

RESERVOIR INFLOW FORECASTING USING NEURAL NETWORKS

CHANDRASHEKAR SUBRAMANIAN
MICHAEL T. MANRY
Department Of Electrical Engineering
The University Of Texas At Arlington
Arlington, TX 76019

JORGE NACCARINO
Bailey Network Management
P.O. Box 5031
Sugar Land, TX 77487-5031

INTRODUCTION

In utilities using a mixture of hydroelectric and non-hydroelectric power, the economics of the hydroelectric plants depend upon the reservoir height and the inflow into the reservoir for several days into the future. Accurate forecasts of reservoir inflow allow the utility to feed proper amounts of fuel to individual plants, and to economically allocate the load between various non-hydroelectric plants.

Neural networks provide an attractive technology for inflow forecasting, because of (1) their success in power load forecasting¹⁻⁶, and (2) because of the availability of relevant measurements, including historical data. Typical load forecasting systems use separate weekly, daily, and hourly neural net modules to model the different load trends. The various estimates are then fused to form a continuous forecast. Neural network load forecasting systems avoid using large neural networks by breaking the forecasting task into many separate networks, each operating at a specific time of day and forecasting a specific number of hours into the future. In addition these systems concatenate forecasts so that forecast powers are used as inputs to networks forecasting powers further into the future.

However, there are several problems that must be overcome before neural network load forecasting systems can be successfully applied to inflow forecasting. First, forecast inflows cannot be used as inputs to inflow forecasting networks, since that has been observed to lead to unstable behavior. Second, the day of the week is of no importance in inflow forecasting. Third, too many networks are required if a separate network is used to forecast each hour in the future, over the period of several days. Fourth, each network requires hundreds of inputs, since we use hourly inputs over the past week in inflow forecasting. Fifth, forecasting systems using multiple networks often have too little training data per network.

In this paper we describe a system which avoids these problems. First we describe the features and the desired outputs related to our system. We also introduce some

notation. Then we discuss compression by Karhunen-Loeve transform and its application in the reservoir inflow forecasting system. Next we describe the neural network used and the training algorithm adopted for the system. Lastly we discuss some forecasts produced by the system.

RAW FEATURES AND DESIRED OUTPUTS

Features

In our system, we are forecasting inflow levels in a reservoir. Therefore past inflow levels form a vital set of inputs for the system. Other feature groups include weather variables such as temperature, rainfall, and snowfall. Time related features such as time-of-year and day-of-year are also used in our system.

In forecasting systems, having just one element per feature group is not very useful. Usually a given feature group will have a large number of elements. Suppose we have K feature groups, then each feature group \mathbf{x}_i , a vector, would have N_i elements, where i varies from 1 to K . These K feature groups are then arranged as $\mathbf{x} = \{\mathbf{x}_1^T, \mathbf{x}_2^T, \dots, \mathbf{x}_K^T\}^T$, to form an input vector. The dimension of the input vector \mathbf{x} is then,

$$N = \sum_{i=1}^K N_i \quad (1)$$

In our system, N_i for a feature group varies from 48 to 120, depending on the number of hours in the past from which data is extracted.

Past inflow levels used in our system were recorded at hourly intervals over a period of three years. The inflow levels were measured in cubic meters per second.

The weather variables were also measured and recorded at hourly intervals over the same interval. Temperature was measured in degree Kelvin. Rainfall and snowfall were measured in centimeters at hourly intervals over the same time period.

Time related features such as time-of-day and day-of-year are coded as sine and cosine terms. Time-of-day features are coded as $\sin(2\pi ni/24)$ and $\cos(2\pi ni/24)$ where i denotes the number of pairs (harmonics) of time features chosen and n denotes the time in hours. n takes values in the range (0, 23). Similarly day-of-year is coded as $\sin(2\pi ni/365)$ and $\cos(2\pi ni/365)$ where i denotes the number of pairs (harmonics) of time features chosen and n in this case denotes day of a year. n takes values in the range (0, 364). Time related

features groups in our system can have at most 3 harmonics. This implies that these features can have dimensions of 2, 4, or 6, depending on the number of harmonics.

