Journal of Hydrology (NZ) 37(2): 81-94, 1998

# Smoothing hydrological data: a neural network approach

**Jane T. Hill and Derek G. Goring**
*National Institute of Water and Atmospheric Research Ltd (NIWA), PO Box 8602, Christchurch.*

## Abstract

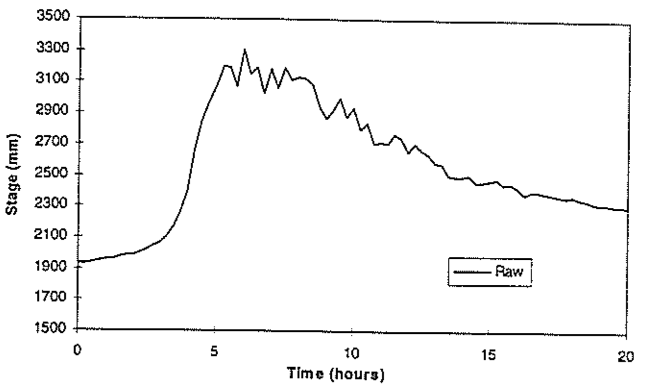
Neural networks were applied to the problem of smoothing flood hydrographs for the Waimakariri River that cannot be smoothed by other automatic methods. A system combining three different neural networks gave satisfactory (and in some cases better) results than manual smoothing. Because of the difficulty in setting up a network, however, the method of neural networks is recommended only for problems that cannot be solved by simpler means.

## Introduction

Hydrological data sequences often contain noise. In many cases, the noise may be eliminated by low-pass filtering (taking means, for example). However, the original data sequence often contains important information that should not be lost by the filtering process. One example is the use of time series of lake levels to calculate inflow to a lake: the lake level series must be differentiated and this exacerbates the noise. However, if daily means are used to smooth the data, most fluctuations in the inflow with periods of 24 hours or less are eliminated. Another example is a flood hydrograph from the Waimakariri Gorge water-level site (Fig. 1). The hydrograph, which is typical of all floods at the Gorge, shows fluctuations in water level that begin just before the crest of the rising limb and continue down the falling limb. The fluctuations are probably caused by waves of gravel pulsing through the Gorge. Goring (1994) analysed flood waves emerging from the Gorge and modelled their propagation down the river to the Old SH Bridge, 51 km away, in an effort to explain the wide variation in observed travel times (6 to 12 hours). He found that the shape of the flood wave at the Gorge, and in particular the slope of the rising limb of the flood wave, was the critical factor in determining how the wave would propagate. To model the propagation, waves such as that shown in Figure 1 had to be smoothed to remove the oscillations on the crest and in the falling limb. Various standard low-pass filtering methods were tried, but none could remove the oscillations without affecting the slope of the rising limb. Therefore, the data had to be smoothed manually. Manual smoothing involves drawing a curve through a set of data by eye, then re-digitising the data. However, this method is unsatisfactory when large numbers of such curves must be processed, as it requires a human operator, and errors may creep in.

This note describes an alternative smoothing algorithm that uses the technique of neural networks; it falls between the two extremes of manual smoothing and low-pass filtering. The objective was to find an automatic way to smooth noisy signals without losing vital information.

**Figure 1** – Raw water-level data from the Waimakariri Gorge water-level site for a typical flood on the Waimakariri River. Fluctuations, probably caused by gravel waves, can be seen at peak flow and on the falling limb.



