KMITL Sci. Tech. J. Vol. 12 No. 2 Jul.-Dec. 2012

Regional Climate Downscaling by Artificial Neural Network

Wachiraporn Permpoonsinsup and Dusadee Sukawat[*]

Department of Mathematics, Faculty of Science, King Mongkut's University of Technology Thonburi, Bangkok, Thailand

Abstract

Global models for climate change show that in the future temperature of the world is raising. However, regions of the world may experience different changes in temperature. Moreover, the regional climate changes are stronger than the global change. The main idea in this paper is to interrelate regional climate parameters to large-scale variables using an interpolation technique. Interpolation is used to downscale the output from global to regional climate models. Network is also used to train temperature data based on neural network technique. The temperature data from the National Center for Environmental Prediction (NCEP) reanalysis data are trained for temperature at Bangkok. In training phase, the error are minimized and artificial neural networks (ANNs) are adjusted for the connect weights. Accordingly, output data from the regional model are compared with observation data of the Thai Meteorological Department.

Keywords: Downscaling, Artificial Neural Network.

1. Introduction

The Intergovernmental Panel on Climate Change (IPCC) has reported that Earth is committed to continue warming over the last century [1]. Warming will not be evenly distributed around the globe. The global climate models (GCM) have been used to predict climate, however, these models are mostly designed to run at coarse resolutions. This paper applies downscaling technique to the outputs from a global climate model to generate surface temperature at Bangkok, Thailand. An artificial neural network with a multilayered approach that approximates complex mathematical functions to process data is applied to simulate a regional model [2]. Tasadduq et al., predicted the hourly mean temperature at coastal locations in Saudi Arabia using applied neural network which is trained off-line by back propagation and a batch learning scheme [3]. Dibike et al., investigated the application of temporal neural networks in a downscaling GCM outputs for the generation of daily precipitation and temperature series at the Chutedes-Passes station located in the Saguenay watershed in Canada [4]. In this paper, the Levenberg-Marquardt backpropagation network is used to train the temperature data from the National Center for Environmental Prediction (NCEP) reanalysis. In training phase, the error are minimized and artificial neural networks (ANNs) are adjusted for the connect weights. Accordingly, output data from the regional model are compared with observation data of the Thai Meteorological Department.

[*]Corresponding author: Tel: 0 2470 8221 Fax: 0 2428 4025

E-mail: dusadee.suk@kmutt.ac.th

2. Materials and Methods

Neuron is a nonlinear, parameterized function of its input variables. The neuron network is composed of the nonlinear functions of two or more neurons [5]. The main type of neural network is feedforward neural network.

2.1 Feedforward Neural Network

In 1940s, McCulloch and Pitts described the architecture of the feedforward neural networks which became the most widely used form of neural network [6]. The backpropagation is a training algorithm that is most used in the feedforward neural network. The backpropagation is a common algorithm which is used to train the architecture of feedforward neural network (Figure 1) [7].

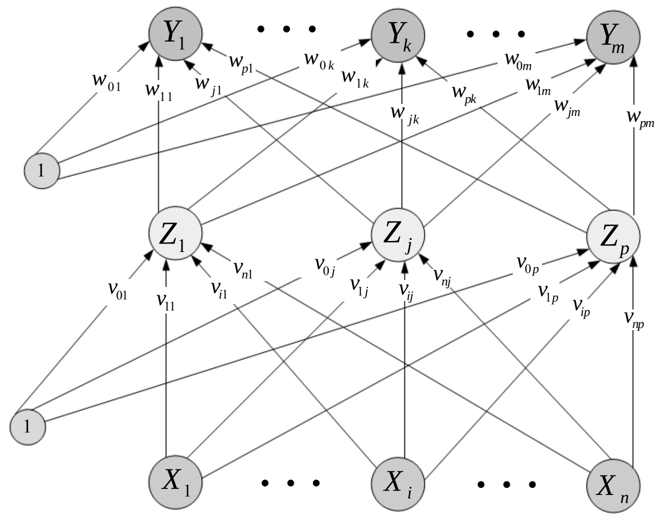

Figure 1 The architecture of feedforward neural network.

Each layer of feedforward neural network has the activation functions. They are the functions that are applied to the net output of any node before feeding to the next layer. There are typical three types of activation functions: linear, threshold or sigmoid [3]. One of the most used activation functions is the binary sigmoid function which has the range of [0,1]. It is very useful in neural networks trained by backpropogation and is defined as

$$f(x) = \frac{1}{1+\exp(-x)} \tag{1}$$

$$f'(x) = f(x)[1-f(x)] \tag{2}$$

where x is the net input.
f is the sigmoid function.