In our system, the input vector \mathbf{x} is quite large, and can have a dimension N as large as 492.

Outputs

The outputs of the system are the forecast inflow values at hourly intervals. There are several ways to make a forecast. We could forecast the inflow at some fixed number of hours in the future, measured from the current time. Such a forecast is not particularly useful considering the amount of computation and data collection involved in developing a forecasting system. A forecasting system is more useful if it forecasts a set of values. In our system we forecast the inflow levels at hourly intervals for a period of say five days.

We denote the desired output by the vector \mathbf{t} and the forecast output by the vector \mathbf{y} . The dimension of the output vector, both desired and forecast, is denoted by L . L for our system is usually 120 or more.

COMPRESSION OF INPUTS AND OUTPUTS

As we saw in the previous section, our forecasting system involves large input and output vectors. Neural nets with large input and output vectors are very difficult to train. In order to solve this dimensionality problem, we use the Karhunen-Loeve transform (KLT) ⁷ to compress both the inputs and the outputs.

Theory

The KLT is an optimal transform that has been widely used to compress signals in many applications. If \mathbf{x} is the input vector of dimension N , define \mathbf{m}_x the mean vector and the auto-covariance matrix of \mathbf{x} respectively as,

$$E[\mathbf{x}] = \mathbf{m}_x,$$

$$\mathbf{C}_x = E[(\mathbf{x} - \mathbf{m}_x)(\mathbf{x} - \mathbf{m}_x)^T]$$

Using the singular value decomposition (SVD)

$$\begin{aligned} \mathbf{C}_x &= \mathbf{U} \cdot \Sigma \cdot \mathbf{V}^T, \\ \mathbf{A} &= \mathbf{U}^T \end{aligned} \quad (2)$$

The KLT is applied as,

$$\mathbf{z} = \mathbf{A} \cdot \mathbf{x} \quad (3)$$

Consider the case where the dimensions of \mathbf{x} and \mathbf{z} are both equal to N . In this case the dimension of \mathbf{A} is $[N \times N]$, i.e., it is square. In this case,

$$\|\mathbf{x}\|^2 = \|\mathbf{z}\|^2 \quad (4)$$

For compression we use only M rows of the matrix \mathbf{A} where $M \ll N$. Therefore $\dim[\mathbf{A}] = M \times N$, and $\dim[\mathbf{z}] = M$. In this case equation (4) can be written as

$$\|\mathbf{x}\|^2 = \|\mathbf{z}\|^2 + \epsilon^2 \quad (5)$$

where ϵ^2 is the error energy due to truncation of the matrix \mathbf{A} . For the KLT, this compressed representation of \mathbf{x} is optimal in the sense that the error energy is minimum. Thus the KLT provides optimal representation of the vector \mathbf{x} with minimum loss of information ⁷.

Application of KLT to Input Data

We have already seen that the number of inputs and outputs is quite large in a forecasting system such as ours. We also saw in the last section, that the KLT provides an optimal representation of a vector if its expansion is truncated to fewer than N orthonormal basis functions. Further redundancies in the vectors allow us to apply the KLT to reduce their sizes.

Each feature group is independent of the others. Therefore we compute the KLT for each of the groups and transform them separately, and arrange them as a vector of inputs. In other words, each of the K feature groups, with an input vector \mathbf{x}_i , is transformed to a vector \mathbf{z}_i by a KLT matrix \mathbf{A}_i as shown in equation (2). Note that the dimension of the compressed vector \mathbf{z}_i is given by M_i ($\ll N_i$). The dimension of the compressed input vector, denoted by \mathbf{z} , is given by,

$$\mathbf{M} = \sum_{i=1}^K M_i \quad (6)$$

The compressed input vector \mathbf{z} , of dimension M , as we saw for the original input vector \mathbf{x} is $\mathbf{z} = \{\mathbf{z}_1^T, \mathbf{z}_2^T, \dots, \mathbf{z}_K^T\}^T = \{(\mathbf{A}_1 \mathbf{x}_1)^T, (\mathbf{A}_2 \mathbf{x}_2)^T, \dots, (\mathbf{A}_K \mathbf{x}_K)^T\}^T$.