## Neural networks

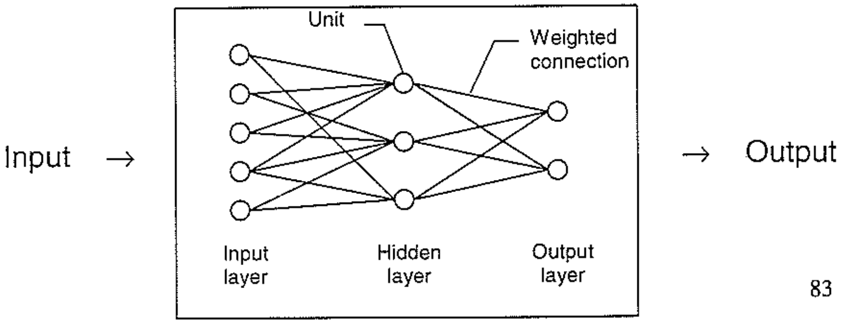
There are many textbooks describing neural networks, e.g. Hertz *et al.* (1991), and their application to time-series data, e.g. Weigend and Gershenfeld (1993), Vemuri and Rogers (1994). There are also many neural network computer packages available. We have used the Matlab Neural Network Toolbox, described by Demuth and Beale (1994). In this section we briefly describe neural networks in a non-mathematical way; more detailed information is available in the above references.

Neural networks have been developed as part of international research into artificial intelligence, being inspired by the neurons and synapses of the human brain. They provide an approach which, at a very basic level, attempts to mathematically simulate the working of the brain.

In broad terms, a neural network can be thought of as a "black box"; input data is related to output data (or causes to effects) without prior knowledge of the processes involved. Such models have been used extensively in New Zealand for many years. Some examples include the linear systems analysis technique of Goring (1984) for flood routing used in several flood forecasting systems, the unit hydrograph method for modelling rainfall and runoff, and the more sophisticated, adaptive Kalman filtering method of Bidwell and Griffiths (1994) for rainfall and runoff. Neural networks provide a generalised framework for these methods. Furthermore, some neural networks have a much more complicated structure than either linear systems or Kalman filtering, and are useful in a wide range of problems.

Fundamentally, a neural network is a group of processing elements, some of which are joined by weighted connections. The number of elements and the connectivity between them defines the architecture of a neural network. A typical example is shown in Figure 2: it consists of a set of layers joined by weighted connections. Data flow from one layer to the next until the output layer is reached. Determination of the weights of the connections between the layers requires a set of desired outputs for a given set of inputs. The process of determining weights is called learning and is analogous to the human process of learning, i.e. we observe how a system responds to stimuli and from this predict how it will respond in future to a similar stimulus. In most cases, learning is "supervised" i.e. for training data, the desired output is known in advance and these values (targets) are used in the training process. Generally, data are split into two groups, a training set and a test set. Targets are known for the input from both sets, but the training process uses only data from the training set. Data from the test set is then used to establish how well the network operates.

**Figure 2** – Schematic diagram neural network architecture.



There are many different learning algorithms (or models); some are based purely on linear calculations while others include non-linear interactions. The model architecture describes how the input is processed through the weighted connections to produce output. A linear network uses a linear combination of the weights of the connections and the input values provided to the network.

The first stage of designing a neural network is to choose the model and architecture; the network is then trained using a suitable data set. This is done by supplying input to the network, and obtaining the output, which is then compared to the target values. The process is iterative and involves changing the weights to minimize the difference between the output from the network and the target values for a set of given input training values. Once training is complete, the network can be applied to the test data and evaluated, and if the trained network performs to requirement, it can be used either on its own, or as part of some other system.

In the next section we illustrate the steps involved in designing a neural network by using the method to smooth Waimakariri flood hydrographs as shown in Figure 1.

## Application

The flood hydrograph for the Waimakariri Gorge (Fig. 1) consists of a steep rising limb, a crest and a falling limb that recedes more slowly. Oscillations start just before the crest and continue over the crest and down the falling limb. Our goal is to remove these oscillations, but to retain the slope of the rising limb and the overall shape of the hydrograph. This can be done manually or by using some other smoothing technique (e.g. one of the family of low-pass filters). A low-pass filter suppresses sudden changes (high frequencies) in the hydrograph, while smooth (low frequency) behaviour is relatively unaffected. Figure 3 illustrates these methods for two typical floods. The first flood hydrograph (Fig. 3 a-c) has a very steep rising limb, while the second (Fig. 3d-f) has a flatter rising limb but much more noise on the crest.

