy (Figure 5.26). Only a few outliers are observed as found in the histogram plot, Figure 5.27, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.25. The data has a maximum and minimum of 17.20 m and 12.99 m respectively. The average value for the data set is 13.57 m with a standard deviation of 0.14. The observed (13.57 m) value is well within the two standard deviation value from the average. The standard deviation of the predicted data set is quite small, indicating a very efficient neural network scheme in the prediction of a single step forward.

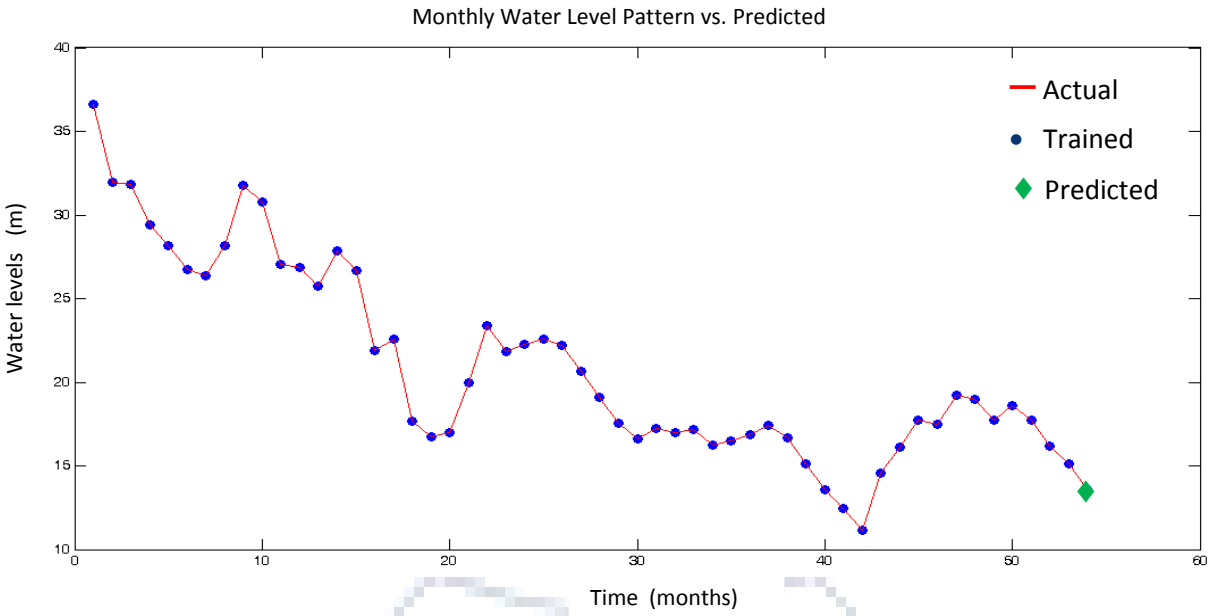


Figure 5.25 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Focussed delay neural network contained two layers with 10 neurons per layer. Data estimation error less than 0.0001 %.

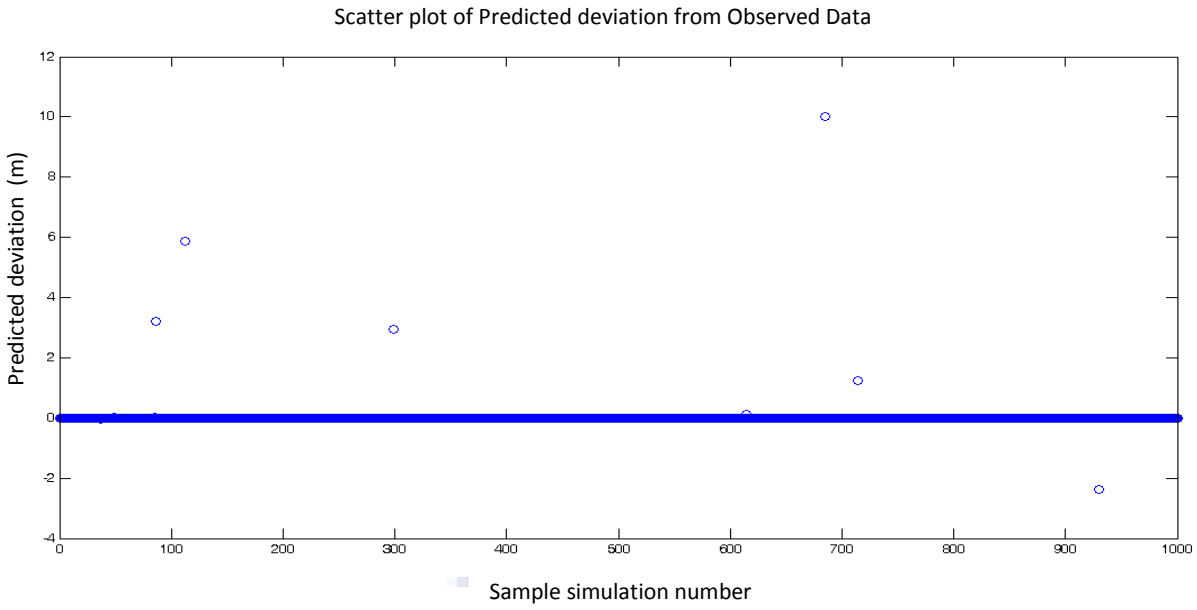


Figure 5.26 Neural network containing 10 neurons. Predicted deviation from observed data point from 1000 sample runs.

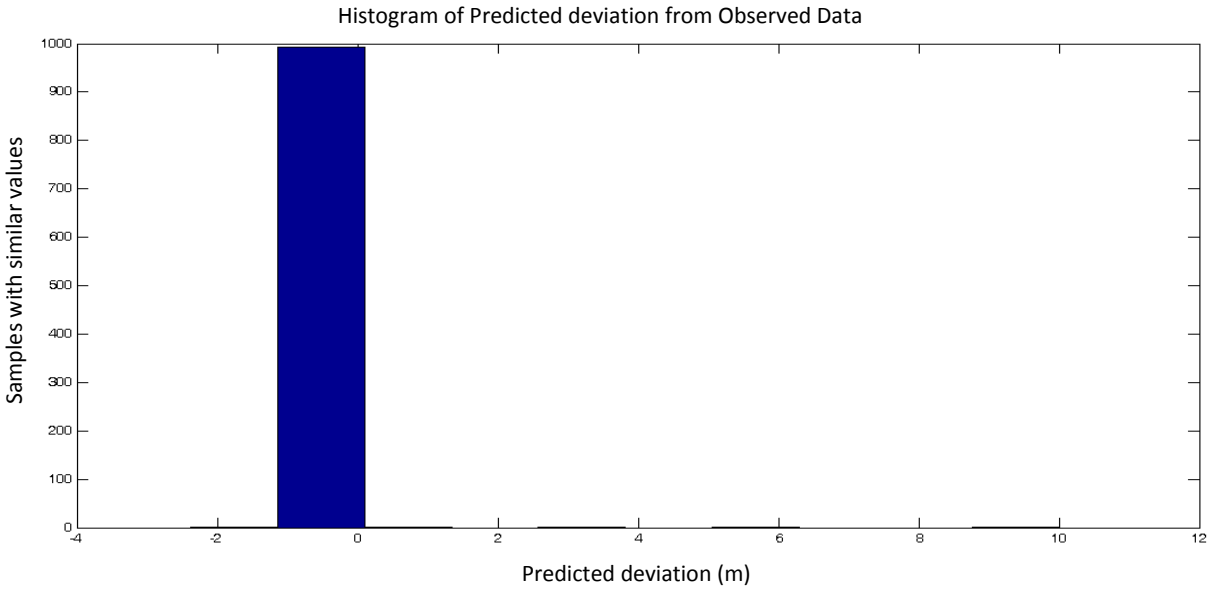


Figure 5.27 Histogram showing the number of results deviating from the observed value.

5.3.2. Layer-Recurrent Neural Network

The predictive power of the layer-recurrent neural network was estimated by running a sample set containing one thousand independently trained neural networks and by observing the predicted value in the final step compared to the actual value. The results of these simulations are shown in Figure 5.29 – Figure 5.33.

Using five neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.29. Only a few outliers are observed as found in the histogram plot, Figure 5.30, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.28. The data has a maximum and minimum of 22.91 m and 11.43 m, respectively. The average value for the data set is 14.50 m with a standard deviation of 2.01. The observed (13.57 m) value is well within the two standard deviation value from the average which indicates that the focused layer-recurrent neural network with five neurons effectively predicts the trend.

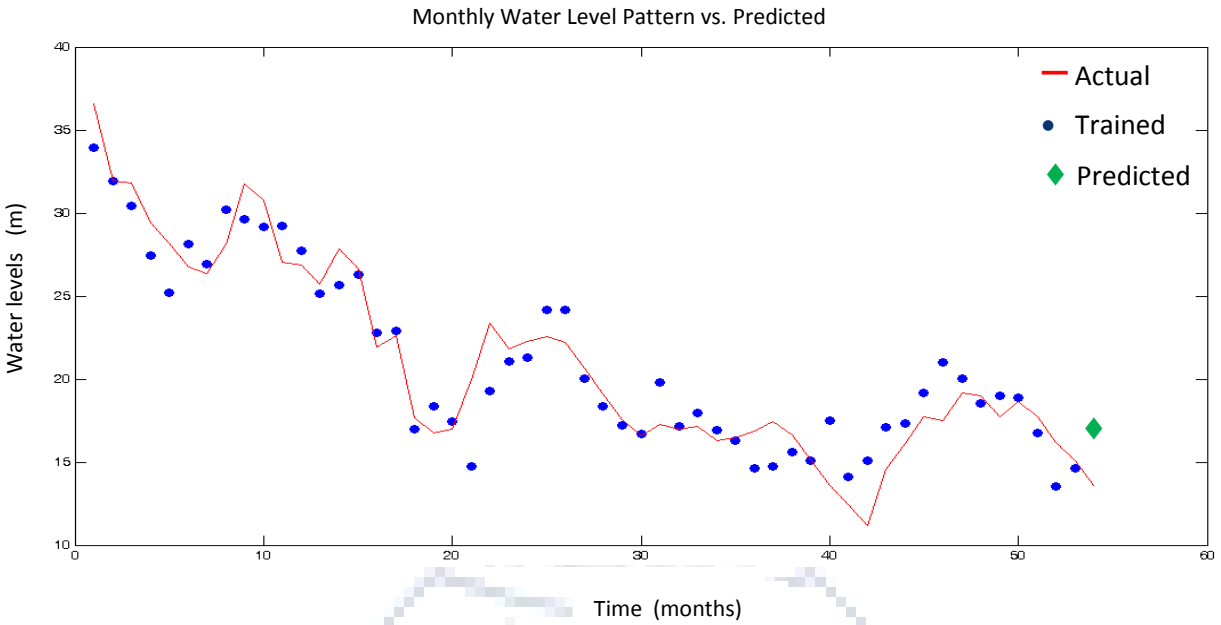


Figure 5.28 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 5 neurons per layer. Data estimation error less than 19 %.

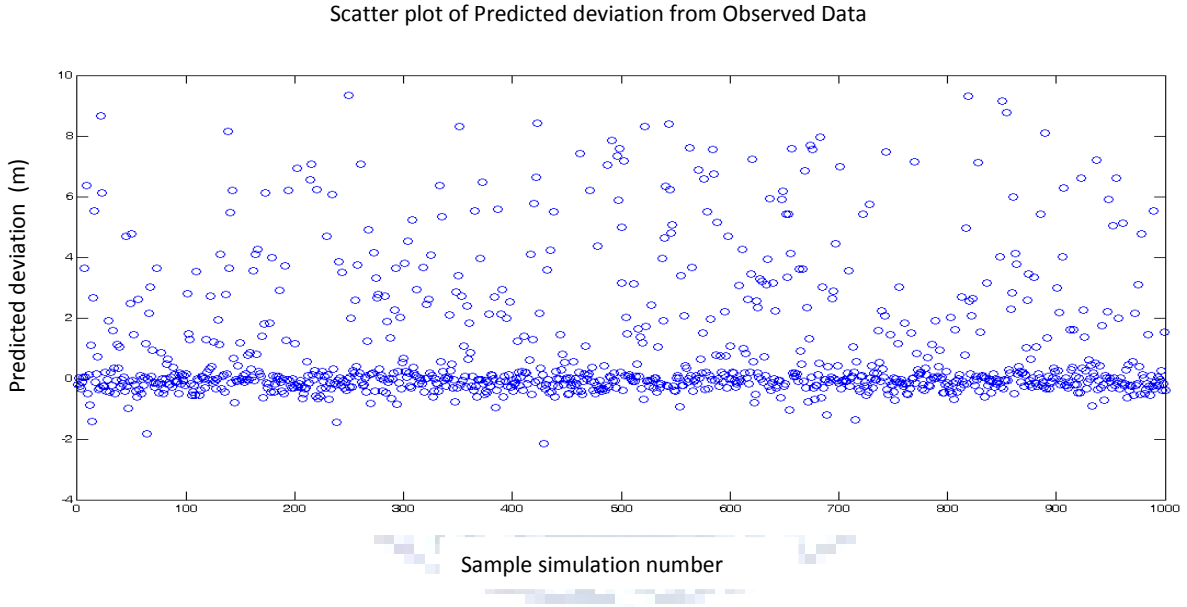


Figure 5.29 Neural network containing 5 neurons. Predicted deviation from observed data point from 1000 sample runs.

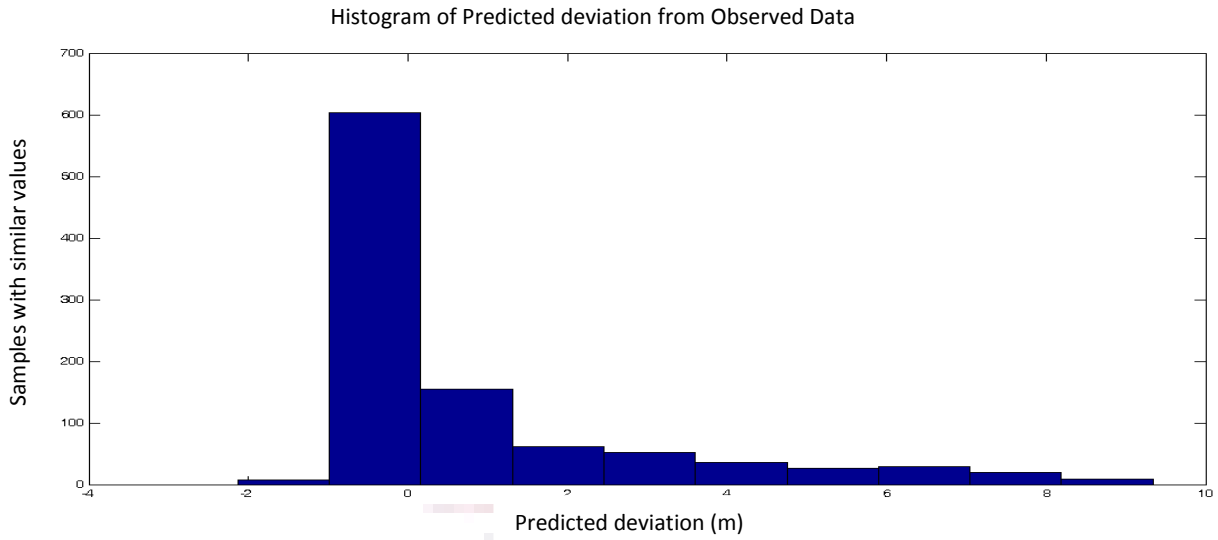


Figure 5.30 Histogram showing the number of results deviating from the observed value.

Using ten neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.32. Only a few outliers are observed as found in the histogram plot, Figure 5.33, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.31. The data has a maximum and minimum of 23.67 m and 9.81 m respectively. The average value for the data set is 14.10 m with a standard deviation of 1.56. The observed (13.57 m) value which is nearly the same. The standard deviation value for the data set is also significantly reduced indicating that the layer-recurrent neural network with ten neurons effectively predicts the trend. Additionally, the spread of solutions are significantly reduced compared to the five neuron network.

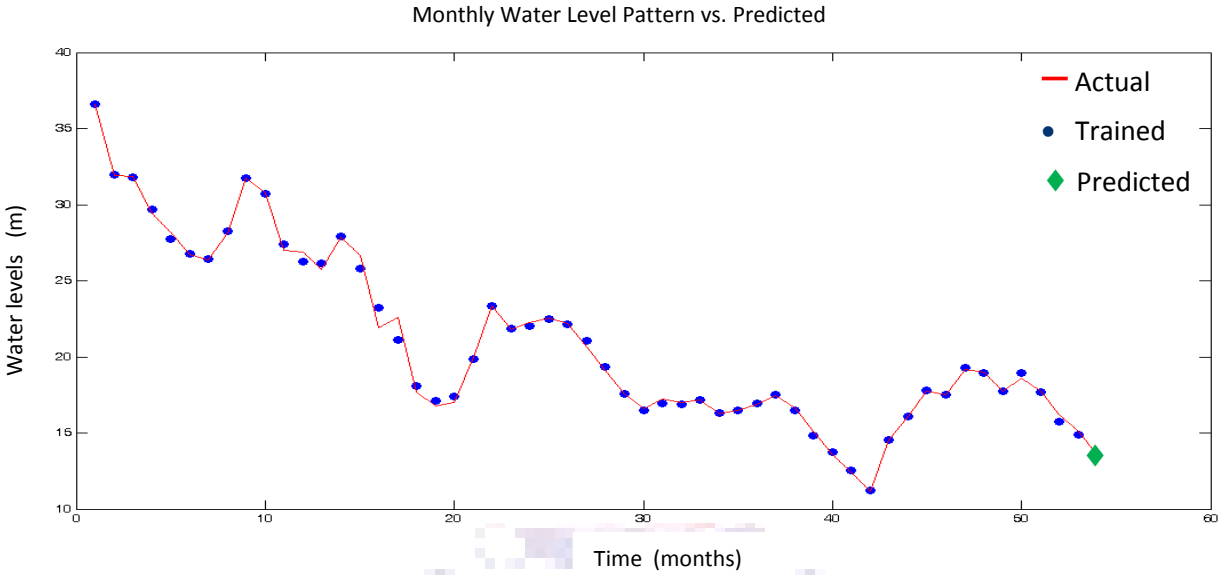


Figure 5.31 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 10 neurons per layer. Data estimation error less than 4 %.

