Petroleum & Coal
ISSN 1337-7027

Available online at www.vurup.sk/petroleum-coal
Petroleum & Coal 57(5) 573-586, 2015

PREDICTION OF CARBON DIOXIDE WATER CONTENT USING A PARTICLE SWARM OPTIMIZATION-ARTIFICIAL NEURAL NETWORK HYBRID MODEL

M. Koolivand- Salooki[1], A. Azarmehr [*2], A. Bahadori[3], Sh. Kord[4], F. Sharifi[5], Sh. Alfkhan[6]

[1]Gas Research Division, Research Institute of Petroleum Industries, Tehran, Iran; [2]Engineering and Development Department, Bidboland Gas Refining Company, Omidiyeh, Iran; [3]Southern Cross University, School of Environment, Science and Engineering, Lismore, NSW, Australia; [4]Department of Petroleum Engineering, National Iranian South Oil Co., Ahwaz, Iran; [5]Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan, Iran;[6]Department of Chemical Engineering ,Institute of Petroleum

Received August 7, 2015, Accepted November 30, 2015

Abstract

It is well known that the simple methods are not accurate for estimating the water content of acid gases, which includes carbon dioxide, because CO_2 exhibits a minimum in the water content which cannot be predicted accurately by existing methods. In the present study, particle swarm optimization (PSO) was coupled to artificial neural network (ANN) to develop a robust hybrid training algorithm with both local and global search capabilities to predict carbon dioxide water content accurately. A database of CO_2 water content is used to train and validate the PSO-ANN hybrid model. The proposed model was used for optimization of ANN structure using particle swarm optimization (PSO). To demonstrate the capacity of the proposed model, we provide performance surface of the model. The results show that the proposed hybrid learning algorithm simulates the CO_2 water content at various conditions accurately.

Keywords: carbon dioxide; particle swarm optimization; artificial neural network; hybrid model; water content, modeling.

1. Introduction

Accurate estimation of water content of gases is very important property in many process engineering applications and design. It is a well-known fact that the phase behavior of the system carbon dioxide + water is very complex as a system because the water content of CO_2 does show a minimum. For carbon dioxide + water the landmark study is that of Wiebe and Gaddy [1]. Included in their study was the measurement of the water content of the CO2-rich phase for pressures up to 10,300 psia. Gillespie *et al.* [2] also measured phase equilibrium data for this system. In the range of interest in this study they measured the water content at two temperatures (167° and 200°F) and for pressure from 100 to about 3000 psia. Two other important studies of the water content of CO_2 are those of Coan and King [3] and King *et al.* [4]. Coan and King included CO_2 in their study of the water content of gases. Their data are for temperatures up to 212°F and for pressure less than 750 psia. King et al. measured VLE in the region near the critical point of CO2. There are a few low-pressure data for the water content of carbon dioxide in the temperature range of this study. Müller *et al.* [5] measured VLE at 212°F at pressures less than 340 psia. Zawisza and Malesinska [6] measured two aqueous dew points at 212°F. These data are useful for testing low-pressure correlations, which in turn are useful for estimating the water content of the acid gas off the amine regenerator. Another important study of the water content of carbon dioxide is that of Song and Kobayashi [7]. They measured a few points for the phase equilibria for this system, Because of the lack of precise knowledge about knowledge base precise modeling, these methods are not suitable for complex systems.

Based upon non-linearity and time-changing behaviour of these systems, artificial intelligence methods have paved a way for producing empirical models. It has been defined for modelling complicated systems.

Application of artificial intelligence methods for the identification of non-linear systems has been studied widely in many fields. Kennedy and Eberhart [8] defined PSO because of incentive by the way food obtained by birds and their searches for finding the optimum point by renewing generations [9-11]. PSO is a population based optimization method like Genetic Algorithm with some differences. Genetic Algorithm, unlike PSO, uses evolution operators like cross over and mutation. Shi and Eberhart [12] compared both these methods and showed PSO reached the optimum point faster than Genetic Algorithm, though the speed of convergence reduced in the proximity of minimum point [13].

Due to the premature convergence of PSO, other intelligent algorithms combined with Particle Swarm Optimisation in order to combat this problem. Liu and Fan [14] said that Although BPSO had a great power to search globally, enhancement needed for its searching ability. As an example, a combination of PSO with differential evolution (DE) algorithm suggested by Epitropakis *et al.* [15].

PSO enhanced greatly the neural network learning [16-17]. Zhang *et al.* [18] and Cui *et al.* [19] developed an interchangeable algorithm. In this algorithm, PSO initiated before Back Propagation used for training the network when PSO cannot make any more progress in the training process.

2. Theory

2.1 Artificial neural network(ANN)

Artificial Neural Network consists of a significant number of neurons so that it can mimick systems' behaviour very accurately. ANNs are divided to many types. They include Multilayer Perceptron, which is greatly used and normally built upon back-propagation of error algorithm training, radial basis function, adaptive linear neuron and adaptive network built upon system of fuzzy interference. Based upon data exchange through the network, ANNs are separated into feed-forward and recurrent. They are also categorised based upon their learning types, which can be supervised or self-organised [20-21].

Figure 1 shows the schematic diagram of a simple model of a neural network. In a feed-forward network, each neuron has a weighted connection to all neurons in the following layer. ANNs compute independent variables via neurons of input layers and transfer the results through neurons of output layers. The number of neurons in input and output layers are dominated by the number of independent and dependent variables respectively. There are hidden layers between input and output layers, which can have any number of neurons. They are connected to input layers which act as distribution terminals transferring their information to the neurons of hidden layers. Inputs of hidden and output layers are computed by caring out a weighted summation of outputs they get from previous layers [22-23].

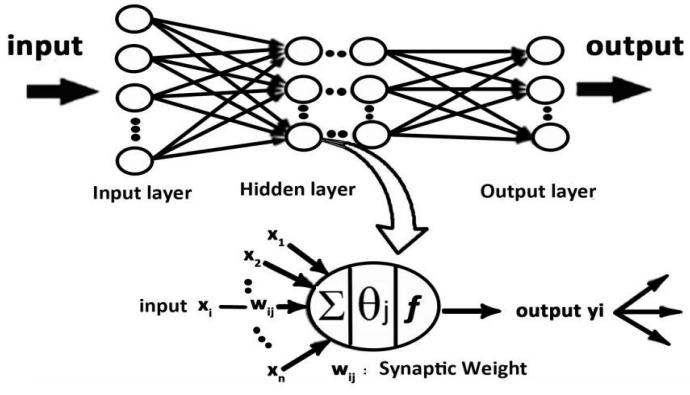

Fig 1 Architecture of the ANN model [41]

Figure 2 depicts the diagram of a training Multi-Layer Perceptron feed-forward ANN which utilised Back Propagation algorithm.