In the manual method, only the sections with oscillations are smoothed; a curve is drawn through the mean of the oscillations and the curve is re-digitised. This method produces an excellent fit (Fig. 3a and d).

Two low-pass filters were examined: moving means (Fig. 3b and e) and a tapered boxcar, frequency domain filter (Goring and Bell, 1996) (Fig. 3c and f). Moving means are unsatisfactory–for one flood (Fig. 3b) the slope of the rising limb is reduced significantly and for the other (Fig. 3e)

**Figure 3** – Comparison of smoothing: (a) manual, (b) 6-term moving means and (c) tapered boxcar filter for a flood hydrograph with a steep rising limb

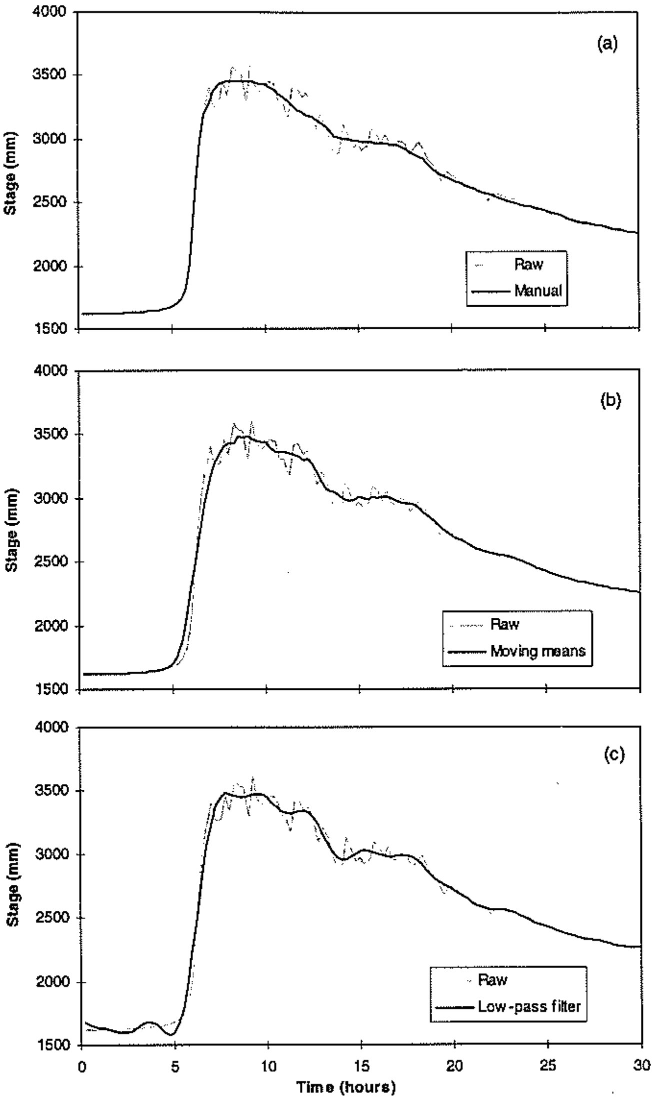
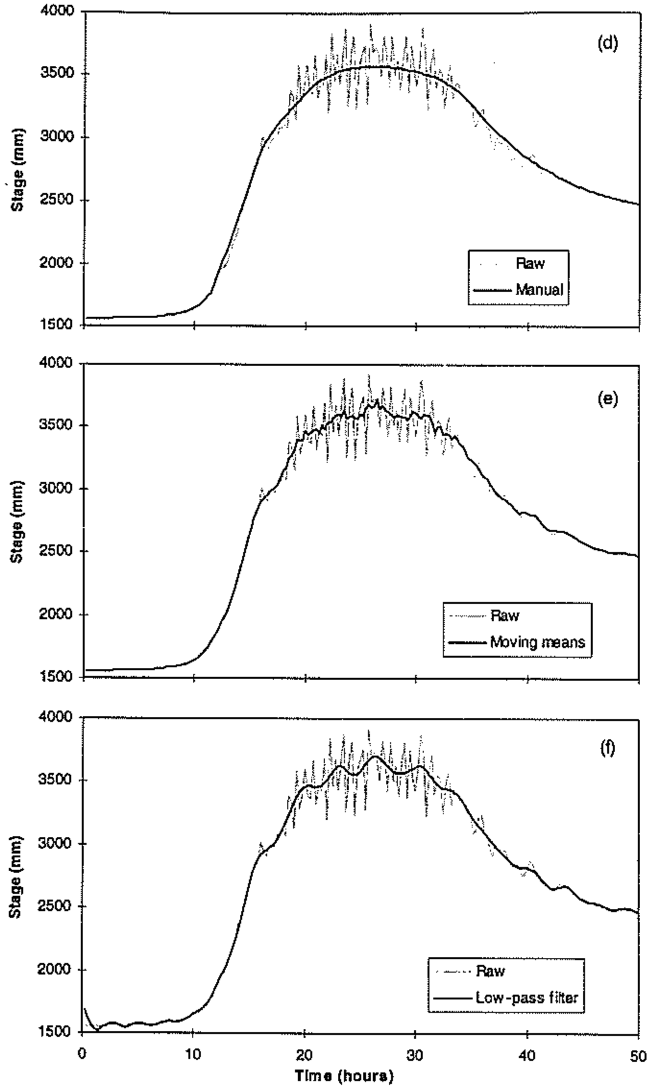