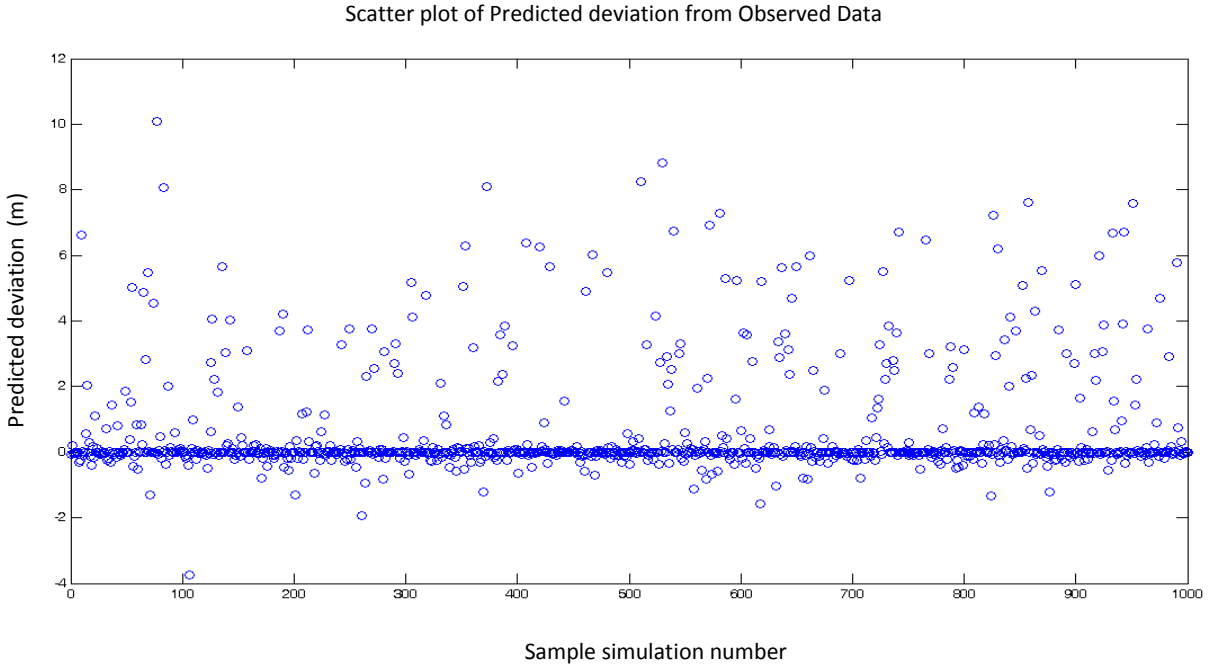


Figure 5.32 Neural network containing 10 neurons. Predicted deviation from observed data point from 1000 sample runs.

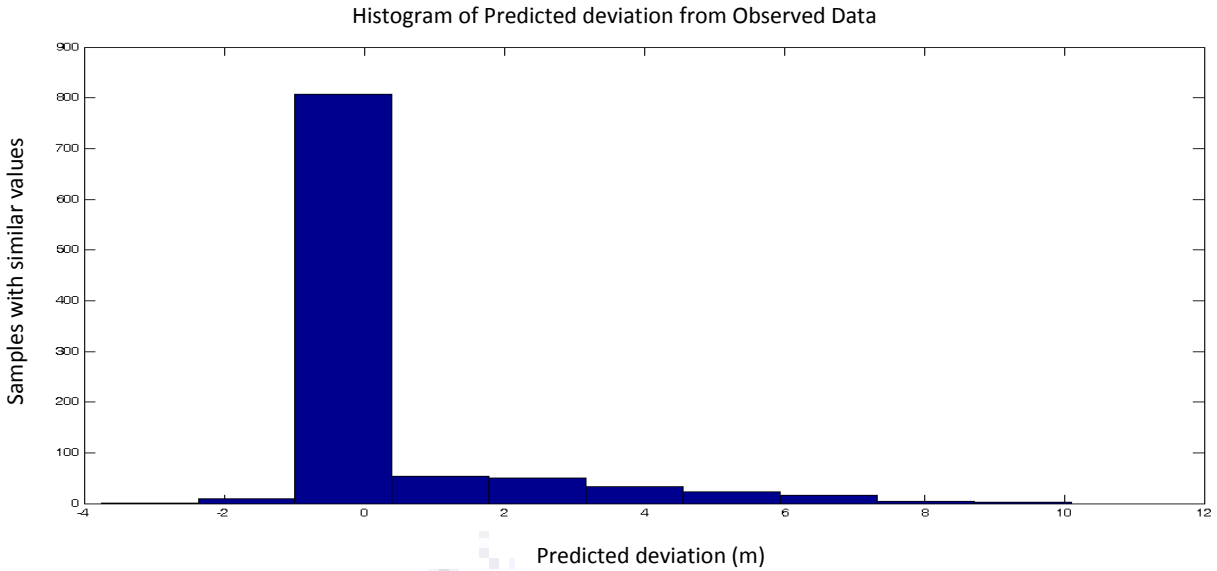


Figure 5.33 Histogram showing the number of results deviating from the observed value.

5.3.3. Radial Basis Neural Network

In order to investigate radial basis networks, the spread of the radial basis function was adjusted from 0.25 to 1.25. This determines the width of an area in the input space to which each neuron will respond. During the training phase fifty nine months of data was used and only the last point in the five year cycle was predicted, i.e., one step-ahead. The results of these simulations are shown in Figure 5.35 – Figure 5.36.

The predictive power of the radial basis neural network was estimated by running a sample set containing one thousand independently trained neural networks and by observing the predicted value in the final step compared to the actual value.

Using an adjustable radial basis function, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.35. Only a few outliers are observed as found in the histogram plot, Figure 5.36, indicating that the neural network did not reliably predict the observed value. An example of one of these neural network simulations are shown in Figure 5.34. The data has a maximum and minimum of 13.66 m and 13.46 m respectively. The average value for the data set is 13.57 m with a standard deviation of 0.016. The

observed (13.57 m) value is exactly estimated by this method, which indicates that the radial basis network has a larger tolerance in respect to the extent of neuron overlap.

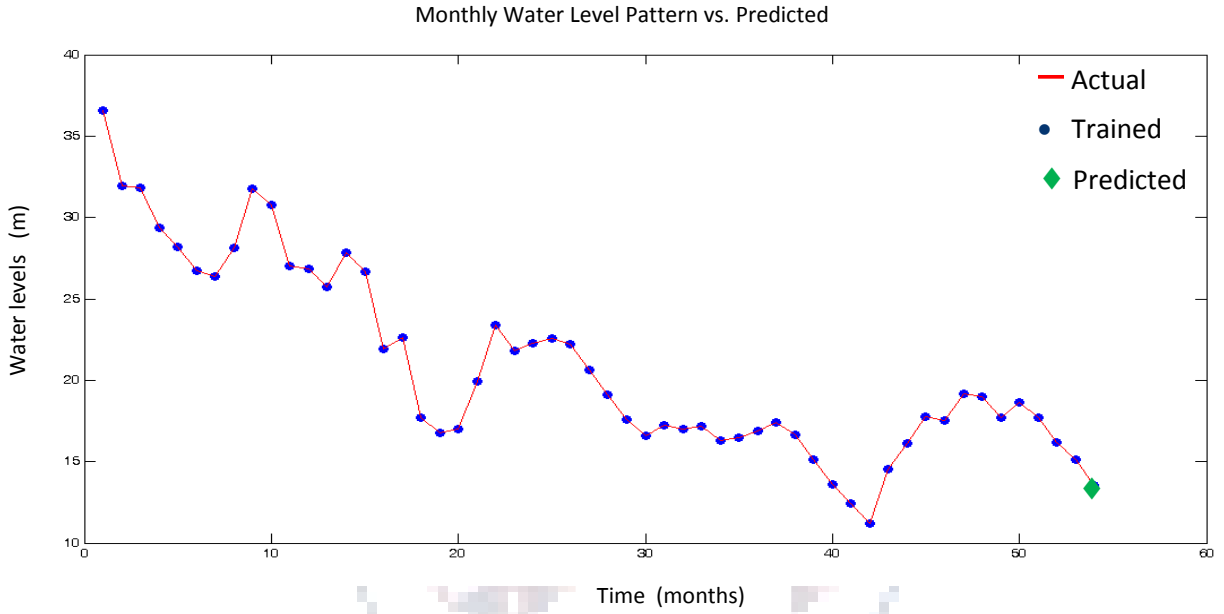


Figure 5.34 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Radial basis neural network with a radial basis function 0.651. Data estimation error less than 0.1 %.

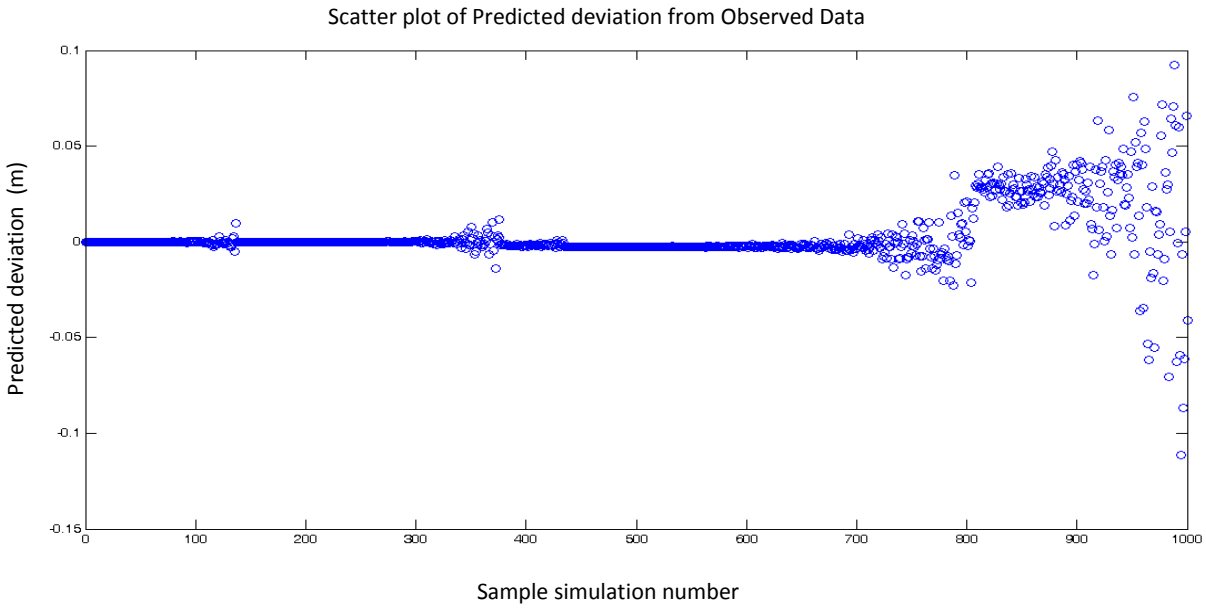


Figure 5.35 Neural network containing radial basis functions adjusted incrementally from 0.25 to 1.25. Predicted deviation from observed data point from 1000 sample runs.

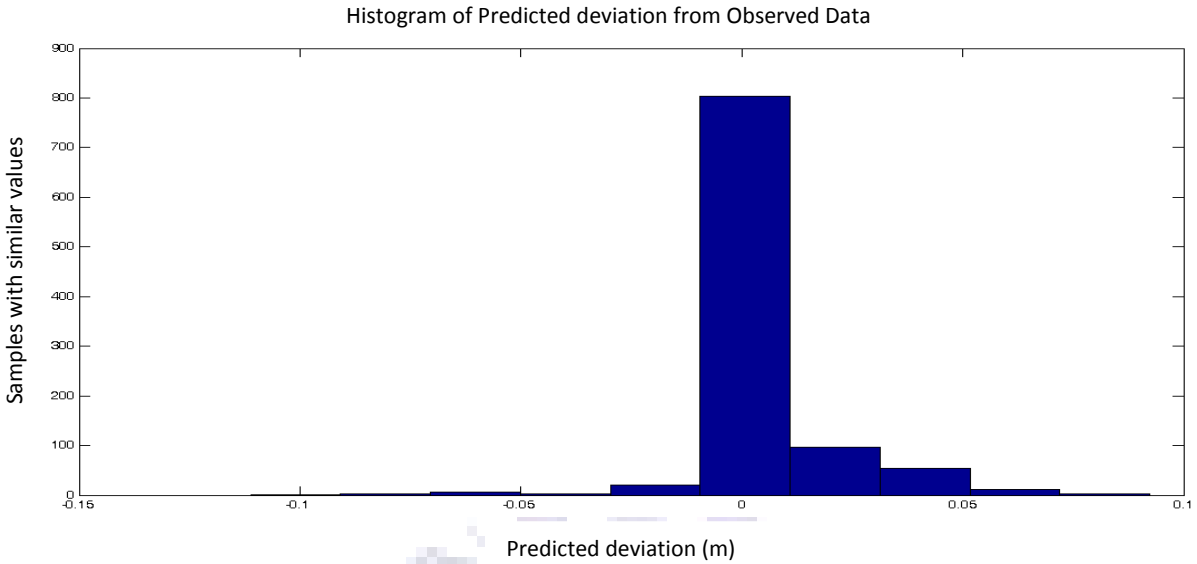


Figure 5.36 Histogram showing the number of results deviating from the observed value.

5.3.4. Probabilistic Neural Network

Probabilistic neural networks can be used for classification problems. When an input is presented, the first layer computes distances from the input vector to the training input vectors and produces a vector whose elements indicate how close the input is to a training input. In these networks only a single output value is presented. The simulation of the data by this network is presented in Figure 5.37. The error in the simulation using the probabilistic network does follow the data with a significant degree of accuracy, resulting in highly accurate fit. Considering only the last step the estimation of the actual value (13.57 m) is 13.57 m which has a consistent underestimation of less than 0.001 m.

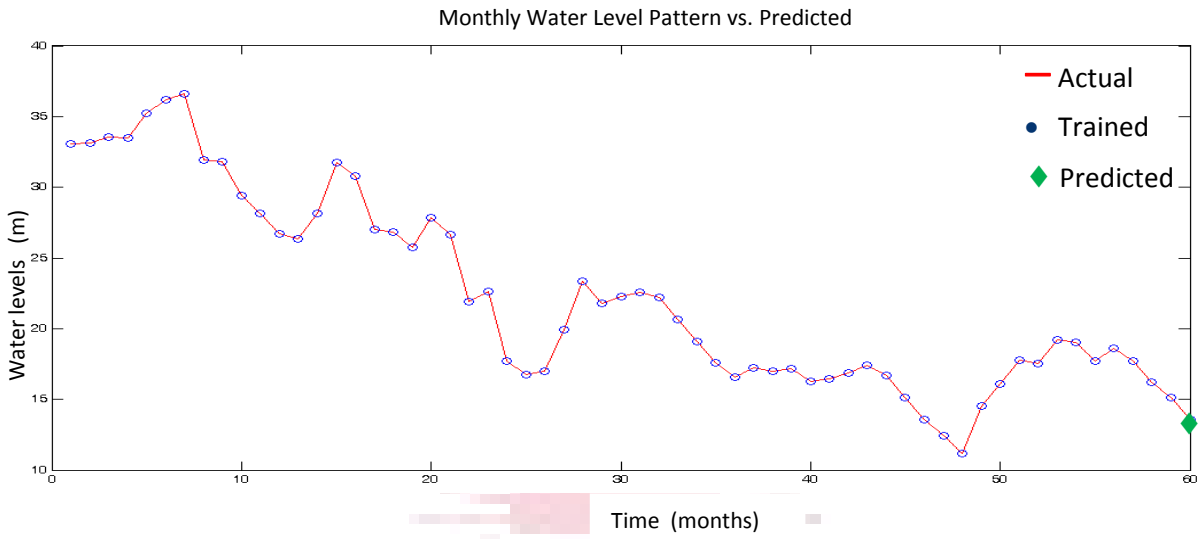


Figure 5.37 Predicted Water levels for the model system using Rainfall and Flow volume in river. Actual data presented as a red line while blue dots indicate estimation values.

5.4. Case Study 3 Vaal River system

In this case study a borehole was identified in proximity to the Vaal River system, which had a constant change in water flow patterns. The borehole coordinates are shown in Figure 5.38.

Figure 5.38 Aerial photo of study area in the Vaal River system. Borehole is ca. 0.3 km from the river edge, which has a weir cage (C2H007) downstream in the Vaal River (green square).

Water levels, flow volumes and rainfall data were collected from the relevant government organisations; this data is represented in Figure 5.39.

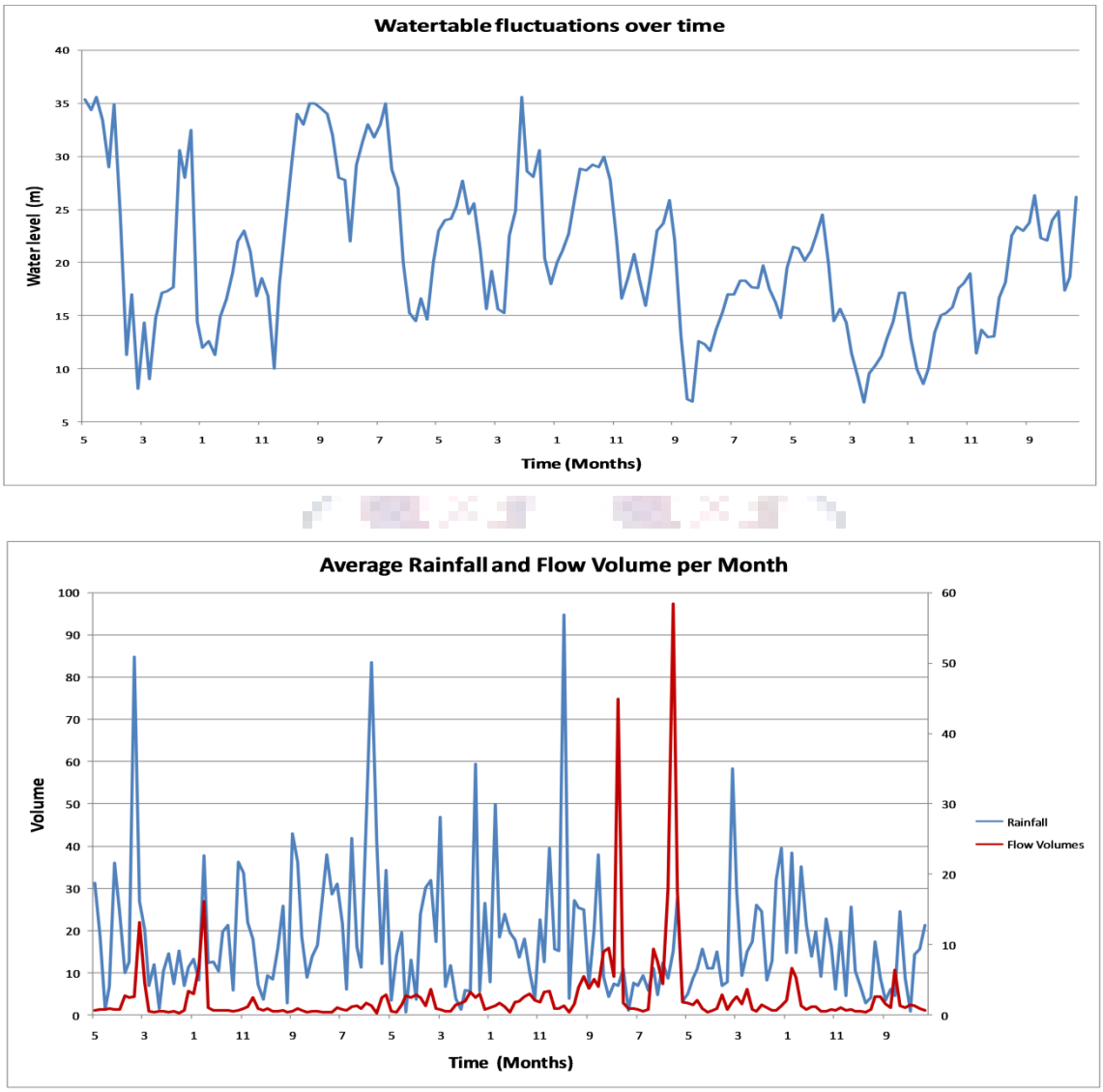


Figure 5.39 Vaal River system data. Top section indicates average water table fluctuations over a 14 year time period in m, while lower section shows average rainfall per month (red) and flow volumes discharged from the river (green) in mm and m^3 per month respectively.