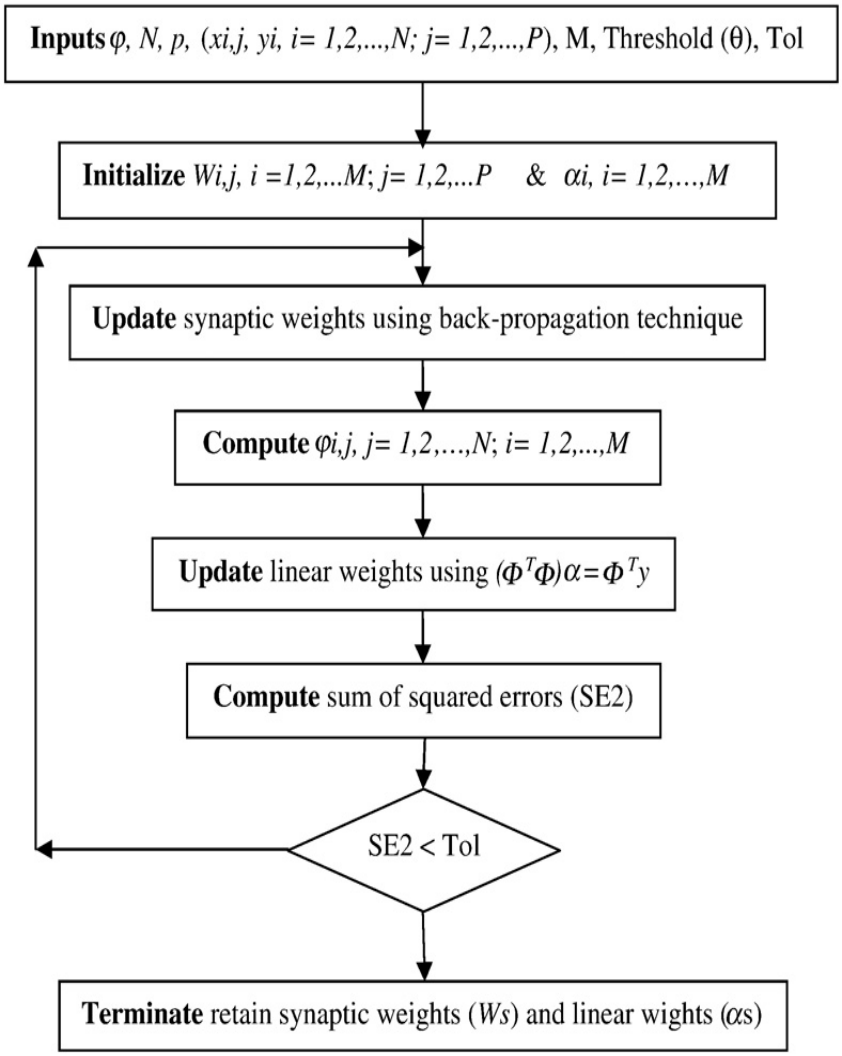

Fig 2 Learning procedure for training MLP networks [41]

2.2 Particle Swarm Optimization (PSO)

In PSO, the potential solutions, called particles are flown through the problem space by following the current optimum particles. Each particle is characterized by its position and represents a candidate solution to the problem at hand. The system is initialized with a population of random solutions and searches for optima by updating generations. In the PSO algorithm, each particle keeps track of its own position (Xid) and velocity (Vid) in the problem space. The initial position and velocity of a particle are generated randomly. During the optimization process, each particle memorizes its own best position encountered so far which is called the local best. On the other hand, the population memorizes the best position among all individual best positions obtained so far which is called the global best. The velocity and position of each particle will be changed by the particle best value and global best value. To balance between the global and local exploration capabilities of the particles. Kennedy and Spears [24] suggested the inertia weighting method, which results in lesser iterations on average to find a sufficiently optimal solution. The PSO algorithm can be described in the following steps:

At the *i*th step of the PSO algorithm four vectors are associated to the *j*th particle, j ∈ {1,…, N}:

X_j^i : the position at step i of the j-th particle;

V_j^i : the velocity at step i of the j-th particle;

P_j^i : the best position visited so far (from i=1 until step i) by the j-th particle.

g^i : is the location of the best solution that the whole swarm has achieved so far (from i=1 until step i).

Step 1: Randomly initialize the positions X_j^i and velocities V_j^i of swarm with the population size N on D dimensions.

Step 2: Compute f(x) for each particle, where f(x) is the objective function in the minimization problem and then update P_j^i and g^i according to the fitness. If the fitness of g^i is larger than a given stopping criterion or the number of iteration is larger than a given threshold, then go to

Step 3: Update the velocities according to the equation 1:

$$V_j^{i+1} = W^{i+1}.V_j^i + C1.rand() \oplus (P_j^i - X_j^i) + C2.rand() \oplus (g^i - X_j^i) \quad (1)$$

where rand() represents a random number in [0, 1], c1 and c2 are called learning factors. The symbol $\oplus$ denotes component-wise product and W^i is called inertia weight which usually adopts a linear decreasing strategy as follows (Equation 2):

$$W^i = W_{max} + \frac{W_{min\,i} - W_{max\,i}}{I} \quad (2)$$

in which i represents ith iteration, I denotes a given maximum of iteration number, $W_{max\,i}$ is the inertia weight at the beginning, and $W_{min\,i}$ is the inertia weight at the end.

Step 4: Update particle positions according to the formula below (Equation 3):

$$X_j^{i+1} = X_j^i + V_j^{i+1} \quad (3)$$

in which i represents ith iteration. Go to step 2.

Step 5: Output g^i and its corresponding fitness.