**Figure 3 (cont.)** – Comparison of smoothing: (d) manual, (e) 6-term moving means and (f) tapered boxcar filter for a flood hydrograph with a lot of noise

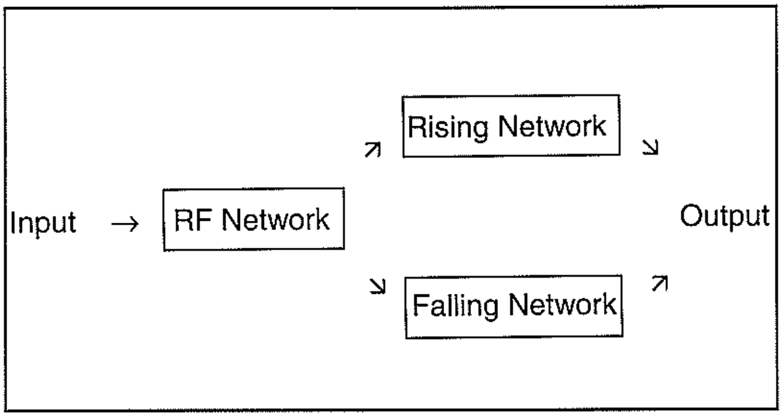


the noise is not eliminated. The boxcar filter smooths the noise better, but extra oscillations remain. Neither method can distinguish between the steepness of the rising limb and the steepness in the oscillations: if the oscillations are removed, the slope of the rising limb is reduced, but if the slope of the rising limb is retained, the oscillations are not eliminated.

To solve this problem, a neural network was trained to recognise rising and falling limbs of a hydrograph, so these sections may be treated separately, just as a human does in the manual method. This network is called the "Rising-Falling" or "RF" network. Two further networks were developed to deal with the rising and falling limbs.

The final system (Fig.4) consists of the three networks combined. The first of the three indicates in what proportion to use the second two, depending on whether the input is from a rising or a falling limb.

**Figure 4** – Architecture of combination of rising and falling limb networks.



## Rising-Falling network

Because there is noise at the peak of a flood, splitting the data into rising and falling limb components is not a trivial task. A linear network is used to classify the data into rising and falling components. The input for a particular value consists of the value itself, and the 12 previous and the 12 subsequent values (where the data interval is 15 minutes), giving 25 input units, while the output layer consists of one unit. The corresponding target value (desired output) is set to unity if that value is part of the rising limb and zero otherwise. These target values are determined by hand (i.e. manual smoothing) for floods in the training set.

