A Comparison of Artificial Neural Networks Algorithms for Short Term Load Forecasting in Greek Intercontinental Power System

G.J. TSEKOURAS[1], F.D. KANELLOS[2], V.T. KONTARGYRI[2], C.D. TSIREKIS[2], I.S. KARANASIOU[2], CH. N. ELIAS[2], A.D. SALIS[2], N. E. MASTORAKIS[1]

[1]Department of Electrical & Computer Science, Hellenic Naval Academy, Terma Hatzikyriakou, Piraeus

[2]School of Electrical and Computer Engineering, National Technical University of Athens, 9 Heroon Polytechniou Street, Zografou, Athens

GREECE

Abstract: - The objective of this paper is to compare the performance of different Artificial Neural Network (ANN) training algorithms regarding the prediction of the hourly load demand of the next day in intercontinental Greek power system. These techniques are: (a) stochastic training process and (b) batch process with (i) constant learning rate, (ii) decreasing functions of learning rate and momentum term, (iii) adaptive rules of learning rate and momentum term, (c) conjugate gradient algorithm with (i) Fletcher-Reeves equation, (ii) Fletcher-Reeves equation and Powell-Beale restart, (iii) Polak-Ribiere equation, (iv) Polak-Ribiere equation and Powell-Beale restart, (d) scaled conjugate gradient algorithm, (e) resilient algorithm, (f) quasi-Newton algorithm, (g) Levenberg-Marquardt algorithm. Three types of input variables are used as inputs: (a) historical loads, (b) weather related inputs, (c) hour and day indicators. The training set is consisted of the actual historical data from three past years of the Greek power system. For each ANN training algorithm a calibration process is conducted regarding the crucial parameters values, such as the number of neurons, etc. The performance of each algorithm is evaluated by the Mean Absolute Percentage Error (MAPE) between the experimental and estimated values of the hourly load demand of the next day for the evaluation set in order to specify the ANN with the smallest value. Finally the load demand for the next day of the test set (with the historical data of the current year) is estimated using the best ANN of each training algorithm, so that the verification of behaviour of ANN load prediction techniques should be demonstrated.

Key-Words: - artificial neural networks, short-term load forecasting, ANN training back-propagation algorithms

1 Introduction

In a deregulated electrical energy market, the load demand has to be predicted with the highest possible precision in different time periods: very short-term (for the next few minutes), short-term (for the next few hours to a week), midterm (for the next few weeks to few months) and long-term (for the next few months to years). Especially, short load forecasting is very crucial problem, because its accuracy effects to other operational issues of power systems, such as unit commitment [1], scheduling of spinning reserve [2], available transfer capability [3], system stability [3], application of load demand programs, etc. Accurate forecasting leads to higher reliability and lower operational costs for power systems.

Several forecasting methods have been implemented for short-term load forecasting with different levels of success, such as ARMAX models [4], regression [5], ANNs [6], fuzzy logic [7], expert systems etc. Despite the diversity of methods used, ANNs are the most common and effective [6].

Especially, in Greece, ANNs have been used successfully either for the intercontinental power system [8-10], or autonomous big islands [8, 11-12]. Some techniques belong to classical ANNs [9-10] or specialized ones (i.e. parallel implementation of recurrent ANN and zero-order regulation radial basis networks [11]) or they are combined with fuzzy logic algorithms [12].

In this paper the comparison of 14 different basic training ANN algorithms is carried out based on the basic structure of the ANN proposed by Kiartzis et al [8-9] for the inputs and outputs neurons for the Greek intercontinental power system. The main goals are:

- the modulation of the internal neural network structure (number of neurons of hidden layer, initial value of learning rate, etc.) for each different training algorithm with respect to the best *Mean Absolute Percentage Error* (MAPE) of the evaluation set,
- the comparison of the respective algorithms according to MAPE and computational time and

- the suggestion of the best training algorithm for this case study.

The respective results are based on actual hour load data of the Greek intercontinental power system for years 1997-2000.

2 Proposed ANN Methodology for Each Training Algorithm for Short-term Load Forecasting

The short-term load forecasting is achieved by applying an ANN methodology through the proper selection of the parameters for each training back-propagation algorithm. This methodology has the following basic steps and its flow chart is shown in Figure 1.