The data obtained was used in all four of the previously identified artificial neural network architectures to determine if the groundwater interaction with the surface water contribution could be simulated. It should be noted that in this case study only fourteen years of data was available. In the ideal case study it was shown that five and ten neurons per layer were the most effective. In the current investigation only these two possibilities will be examined.

5.4.1. Focused Time-Delay Neural Network

The predictive power of the focused time-delay neural network was estimated by running a sample set containing one thousand independently trained neural networks and observing the predicted value in the final step compared to the actual value.

Using five neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.41. Only a few outliers are observed as found in the histogram plot, Figure 5.42, indicating that the neural network reliably predicts the observed value within error. An example of one of these neural network simulations are shown in Figure 5.40. The data has a maximum and minimum of 50.02 m and -5.23 m, respectively. The average value for the data set is 22.29 m with a standard deviation of 5.95. The observed (26.20 m) value is well within the two standard deviation value from the average. However, the standard deviation of the predicted data set is quite large.

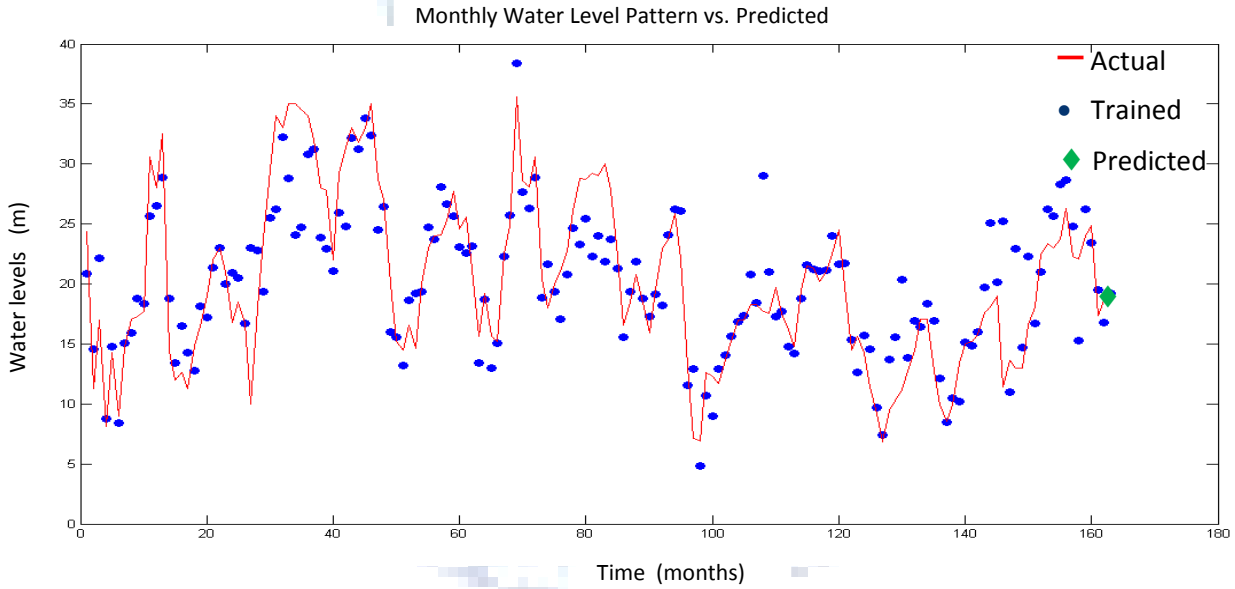


Figure 5.40 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 5 neurons per layer. Data estimation error less than 25 %.

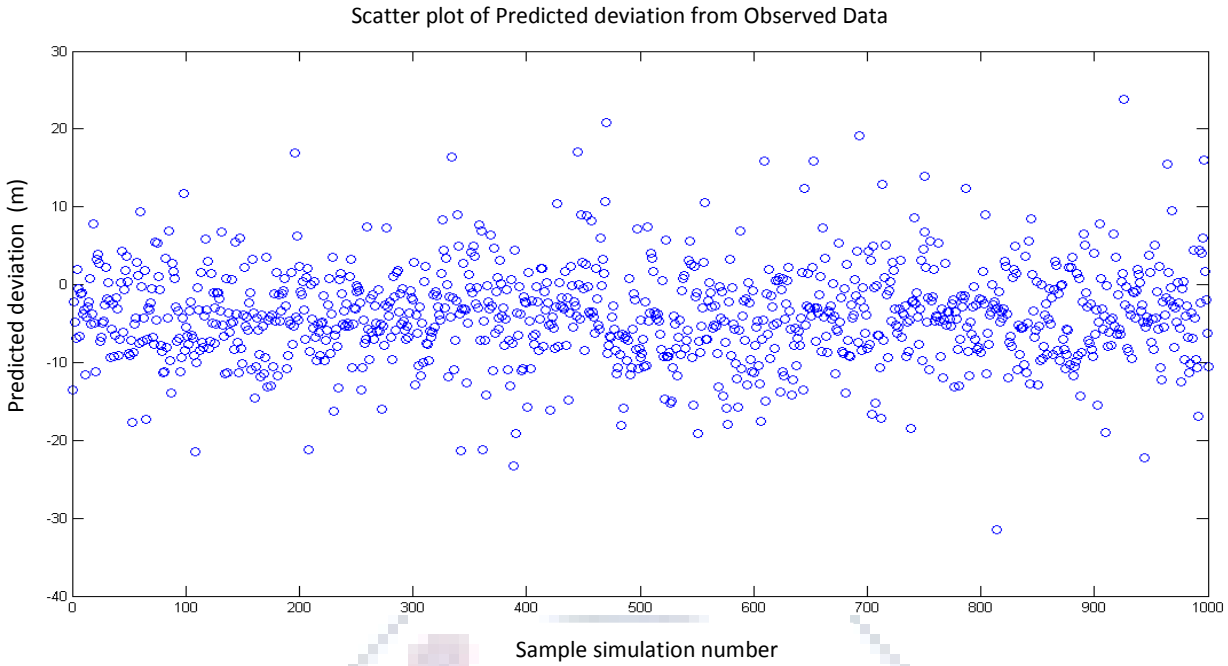


Figure 5.41 Neural network containing 5 neurons. Predicted deviation from observed data point from 1000 sample runs.

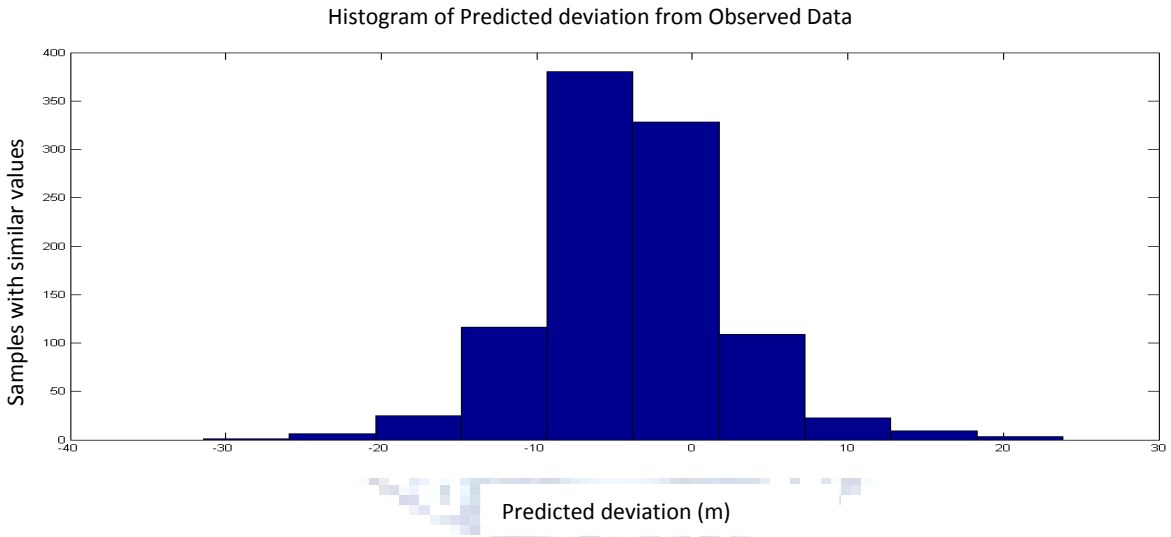


Figure 5.42 Histogram showing the number of results deviating from the observed value.

Using ten neurons per layer, the predicted values for the water level increases in accuracy (Figure 5.44). Only a few outliers are observed as found in the histogram plot, Figure 5.45, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.43. The data has a maximum and minimum of 141.18 m and -44.67 m respectively. The average value for the data set is 20.82 m with a standard deviation of 14.47. The observed (26.20 m) value is well within the two standard deviation value from the

average. However, the standard deviation of the predicted data set is quite large. Additionally, the spread of solutions are significantly increased (50 %) compared to the five neuron network.

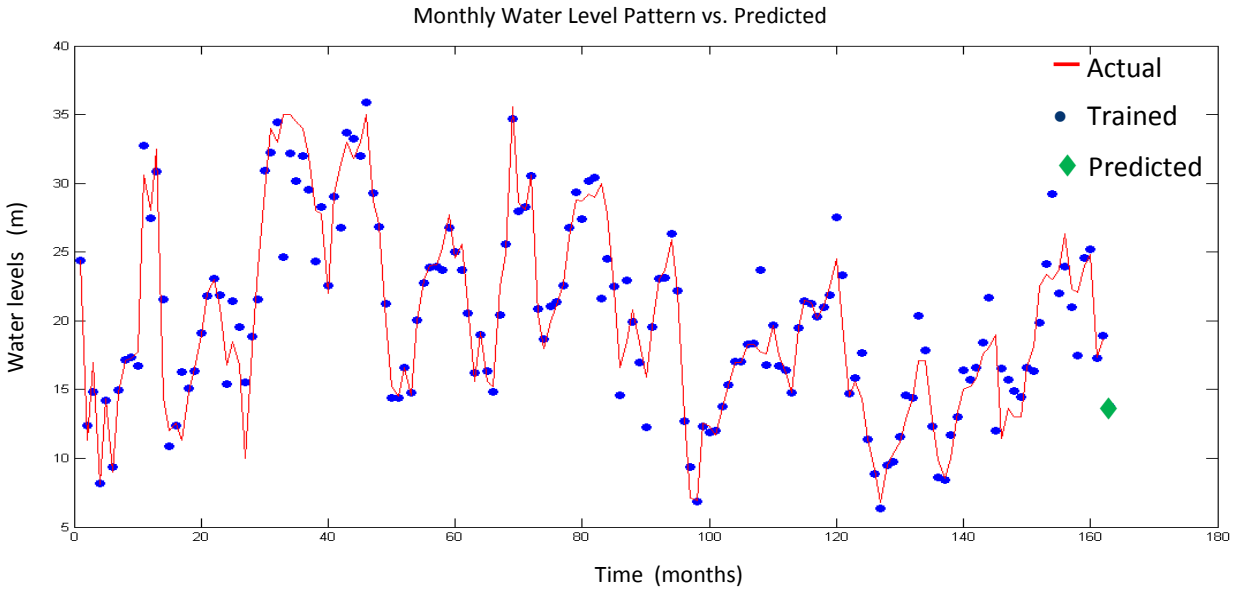


Figure 5.43 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 5 neurons per layer. Data estimation error less than 30 %.

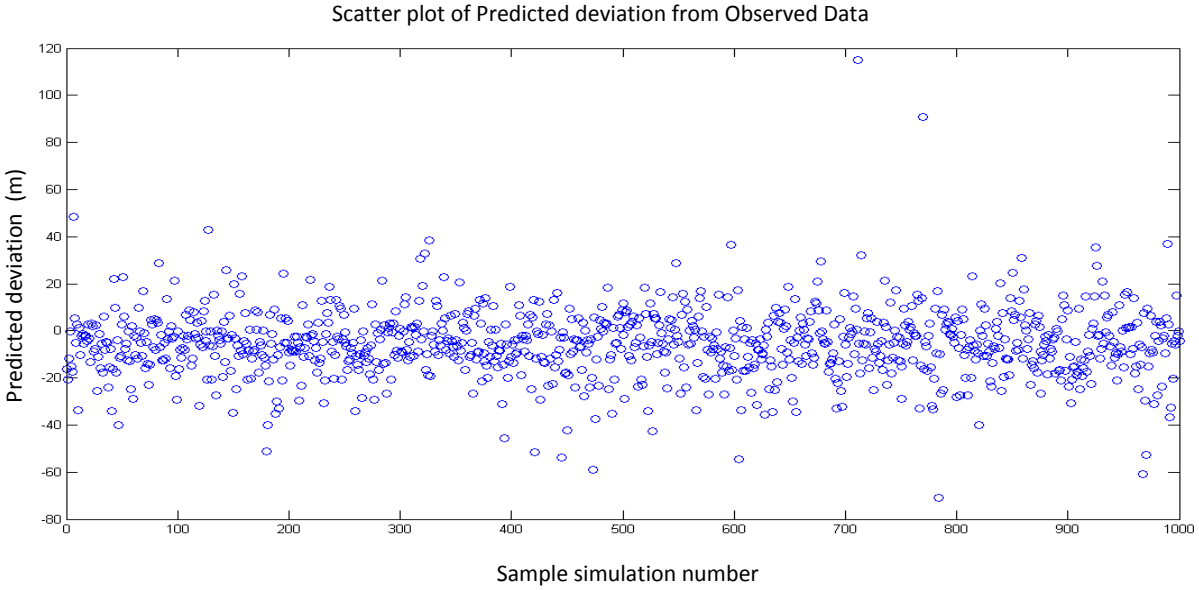


Figure 5.44 Neural network containing 10 neurons. Predicted deviation from observed data point from 1000 sample runs.

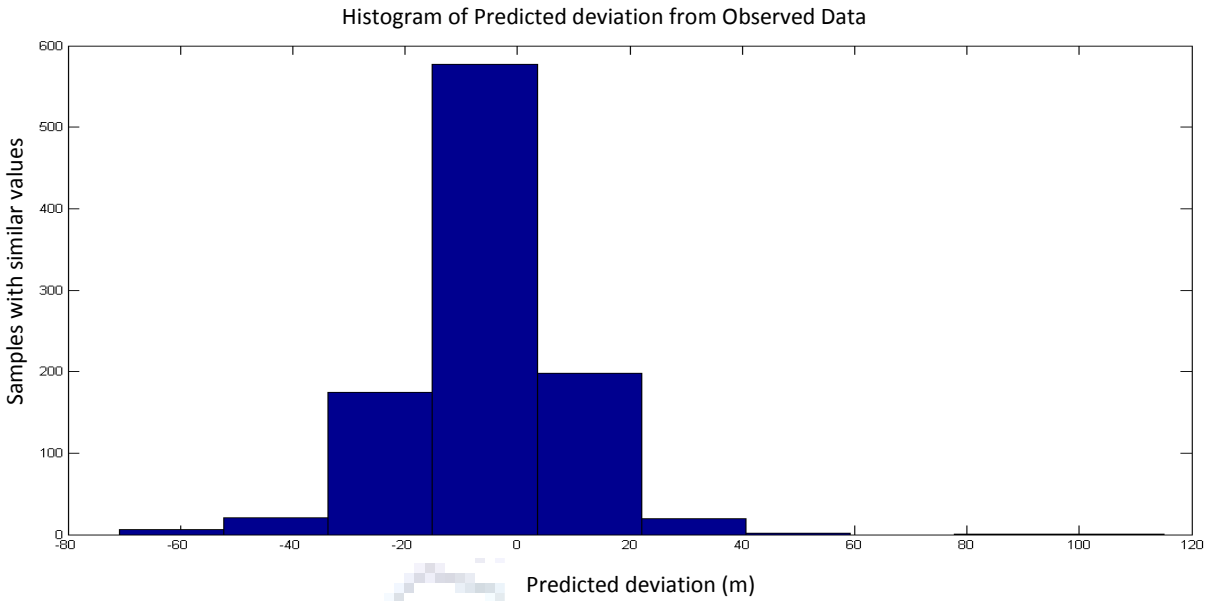


Figure 5.45 Histogram showing the number of results deviating from the observed value.

5.4.2. Layer-Recurrent Neural Network

The predictive power of the layer-recurrent neural network was estimated by running a sample set containing one thousand independently trained neural networks and by observing the predicted value in the final step compared to the actual value. The results of these simulations are shown in Figure 5.47 – Figure 5.51.

Using five neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.47. Only a few outliers are observed as found in the histogram plot, Figure 5.48, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.46. The data has a maximum and minimum of 31.77 m and 11.72 m respectively. The average value for the data set is 22.73 m with a standard deviation of 3.10. The observed (26.20 m) value is well within the two standard deviation value from the average which indicates that the focused layer-recurrent neural network with five neurons effectively predicts the trend.