In Figure1, the structure of the feedforward neural network shows threee layers: input layer, hidden layer and output layer. The index variable, *i*, represents the input layer and x_i is the input signal for units X_i which is propagated through the connection weights $v_{i,j}$ from unit X_i to unit Z_j. The net input to hidden nodes, *j* th, is expressed by Equation (1) and the output signal of hidden node, *j* th is denoted by Equation (2)

$$net_j = \sum_{i=1}^{n} x_i v_{ij} + v_{0j} \tag{3}$$

$$z_j = f_j(net_j) \tag{4}$$

where x_i is the input signal

v_{ij} is the weight on the connection from input node, *i*th, to hidden node, *j*th

v_{0j} is a bias

p is the number of input nodes.

In the hidden layer, the input signal z_j for units Z_j is propagated through the connection weights $w_{j,k}$ from unit Z_j to unit Y_k to output layer. The net input to output nodes, *k*th, is expressed by Equation (4) and the output signal from hidden node, *jth* is denoted by Equation (6).

$$net_k = \sum_{j=1}^{p} z_j w_{jk} + \theta_k \tag{5}$$

$$y_k = f_k(net_k) \tag{6}$$

where z_j is the input signal

w_{ij} is the weight on connection from input node, *j*th, to hidden node, k*j*th

w_{0j} is a bias

p is the number of input nodes.

2.2 Levenberg-Marquardt Backpropagation Methods

The tranining of a neural network implies finding a set of weights that minimize error between the target output and the output calculated by the neural network. A backpropagation algorithm is used for the training of the neural networks. The objective of training is to reduce the error between the desired output and the neural network output and defined as [8] :

$$E = \sum_{p=1}^{P} \sum_{k=1}^{K} \left(o_{kp} - t_{kp}\right)^2 \tag{7}$$

where p is number of patterns

k is number of output

o_{kp} is network output at the k^{th} output node when applying pattern *p*

t_{kp} is target output at the k^{th} output node when applying pattern *p*.

To update weights, the update rule of Levenberg-Marquardt algorithm can be presented as [7]

$$\Delta w = -(JJ + \mu I) \cdot JE^{-1} \tag{8}$$

where J is Jacobian matrix

E is all errors

I is identity matrix

μ is learning rate.

Jacobian matrix:

$$J = \begin{bmatrix} \frac{\partial e_{1,1}}{\partial w_1} & \frac{\partial e_{1,1}}{\partial w_2} & \cdots & \frac{\partial e_{1,1}}{\partial w_N} \\ \frac{\partial e_{2,1}}{\partial w_1} & \frac{\partial e_{2,2}}{\partial w_2} & \cdots & \frac{\partial e_{2,1}}{\partial w_N} \\ & \vdots & & \\ \frac{\partial e_{p,M}}{\partial w_1} & \frac{\partial e_{p,M}}{\partial w_2} & \cdots & \frac{\partial e_{p,M}}{\partial w_N} \end{bmatrix} \tag{9}$$

The steps of Levenberg-Marquardt Backpropagation method are,

Step 1 : Input all initial input vectors, $\boldsymbol{x}$ and weight vector, $\boldsymbol{w}$ and the learning rate, μ.

Step 2 : Calculate the net input to hidden layer and the output signal to hidden node by Equation (3) and Equation (4).

Step 3 : Calculate the net input to output nodes and the output signal to output nodes by Equation (5) and Equation (6).

Step 4 : Calculate the mean square error between the target and the neural network output and calculate the Jacobian matrix by Equation (7) and Equation (9)

Step 5 : Calculate Δw as in Equation (8) that solve the Levenberg-Marquardt weight update.

Step 6 : Recalculate the error by Equation (7) and output the new error E_{k+1}

Step 7 : If the new error $E_{k+1} \leq E_{\max}$ then stop, if not check if $E_{k+1} < E_k$ then go to Step 1 and update μ by multiply with 0.1 and restore $w_k = w_{k+1}$, if not go to Step 5 and update μ by multiply with 10 [9].