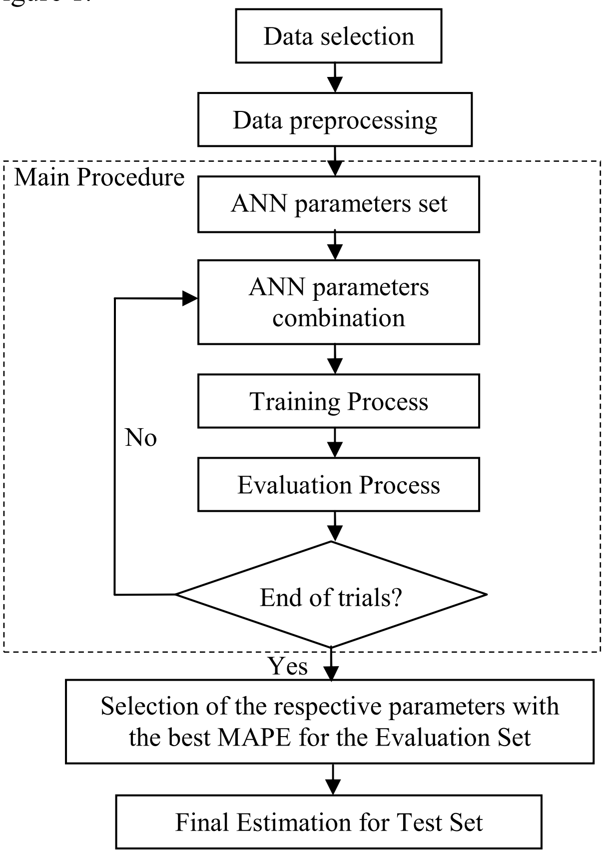

Fig. 1. Flowchart of the ANN methodology for the proper selection of ANN parameters per training algorithm for short-term load forecasting

(a) *Data selection*: The input variables for load forecasting of d-th day are the following according to Kiartzis et al [9]:

- the hourly actual loads of the two previous days: L(d-1,1),…, L(d-1,24), L(d-2,1),…, L(d-2,24) (in MW),
- the maximum mean temperature per three hours and the minimum mean temperature per three hours for Athens for the current day and the last day max_$temp_{\mathrm{Ath}}$(d), min_$temp_{\mathrm{Ath}}$(d), max_$temp_{\mathrm{Ath}}$(d-1), min_$temp_{\mathrm{Ath}}$(d-1) respectively (in ^{o}C),
- the maximum mean temperature per three hours and the minimum mean temperature per three hours for Thessalonica for the current day and the last day max_$temp_{\mathrm{Th}}$(d), min_$temp_{\mathrm{Th}}$(d), max_$temp_{\mathrm{Th}}$(d-1), min_$temp_{\mathrm{Th}}$(d-1) respectively (in ^{o}C),
- the temperature difference between the maximum mean temperature per three hours of the current day and the respective one of the last day for Athens dif_$temp_{\mathrm{Ath}}$ and Thessalonica dif_$temp_{\mathrm{Th}}$ respectively:

$$dif\_temp_{Ath} = max\_temp_{Ath}(d) - max\_temp_{Ath}(d-1) \quad (1)$$

$$dif\_temp_{Th} = max\_temp_{Th}(d) - max\_temp_{Th}(d-1) \quad (2)$$

- the temperature dispersion from comfortable living conditions temperature for Athens for the current day $T^2_{Ath}(d)$ and the previous day $T^2_{Ath}(d-1)$, where:

$$T^2_{dispersion} = \begin{cases} (T_c - T)^2, & T < T_c \\ 0, & T_c < T < T_h \\ (T - T_h)^2, & T_h < T \end{cases} \quad (3)$$

where T_c=18 ^{o}C, T_h=25 ^{o}C.

- the temperature dispersion from comfortable living conditions temperature for Thessalonica for the current day $T^2_{Th}(d)$ and the previous day $T^2_{Th}(d-1)$,
- seven digit numbers, which express the kind of the week day, where Monday corresponds to 1000000, Tuesday to 0100000, etc,
- two sinusoidal functions ($\cos(2\pi d/T)$, $\sin(2\pi d/T)$), which express the seasonal behavior of the current day, where T is the number of the days of the current year.

The output variables are the 24 hourly actual load demand of the current day $\hat{L}$(d,1),…, $\hat{L}$ (d,24).

(b) *Data preprocessing*: Data are examined for normality, in order to modify or delete the values that are obviously wrong (noise suppression). Due to the great non linearity of the problem, non linear activation functions are preferably used. In that case, saturation problems may occur. These problems can be attributed to the use of sigmoid activation functions that present non-linear behavior outside the region [-1, 1]. In order to avoid saturation problems, the input and the output values are normalized as shown by the following expression:

$$\hat{x} = a + \frac{b-a}{x_{\max} - x_{\min}}\left(x - x_{\min}\right) \quad (4)$$

where $\hat{x}$ is the normalized value for variable x, $x_{\min}$ and $x_{\max}$ are the lower and the upper values of variable x, a and b are the respective values of the normalized variable.