However the time related features, which have very small dimension as we have already seen, need not be transformed. In our system we substitute the identity matrix \mathbf{I} for the KLT matrices \mathbf{A}_5 and \mathbf{A}_6 .

Reconstruction of Forecast from Compressed Output

For the desired output of the system, a similar compression is employed so that the number of outputs is also significantly reduced. The desired output vector \mathbf{t} , of dimension L , is compressed to a vector $\mathbf{T} = \mathbf{A}\mathbf{t}$, of dimension $L_o < L$. Note that the output of the neural network is the vector \mathbf{Y} , which is actually the transform of the vector \mathbf{y} . Thus to obtain the vector \mathbf{y} , we have to reconstruct it using the inverse transform,

$$\mathbf{y} = \mathbf{A}^{-1}\mathbf{Y} \quad (7)$$

The value of L_o , the dimension of the compressed output vector \mathbf{Y} , is chosen by decreasing its dimension from L until the reconstruction error energy, ϵ^2 , becomes significant. The reconstruction error energy is related to the original and transformed vector energies by Parseval's equation as,

$$\|\mathbf{t} - \mathbf{y}\|^2 = \|\mathbf{T} - \mathbf{Y}\|^2 + \epsilon^2 \quad (8)$$

Thus we obtain the forecast output in the time domain with minimum reconstruction error energy.

NEURAL NETWORKS

Network Sizing

In our system a three-layer neural net is used. The number of inputs and outputs of the neural net were determined by the number of KLT coefficients that were used during compression of the input and output vectors. The number of hidden units however could not be readily determined. An experiment was carried out where the number of hidden units (N_h) of the neural net was varied and the training error was measured when the inputs were not compressed and when they were compressed. The resultant graph is shown in figure 1.

From figure 1, we make the following observations:

1. In both domains the training error tends to oscillate, i.e., as the training error tends to increase and decrease and increase again as N_h is varied. The training error would decrease monotonically with N_h , if there were several networks designed for each value of N_h .
2. As N_h increases the oscillation in the training error tends to be damped in the sense that the oscillation is less pronounced as N_h increases. Observe that for $N_h = \{13, 15, 17\}$ The error values are almost the same.
3. The most useful observation is that as N_h increases the difference in the training errors in both the domains tend to converge to the same value. This is very useful because training using compressed data is much faster than with uncompressed data. Also we note that even though for $N_h = 5$, the errors are identical in both

domains, for higher values of N_h the training errors are much lower. This is desirable.

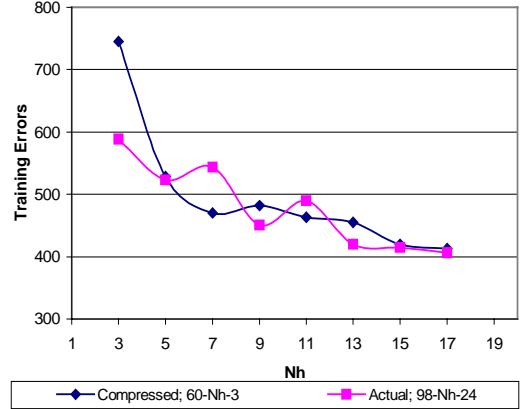


Fig. 1: Effect of training error when N_h is varied.

For all future forecast trials the number of hidden units of the neural network is set to 15 (i.e., $N_h = 15$).

Training Method

Multilayer perceptron (MLP) networks have been widely applied in the fields of pattern recognition, signal processing, and remote sensing. One of the critical problems in MLP applications has been the long training time required. The fast-training module, described in this section, has been designed to overcome this problem.

Consider a multilayer perceptron (MLP) neural network in which every unit connects to all units in previous layers. Assume that the output layer units have linear activations, so that the outputs are actually net functions. The weights and thresholds in such a network naturally divide into two sets. First we have output weights (weights and thresholds) which are those connecting to the output layer. Then we have the hidden weights, which are weights and thresholds connecting to hidden layer net functions.