2.3 Downscaling

In this study, Linear interpolation is applied to project the large-scale to small-scale climate. Linear interpolation can calculate an unknown value by a simple slope formula. Let the values of *f*(*x*) at two points, x_0 and x_1 are known, a first-degree polynomial for *x* can be written as [10],

$$p_1(x) = \left(\frac{x - x_1}{x_0 - x_1}\right) f(x_0) + \left(\frac{x - x_0}{x_1 - x_0}\right) f(x_1) \tag{10}$$

2.4 Experiment Cases

The interpolated temperature at the 850 hPa level from the GCM is fed into the neural network model. The local temperature and the feedforward neural network are used to calculate and minimize the error between temperature from the GCM and the observed data.

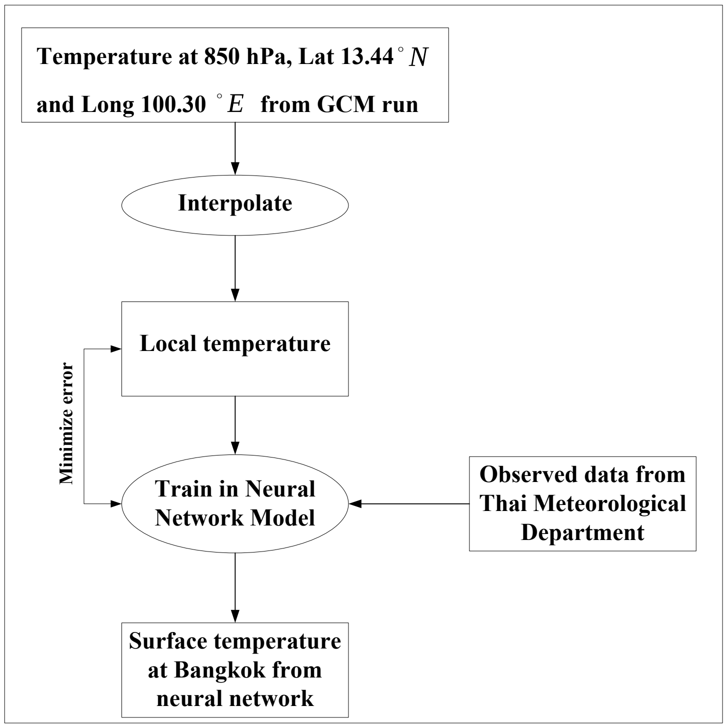
Temperature at 850 hPa, Lat 13.44 °N and Long 100.30 °E from GCM run
Interpolate
Local temperature
Minimize error
Train in Neural Network Model
Observed data from Thai Meteorological Department
Surface temperature at Bangkok from neural network

Figure 2 Downscaling step.

Table 1 shows the temperature data from the National Center for Environmental Prediction (NCEP) reanalysis at 850 hPa. In the interpolation process, the grid point of Bangkok at lat $13^{\circ} N$ and long $100^{\circ} E$ are choosen to train, test and validate in the downscaling neural network model.

Table 1 The experiment setting.

GCM grid points	lat: $12^{\circ} N$ to $14^{\circ} N$, lon: $99^{\circ} E$ to $101^{\circ} E$
Point of interest	Bangkok at 13º lat and 100º long.
NCEP data (at 0000UTC, 850hPa)	1. 1 January to 31 December, 2001 to 2003.
	2. 1 January to 31 December, 2001 to 2005.
	3. 1 March to 31 May, 2001 to 2003.
	4. 1 March to 31 May ,2001 to 2005.
Downscale grid size	Interpolate 1º lat × 1º long to 0.1ºlat × 0.1º long.

NCEP data are 1.0x1.0 degree resolution grids and available every six hours. Seventy percent the data are used for training ,15% for test and another 15% for validation in neural network model. Eventually, the results are compared with the observed data of the Thai Meteorological Department.

3. Results and Discussion

In this paper, the input data are passed through the feedforward neural network and trained by Levenberg-Marquardt backpropagation. It can minimize the error between the network output and the target. In this study, sum square error (SSE) is used to measure error of the feedforward neural network. The fitting between network output and the desired output is determined by considering the size of SSE. The regression is limited within the range [-1, 1]. The positive linear regression (R=1) means that the network output and the target vary in the same direction. The linear negative regression (R=-1) means that the network output and target output vary in opposite direction. If R=0, It means no relation between network output and target output. The learning rate μ is varied from 0.1 to 1.0. The number of neurons in each layer is shown in Table 2.