The network is trained using the floods from the training set and the target values for those floods. Once training is complete, the weights of the network are fixed and no further changes are made. Then, given data from a flood, for each value on the hydrograph, the input (consisting of the value itself and its neighbouring values) is supplied to the network. The resulting value should indicate whether the original value is on a rising or falling limb. If the result is approximately one, then the value is likely to be on a rising limb, while if the result is approximately zero, it is likely to be part of a falling limb.

These outputs are turned into a step function based on a threshold of 0.5, with ones corresponding to rising values and zeroes otherwise. The step function is then smoothed to achieve a smooth transition between rising and falling states. This removes spikes in the step function, and the resulting function has no sharp changes between states. For each flood hydrograph, we call this corresponding smoothed step function the "rf-function".

### 'Rising' and 'falling' networks

Since the rising limb is steep and not very noisy, the 'rising' network needs to closely follow the movements of the original data, while the 'falling' network needs to perform more smoothing of the local oscillations.

The training data are split into two categories on the basis of the step functions derived to separate the rising and falling limbs. A linear network is trained, using as input the data corresponding to rising limbs, and as targets, the corresponding values obtained from smoothing the data by hand. Similarly, another network is trained using the data from the falling limbs. These networks are known as the 'rising' network and the 'falling' network respectively.

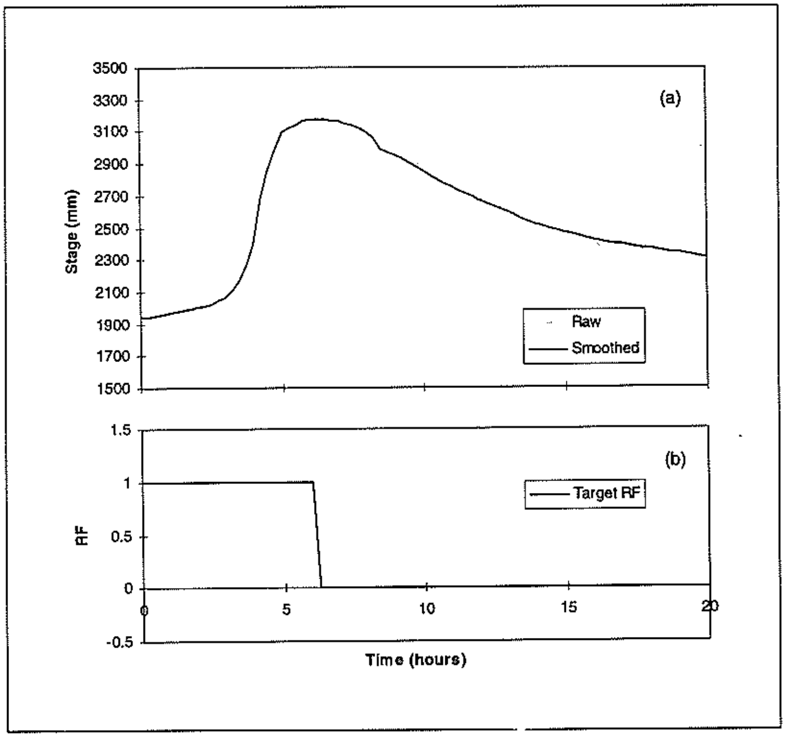
### Combining the networks

To obtain a smoothed flood wave we pass the unsmoothed data through all three networks to produce the rf-function, $rf$, from the rising-falling network, the rising limb, $x_r$ from the rising limb network and the falling limb, $x_f$ from the falling limb network. Then we combine these to produce the smoothed flood wave using: $rf \cdot x_r + (1 - rf) \cdot x_f$.

## Data

Water-level data sets from 32 floods of the Waimakariri River were analysed. These sets were split into two groups: a training group of 20 floods and a test group of 12 floods. For each of the 32 floods, the raw data were smoothed by hand to give target data that preserved the steepness of the rising limb and smoothed the data on the falling limb (e.g. Fig. 5a). For each of the 20 floods in the training set, a target step function was produced, assigning a value of unity to the rising component and zero to the remaining data (Fig. 5b).