In Ref. 8, a fast training method for the MLP was introduced, called output weight optimization-hidden weight optimization or OWO-HWO. In this training technique, we alternately solve linear equations for output weights and use a modified backpropagation technique called HWO to improve hidden weights. The training error for the i^{th} output unit can be written as

$$E(i) = \sum_{p=1}^{N_v} [T_{pi} - Y_{pi}]^2 \quad (9)$$

Here Y_{pi} and T_{pi} denote $Y_i(\mathbf{x})$ and the desired output net function T_i for the p^{th} training vector. N_v denotes the number

of available training patterns. For each of several random networks, find the set of output weights $w(i,k)$ which minimizes the error function. Taking the gradient of $E(i)$ with respect to the output weights and setting the gradient to zero, we obtain the linear set of equations

$$\mathbf{R} \cdot \mathbf{w} = \mathbf{T} \quad (10)$$

where the $N \times N$ matrix \mathbf{R} and the $N \times 1$ vectors \mathbf{w} and \mathbf{T} have elements $r(k,m)$, $w_o(i,k)$, and $T(m)$ respectively. Solving this set of linear equations for the output weights and using HWO for to improve the hidden weights allows us to train the neural network faster than using conventional BP⁸.

NUMERICAL RESULTS

In this section we present some results of our reservoir inflow forecasting system.

Experimental Feature Vectors

Our objective was to forecast inflow levels for a period of 5 days at hourly intervals beginning 3 days in the future from the current time. The number of outputs of the system is $L = 120$ (5 days X 24 hours). The time delay of the first output is 72 hours (3 days). For generating the training data we used data collected over a period of two years. For testing, data from the following year was used.

The feature groups used are past inflow levels, temperature, rainfall, snowfall and time-of-day. Inflow and weather information was obtained from the past 48 hours with respect to the current time. $N_i = 48$, $i = 1, 2, 3$, and 4. One time-of-day feature was used, this means that the number of time related features is $N_5 = 2$ taking into account one sine and one cosine component for time-of-day. Therefore the dimension of the input vector before compression is $N = 194$ ($48 + 48 + 48 + 48 + 2$) as given in equation (1). This large input vector was compressed as follows: inflow vector of dimension $N_1 = 48$ was compresses to a vector of dimension $M_1 = 15$, each of the weather feature groups (temperature, rainfall and snowfall) of dimension $N_i = 48$, $i = 2, 3$, and 4, was compressed to vectors of dimension $M_i = 5$. Finally, the time-of-day vector of dimension 2 is not compressed, i.e., $M_5 = 2$. The total dimension of the compressed input vector is then $M = 32$ ($15 + 5 + 5 + 5 + 2$) as given in equation (6).

On the output side, the desired inflow vector of dimension 120 was compressed to a vector of dimension $L_o = 3$. Using 3 KLT coefficients for output compression was sufficient to reconstruct the outputs in the time domain with reasonable reconstruction error energy.

Experimental Results

The data used was collected from 1993 through 1995 by Lyse Kraft, at their reservoirs in Norway. We trained the neural network using data collected for 1993 and 1994. The trained network was tested using data collected in 1995. These results are shown in the next section. The number of training patterns was $N_v = 17,520$. The number of testing patterns was $N_v = 8,760$.