Table 2 The number of data to the feedforward neural network model

Provinces	Case	Period	Number of data	
			Input layer	Hidden layer
Bangkok	1	January 2001 – December 2003	894	447
	2	January 2001 – December 2005	1413	706
	3	Summer of 2001 – 2005 (March-June)	528	264
	4	Winter of 2001 – 2005 (October-Febuary)	414	207

Only summer and winter are selected as study cases in this paper. The rainy season is not used as study case because temperature in this season is contained by local effects that are not included in the global climate model. These results in low accuracy of forcast temperature in the global climate model. Four cases of traning data are fed into the feedforward neural network as depicted in Figures 3 to 6.

Training: R=0.90475

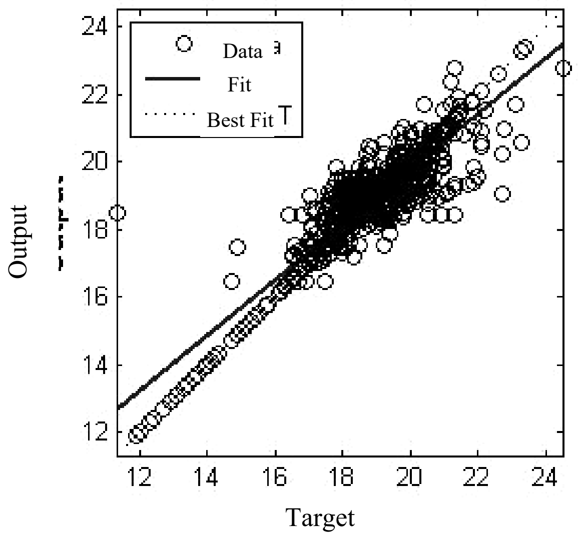

Figure 3 Output data (Fit) and observed data (data) for the 3 full-year of Bangkok.

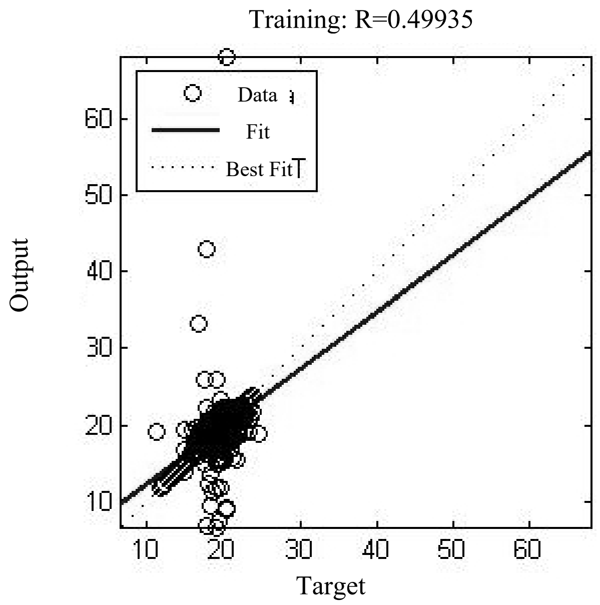

Figure 4 Output data (Fit) and observed data (data) for the 5 full-year of Bangkok.

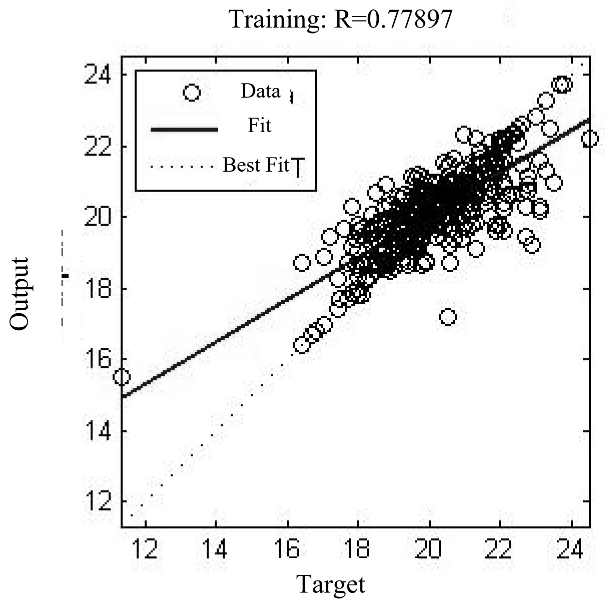

Figure 5 Output data (Fit) and observed data (data) for the 5-year in summer of Bangkok.