2.2.1 Inertia weight and constriction factor

From the PSO, we can see the main factors influencing on the performance are inertia weight, learning factor. The function of inertia weight is to balance global exploration and local exploittation. A larger value of W encourages global exploration (searching new areas); a smaller inertial weight promotes local exploitation. A linearly decreasing W over the search process is a good choice [12]. 'W' can be a positive constant or positive, linear or nonlinear function of time [25]. Tang and Zhao [26] suggested a strategy of self-adaptive dynamic inertia weight as per the given expression (Equation 4):

$$W1 = (W1 - W2)\frac{f - f_2}{f_1 - f_2} \quad (4)$$

Here$f$ is current objective function value, f_1 is current particle's average objective function value and f_2 is minimum objective function value and W1 and W2 are initial and final values of inertia weight. This method is effective for global and local exploration capability, dealing with multi peak functions. Some other choices of formula for inertia weight are in [27-29].

Initially in earlier experiments acceleration factors were given a constant value '2' giving an average of 'one', so that particles overfly only half the time of search. Instead of having a fixed value of acceleration factors Jie *et al.* [30] through empirical studies suggested that acceleration coefficients should not be always equal to 2 for obtaining better solutions. A time varying value of acceleration coefficients (TVAC) proved to give improved performance of particle swarm optimization after certain number of initial iterations, in order to have fine tuning and problem based tuning of particle swarm optimization [31].

3. Hybrid PSO- ANN modeling

This study used the PSO-based artificial neural network, which is characterized by good training performance, a high convergence rate, few repetitive computational steps, improved

estimation accuracy, and the ability to handle nonlinear problems, for the prediction of water content of carbon dioxide. Based on the nonlinear approximation ability of the artificial neural network, the BP neural network approach is established for estimating water content of carbon dioxide. The routine BP or other gradient algorithms are adopted, the neural network model requires more training time, and may generate local minimum value [32]. Therefore, this experiment introduced the rapid PSO algorithm into this approach for training the artificial neural network. Following this, the convergence effect was significantly improved. In this section The PSO method is adopted to determine the optimal weight of the Neural Network. Then, the optimal Neural Network is used to perform the water content of carbon dioxide forecasting. While utilizing the proposed PSO method to train the Neural Network, the positions of the particles represent the values of parameters w_{ij} in Fig 2. Each particle has its own position, velocity, and fitness value. In this paper, the fitness function is defined in term of the mean square error (MSE) of the testing data, shown in equation 5:

$$\mathrm{MSE} = \frac{\sum_{i=1}^{N}(O_i - T_i)^2}{N} \quad (5)$$

The procedure of the PSO-ANN is shown in Fig 3. The searching procedure of the proposed PSO-ANN is describe as follows;

- Preprocessing of input data and output data: Different variables may have various magnitudes, and some of them could have unmerited, but favorable, and effective on the monitored quantity. In this research, all inputs and outputs are normalized within a uniform range of [(-1) – (1)]. Without normalizing the data, the small-value neurons may have a far less influence on the network than the large-value neurons do, which also impacts network learning, there are 2100 groups of data for this research, all of them were normalized, the normalization formula is presented as shown in equation 6 [33-34]:

$$X_{norm} = \frac{(x - X_{Min})}{(X_{Max} - X_{Min})}(\lambda_2 - \lambda_1) + \lambda_1 \quad (6)$$

with the normalized source data, the output values from network training were denormalized so as to reflect the actual scale, and the denormalization formula is defined in equation7:

$$Z_{denorm} = \frac{(x - \lambda_1)}{(\lambda_2 - \lambda_1)}(z_{Max} - z_{Min}) + z_{Min} \quad (7)$$

where x is variable, X_{Max} is maximum value, X_{Min} is minimum value, λ_1 is the lower bound, λ_2 is the upper bound, z_{Max} is the maximum value in the source data and z_{Min} is the minimum value in the source data. λ_2 and λ_1 were set 1 and -1 respectively in this study.

- Set PSO-ANN parameters: there are several important parameters for artificial neural network and particle swarm optimization, including: number of hidden nodes, number of hidden layers, activation function, learning rate, PSO population.

Number of hidden layers: More layers would result in greater complexity of calculation, and a local minimum value would occur more easily. In past studies, it was generally agreed that one layer may be chosen for the general problems and two layers may be used for the more complex problems, hence the number of hidden layers was specified 1 in this study [35].

Number of hidden nodes: According to Kumar [35], the selected number of processing units of hidden layer is as follows (Equation 8):

$$\mathrm{NHN} = \frac{(N_{input} + N_{output})}{2} \quad (8)$$

NHN is the Number of hidden nodes; N_{output} is the Number of neurons in output layer and N_{input} is the Number of neurons in input layer.

But there is no such general rule that could be followed to determine the exact number of neurons in the hidden layer. Hence, the optimum number of neurons in these layers has been determined based on the experience through ''trial and error'' experiments [36-38]. The number of hidden nodes was set at between 3- 20 for the adopted approach in this study.