**Figure 5** – Data supplied to network for flood in training phase: (a) flood hydrograph with a manually fitted curve and (b) target set function for classifying rising and falling limb.

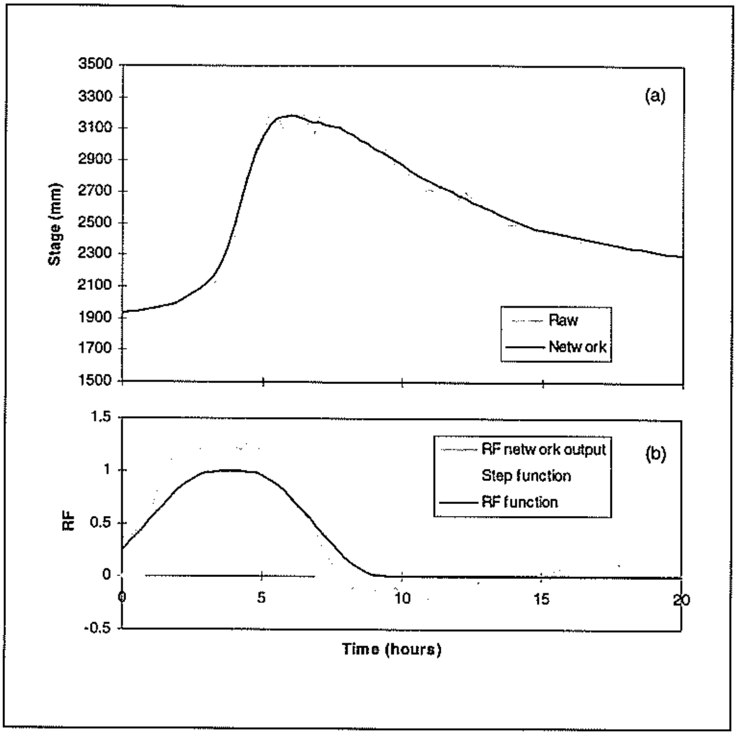


## Results

### Training

In the training phase, the three networks (Fig. 4) were trained, using 20 of the 32 floods. Training was linear, and using the Matlab Neural Network Toolbox, weights were the input and corresponding target data. The resulting networks were tested against the data, and typical results are presented in Figure 6. Figure. 6b shows how a step function is derived from the output from the rising-falling network, which is subsequently smoothed to give the rf-function. This function governs the proportions of the output from the rising and falling networks which are used to produce the smoothed hydrograph curve in Figure 6a.

**Figure 6** – Results of training the network for a flood from the training set: (a) flood hydrograph and (b) r-f network output with derived functions.



## Testing

The fit of the smoothed data with the target data were compared for both the events used in the training phase and for the 12 test events. Figure 7 shows typical results for the two floods considered earlier (Fig. 3).

We also compared the success of the neural network with the other methods of smoothing (moving means and boxcar filter) by calculating the error as the difference between the estimated and target wave, then calculating statistics of that error. The parameter used was average relative variance, *arv*, described in Weigend *et al.* (1990), defined for a set, *S*, by

$$arv_s = \frac{\sum_{k \in S} (x_k - \bar{x}_k)^2}{\sum_{k \in S} (x_k - \mu_s)^2} = \frac{1}{\sigma_S^2} \frac{1}{N_S} \sum_{k \in S} (x_k - \bar{x}_k)^2 ,$$