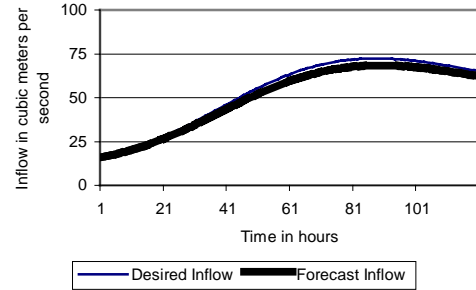


Fig. 2: 120-hour forecast made on Feb. 27, 1995

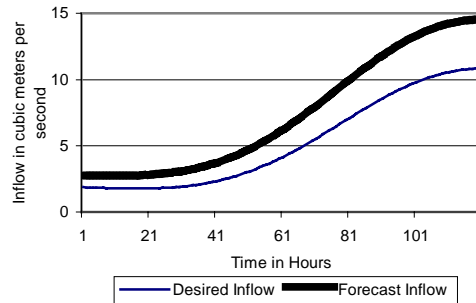
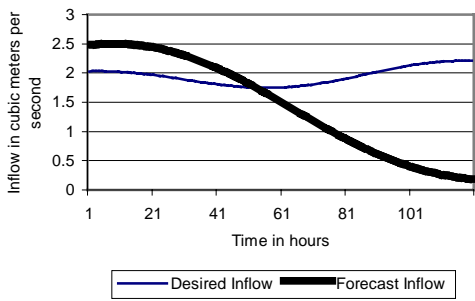


Fig. 3: 120-hour forecast made on Apr. 11, 1995.

In figures 2-4, we show 120-hour forecasts made on several different dates. All forecasts start 72 hours in the future from the current time, which is time zero. The x-axes on these figure are numbered from 1 to 120 to indicate the number of hours forecast. In all figures we see that the forecasts are continuous, and, with the exception of figure 4, follow the same shape as the desired waveform.

In figures 5-8, we show the forecast at quarterly intervals. Each sample in these plots is 82 hours in the future from the current time. These plots were obtained from the 120-hour forecasts made by the system for the year 1995.



Using the 120-hour forecasts for all of 1995, we calculated the RMS inflow error, over the 120-hour periods, as 5.092 m³/sec.

Fig. 4:120-hour forecast made on Dec. 3, 1995.

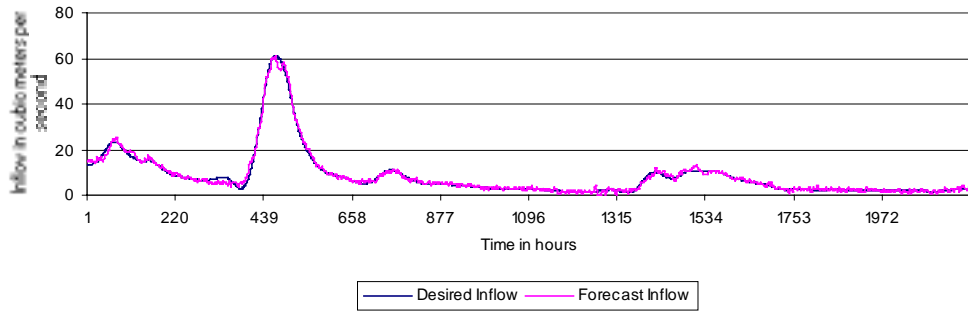


Fig. 5: First Quarter Forecast for 1995.

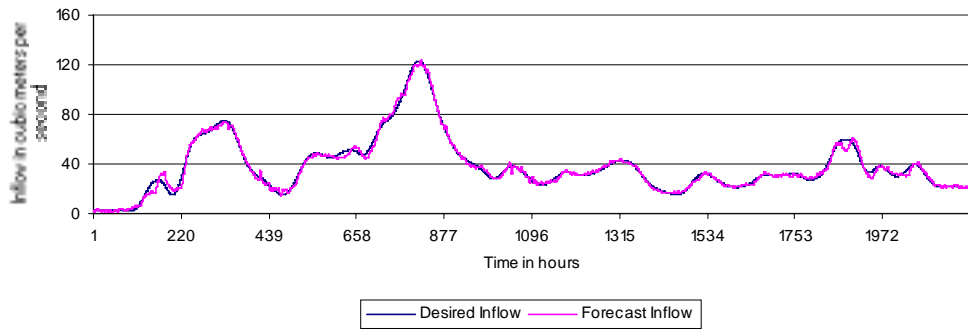


Fig. 6: Second Quarter Forecast for 1995.