(c) *Main procedure*: For each ANN training algorithm the respective parameters of the neural network are selected through a set of trials. Specifically for each ANN parameter (such as the neurons of the hidden layer) the training algorithm is separately executed for the respective range of values (i.e. 20 to 70 neurons with step 1) based on the error function (sum of the square of errors for all neurons per epoch) for training set and the regions with satisfactory results (minimum *MAPE* for evaluation set) are identified. Following, the training algorithm is repeatedly executed, while all parameters are simultaneously adjusted into their respective regions, so that the combination with the smallest *MAPE* for the evaluation set is selected. It is noted that the *MAPE* index between the measured and the estimated values of hourly load demand for the evaluation set' days is given by:

$$MAPE_{ev} = 100\% \cdot \frac{1}{m_{ev}} \cdot \sum_{d=1}^{m_{ev}} \sum_{i=1}^{24} \frac{\left| \hat{L}(d,i) - L(d,i) \right|}{L(d,i)} \quad (5)$$

where $L(d,i)$ is the measured value of load demand for the i-th hour of d-th day for the evaluation set, $\hat{L}(d,i)$ the respective estimated value, m_{ev} the population of the evaluation set. This index is a practical measure, which reflects the approximation of the actual load demand independently from its units.

(d) *Final estimation for the test set*: The actual load demand (in MW) for the days of the test set is finally estimated by using the respective ANN parameters of the current training algorithm.

This process is repeated for 14 different training algorithms, which are synopsized in Table 1. A short description of all training algorithms is presented in [13], while more analytical representations can be found in references of Table 1 [14-23]. The basic steps of the back-propagation algorithm have been described in several textbooks [24-25]. According to Kolmogorov's theorem [24], an ANN can solve a problem by using one hidden layer, provided it the proper number of neurons. Under these circumstances one hidden layer is used, but the number of neurons must properly be selected properly. It is also mentioned that there are several parameters to be selected, depending on the variation of the back-propagation algorithm that is being used each time in order to train the ANN. The parameters that are common in all algorithms are:

- the number of neurons N_n in the hidden layer,
- the type of the activation functions (hyperbolic tangent, logistic, linear),
- the parameters (a, b) of the activation functions, i.e. $\phi(x) = \tan(a \cdot x + b)$ for hyperbolic tangent and
- the maximum number of epochs (max_epochs).

For methods 1-6 (see Table 1) the additional parameters are:

- the time parameter and the initial value of the learning rate T_η , η_0 ,
- the time parameter and the initial value of the momentum term T_a , a_0 (not for methods 3, 4).

For methods 7-10 (see Table 1) the additional parameters are the initial value s, the number of iterations T_{bn} for the step of the basic vector's calculation, the number of iterations T_{trix} for the tri-section step using the golden method for minimizing the error function.

TABLE 1
DESCRIPTION OF ANN'S TRAINING ALGORITHMS

No.	Description of ANN's training algorithms
1	Stochastic training with learning rate and momentum term (decreasing exponential functions) [14]
2	Stochastic training, use of adaptive rules for the learning rate and the momentum term [14]
3	Stochastic training, constant learning rate [14]
4	Batch mode, constant learning rate [14]
5	Batch mode with learning rate and momentum term (decreasing exponential functions) [14]
6	Batch mode, use of adaptive rules for the learning rate and the momentum term [14]
7	Batch mode, conjugate gradient algorithm with Fletcher-Reeves equation [15-16]
8	Batch mode, conjugate gradient algorithm with Fletcher-Reeves equation & Powell-Beale restart [15-17]
9	Batch mode, conjugate gradient algorithm with Polak-Ribiere equation [15, 18]
10	Batch mode, conjugate gradient algorithm with Polak-Ribiere equation & Powell-Beale restart [15, 17-18]
11	Batch mode, scaled conjugate gradient algorithm [19]
12	Batch mode, resilient algorithm [20]
13	Batch mode, quasi-Newton algorithm [21]
14	Batch mode, Levenberg-Marquardt algorithm [22-23]

For method 11 the additional parameters are σ and λ_0, for method 12 the increasing δ_1 and the decreasing δ_2 factor of change in the value of the weights between two successive epochs and for method 14 the initial factor $\lambda(0)$ and the multiplicative parameter β respectively.

It is also noted that during the training process for each ANN three stopping criterions are used [25]:

- weights stabilization (smaller than $\ell imit_1$),
- the respective error function not to be improved (the variation between two epochs should be smaller than $\ell imit_2$) or
- the maximum number of epochs to be exceeded (bigger than $\max\_epochs$).