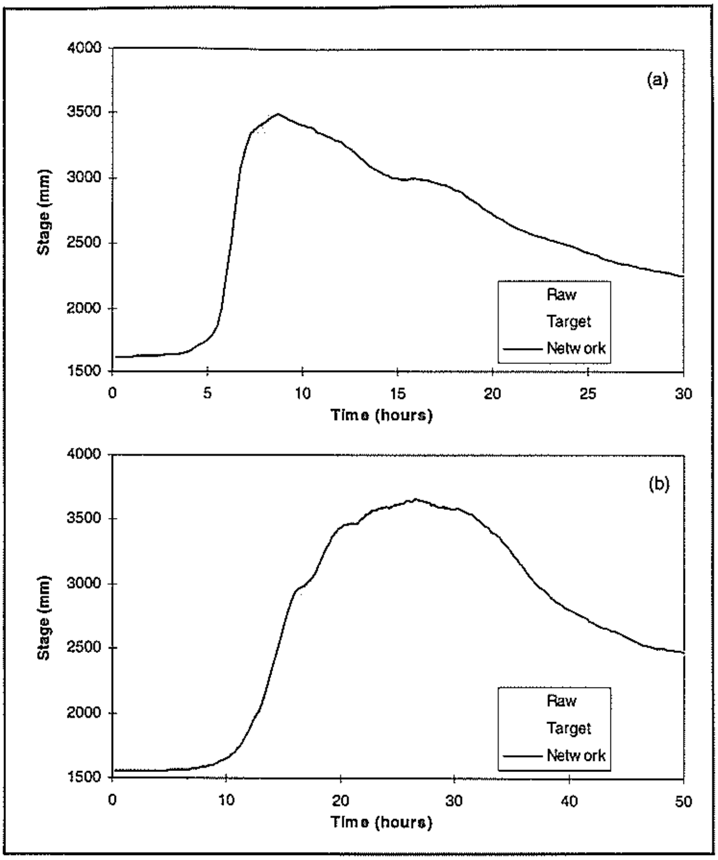
where $x_k$ is the measured value, $\bar{x}_k$ is the estimated value, $N_s$ is the number of elements in the set, $\mu_s$ is the mean and $\sigma_S^2$ is the variance. The normalisation (division by the estimated variance of the data) removes the dependence on the range of the data. Figure 8 shows the average relative variance for the three smoothing methods for each of the 32 floods: the 20 floods in the training set and the 12 in the test set.

## Discussion

The fit between target and network data (Fig. 7) is generally very good. Indeed, in some cases for parts of the wave the neural network does a better job of smoothing the curve than the manual method used to determine the target wave. This raises an important factor in the use of any of the automatic methods for smoothing, namely, that they are objective. This has advantages and disadvantages. Manual smoothing is subject to human error while automatic methods are not, but manual smoothing automatically adapts to differences in the shape of the flood waves. Designing automatic methods which do this is difficult. Neural networks are an attempt, but the design of the architecture of the network and the model applied is critical.

The neural network presented here smooths data better than either the moving mean or boxcar filter methods. However, it comes with a price. To apply low-pass filtering, we need only pass the data through the respective algorithms to produce the results (Fig. 3a-f). Some experimentation may be required to optimise the parameters (i.e., the number of terms for the moving mean and the cutoff frequency for the boxcar filter), then the goodness of fit can be assessed by eye. However, for a neural network we need a set of target data i.e. data that have been optimally smoothed. In most cases, this must be done by hand by someone familiar with the data. Experimentation is required to optimise the neural network, i.e., determine the number of units in the network and the learning