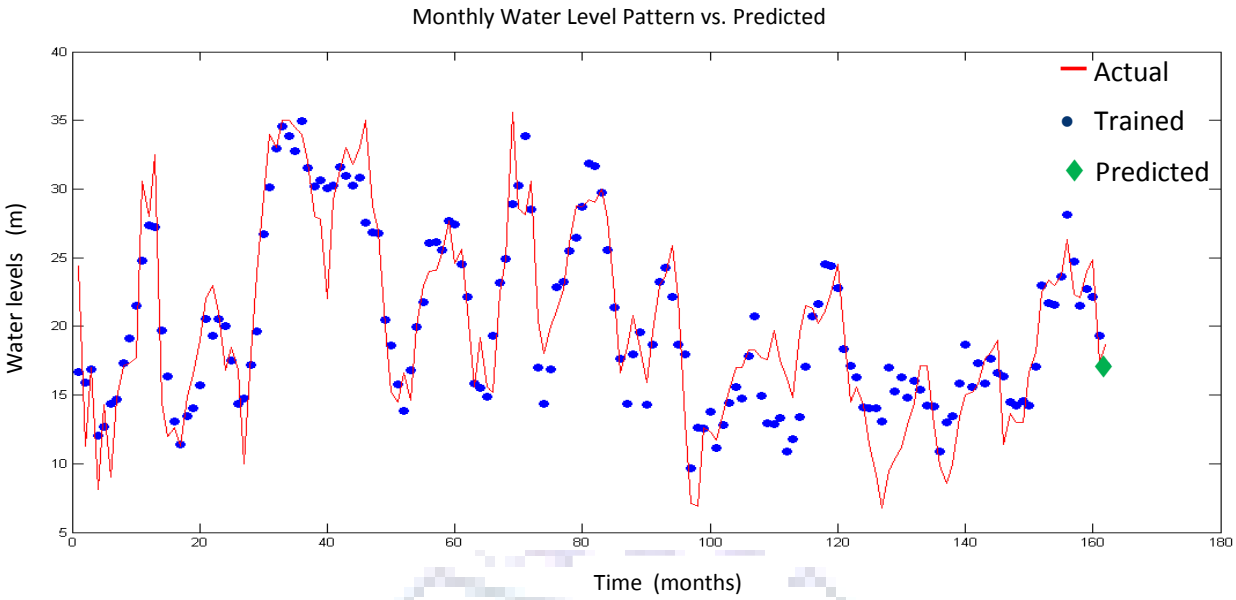


Figure 5.46 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 5 neurons per layer. Data estimation error less than 25 %.

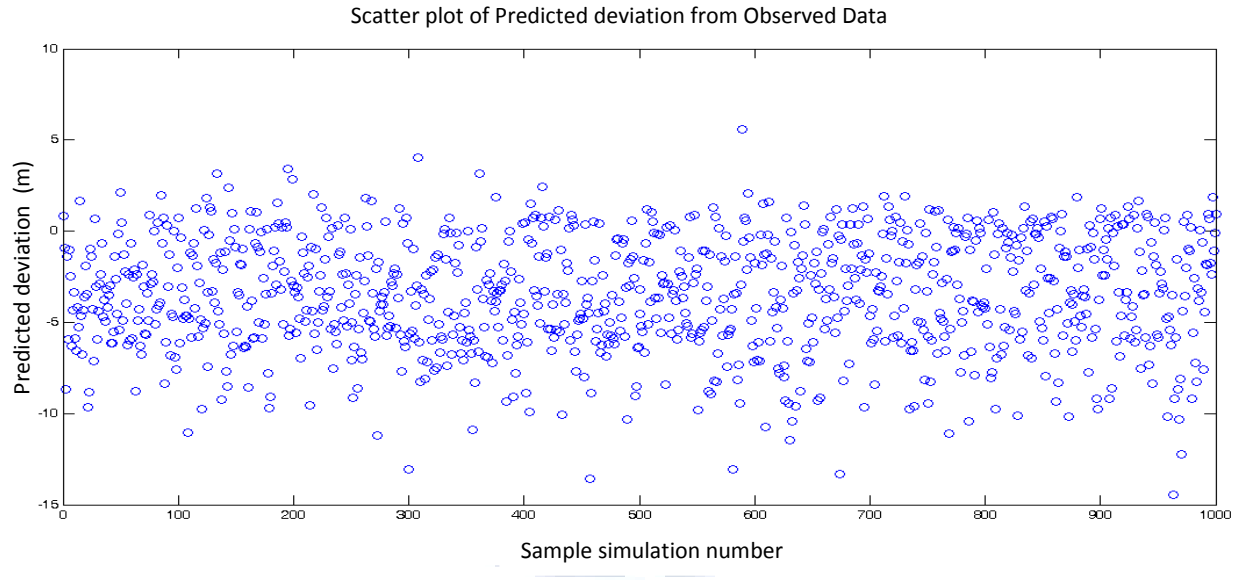


Figure 5.47 Neural network containing 5 neurons. Predicted deviation from observed data point from 1000 sample runs.

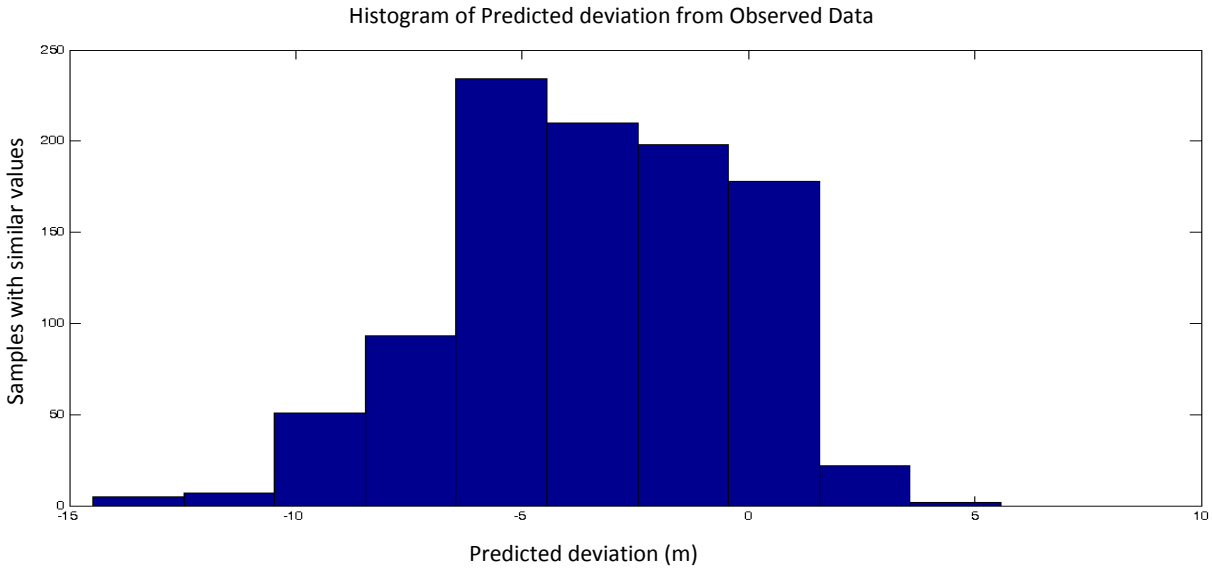


Figure 5.48 Histogram showing the number of results deviating from the observed value.

Using ten neurons per layer, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.50. Only a few outliers are observed as found in the histogram plot, Figure 5.51, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.49. The data has a maximum and minimum of 64.95 m and -11.21 m respectively. The average value for the data set is 21.49 m with a standard deviation of 4.13. The observed (26.20 m) value is within two standard deviation values from the predicted. The standard deviation value for the data set is not significantly reduced indicating that the layer-recurrent neural network with ten neurons does a less effective simulation of the predicted the trend. Additionally, the spread of solutions are significantly increased compared to the five neuron network.

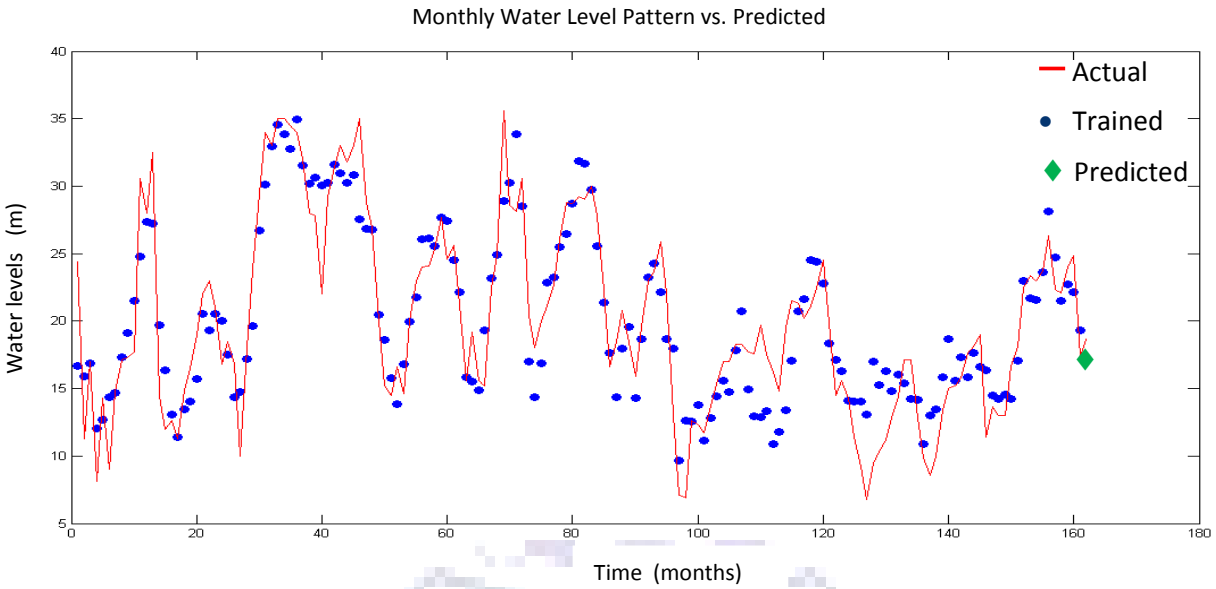


Figure 5.49 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Layer recurrent neural network contained two layers with 10 neurons per layer. Data estimation error less than 15 %.

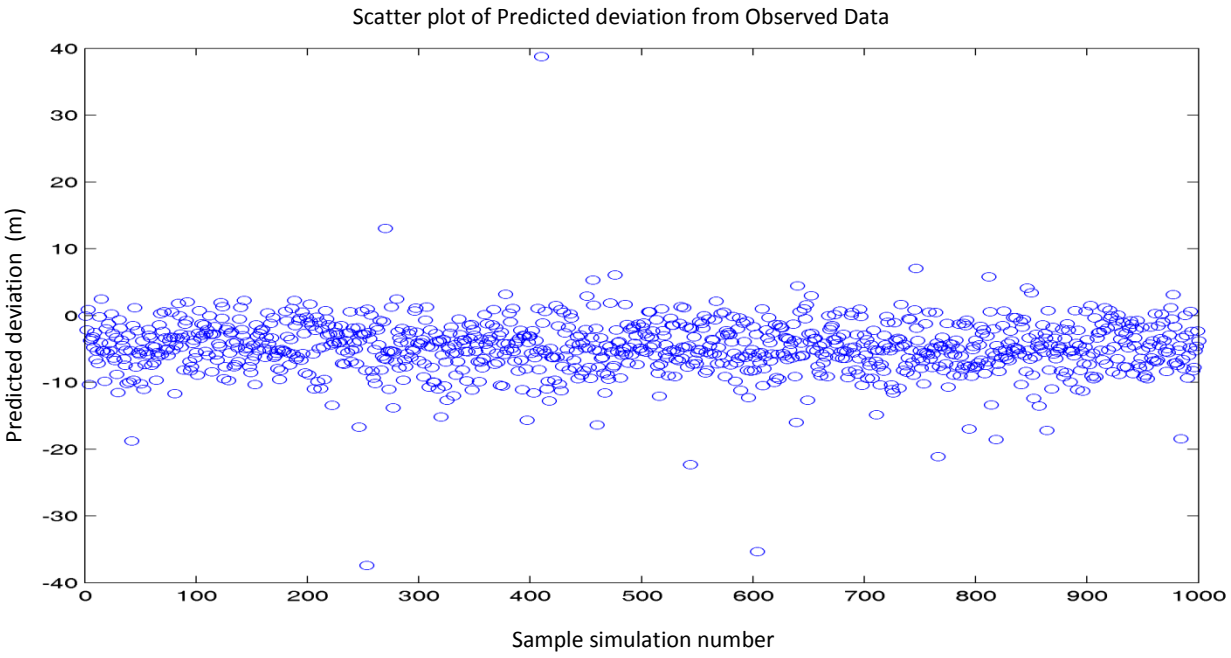


Figure 5.50 Neural network containing 10 neurons. Predicted deviation from observed data point from 1000 sample runs.

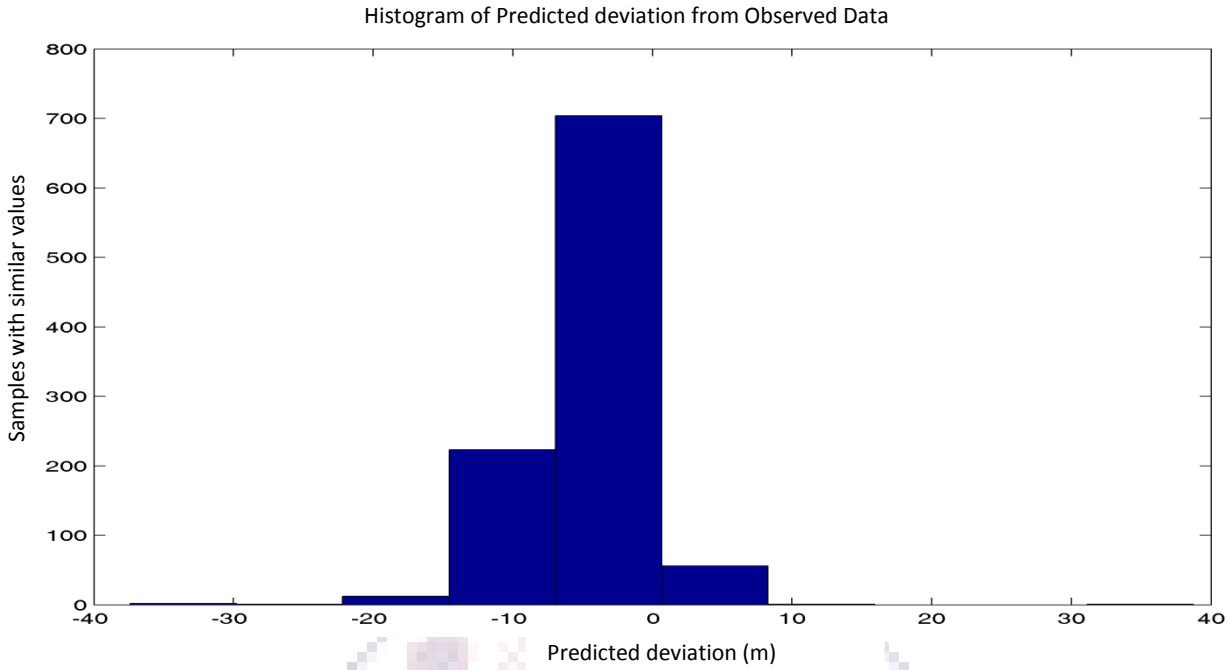


Figure 5.51 Histogram showing the number of results deviating from the observed value.

5.4.3. Radial Basis Neural Network

The predictive power of the radial basis neural network was estimated by running a sample set containing one thousand independently trained neural networks and by observing the predicted value in the final step compared to the actual value. The results of these simulations are shown in Figure 5.53 – Figure 5.54.

Using an adjustable radial basis function, a sample set of one thousand simulations were calculated resulting in an absolute deviation from observed data (A_{obs} - A_{sym}) being plotted in Figure 5.53. Only a few outliers are observed as found in the histogram plot, Figure 5.54, indicating that the neural network reliably predicts the observed value. An example of one of these neural network simulations are shown in Figure 5.52. The data has a maximum and minimum of 30.00 m and 18.49 m respectively. The average value for the data set is 23.79 m with a standard deviation of 3.07. The observed (26.20 m) value is within the two standard deviation values from the average which indicates that the radial basis neural network with adjusted radial basis function does effectively predict the trend.

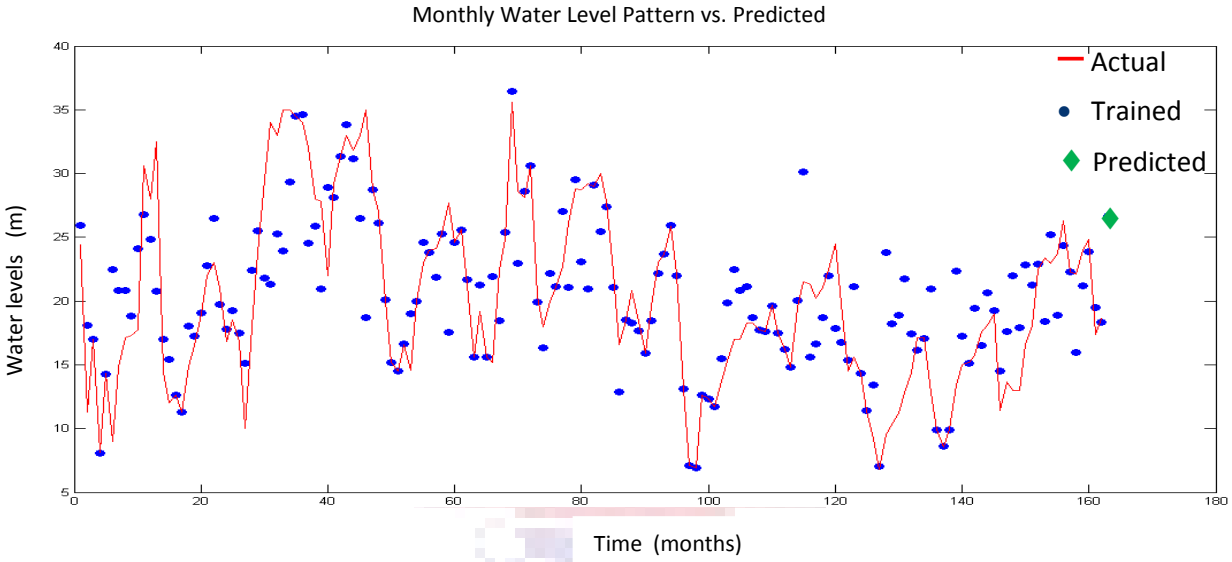


Figure 5.52 Predicted Water levels for the model system using Rainfall and Flow volumes in the river. Radial basis neural network with a radial basis function 0.651. Data estimation error less than 45 %.