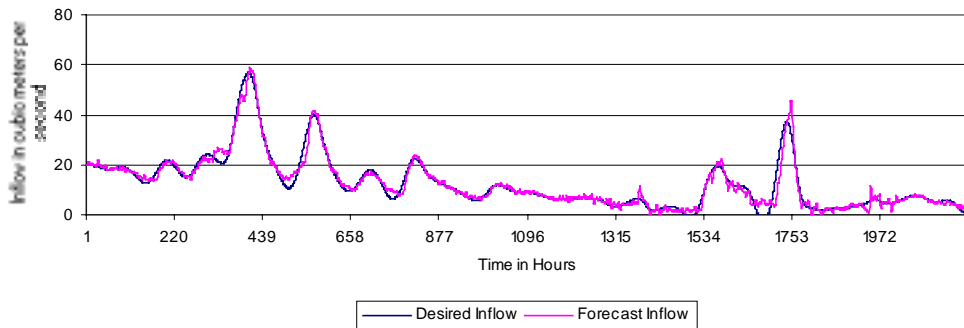


Fig. 7: Third Quarter Forecast for 1995.

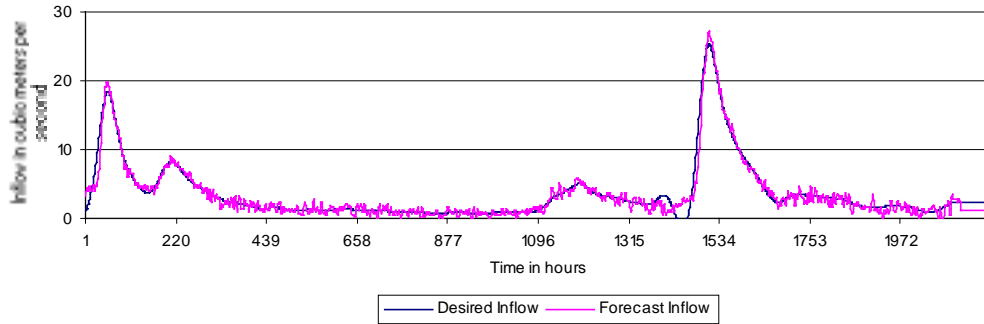


Fig. 8: Fourth Quarter Forecast for 1995.

ACKNOWLEDGEMENTS

This work was supported by the state of Texas through the Advanced Technology Program under grant number 003656-063.

REFERENCES

1. A. Khotanzad, R-C Hwang, and D. Maratukulam, "Hourly Load Forecasting by Neural Networks", Proceedings of the IEEE PES 1993 Winter Meeting, Columbus Ohio, February 1993.
2. A.D. Papalexopoulos, S. Hao, and T-M Peng, "An Implementation of a Neural Network Based Load Forecasting Model for the EMS", Proceedings of the IEEE/PES 1994 Winter Meeting, New York, New York, January 30 - February 3, 1994, pp. 1956-1962.
3. D.D. Highley and T.J. Hilmes, "Load Forecasting by ANN", IEEE Computer Applications in Power, July 1993, pp. 10-15.
4. K. Liu, S. Subbarayan, R.R. Shoults, M.T. Manry C. Kwan, F.L. Lewis, and J. Naccarino, "Comparison of Very Short-Term Load Forecasting Techniques", IEEE Power Engineering Society Summer Meeting, Portland, OR, 1995.
5. K. Liu, S. Subbarayan, R.R.Shoults, M.T.Manry C.Kwan, F.L.Lewis, and J.Naccarino, "Comparison of Very Short-Term Load Forecasting Techniques", IEEE Transactions on Power Systems, Vol. 11, No. 2, May 1996, pp. 877-882.
6. M.T. Manry, Ray Shoults, and Jorge Naccarino, "An Automated System for Developing Neural Network Short Term Load Forecasters", Proceedings of the American Power Conference, Chicago IL., 1997.
7. Douglas F. Elliot and K. Ramamohan Rao, Fast Transforms – Algorithms, Analyses, Applications, Academic Press, pp. 382-385, 1982.
8. H-H Chen, M.T. Manry, and Hema Chandrasekaran , "A Neural Network Training Algorithm utilizing Multiple Sets of Linear Equations", accepted by Neurocomputing