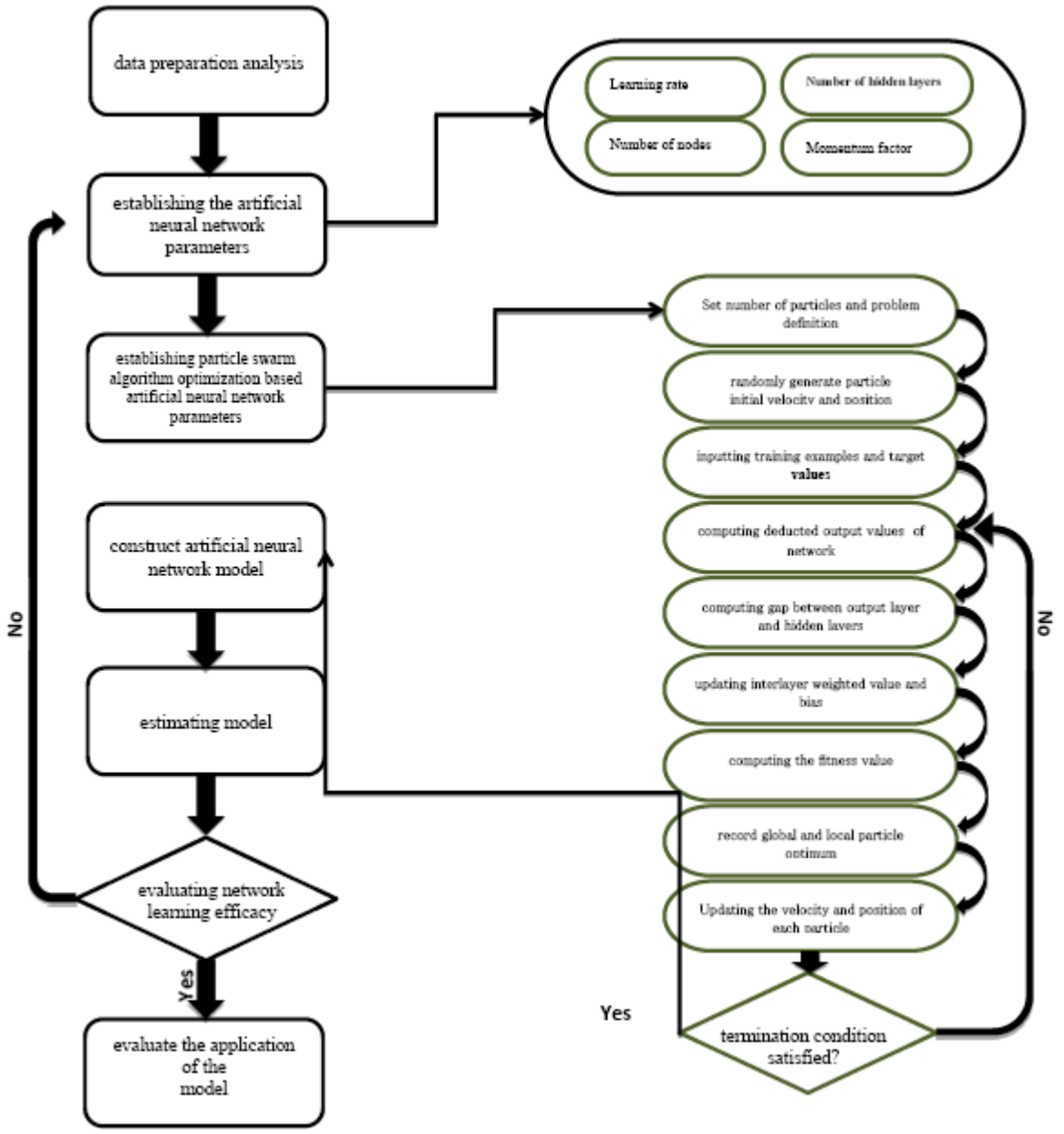

Fig 3. Execution procedures of the proposed approach.

Activation function: Sigmoid function was used as an activation function [39-40]. The output of the hidden node is calculated as following equation 9 [41]:

$$h(j) = 1/(1 + \exp(-\lambda(\sum_{n=0}^{N} w_{nj}\, x_n - \theta_j))),\ \text{for } j = 1, 2, \ldots, NHN \quad (9)$$

where w_{nj} is the weight between the nth node of input layer and jth node of hidden layer, θ_j the threshold of the hidden layer, x_n the nth input, λ the activation gain and NHN the number of node in hidden layer. The desired output is calculated as follow in equation 10:

$$f(d) = 1/(1 + \exp(-\lambda(\sum_{j=0}^{NHN} w_{jd}\, x_j - \theta_d))),\ \text{for } d = 1, 2, \ldots, D \quad (10)$$

where w_{jd} is the weight between the jth node of hidden layer and dth node of output layer, θ_d the threshold of the output layer, x_j the jth node of hidden layer, λ the activation gain and D the number of node in output layer.

Learning rate: Learning rate which is too small or too large may not be favorable for convergence. To prevent over-learning rate, which would lead to error vibration, the learning rate was between 0.1 and 1 to test the network [42].

- PSO population: Population size is generally problem dependent and generally kept in between 20 and 50 [43]. Population size was set at between 5- 120 for the adopted approach in this study.
- Initialization of particle swarms velocity and position: Set $X_j = (X_j1, X_j2, \dots, X_jD)$ as the current position for particle j; $V_j = (V_j1, V_j2, \dots, V_jD)$ as the current velocity for particle j; Hence the velocity and position in a D-dimensional space for every particle in the swarm are randomly generated in a range between (-1, 1).
- Inputting training examples and target values: Input values and target output values of the training examples input the artificial neural network.
- Computing deducted output values of network.
- Calculate hidden layer output vector H by Equation 9.
- Calculate inferred output vector F by Equation 10.

As shown in equations 11 and 12 computing gap between output layer and hidden layers and updating interlayer weighted value and bias. δ is the sum of squares of the error over all output units and M the set of trained example [44-45]:

$$\delta^n(d) = T^n(d) - f^n(d) \qquad \text{for } d = 1, 2, \dots, D; \tag{11}$$

$$\delta = \frac{1}{2}\sum_{n=1}^{M}\sum_{d=1}^{D}\delta^n(d)^2 \tag{12}$$

$T(d)$, target value of the output unit d of nth training example; $\delta(d)$, error of the dth output unit of nth training example.

➢ Fitness calculation for each particle: The particles' fitness values are measured based upon objective function defined in the problem, the fitness function value is compared with the optimum point in the memory, afterwards the velocity for the following step of search will be modified based upon particle best value stored in memory

➢ Record global and local particle best point: The particle best value is compared with the global best value. Modification of the global best value is carried out when the particle best is better than the global best. The current state will be assumed as local best if the current fitness value is better than earlier best value in memory

➢ Renewing particle' velocity and position: The updated approach is shown in Equations1 and 3 [24, 46].

➢ End the search if the condition specified for ending is reached, otherwise start again from step 5 (ending condition is considered as global optimum or maximum number of epoch)

In this step performance efficiency of the network was evaluated using the experimental values and PSO-ANN estimated values. In order to assess the fitting and prediction accuracy of models made, MSE (mean squared error), RMSE (Root Mean Squared Error),MAE (Mean Absolute Error) , SSE (sum of squared error) and R^2 (coefficient of determination) are used. For the network, the smaller the error the closer the predicted value to the actual value, and the higher R^2 the closer the output to the input value. Therefore, the R^2 and error can represent the learning quality indices of different meanings, with expressions as follows:

$$\text{RMSE} = \frac{\sqrt{\sum_{i=1}^{N}(O_i - T_i)^2}}{N} \tag{13}$$

$$\text{MAE} = \frac{\sum_{i=1}^{N}|O_i - T_i|}{N} \tag{14}$$

$$\text{SSE} = \sum_{i=1}^{N}(O_i - T_i)^2 \tag{15}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(O_i - T_i)^2}{\sum_{i=1}^{N}(O_i - T_m)^2} \tag{16a}$$

$$T_m = \frac{\sum_{i=1}^{N} O_i}{N} \quad (16b)$$

where Oiis the *i* th experimental value, T_i is the *i*th predicted value and N is the number of data.