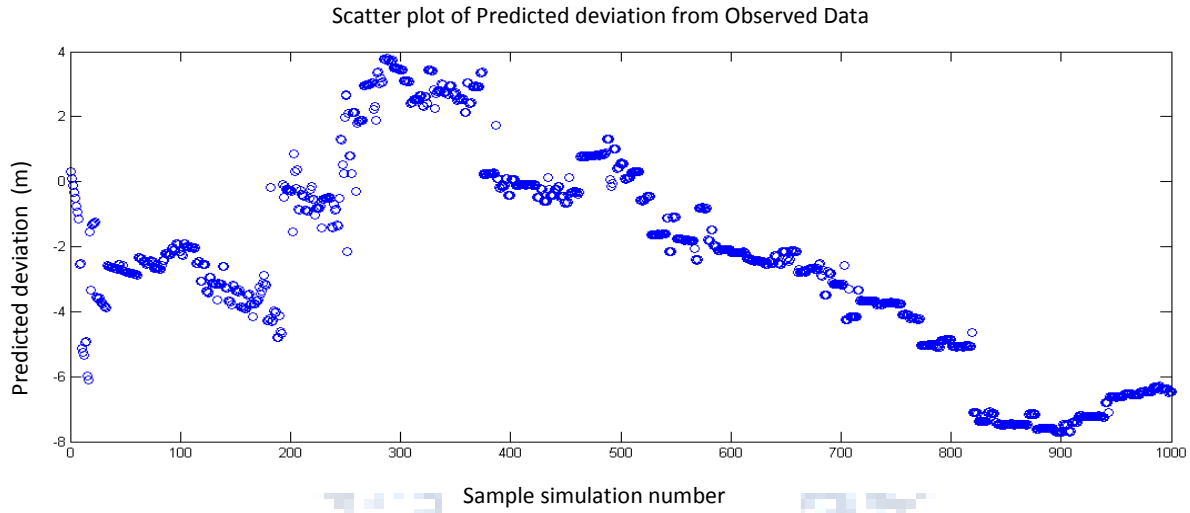


Figure 5.53 Neural network containing radial basis functions adjusted incrementally from 0.25 to 1.25. Predicted deviation from observed data point from 1000 sample runs.

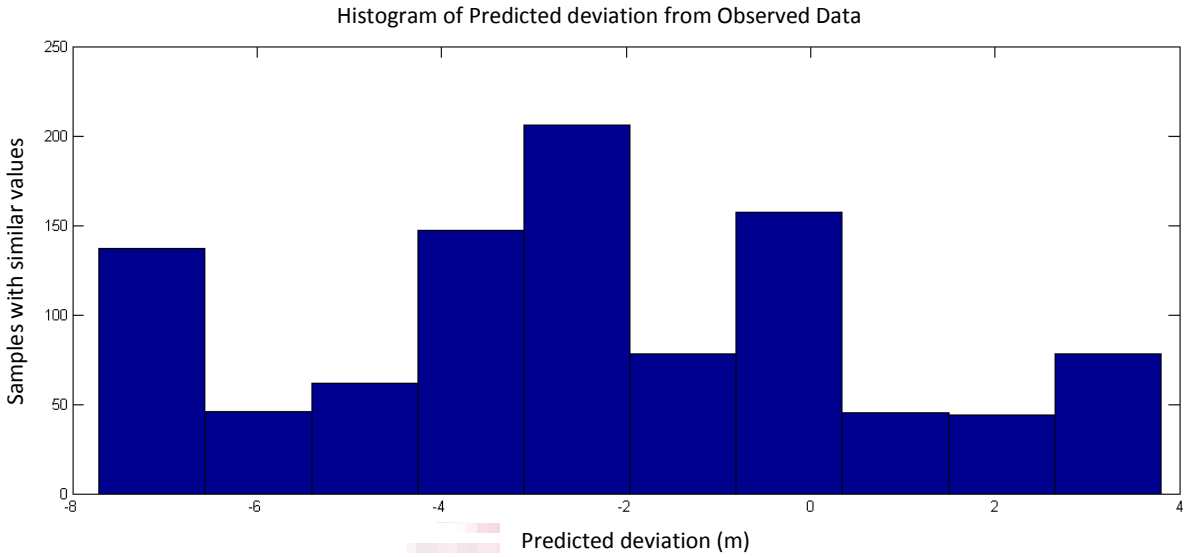


Figure 5.54 Histogram showing the number of results deviating from the observed value.

5.4.4. Probabilistic Neural Network

Probabilistic neural networks can be used for classification problems. When an input is presented, the first layer computes distances from the input vector to the training input vectors and produces a vector whose elements indicate how close the input is to a training input. In these networks only a single output value is presented. The simulation of the data by this network is presented in Figure 5.55. The error in the simulation using the probabilistic network does not follow the data with a significant degree of accuracy, resulting in highly variable fit. Considering only the last step the estimation of the actual value (26.20 m) is 26.20 m which has a consistent estimation error of 0.00 m.

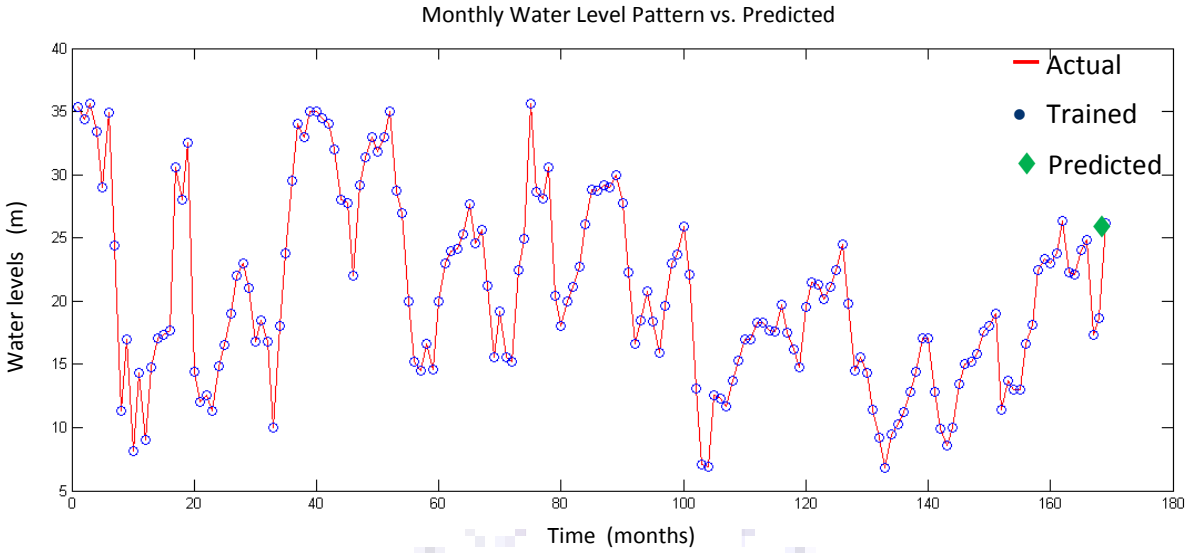


Figure 5.55 Predicted Water levels for the model system using Rainfall and Flow volume in river. Actual data presented as a red line while blue dots indicate estimation values.

5.5. Discussion

During the course of this section three main points of interest will be focussed on. Firstly, the general applicability of neural networks to model surface water and groundwater interactions. Secondly, the ability of predicting more than one step ahead. Finally, using only one of the input variables (rainfall or flow volumes) to train artificial neural networks to predict water levels in a borehole. To estimate the applicability of neural networks to model surface water and groundwater interactions, more than one neural network architecture was used in data simulation. These results are tabulated in Table 5.1, with both five and ten neurons per layer being represented, for the three distinct scenarios.

The performance of the Ideal Model data was expected to be above average since a clear correlation (and empirical) existed between the rainfall and flow volumes compared to the water level in the borehole. In general all four the neural networks performed well, with very small errors in average estimated water level values (Table 5.1). The range of solutions depended heavily on the number of neurons, with the five neurons per layer winning out over the ten neuron per layer architecture, in both the Time-Delay and Layer-Recurrent Neural Networks. One of the most significant causes of this effect is the loss of generality in the neural network and over-fitting of the data present. Both of

the statistical networks failed to accurately estimate the final value for the Ideal Model. In this instance the variability and macro properties of the neural networks hampered the training of the optimal set, causing either an over estimation or under estimation to occur (Figure 5.17 and Figure 5.19).

Table 5.1 Compilation of predicted values for the respective methods and number of neurons.

Scenario / Method		Neurons							
		5				10			
	Real	Min	Max	Ave.	Std. Dev.	Min	Max	Ave.	Std. Dev.
Ideal Model									
Time-delay	16.22	3.00	18.84	15.98	0.58	8.88	19.59	15.71	0.83
Recurrent	16.22	15.89	17.07	16.22	0.06	14.76	16.84	16.22	0.07
Radial Basis	16.22	16.37	16.38	16.38	0.001	-	-	-	-
Probabilistic	16.22	15.81	15.81	15.81	-	-	-	-	-
Dwars River System									
Time-delay	13.57	10.94	24.68	14.37	2.05	12.99	17.20	13.57	0.14
Recurrent	13.57	11.43	22.91	14.50	2.01	9.81	23.67	14.10	1.56
Radial Basis	13.57	13.46	13.66	13.57	0.016	-	-	-	-
Probabilistic	13.57	13.57	13.57	13.57	-	-	-	-	-
Vaal River System									
Time-delay	26.2	-5.23	50.02	22.29	5.95	-44.67	141.18	20.82	14.47
Recurrent	26.2	11.72	31.77	22.73	3.10	-11.21	64.95	21.49	4.13
Radial Basis	26.2	18.49	30	23.79	3.07	-	-	-	-
Probabilistic	26.2	26.2	26.2	26.2	-	-	-	-	-

Moving to real world data the Dwars River system was simulated. The Time-Delay and Layer-Recurrent Neural Networks used and performed well with the ten neuron per layer setup having a more accurate predictive capability. However, the increase in accuracy is only slight compared to the total simulation result variance. The Radial Basis neural network was stable and gave good quality results, which would make this network suitable for general simulations. In contrast the Probabilistic network gave nearly the exact value of the observed water level.

The Vaal River system was used to determine if only small correlation between surface water and groundwater could be observed. The Time-Delay and Layer-Recurrent Neural Networks performed well with the five neuron per layer setup having a more accurate predictive capability. However, the

increase in accuracy is ca. three times that observed in the Time-Delay neural network. The Layer-Recurrent neural network indicates a preference for ten neurons per layer. The Radial Basis neural network gave a reasonable approximation of the observed value. The Probabilistic network gave a flawless estimation of the final value.

Considering all the variables, the Time-Delay neural network with a six month delay gave the most acceptable values for all three the scenarios. The Layer-Recurrent network in general resulted in the most accurate predictions, however these networks typically require hours to train and run. In the instance of Time-Delay neural networks results can be obtained in less than a minute. Both the Probabilistic and Radial Basis neural networks indicated flaws in estimating the correct value.

Assuming that the Time-Delay neural networks were general and fast enough to predict one step ahead values, a set of one hundred training and simulation sets were used to estimate forward predictions of differing time periods. A systematic increase of one to six months a-head prediction was done and is reported in Table 5.2.

Table 5.2 Layer recurrent (5 neurons, 6 months) prediction for the final value (26.2 m) in Vaal River system.

Period	1	2	3	4	5	6
Min	-6.45	-5.93	1.51	-10.67	-13.53	-1.09
Max	34.99	46.80	39.59	48.18	32.03	50.39
Average	18.59	19.02	19.29	17.76	17.53	19.76
Std. Dev.	7.59	6.96	6.99	8.49	7.12	7.79

It was expected that the neural network would perform worse as the prediction time frame was extended. However, since the five neuron network has a significant generalization component; it fares reasonably well in predicting future values. Considering that during each step presented in Table 5.2, one hundred independently trained networks were constructed. A single network simulation result is shown in Figure 5.56 and Figure 5.57. A general trend can be observed in the predicted and actual data (Figure 5.56). However, on closer inspection of the trend of the final data points a counter direction is observed in the values (Figure 5.57). This observation is largely due to the general predictive capabilities of the neural network which follows the general pattern of the Rainfall and Flow Volume data. This clearly shows the success and failure of neural networks which can only predict a general trend in data.

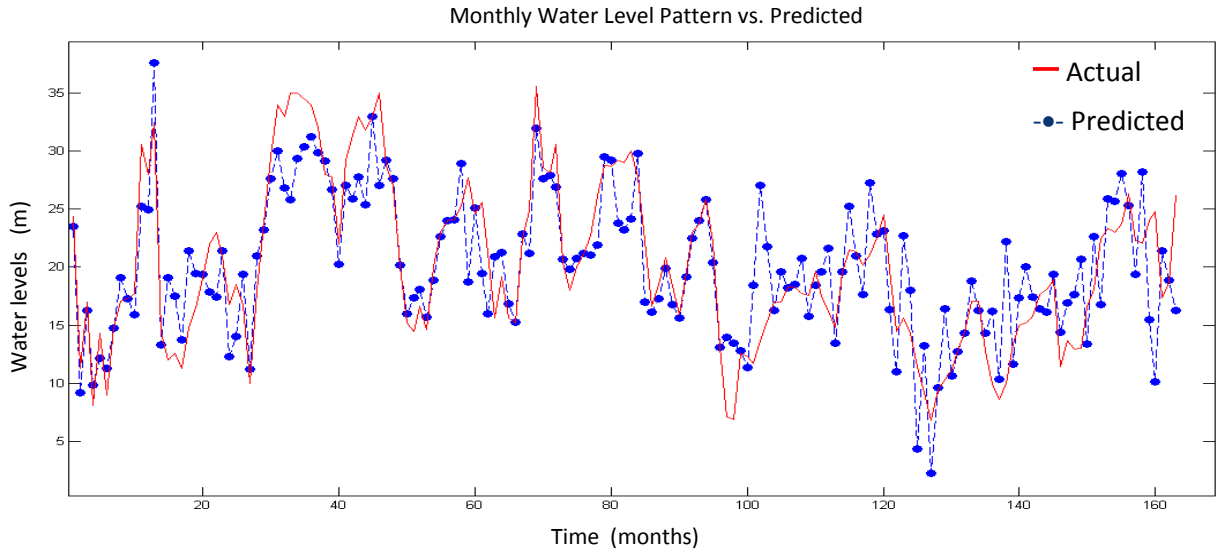


Figure 5.56 Neural network simulation of Vaal River system borehole data. Red line indicates actual observed, while blue dotted line indicates fitted data points.

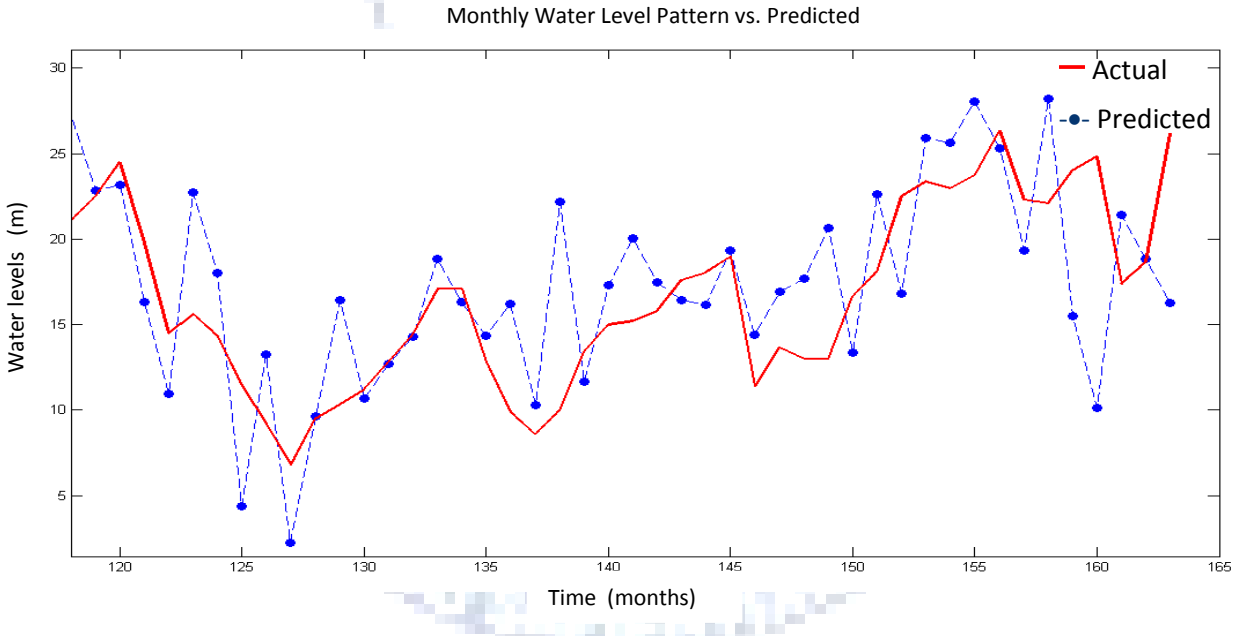


Figure 5.57 Close-up of the last fifty months of the neural network simulation of Vaal River system borehole data (Figure 5.56). Red line indicates actual observed, while blue dotted line indicates fitted data points.

A further consideration in predicting future values was the use of statistical data to create possible inputs for the artificial neural network, since layer recurrent neural networks had shown a high accuracy, this type of network was chosen. The Dwars River data set was used, due to the high correlation between water levels, rainfall and flow volumes (Figure 5.21) as observed from the

trainability and predictions of the four different neural networks. A systematic method was used, in which the input data was reduced from sixty months to forty months and the remaining twenty months data was generated using stochastic assumptions. The artificial neural network was trained on only the initial forty months data and then used in conjunction with the stochastic data to predict the remaining twenty months of data. It should be noted that three data sets were generated in the case of the stochastic method. An average data set (A_i) of each of the previous monthly data points (i.e. Jan 1999, Jan 2000, Jan 2001 etc.) was generated, from which a standard deviation (σ_i) was calculated. Using the standard deviation (σ_i) a maximum ($Max_i = A_i + 2\sigma_i$) and minimum ($Min_i = A_i - 2\sigma_i$) data set could be created to give a range of possible outcomes for future water levels. These results are plotted in Figure 5.58 – Figure 5.60, to illustrate the estimated variance in data compared to the actual data set.