**Figure 7** – Results of training the network for the two floods shown in Figure 3.

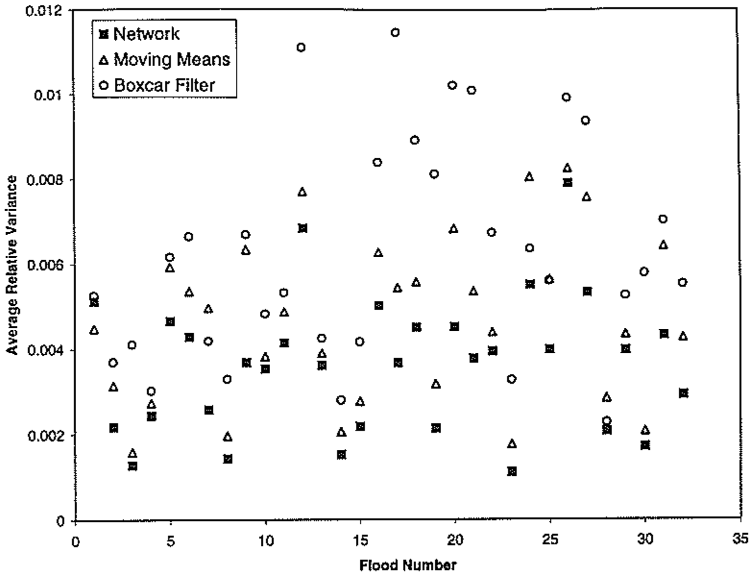


algorithm that should be used. Because of these disadvantages, the use of neural networks will probably be restricted to those problems which cannot be solved satisfactorily by other methods.

The neural network used in this application was a very simple, linear one. Other neural networks can incorporate nonlinear interactions and can be adaptive i.e., the network learns as it passes through the data, as opposed to learning once and applying those rules always.

Research is continuing on the use of neural networks for real-time detection of water-level recording measurement errors such as spikes and gaps in the data and drifting.

**Figure 8** – Average relative variance of three smoothing methods for 32 floods.



Following the suggestion of one of the reviewers (E. Bardsley, pers. comm.), investigation is also underway into the efficacy of using wavelets for smoothing hydrological data. Wavelets have the advantage that they are localised. This means that they may automatically detect the difference between rising and falling limbs of hydrographs, and adjust the smoothing algorithm accordingly.

## Conclusions

The method of neural networks can be used to smooth flood hydrographs that could not be smoothed automatically using traditional methods, while still preserving the slope of the rising limb. A network was designed which distinguished between the rising limb and other parts of the flood wave. This network, in conjunction with separate networks to handle the rising and falling limbs, gave much better results for smoothing than other automatic systems. In some cases the neural network performed the smoothing better than the manual method. Neural networks however are more difficult to use than other automatic methods and should be considered only for those problems which cannot be solved by routine methods such as low-pass filtering.

## Acknowledgements

This work was carried out under contract CO1417 for the Foundation of Research, Science and Technology (New Zealand). The Canterbury Regional Council is thanked for provision of the Waimakariri River water-level data. Glenn Carter and Charles Pearson (NIWA) are thanked for constructive reviews and comments. Neural network processing was done using Matlab (The Math Works Inc), with the Neural Network Toolbox.

## References

Bidwell, V.J.; Griffiths, G.A. 1994: Adaptive flood forecasting: an application to the Waimakariri River. *Journal of Hydrology New Zealand: 32(2)*: 1-15.

Demuth, H.; Beale, M. 1994: *Neural Network Toolbox User's Guide.* The Math Works, Inc. 359p.

Goring, D.G. 1984: Flood routing by a linear systems analysis technique. *Journal of Hydrology 69*: 59-76.

Goring, D.G. 1994: Kinematic shocks and monoclinal waves in the Waimakariri: a steep, braided, gravel-bed river. *Proceedings of International Symposium: Waves – Physical and Numerical Modeling, Vancouver, August 1994*: 336-345.

Goring, D.G.; Bell, R.G. 1996: Distilling information from patchy tide gauge records: the New Zealand experience. *Marine Geodesy 19*: 63-76.

Hertz, J.A.; Palmer, R.G.; Krogh, A.S. 1991: *Introduction to the theory of neural computation.* Addison-Wesley. 327p.

Vemuri, V.R.; Rogers, R.D. 1994: *Artificial Neural Networks, Forecasting Time Series.* IEEE Computer Society Press. 110p.

Weigend, A.S.; Gershenfeld, N.A. 1993*: Time Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley. 643p.

Weigend, A.S.; Huberman, B. A.; Rumelhart, D. E. 1990: Predicting the Future: A Connectionist Approach. *International Journal of Neural Systems 1(3)*: 193-209.

**Manuscript received: 10 July 1997; accepted for publication: 18 April 1998.**