4. Results and discussion

4.1 Framework settings for the PSO-based neural network

The total number of data's acquired at the time of this study was added up to 218. The database [1] to be introduced to PSO-ANN model was broken down randomly into two groups: training and testing. The PSO-ANN was trained using the training set data. The test set was used to evaluate the predictive ability of the PSO-ANN model. Training continued as long as the computed error between the actual and predicted outputs for the test set was decreased. Typically, about 70% of the data is used for training and the rest is categorized as testing. The test set is used to evaluate the accuracy of the newly trained PSO-ANN by providing a set of data it has never seen. During the testing, the learning is turned off and the chosen data set is fed through the PSO-ANN. The PSO-ANN outputs are collected and a report is then generated showing the testing results.

Network architecture has important influences on the predicted values. The number of input and output nodes is equal to the number of input and output data, respectively (2 and 1 in this study). However, the number of neurons in hidden layer was recognized by training several feed forward neural networks (FFNNs) of different architecture and selecting the optimal one based on minimization of MSE and improving generalization ability of the topology. In this study, the optimal topology of the ANN model developed involves two inputs, one hidden layer with 5 neurons and one output layer (2:5:1).

The values and the short description of parameters in the algorithm have been shown in Table 1. Many experiments have been done to tune these parameters.

Table 1 Description and values of the parameters in the algorithms

Parameters	Description	Considered value
w_{max}	Maximum inertia weigh	1
w_{min}	Minimum inertia weigh	0.5
C_1	Local search coefficient	2
C_2	Social search coefficient	2
T	Number of iterations	150
Population size	Number of particles	70
Number of runs	–	30

4.2 Modeling results

Efficiency and accuracy of the PSO-ANN architecture was evaluated comparing CO_2 water content and measured values with estimated PSO-ANN values. The network model was tested respectively under different particles numbers: 5, 10, 20, 35, 50 and 70 and a complete and smooth convergence was obtained under a 70 particles (refer to Fig. 5), where MSE, SSE, MAE, RMSE hit minimum and R^2 hit maximum, refer to Tables 2 and 3.

Table 2 Summary of training results for various particles no

Particles	MSE	SSE	MAE	RMSE	R
5	0.0812	13.8911	0.2196	0.2850	0.8788
20	0.0147	2.5093	0.1039	0.1211	0.9729
50	0.0090	1.5471	0.0813	0.0951	0.9839
70	0.0062	1.0559	0.0635	0.0786	0.9892
90	0.0045	0.7718	0.0538	0.0672	0.9908
120	0.0058	0.9912	0.0614	0.0761	0.9894

Table 3 Summary of testing results for various particles no

Particles	MSE	SSE	MAE	RMSE	R
5	0.0614	4.4839	0.2035	0.2478	0.9075
20	0.0131	0.9548	0.0981	0.1144	0.9804
50	0.0110	0.0813	0.0796	0.1048	0.9760
70	0.0040	0.2911	0.0489	0.0632	0.9913
90	0.0083	0.6089	0.0662	0.0913	0.9877
120	0. 0068	0.4979	0.0637	0.0826	0.9860

Table 2-3 show the comparisons of MSE, SSE, MAE, RMSE with the number of particle 5-20. The experimental results are described as follows:

1. The values of R of testing data for water content of carbon dioxide obtained from 5, 20, 50, 70, 90 and 120 particles are 0.9075, 0.9804, 0.9760, 0.9913, 0.9877 and 0.9860 respectively. We can find the PSO-ANN approach have better responses with 70 particles for water content of carbon dioxide molding.
2. The RMSEs of the testing data for water content of carbon dioxide obtained from 5, 20, 50, 70, 90 and 120 particles are 0.2478, 0.1144, 0.1048, 0.0632, 0.0913 and 0.0826 respectively. It is shown the PSO-ANN RMSE with 70 particles is superior to the others. Similarly, the responses of 70 particles on SSE, MSE, RMSE, and R^2 are better than the responses of others.

The cost percentage error between estimated cost and actual cost are presented in Fig. 4.

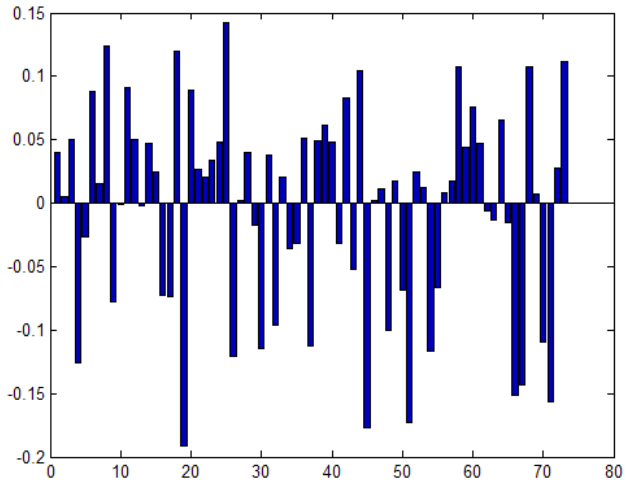

Figure 4 Percentage error between estimated value and data [1]

Fig. 4 suggests that only 18 members of data have a cost percentage error over ±1 (24% all of data), 12groups of data have a cost percentage error between ±0.5 _ ±1 (16% all of data), and 44 members of data have a cost percentage error less than ±0.5 (60% all of data) Through interviews with experts we confirmed that the errors are acceptable.

To show the improvement of ANN–PSO number of particles, comparison plots of objective function for particle numbers of 5, 10, 20, 35, 50 and 70 are presented in Fig. 5. For particle numbers of 5, 10 and 20,PSO-ANN algorithm starts its convergence at 20rd epoch and attains maximum convergence to a minimum error of 0.08, 0.015 and 0.02 respectively. The errors stay there even up to 200 epochs and more. Also the convergence of PSO-ANN algorithm in particle numbers of 50 and 70, starts at 20rd epoch but reaches its convergence to minimum error of 0.009 and 0.006 at 50th and 120th epoch respectively as shown in Fig 5.