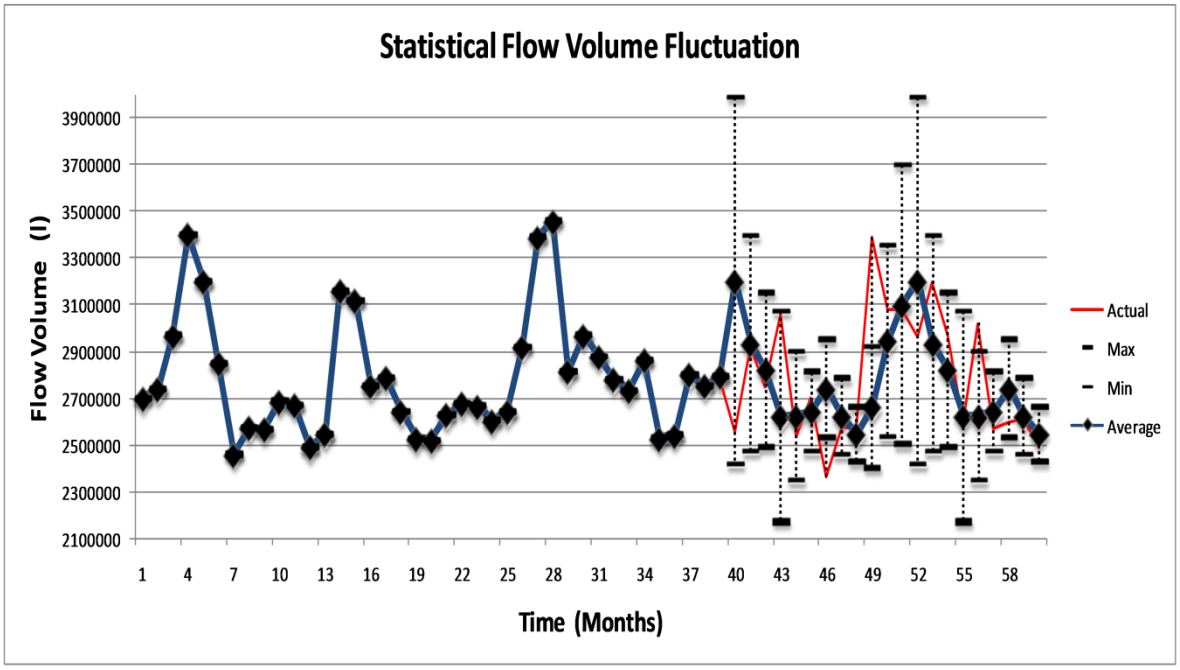


Figure 5.58 Flow volume graph with the first 40 months being the actual data and subsequently averaged data from previous unit month (blue line with black diamond's). Actual data after 40 months represented with a red line.

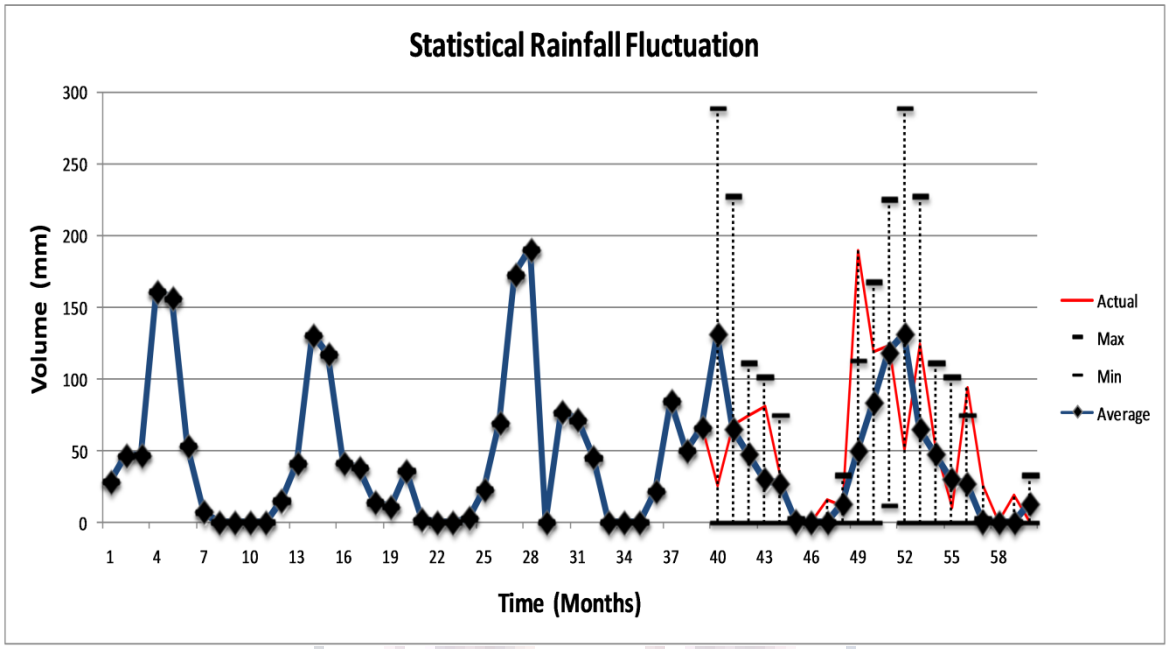


Figure 5.59 Rainfall graph with the first 40 months being the actual data and subsequently averaged data from previous unit month (blue line with black diamond's). Actual data after 40 months represented with a red line.

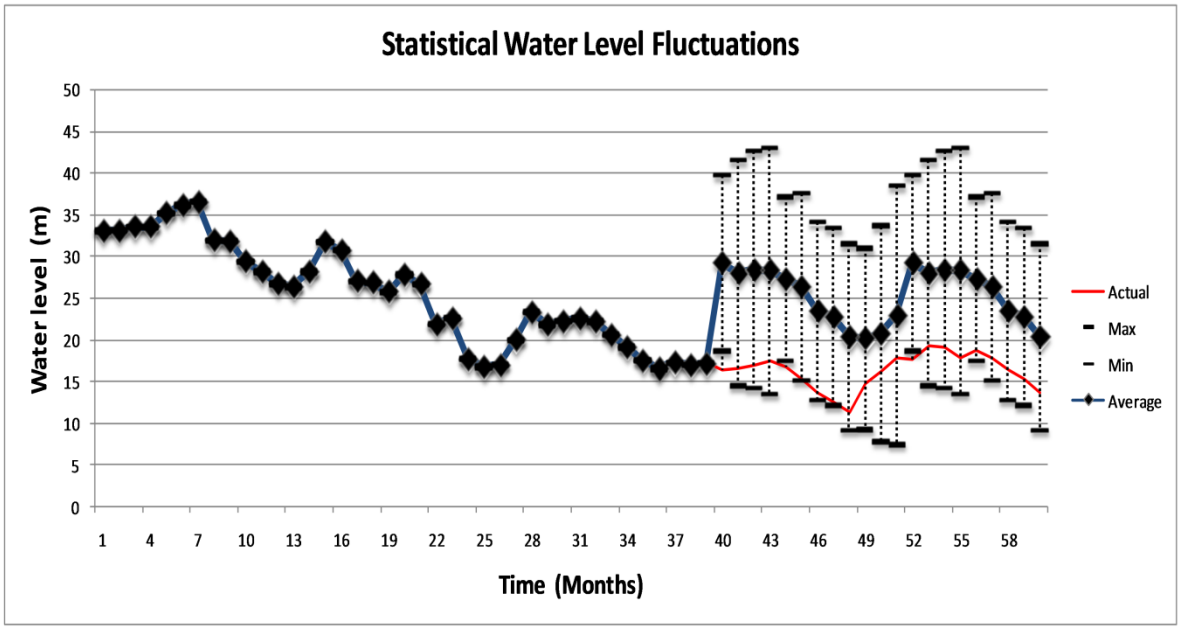


Figure 5.60 Water level graph with the first 40 months being the actual data and subsequently averaged data from previous unit month (blue line with black diamond's). Actual data after 40 months represented with a red line.

The predicted average values for both rainfall and flow volumes were used to train the neural network. The results of using the minimum, maximum and average data sets to predict the following twenty months of water levels are shown in Figure 5.61. In order to get a reflection of the average contribution from the neural network results, the last twenty months of data was averaged and plotted in Figure 5.62

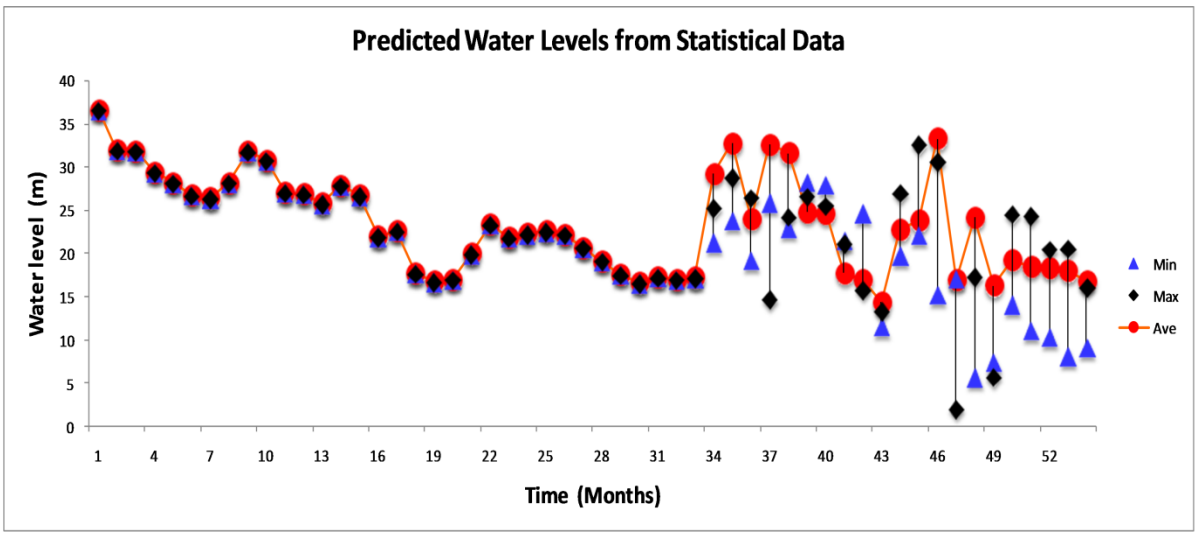


Figure 5.61 Water level graph with the first 40 months being the training data and the following 20 months the predicted value using the minimum, maximum and averaged data sets.

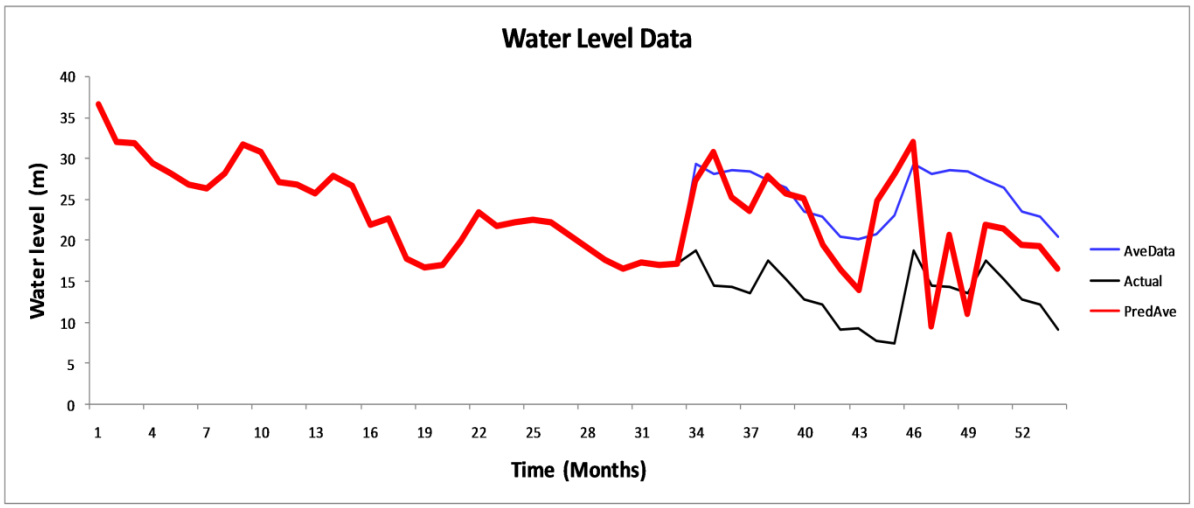


Figure 5.62 Water level graph with the first 40 months being the training data. The red line indicated an averaged value of the results of the neural network simulation of the minimum, maximum and averaged data sets. Black line represents the actual observed data while the blue line shows the three point moving average values.

The averaged predicted values (Figure 5.62) from the neural network simulation data show a similar trend in the movement of the water levels as the actual data. In contrast the normal averaged data does not contain this movement pattern. Although the predicted average data does not simulate the observed data very well, the simulation results are significantly closer than the averaged data values. Using the actual rainfall and flow volume data in the trained neural network results in only a ten month period in which the predicted values are in good correlation with the predicted data set, see Figure 5.63. However, subsequently a similar behaviour or trend in the predicted and actual data can be observed.

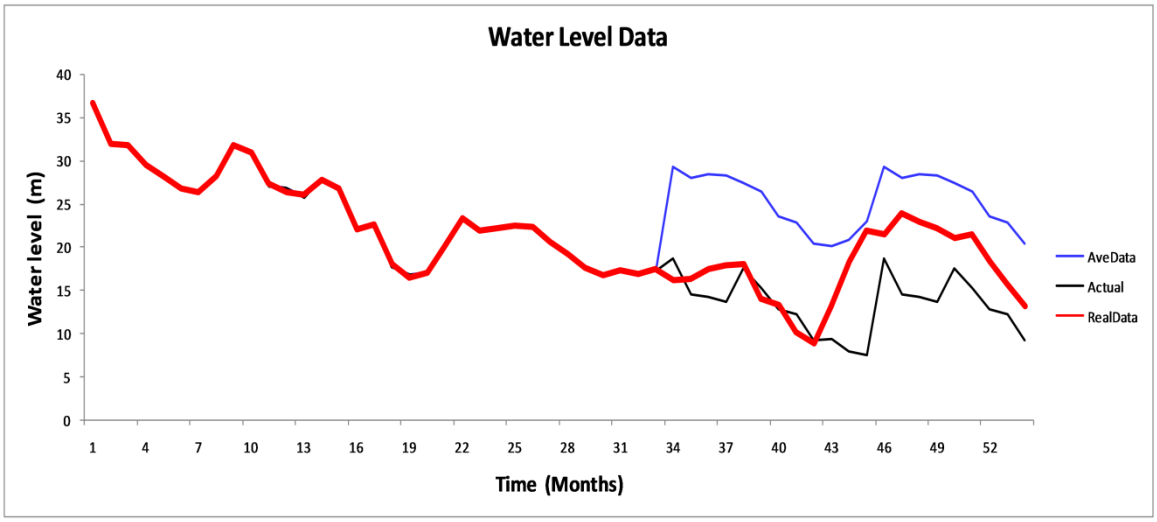


Figure 5.63 Water level graph with the first 40 months being the training data. The red line indicated an averaged value of the results of the neural network simulation of the minimum, maximum and averaged data sets. Black line represents the actual observed data wh

One important observation in the use of artificial neural networks is that the amount of historical data to train on is of considerable importance, if more data was available the simulation results would have converged to give a better prediction.

Finally, it was suggested that one of the two input data sets could be used as a sole source for the training of the neural networks. If no significant interaction is observed from the Flow Volume data then the network should have a significant bias to the Rainfall data and conversely the opposite might also be applicable. To test this hypothesis the most variable data set was used, i.e, the Vaal River system. A representative simulation was chosen for each of the two scenarios and presented in Figure 5.64 and Figure 5.67. The results of training and simulations of one thousand Time-Delay

neural networks (10 neurons per layer) are presented in Figure 5.65 – Figure 5.69 for each of the respective training methodologies. In the training of both sets a significant difference in output results were expected.

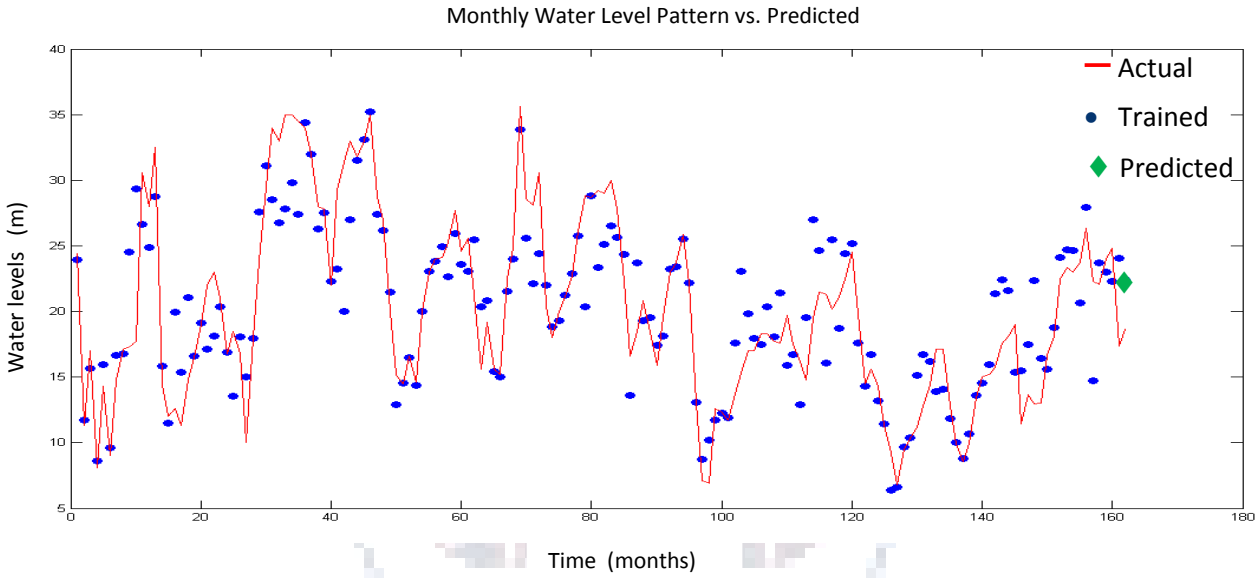


Figure 5.64 Neural network simulation of Vaal River system borehole data using only rainfall data. Red line indicates actual observed, while blue dotted line indicates fitted data points.