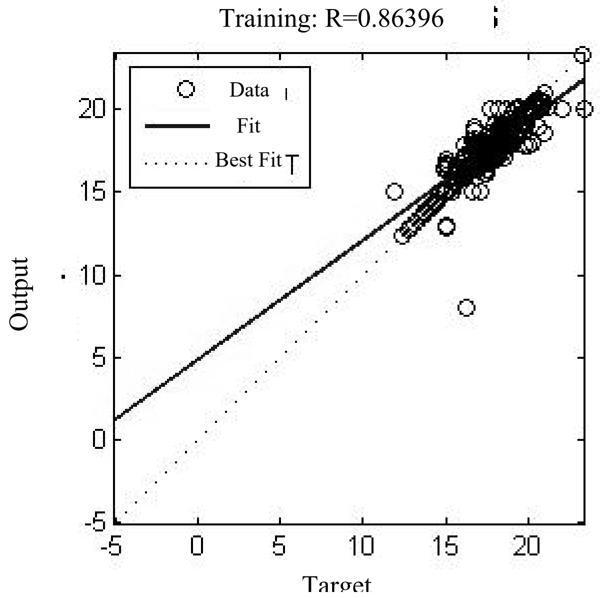

Figure 6 Output data (Fit) and observed data (data) for the 5-year in winter of Bangkok.

In the experiments, Figure 3 shows the result of Case 1 that has the training regression R= 0.90475. The regression is positive and closed to 1, it means that network output matches the target very well. For Case 2, Figure 4 shows the training regression R=0.49935. There are many differences between the network output and the target output. In Case 3, the training regression is 0.77997 in Figure 5 and Case 4 has R=0.86396 in Figure 6.

4. Conclusions

In this paper, the temperature data at Bangkok, Thailand are interpolated from GCM output. The feedforward neural network is designed for this model and the Levenberg-Marquardt backpropagation is provided for training data. There are four sets of training data that fed into the feedforward neural network model. In traning, the error are compared between model output and observed data using the sum square error (SSE). The weight-update are adjusted by the Levenberg-Marquardt algorithm. The regression of the full 3-year experiment shows that the output data are related to the observed data more than the regression of the full 5-year experiment. Moreover, the regression of the full 5-year experiment (R=0.49935, Figure 4) shows that the output data are related to the observed data less than the regression of the 5-year in summer (R=0.77997, Figure 4) and winter experiments (R=0.86396, Figure 4). This could be because the full-year experiments also include rainny seasons which have low-accuracy of forcast temperature as mentioned in Section 3. Therefore, the results show that season affects the performance of the model.

5. Acknowledgements

This paper has been supported by the Centre of Excellence in Mathematics, the Commission on Higher Education, Thailand and Department of Mathematics, King Mongkut's University of Technology Thonburi.

References

[1] Solomon, S., Qin, D., Manning, M., Chen, Z., Marquis, M., Averyt, K.B., Tignor, M. and Miller, H.L. eds, **2007**. *Climate Change 2007-The Physical Science Basis, 4th ed*. New York: Cambridge University Press.

[2] Hsu, C.C. and Chen, C.Y., **2002**. Regional load forecasting in Taiwan applications of artificial neural networks, *Energy Conversion and Management*, 44(12), 1941-1949.

[3] Tasadduq, I., Rehman, S. and Bubshaita, K., **2002**. Application of neural networks for the prediction of hourly mean surface temperatures in Saudi Arabia, *Renewable Energy*, 25(4), 545-554.

[4] Dibike, Y.B. and Coulibaly, P., **2006**. Temporal neural networks for downscaling climate variability and extremes, *Neural Networks*, 19(2), 135-144.

[5] Kumar, S., **2010**. *Neural networks*, New Delhi: Tata McGraw Hill Education Private Limited.

[6] Fausett, L.V., **1994**. *Fundamentals of Neural Networks*, New Jersey: Prentice-Hall.

[7] Dohnal, I.J., **2004**. *Using of Levenberg-Marquardt method in Identification by Neural Networks*, STUDENT EEICT.

[8] Pradeep, T., Srinivasu, P., Avadhani, P.S. and Murthy, Y.V.S., **2007**. Comparison of variable learning rate and Levenberg-Marquardt back-propagation training algorithms for detecting attacks in Intrusion Detection Systems, *International Journal on Computer Science and Engineering (UCSE)*, 3(11), 3572-3581.

[9] Hagan, M.T. and Menhaj, M.B., **1994**. Training feedforward networks with the Marquardt Algorithm, *IEEE Transactions on Neural Networks*, 5(6), 989-993.

[10] Atkinson, Han, and Weimin, **2006**. *Elementary Numerical Analysis*, 3 rd ed, New Jersey: Upper Saddle River.