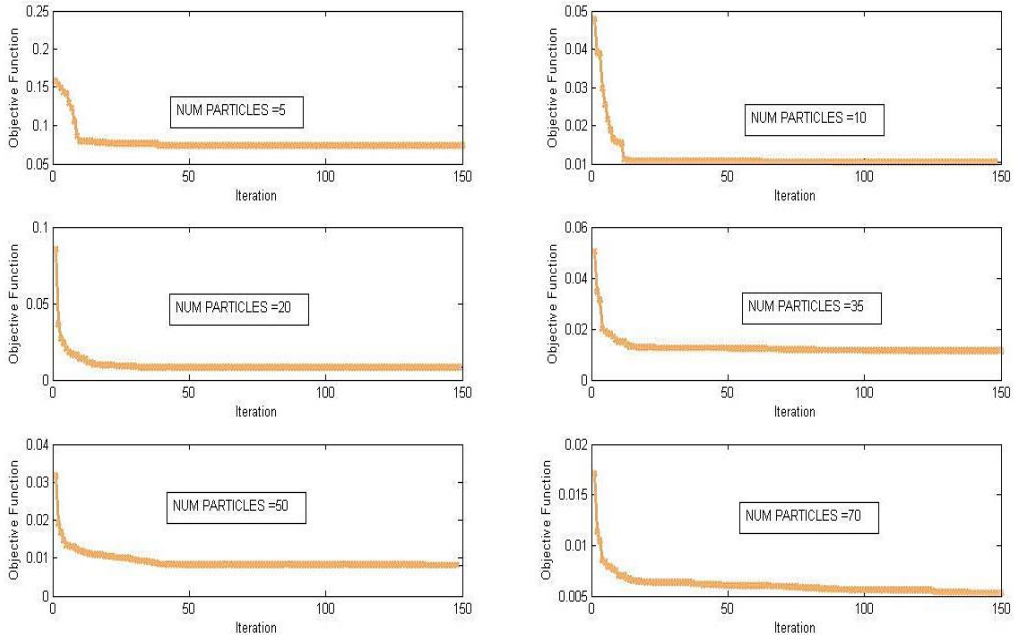

Figure 5 PSO-ANN objective function for various particles no

In Figs. 6 and 7, the CO_2 water content values estimated by PSO-ANN models were compared with their corresponding experimental values. The figures allow easily visual comparison between the predicted and actual values of them for train set and test set respectively.

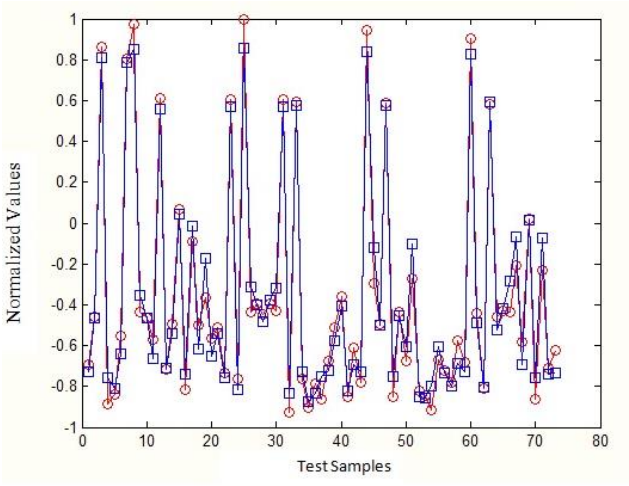

Figure 6 PSO-ANN prediction of optimized model for testing data

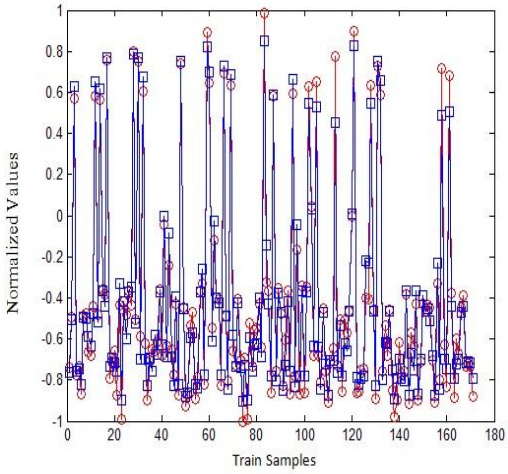

Figure 7 PSO-ANN prediction of optimized model for training data

The goal of regression analysis is to determine the values of parameters for a function that cause the function to best fit a set of data observations that is provided. In linear regression, the function is a linear (straight-line) equation. Regression analysis is widely used for prediction and forecasting, where its use has been substantial overlap with the field of machine learning. The regression in Fig. 8 and Fig. 9 is 0.9784 and 0.9785 for training and testing data respectively. That means the fitting is perfect and almost completely overlapped the target.

The quality of the PSO-ANN learning is assessed by comparing the training data and the network output, while the network's prediction accuracy is determined by comparing the testing data and the network predictions. In Figs. 8 and 9, the prediction results of the CO_2

water content trained using the existing PSO–ANN algorithm, are plotted alongside the training and testing data respectively. As indicated in these figures, the coefficient of determination was found to be 0.9784 and 0.9785 for CO_2 water content that are all well close to 1.

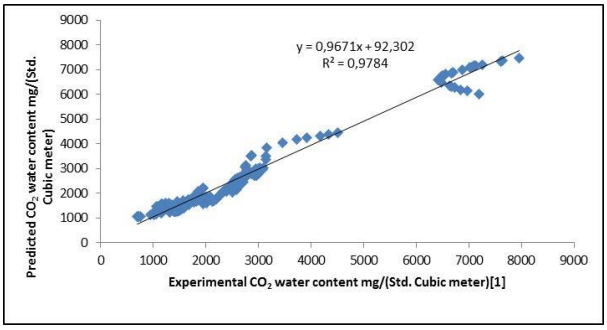

Figure 8 PSO-ANN train regression

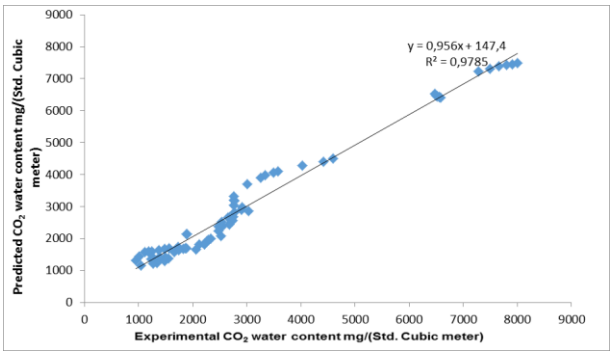

Figure 9 PSO ANN test regression

Water content PSO-ANN model is demonstrated in Fig. 10 and 11. In these figures modeled CO_2 water contentis plotted versus binary groups of temperature and pressure. As indicated in these figures.