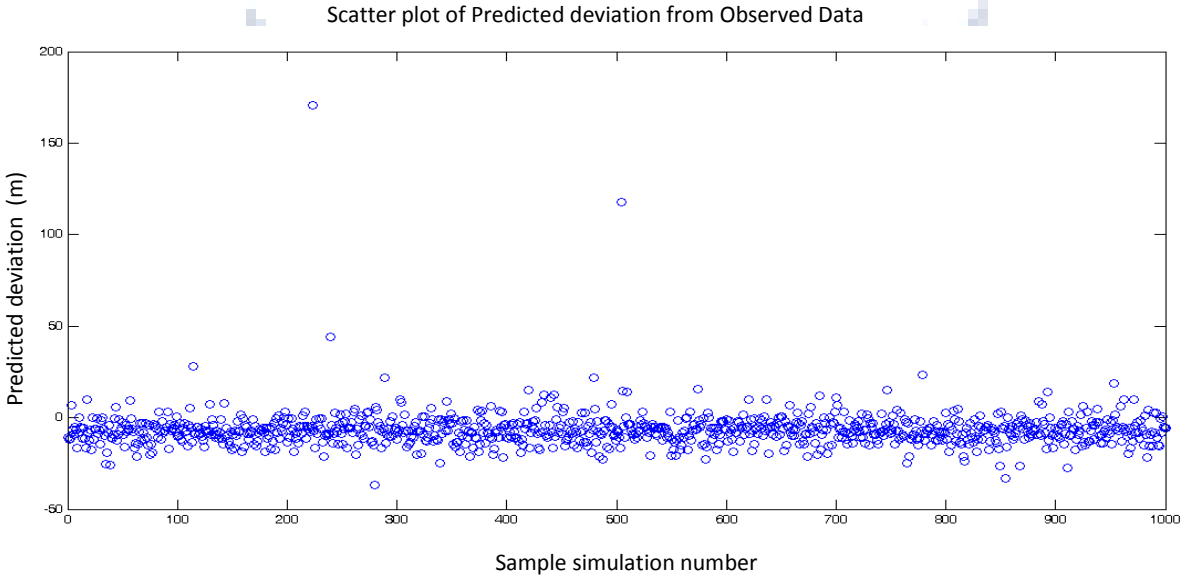


Figure 5.65 Scatter plot of estimated error in Time-Delay neural network training and simulation results were using only rainfall to estimate water levels.

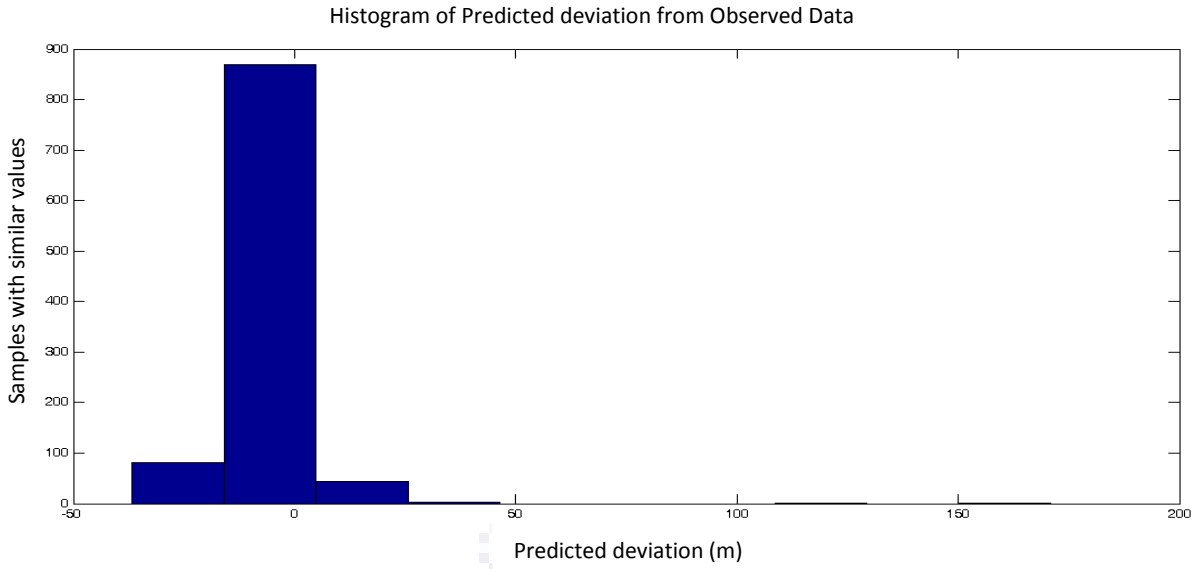


Figure 5.66 Histogram plot of estimated error in Time-Delay neural network training and simulation results were using only rainfall to estimate water levels.

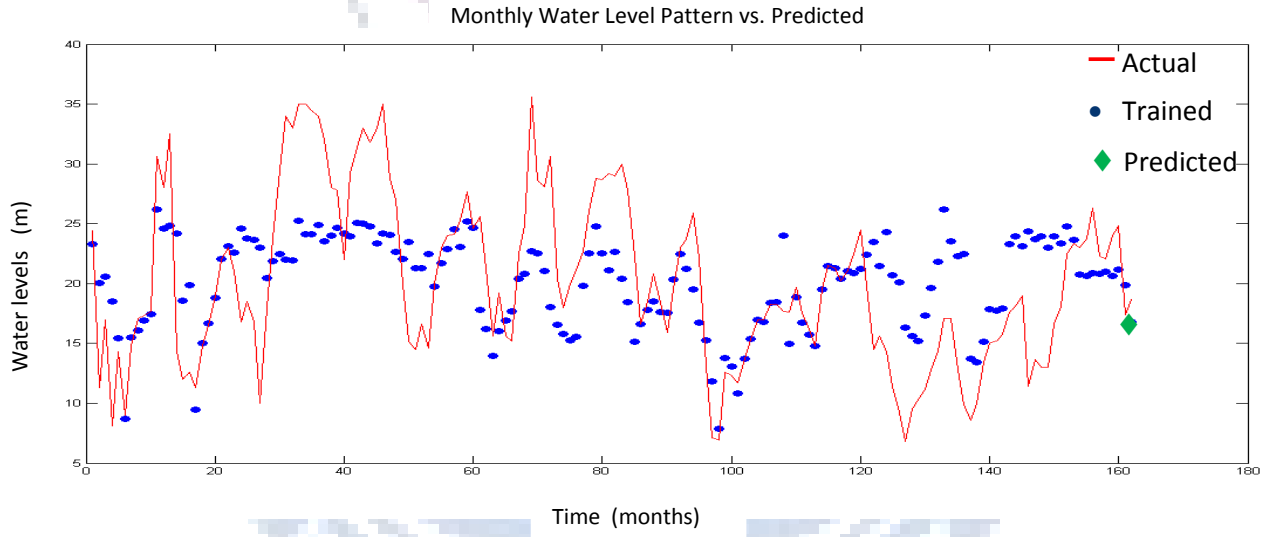


Figure 5.67 Neural network simulation of Vaal River system borehole data using only flow volumes data. Red line indicates actual observed, while blue dotted line indicates fitted data points.

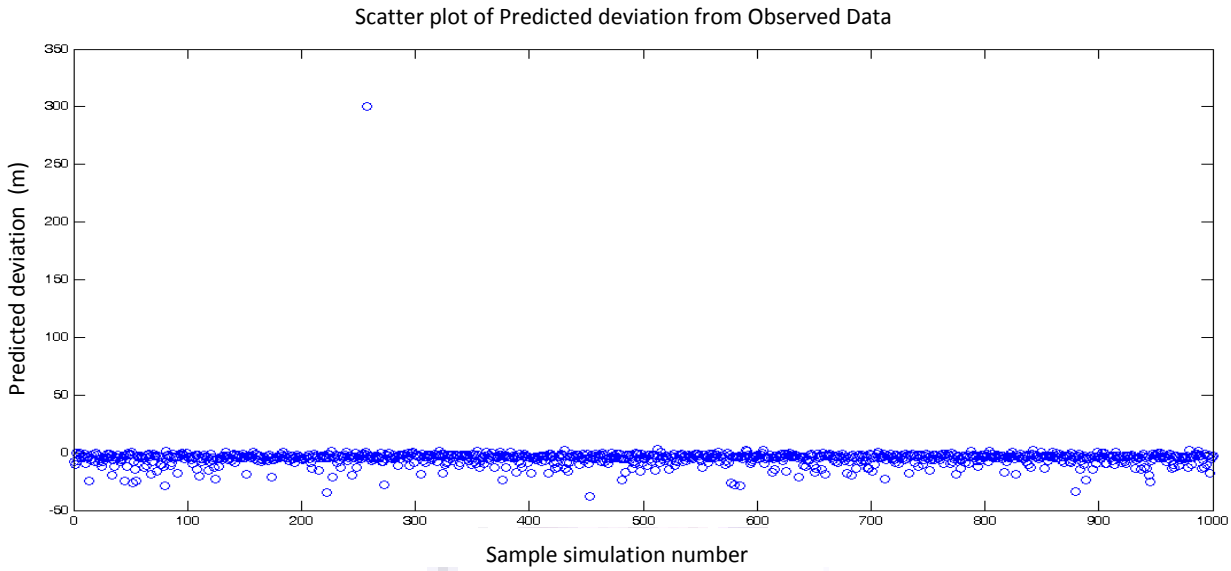


Figure 5.68 Scatter plot of estimated error in Time-Delay neural network training and simulation results were using only flow volumes to estimate water levels.

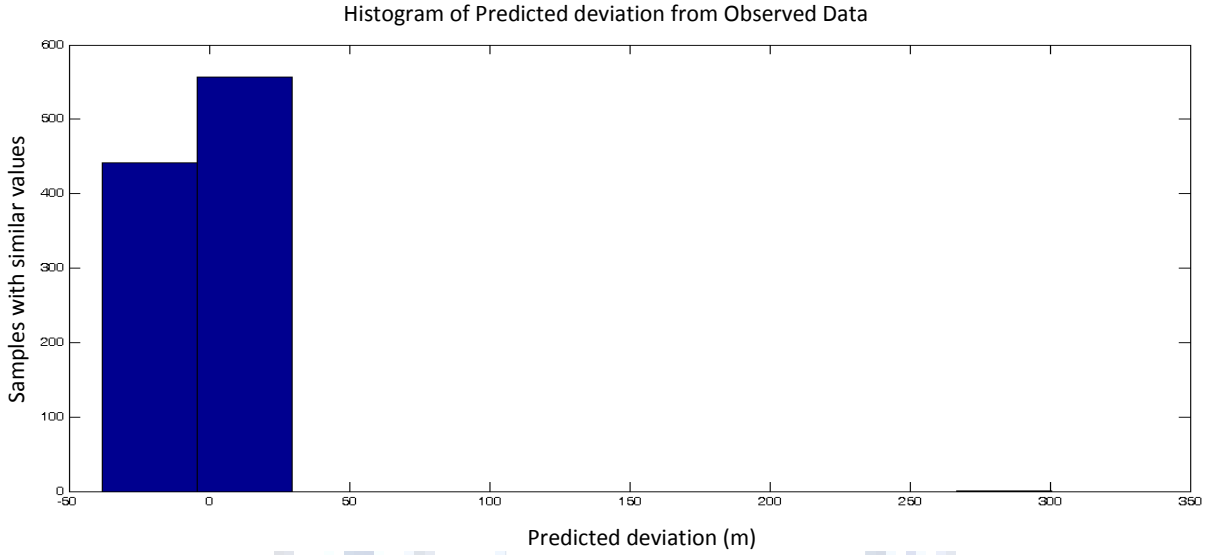


Figure 5.69 Histogram plot of estimated error in Time-Delay neural network training and simulation results were using only flow volumes to estimate water levels.

The rainfall only input resulted in only a few outliers being observed as found in the histogram plot, Figure 5.66, indicating that the neural network reliably predicts the observed value within error. The data has a maximum and minimum of 197.12 m and -10.47 m respectively. The average value for the data set is 19.53 m with a standard deviation of 9.89. The observed (26.20 m) value is well within the two standard deviation values from the average.

The flow volume only input resulted in only a few outliers being observed as found in the histogram plot, Figure 5.69, indicating that the neural network reliably predicts the observed value within error. The data has a maximum and minimum of 326.80 m and -11.88 m respectively. The average value for the data set is 21.27 m with a standard deviation of 10.89. The observed (26.20 m) value is well within the two standard deviation values from the average. It should also be noted that the maximum of 326.80 m only occurred once and might be a result of early stopping in the training algorithm.

The above results indicate that either rainfall or flow volume data can be used to predict water levels in a borehole in the Vaal River scenario, although the results vary significantly. The addition of a second input only assists in training the neural network and allows for a more generalised prediction scheme.

5.6. Conclusion

In this chapter the use of artificial neural networks to predict surface water and groundwater interactions have been reported. Four different artificial neural network architectures have been investigated, with the most effective forward prediction networks proving to be Time-Delay and Layer Recurrent neural networks. Both the Probabilistic and Radial Basis neural networks were shown to have dependencies which might make these network configurations unusable in general applications. Since performance and speed was important Time-Delay networks were identified as the most optimal solution for forward prediction. Multiple time step forward predictions were also reported, with the five neuron per layer network performing the best in predicting multiple time periods ahead. If a more generalised approach was required with more unknown a simple stochastic method resulted in enough data points to make a viable prediction of water levels over a twenty month period. The issue of rainfall and/or flow volumes to predict groundwater levels were also addressed and shown that in either situation a viable solution could be obtained although to a lesser degree of accuracy.

Chapter 6

Conclusions and Recommendations

A brief summary of the historical context of the development of artificial neural networks have been reported. Mathematical concepts and implementation procedures in artificial neural networks have been presented which allows the use of these networks in complicated real world problems. Estimation of computational power and capacity of neural networks to solve problems were briefly visited. The problem of convergence and over-fitting was discussed with likely solutions to the dilemma. Generalisation and confidence values in artificial neural networks were highlighted, with a focus on statistical methods.

The development of an interaction model between surface water and groundwater has been presented and the method by which this forms a component of the total hydrological cycle. A summary of likely mechanisms that affect the groundwater and surface water interaction was discussed. Bank storage on groundwater recharge was briefly summarised.

The design and construction of neural networks were discussed, ranging from the simple neurons to training methods. Transfer functions which enable the neuron to make a decision have been presented. Four popular transfer functions (hardlim, purelin, logsig and tansig) have been highlighted.

The use of artificial neural networks to patch data (rainfall, waterlevels etc.) has been reported. The effect of number of neurons, layers and transfer functions have been described. In order to simulate data a comparison with statistical methods had been made and fitting of these values were far inferior to that observed by neural networks. Using artificial neural networks to patch rainfall data does seem to be an option, with simulated data having an average confidence value of 77 %. In

contrast the most bothersome aspect of statistical methods is the presence of negative values, which must be zeroed before the data can be used. One shortfall of artificial neural networks in patching of data is the presence of non-perennial or anomalous data points; these events are typically ignored and fitted to an observed average value from previous data.

In the application of artificial neural networks to predict surface water and groundwater interactions, four different artificial neural network architectures have been investigated. The most effective forward prediction networks proved to be Time-Delay and Layer Recurrent neural networks. Both the Probabilistic and Radial Basis neural networks were shown to have dependencies which might make these network configurations unusable in general applications. Since performance and speed was important Time-Delay networks were identified as the most optimal solution for forward prediction. Multiple time step forward predictions were also reported, with the five neuron per layer network performing the best in predicting multiple time periods ahead. The issue of rainfall and/or flow volumes to predict groundwater levels were also addressed and shown that in either situation a viable solution could be obtained.

One of the aims of this study was to investigate the possibility of artificial neural networks as a prediction method in the general work environment, the best possible neural network which can be applied in this instance would be the Time-Delay neural network. It is relatively robust and has a low memory and calculation footprint, which can make it ideal in the application of a black box program. It should also be noted that artificial neural networks cannot predict anomalous events. The possibility to predict surface water and groundwater systems on a quaternary scale does not seem to be probable from this study, due to the specialised nature of each neural network.

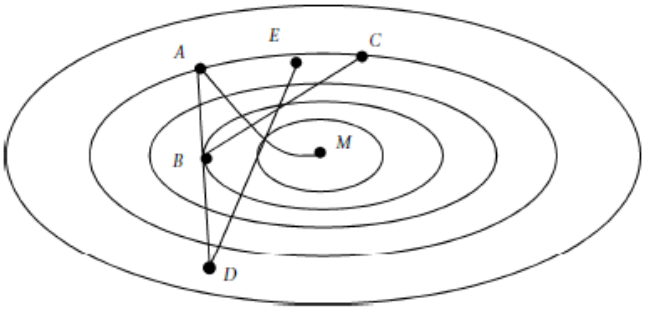
Finally, the implementation of the artificial neural networks can be adapted through the use of an appropriate compiler to create dynamic link libraries (DLL) or executables (EXE). These programs can be used as independent code tools for other programs. Furthermore, these modules can be trained and used to create results from other programs and in turn create future values for management decisions.

Appendix

A.1. Steepest Descent Method

The steepest descent method is a first order minimizer. It uses the first derivative of the potential energy with respect to the Cartesian coordinates. The method moves down the steepest slope of the energy vector on the potential energy surface. The descent is accomplished by adding an increment to the coordinates in the direction of the negative gradient of the potential energy, or the force.

Example: A potential energy surface has a minimum at M. If the minimization begins at point A and proceeds with infinitesimally small steps, the structure follows the overall path A–M during a steepest descent optimization. If the first step is larger, it might proceed along A–B, then the next step would proceed along B–C. A larger initial step might place the system at D. A second step could proceed along the path D–E, which diverges from the minimum.



The steepest descent method rapidly alleviates large forces in the potential energy surface. This is especially useful for eliminating local minima along the pathway of optimisation. Each step in a steepest descent requires minimal computing time. Its disadvantage is that convergence toward a minimum is very slow.

A.2. Back Propagation Algorithm

Backpropagation is the generalization of the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by you. Networks with biases, a sigmoid layer, and a linear output layer are capable of approximating any function with a finite number of discontinuities. Standard backpropagation is a gradient descent algorithm, as is the Widrow-Hoff learning rule, in which the network weights are moved along the negative of the gradient of the performance function. The term backpropagation refers to the manner in which the gradient is computed for nonlinear multilayer networks. There are a number of variations on the basic algorithm that are based on other standard optimization techniques, such as conjugate gradient and Newton methods.