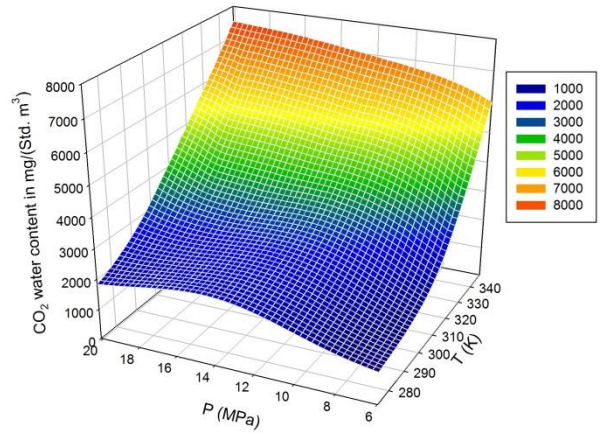

Fig. 10 Train performance of hybrid PSO-ANN model CO_2 water content in mg/(Std. Cubic meter)

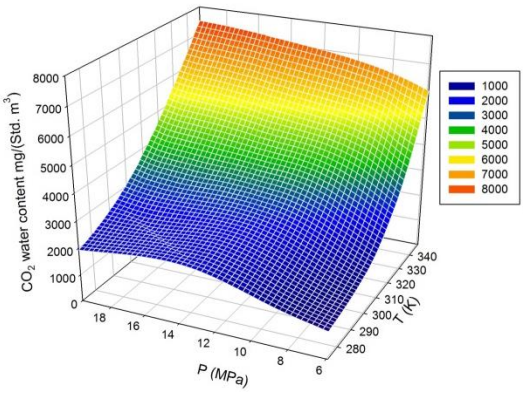

Fig.11 Test performance of hybrid PSO-ANN MODEL CO_2 water content in mg/(Std. Cubic meter)

5. Conclusion

In this study, a CO_2 water content estimating system was constructed with PSO-based neural network; the optimum parameters from particle swarm were input into the neural network to reduce problems concerned with parameter settings, and it was found that the resulting cost percentage error is within ±1%. From the experimental results we concluded that PSO-based neural network can accelerate network convergence rate. As to parameter setting for neural network, generally, optimum parameter settings are obtained through experiments, and the range of parameter is very likely to have a influence on the training results from artificial neural network, hence most problems in respect to artificial neural network are concerned with the parameter specification. In this study, optimum parameters were searched through particle swarm algorithm.

The results showed an excellent agreement between the model predictions and experimental data at all the operatingconditions considered in this investigation (R^2>0.97 in all models).

The optimization of operating conditions of the obtained modeling was performed using particle swarm optimization (PSO). The objective functions were the mean square error. The optimization of these objective functions was investigated in different PSO-ANN parameters.

The optimal operating conditions obtained with PSO-ANN were the number of particles is 70 there is a three layer for network.

References

[1] Wiebe R, Gaddy VL. Vapor Phase Composition of Carbon Dioxide-Water Mixtures at Various Temperatures and at Pressures to 700 Atmospheres.Journal of the American Chemical Society1941; 63: 475-477.

[2] Gillespie PC, Owens JL, Wilson GM. Sour Water Equilibria Extended to High Temperature and with Inserts Present. *AIChE Winter National Meeting; 1984* Mar. 11-14*;* Atlanta, GA;1984. and Gillespie PC, Wilson GM. Vapor-Liquid Equilibrium Data on Water-Substitute Gas Components: N_2-H_2O, H_2-H_2O, CO-H_2O, H_2-CO-H_2O, and H_2S-H_2O. *Research Report RR-41;* 1980; Tulsa, OK. Gas Processors Association; 1980.

[3] Coan CR, King AD. Solubility of Water in Compressed Carbon Dioxide, Nitrous Oxide, and Ethane. Evidence for Hydration of Carbon Dioxide and Nitrous Oxide in the Gas Phase. Journal of the American Chemical Society 1971; 98; 1857-1862.

[4] King MB, Mubarak A, Kim JD, and Bott TR. The Mutual Solubility's of Water with Supercritical and Liquid Carbon Dioxide. Journal of Supercritical Fluids 1992; 5; 296-302.

[5] Müller G, Bender E, Maurer G. Das Dampf-Flüssigkeitsgleichgewicht des ternären Systems Ammoniak-Kohlendioxid-Wasser bei hohen Wassergehalten im Bereich zwischen 373 und 473 Kelvin. Berichte der Bunsengesellschaft für physikalische Chemie 1988; 92; 148-160.

[6] Zawisza A, Malesinska B. Solubility of Carbon Dioxide in Liquid Water and of Water in Gaseous Carbon Dioxide in the Range 0.2-5 MPa and at Temperatures up to 473 K. Journal of Chemical & Engineering Data1981; 26; 388-391.

[7] Song KY, Kobayashi R. The Water Content of CO2-rich Fluids in Equilibrium with Liquid Water and/or Hydrates, Research Report RR-99, Gas Processors Association, Tulsa, OK; 1989.

[8] Kennedy J, Eberhart R. Particle swarm optimization. International Conference on Neural Networks1995; IV; 1942–1948.

[9] Chen GC, Yu JS. Particle swarm optimization algorithm. Information and Control 2005; 34; 318–324.

[10] Fan S, Zahara E. A hybrid simplex search and particle swarm optimization for unconstrained optimization. European Journal of Operational Research2007; 181; 527–548.

[11] Lin CJ, Hong SJ. The design of neuro-fuzzy networks using particle swarm optimization and recursive singular value decomposition. Neurocomputing2006; 2887-2893.