Properly trained backpropagation networks tend to give reasonable answers when presented with inputs that they have never seen. Typically, a new input leads to an output similar to the correct output for input vectors used in training that are similar to the new input being presented. This generalization property makes it possible to train a network on a representative set of input/target pairs and get good results without training the network on all possible input/output pairs. There are two features of Neural Network Toolbox (Matlab) software that are designed to improve network generalization: regularization and early stopping.

There are generally four steps in the training process:

- Assemble the training data.
- Create the network object.
- Train the network.
- Simulate the network response to new inputs.

The default backpropagation training algorithm is Levenberg-Marquardt (trainlm). This is the fastest method in the toolbox, but it can use large amounts of memory. There are ways to reduce this

memory requirement, and there are other algorithms that require less memory and might improve generalization.

Backpropagation Algorithm Method

There are many variations of the backpropagation algorithm, several of which are described in this chapter. The simplest implementation of backpropagation learning updates the network weights and biases in the direction in which the performance function decreases most rapidly, the negative of the gradient. One iteration of this algorithm can be written

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha_k \boldsymbol{g}_k$$

Where $\boldsymbol{x}_k$ is a vector of current weights and biases, $\boldsymbol{g}_k$ is the current gradient, and α_k is the learning rate.

There are two different ways in which this gradient descent algorithm can be implemented: incremental mode and batch mode. In incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. In batch mode, all the inputs are applied to the network before the weights are updated.

A.3. Newton Raphson Method

In numerical analysis, Newton's method or the Newton–Raphson method is perhaps the best known method for finding successively better approximations to the roots of a real-valued function. Newton's method can often converge remarkably quickly, especially if the iteration begins sufficiently near to the desired root. Unfortunately, when iteration begins far from the desired root, Newton's method can easily lead an unwary user astray with little warning. Thus, good implementations of the method embed it in a routine that also detects and perhaps overcomes possible convergence failures.

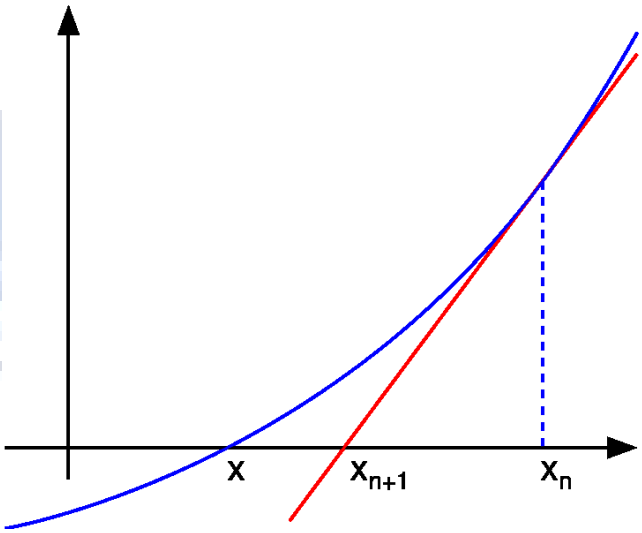
Given a function $f(x)$ and its derivative $f'(x)$, an initial guess of x_0 can be used to estimate the next value x_1 which might be closer to the real minimum value using the formula of $x_1 = x_0 - \frac{f(x)}{f'(x)}$.

Description of the Method

The idea of the method is as follows: one starts with an initial guess which is reasonably close to the true root, then the function is approximated by its tangent line (which can be computed using the tools of calculus), and one computes the x-intercept of this tangent line (which is easily done with elementary algebra). This x-intercept will typically be a better approximation to the function's root than the original guess, and the method can be iterated.

Suppose f : [a, b] → R is a differentiable function defined on the interval [a, b] with values in the real numbers R. The formula for converging on the root can be easily derived. Suppose we have some current approximation x_n. Then we can derive the formula for a better approximation, x_{n+1} by referring to the diagram on the right. We know from the definition of the derivative at a given point that it is the slope of a tangent at that point.

$$f'(x) = \frac{\Delta y}{\Delta x} = \frac{f(x_n) - 0}{x_n - x_{n+1}}$$



Here, $f'(x)$ denotes the derivative of the function $f(x)$. Then by simple algebra we can derive $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$.

The method will usually converge, provided the initial guess is close enough to the unknown zero, and that $f'(x_0) \neq 0$. Furthermore, for a zero of multiplicity 1, the convergence is at least quadratic in the neighbourhood of the zero, which intuitively means that the number of correct digits roughly at least doubles in every step.

A.4. Wavelet Transform

In mathematics, a wavelet series is a representation of a square-integrable (real- or complex-valued) function by a certain orthonormal series generated by a wavelet.

It is well known from Fourier theory that a signal can be expressed as the sum of a, possibly infinite, series of sines and cosines. This sum is also referred to as a Fourier expansion. The big disadvantage of a Fourier expansion however is that it has only frequency resolution and no time resolution. This means that although we might be able to determine all the frequencies present in a signal, we do not know when they are present. To overcome this problem in the past decades several solutions have been developed which are more or less able to represent a signal in the time and frequency domain at the same time.

The wavelet transform or wavelet analysis is probably the most recent solution to overcome the shortcomings of the Fourier transform. In wavelet analysis the use of a fully scalable modulated window solves the signal-cutting problem. The window is shifted along the signal and for every position the spectrum is calculated. Then this process is repeated many times with a slightly shorter (or longer) window for every new cycle. In the end the result will be a collection of time-frequency representations of the signal, all with different resolutions. Because of this collection of representations we can speak of a multi-resolution analysis. In the case of wavelets we normally do not speak about time-frequency representations but about time-scale representations, scale being in a way the opposite of frequency, because the term frequency is reserved for the Fourier transform.

The wavelet transform replaces the Fourier transform's sinusoidal waves by a family generated by translations and dilations of a window called a wavelet. It takes two arguments (time and scale). The

wavelet transform is defined as follows: $Wf(u,s) = \langle f, \psi \rangle = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{s}} \psi^* \left(\frac{t-u}{s} \right) dt$. Where the base ψ is a zero average function, centred on zero with a finite energy.

A.5. Bayesian Classifier

A naive Bayes classifier is a term in Bayesian statistics dealing with a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be independent feature model.

In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. Even though these features depend on the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple.

Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods.

In spite of their naive design and apparently over-simplified assumptions, naive Bayes classifiers often work much better in many complex real-world situations than one might expect. Recently, careful analysis of the Bayesian classification problem has shown that there are some theoretical reasons for the apparently unreasonable efficacy of naive Bayes classifiers. An advantage of the naive Bayes classifier is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

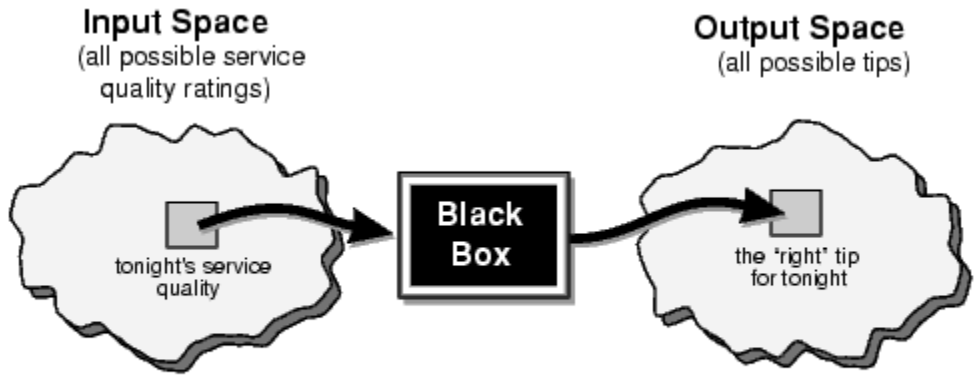
A.6. Fuzzy Logic

Fuzzy logic is a superset of conventional or classic logic that has been extended to handle the concept of partial truth (truth values between *completely true* and *completely false*). Fuzzy logic is discussed in detail in Matlab (2008). Fuzzy logic concerns the relative importance of precision. How important is it to be exactly correct when a rough answer will do? Fuzzy logic balances significance and precision, something that humans have been managing for a very long time.

Fuzzy logic is a convenient way to map an input space to an output space, thereby capturing the knowledge of experts. For example:

- A user states how good the service was at a restaurant, and fuzzy logic tells the user what the tip should be.
- A user states how far away the subject of the photograph is, and fuzzy logic will focus the lens.
- A user states how fast the car is going and how hard the motor is working, and fuzzy logic will shift the gears.

A graphic example of an input-output map is shown below.



An input-output map for the tipping problem:
"Given the quality of service, how much should I tip?"

Between the input and the output, there is a black box that maps the input to the correct output. In the black box there can be a number of systems for example fuzzy systems, linear systems, expert

systems, neural networks, differential equations and interpolated multi-dimensional lookup tables. However, in the black box will contain fuzzy logic for the following reasons:

- It is conceptually easy to understand. The mathematical concepts behind fuzzy reasoning are simple.
- It is flexible. With any given system, it is easy to layer more functionality without starting from scratch.
- It is tolerant of imprecise data. Fuzzy reasoning compensates for imprecise data sets in its processes.
- It can model nonlinear functions of arbitrary complexity.
- It can be build on the experience of experts. In direct contrast to neural networks, which take training data and generate opaque, impenetrable models, fuzzy logic relies on the experience of people who already understand the system.
- Fuzzy systems do not necessarily replace conventional control methods. In many cases, fuzzy systems augment them and simplify their implementation.
- Fuzzy logic is based on natural language. The basis for fuzzy logic is the basis for human communication.

A.7. Universal Turning Machine

Turing machines are basic abstract symbol-manipulating devices which, despite their simplicity, can be adapted to simulate the logic of any computer algorithm. They were described in 1936 by Alan Turing. Turing machines are not intended as a practical computing technology, but a thought experiment about the limits of mechanical computation. Thus they were not actually constructed. Studying their abstract properties yields many insights into computer science and complexity theory.

A Turing machine that is able to simulate any other Turing machine is called a Universal Turing machine (UTM, or simply a universal machine). A more mathematically-oriented definition with a similar "universal" nature was introduced by Alonzo Church, whose work on lambda calculus intertwined with Turing's in a formal theory of computation known as the Church-Turing thesis. The thesis states that Turing machines indeed capture the informal notion of effective method in logic and mathematics, and provide a precise definition of an algorithm or 'mechanical procedure'.

A.8. Divide and Conquer Approach

In computer science, divide and conquer (D&C) is an important algorithm design paradigm based on multi-branched recursion. A divide and conquer algorithm works by recursively breaking down a problem into two or more sub-problems of the same (or related) type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem. This technique is the basis of efficient algorithms for all kinds of problems, such as sorting (e.g., quicksort, merge sort), multiplying large numbers (e.g. Karatsuba), syntactic analysis (e.g., top-down parsers), and computing the discrete Fourier transform (FFTs).

The name "divide and conquer" includes cases where each problem generates only one subproblem, such as the binary search algorithm for finding a record in a sorted list (or its analog in numerical computing, the bisection algorithm for root finding),[1] although some authors call this case decrease and conquer instead.[2] (Such cases can be implemented more efficiently than general divide-and-conquer algorithms, because they can be expressed as tail recursion and/or converted into simple loops.)

The correctness of a divide and conquer algorithm is usually proved by mathematical induction, and its computational cost is often determined by solving recurrence relations.

References

1 D. O. Hebb, “The Organization of behaviour”, New York, Wiley VCH, pp. 62, 1949.

2 F. Rosenblatt, “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, pp. 386-408, 1958.

3 K. Fukushima, “Cognitron: A Self-organizing Multilayered Neural Network, Biological Cybernetics”, v.20, 121-136, 1975.

4 J. J. Hopfield, “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”, Proceedings of the National Academy of Sciences, v79, pp. 2554-2558, 1982.

5 D. D. Rumelhart, G. E. Hinton, R. J. Williams, “Learning Representations by Back-Propagating Errors”, Nature v323, pp. 533-536, 1986.

6 G. V. Cybenko, “Approximation by Superpositions of a Sigmoidal function”, Mathematics of Control, Signals and Systems, v2(4), pp. 303-314, 1989.

7 H. T. Siegelmann, E. D. Sontag, “On the Computational Power of Neural Nets”, J. Comput. Syst. Sci., v50(1), pp. 132-150, 1995

8 A. M. Turing, “On Computable Numbers, with an Application to the Entscheidungsproblem: A correction”, Proceedings of the London Mathematical Society, v2(43), pp. 544–546, 1937.

9 P. J. Werbos, “Back-propagation: Past and future”, Proceeding of International Conference on Neural Networks, San Diego, CA, v1, pp. 343-354, 1988.

10 A. Sperduti, A. Starita, “Speed up learning and network optimization with extended back-propagation”, Neural Networks, v6, pp. 365-383, 1993.

11 R. A. Jacobs, “Increased rates of convergence through learning rate adaptation”, Neural Networks, v1(4), pp. 295-308, 1988.

12 A. Salvetti, B. M. Wilamowski, “Introducing Stochastic Process within the Backpropagation Algorithm for Improved Convergence”, presented at ANNIE'94 - Artificial Neural Networks in Engineering, St. Louis, Missouri, USA, November pp. 13-16, 1994

13 R. Battiti, “First- and second-order methods for learning: between steepest descent and Newton’s method”, Neural Computation, v4(2), pp. 141-166, 1992.

14 F. Coetzee, Media24, Volksblad, 26 November, pp. 1, 2008.

15 T. C. Winter, J. W. Harvey, O. L. Franke, W. M. Alley, “Ground water and surface water, A single resource”, U.S. Geological Survey Circular 1139, Denver, Colorado, 1998.

16 W. Wang, P. H. A. J. M. Van Gelder, J. K. Vrijling, J. Ma, “Forecasting daily streamflow using hybrid ANN models”, J. Hydrology, v324, pp. 383-399, 2006.

17 M. I. Jordan, R. A. Jacobs, “Hierarchical mixture of experts and the EM algorithm”, Neural Comput., v6, pp. 181-214, 1994.

18 K. W. Hipel, A. I. Mcleod, “Time Series Modelling of Water Resources and Environmental Systems”, Elsivier, Amsterdam, 1994.

19 H. R. Maier, G. C. Dandy, “Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications”, Environ. Model. Softw., v15, pp. 101-123, 2000.

20 C. W. Dawson, R. L. Wilby, “Hydrological modelling using artificial neural networks”, Prog. Phys. Geography, v25(1), pp. 80-108, 2001.

21 A. W. Minns, M. J. Hall, “Artificial neural networks as rainfall-runoff models”, J. Hydrol., v10, pp. 282-290, 1996.
22 M. Hagan, H. Demuth, M. Beale, “Neural Network Design”, Boston, MA, PWS Publishing, 1996.
23 H. Demuth, M. Beale, M. Hagan, “Neural Network Toolbox™ 6, User’s Guide”, The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760-2098.
24 M. Hagan, "Backpropagation ", Chapter 5 of the User's Guide for the Neural Network Toolbox for MATLAB, ver. 3, H. Demuth and M. Beale, Natick, MA, The Mathworks Inc., 1998.
25 T. Kohonen, “Self-Organizing Maps”, v30, Springer, Berlin, 2001.
26 J. L. Elman, E. A. Bates, M. H. Johnson, A. Karmiloff-Smith, D. Parisi, K. Plunkett, “Rethinking Innateness: A Connectionist Perspective on Development”, Cambridge, MA: MIT Press., 1996.
27 J. D. Hamilton, “Regime-switching models”, Palgrave Dictionary of Economics, San Diego, 2005.
28 P. Coulibaly, F. Anctil, R. Aravena, B. Bobee, “Artificial neural network modelling of water table depth fluctuations”, Water Resources Research, v. 37(4), pp. 885-896, 2001.
29 M. N. French, W. F. Krajewski, R. R. Cuykendall, “Rainfall forecasting in space and time using a neural network”, J. Hydrol., v137, pp. 1-31, 1992.
30 CB Predictor 1.6, Crystall Ball, http://www.crystalball.com/, +1 303-534-1515, Oracle, 2007.
31 P. Coulibaly, F. Anctil, R. Aravena, B. Bobee, “Artificial neural network modelling of water table depth fluctuations”, Water Resources Research, v37(4), pp. 885-896, 2001.
32 C-C. Yang, S. O. Prasher, R. Lacroix, S. Sreekanth, N. K. Patni, L. Masse, “Artificial Neural Network Model for Subsurface-Drained Farmlands”, J. Irrig. Drain. Eng., v123(4), pp. 285-292, 1997.
33 E. El Tabacha, L. Lancelot, I. Shahrour, Y. Najjar, “Use of artificial neural network simulation metamodelling to assess groundwater contamination in a road project”, Math. Comp. Mod., v45, pp. 766–776, 2007.

Abstract - Uitreksel

Abstract

Groundwater has been identified as a viable alternative for future freshwater production in South Africa. The management thereof is steadily gaining more recognition from governmental institutions. A significant obstacle in the development of this resource is the conceptual understanding of surface water and groundwater interaction. The availability of reliable data for rainfall, flow volumes in rivers and water levels in boreholes have prompted an investigation into patching incomplete data sets. This study also focused on predicting the influence of rainfall and flow volumes in a river on the surrounding groundwater levels. Neural networks have been used to investigate both data patching and forward prediction of water levels in selected data sets.

Uittreksel

Die gebruik van grondwater as 'n alternatiewe bron van varswater in Suid Afrika is geïdentifiseer as 'n heel bruikbare opsie. Die bestuur van die hulpbron is stadig maar seker besig om weier impak en erkenning te kry by regeringsinstansies. 'n Werklike probleem in die ontwikkeling van hierdie hulpbron is die konsepsiuele interaksie tussen oppervlakswater en grondwater. Die beskikbaarheid van 'n volledige stel data van reënval, vloeivolumes en watervlakke in boorgate het tot gevolg gehad dat opvulling van verlore data benodig word deur die gebruik van neurale netwerke. In hierdie ondersoek is gefokus op die voorspelling van die invloed wat reënval en vloeivolumes in riviere op die omliggende boorgat watervlakke het, dus word neurale netwerke ingespan om beide onvolledige data stelle te voltooi en om voorwaartse skattings te maak van watervlakke in boorgate.

Keywords: Neural Network, Geohydrology, Forecasting, Patching, Surface Water And Groundwater Interaction