[12] Shi Y, Eberhart R. Parameter Selection in Particle Swarm Optimization. Seventh IEEE Annual conference on Evolutionary Programming1998; 69–73.

[13] Shi Y, Eberhart RC. Empirical study of Particle Swarm Optimization. Twelfth IEEE International Conference on Artificial Intelligence (IJCA)1999; 1945–1950.

[14] Liu J, Fan X. The analysis and improvement of binary particle swarm optimization. International Conference on Computer Security2009; 254-258.

[15] Epitropakis MG, Plagianakos VP, Vrahatis MN. Evolving cognitive and social experience in particle swarm optimization through differential evolution. IEEE Transactions on Evolutionary Computation2010; 1-8.

[16] Qasem SN, Shamsuddin SM. Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. Applied Soft Computing2011; 11; 1427–1438.

[17] Qasem SN, Shamsuddin SM, Hashim SZM, Darus M, Al-Shammari E. Memetic multi objective particle swarm optimization-based radial basis function network for classification problems. Information Sciences2013; 239; 165–190.

[18] Zhang JR, Zhang J, Lok TM, Lyu MR. A hybrid particle swarm optimization back propagation algorithm for feed forward neural network training, Applied Mathematics and Computation2007; 185; 1026–1037.

[19] Cui Y, Ju SG, Han F, Gu TY. An improved approach combining random PSO with BP for feed forward neural networks. Artificial Intelligence and Computational Intelligence2009; 5855; 361–368.
[20] Ashoori S, Abedini A, Abedini R, QorbaniNasheghi Kh. Comparison of scaling equation with neural network model for prediction of asphaltene precipitation. Journal of Petroleum Science and Engineering2010; 72; 186-194.
[21] Bakhshi R, Utojo U. Unification of neural and statistical modeling methods that combine inputs by linear projection. Computers and Chemical Engineering1998; 22; 1859-1878.
[22] Darwish NA, Hilal N. Sensitivity analysis and faults diagnosis using artificial neural networks in natural gas TEG-dehydration plants. Chemical Engineering Journal2008; 137; 189-197.
[23] Mohammadi MA, Soleimani R, Bahadori A. A computational intelligence scheme for prediction equilibrium water dew point of natural gas in TEG dehydration systems. Fuel2014; 137; 145-154.
[24] Kennedy J, Spears MW. Matching algorithms to problems: An experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. IEEE Evolutionary Computation1998; 19; 78–83.
[25] Shi Y, Eberhart R. A modified particle swarm optimizer, IEEE Evolutionary Computation 1998; 69–73.
[26] Tang J, Zhao X. A fine tuning hybrid particle swarm optimization algorithm. International Conference on Biomedical Engineering2009; 296-299.
[27] Houz X. Wiener model identification based on adaptive particle swarm optimization. IEEE Machine Learning and Cybernetics2008; 1041–1045.
[28] Zhang X, Wen S, Li H. A novel particle swarm optimization with self-adaptive inertia weight. Chinese Control Conference2005; 1373–1376.
[29] Chao X, Duo Z. An adoptive particle swarm optimization algorithm with dynamic nonlinear inertia weight variation. *Int Conf Enhan Promot Comput Methods Eng Sci Mech* 2006; 672–676.
[30] Jie X, Deyun X. New metropolis coefficients of particle swarm optimization. *IEEE CCDC* 2008; 3518-3521.
[31] Ratanweera A, Halgamuge SK, Watson HC. Self organizing hierarchical particle swarm optimizer with time varying acceleration coefficients. IEEE T Evolutionary Computation2004; 240–252.
[32] Guo LX, Deng CH. Prediction model for dissolved oxygen in fish pond based on fuzzy neural network. Journal of Fishery Sciences of China2006; 30; 225–229.
[33] Hassoun MH. Fundamentals of artificial neural networks. Cambridge, MA, London: MIT Press; 1995.
[34] Masters T. Practical neural network recipes in C++. New York: Morgan Kaufmann Publishers; 1993.
[35] Kumar S. Neural networks, *New York: McGraw-Hill; 2005, p.* 186–193.
[36] Bashir ZA, El-Hawary ME. Applying wavelets to short-term load forecasting using PSO-based neural networks. IEEE Transactions on Power Systems2009; 24; 20–27.
[37] Khotanzad A, Reza AR, Lu TL, Abaye A, Davis M, Maratukulam DJ. A neural-network-based electric load forecasting system. IEEE Transactions on Neural Networks1997; 8; 835–846.
[38] Yang Che, Cheng KW. Genetic algorithm-based RBF neural network load forecasting model. IEEE Power and Energy Society2007; 1–6.
[39] Chiang WK, Zhang D, Zhou L. Predicting and explaining patronage behavior toward web and traditional stores using neural networks: A comparative analysis with logistic regression. Decision Support Systems2006; 41; 514–531.
[40] Tabach EE, Lancelot L, Shahrour I, Najjar Y. Use of artificial network simulation metamodeling to assess groundwater contamination in a road project. Mathematical and Computer Modelling2007; 45; 766–776.

[41] Carlisle A, Dozier G. Adapting particle swarm optimization to dynamic environments. International Conference on Artificial Intelligence2000; 429–434.
[42] Fu L. Neural networks in computer intelligence. New York: McGraw-Hill; 1995.
[43] Eberhart R. A new optimizer using particle swarm theory. IEEE IntSymp Micro Mach HumanSci1995; 39–44.
[44] Bose NK, Liang P. Neural Network Fundamentals with Graphs, Algorithms, and Applications. McGraw-Hill Series in Electrical and Computer Engineering; 1996.
[45] Krose B, Van der Smart P. An Introduction to Neural Network. University of Amsterdam; 1996.
[46] Huang CM, Huang CJ, Wang ML. A particle swarm optimization to identifying the ARMAX model for short-term load forecasting. IEEE Transaction on Power Systems 2005; 20;1126–1133.

F.Sharifi (mgasengineer@gmail.com), Sh. Alfkhani (Alfkhani@ut.ac.ir), A. Azarmehr (abbas.azarmehr.put@gmail.com), M. Koolivand-Salooki (mehdi.koolivand@gmail.com), A. Bahadori (bahadori.al@gmail.com), Sh. Kord (Sh.kord@gmail.com)*