In each training algorithm, the error function is the root mean square error $RMSE_{tr}$ for the training set according to:

$$RMSE_{tr} = \sqrt{\frac{1}{m_1 \cdot q_{out}} \sum_{m=1}^{m_1} \sum_{k=1}^{q_{out}} e_k^2(m)} \qquad (6)$$

where q_{out} is the number of neurons of the output layer, $e_k(m)$ is the error of the *k-th* output neuron for the *m-th* pattern of the training set. If one of the three criteria is satisfied, the main core of back propagation algorithm finishes. Otherwise, the number of epochs is increased by one and the feed forward and reverse pass calculations are repeated.

Afterwards, the results of all ANN training algorithms with the respective optimized parameters are compared, in order to choose the one leading to the smallest *MAPE* index within a logical computational time.

3 Application for Stochastic Training Algorithm with Decreasing Exponential Functions for Learning Rate and Momentum Term

Following, the aforementioned method is applied for the short-term load forecasting in Greek intercontinental power system. The training and the evaluation sets consist of the 90% and 10% of the normal days (no holidays) from the years 1997-1999 respectively, while the respective test set consists of the normal days from the year 2000. The input vector $\vec{x}_{in}(n)$ is formed with the 71 input variables of section 2, where the load and the temperature data are normalized, while the output vector $\vec{t}(n)$ is formed by the normalized 24 output actual load demand of the day under prediction.

There are several crucial ANN parameters to be selected, such as:

- the number of the neurons of the hidden layer, which ranges from 20 to 70 with incremental step 1,
- the initial value $\eta_0 = \eta(0)$ and the time parameter T_η of the training rate, which get values from the sets {0.1, 0.2, …, 0.9} and {1000, 1200,…, 2000} respectively,
- the initial value $a_0 = a(0)$ and the time parameter T_a of the momentum term, which get values from the sets {0.1,0.2,…,0.9} and {1000, 1200,…, 2000} respectively,
- the type and the parameters of the activation functions of the hidden and the output layers, where the type can be *hyperbolic tangent*, *linear or logistic*, while the a_1, a_2 parameters get values from the set {0.1,0.2,…, 0.5} and b_1, b_2 from the set $\{0.0, \pm 0.1, \pm 0.2\}$.

The parameters of the stopping criteria are defined after a few trials as $\max\_epochs$=10000, $\ell imit_1$=10^{-5}, $\ell imit_2$=10^{-5}.

The development of the abovementioned method in Visual Fortran 6.0 gives the capability to realize all possible combinations of the values of the crucial parameters. In this study the respective combinations account to 836,527,500, which practically can not be examined. This forced the authors to apply the proposed calibration process gradually through consecutive steps in order to determine the values of the ANN's parameters.

As a first step, the number of neurons varies from 20 to 70, while the remaining parameters are assigned with fixed values ($a_0 = 0.4$, $T_a = 1800$, $\eta_0 = 0.5$, $T_\eta = 2000$, activation functions in both layers: hyperbolic tangent, a_1=a_2=0.25, b_1=b_2=0.0). In Fig. 2 the *MAPE* indexes for the training, the evaluation and the test set are presented. The *MAPE* indexes of the evaluation and the test set keep step with the respective one of the training set, even if the following relationship is valid for every neuron (from 20 to 70):

$$MAPE_{training} < MAPE_{evaluation} < MAPE_{test} \qquad (7)$$

With the neurons numbered from 20 to 45 the *MAPE* index for the evaluation set has small values (the smallest is for 45), while for bigger values it rapidly increases.

As a second step the initial value η_0 and the time parameter T_η of the training rate change simultaneously in the respective regions, while the other parameters remain constant (neurons=45, $a_0 = 0.5$, $T_a = 1800$, activation functions in hidden

& output layers: hyperbolic tangent, a_1=a_2=0.25, b_1=b_2=0.0). In Fig. 3 the results of the *MAPE* index for the evaluation set are satisfactory for $0.5 \leq \eta_0 \leq 0.8$ and $1000 \leq T_\eta \leq 1400$. The best result is obtained for $\eta_0 = 0.5$, $T_\eta = 2000$. It is mentioned that *MAPE* increases dramatically for $\eta_0 \geq 0.7$ and $T_\eta \geq 1600$.

As a third step the initial value a_0 and the time parameter T_a of the momentum term change simultaneously in the respective regions, while the other parameters are constant (neurons=45, $\eta_0 = 0.5$, $T_\eta = 2000$, activation functions in hidden & output layers: hyperbolic tangent, a_1=a_2=0.25, b_1=b_2=0.0). The results of the *MAPE* index for the evaluation set are satisfactory for $a_0 \geq 0.6$ and $T_a \geq 1600$, while the best result is obtained for $a_0 = 0.9$, $T_a = 2000$. It is mentioned that *MAPE* increases dramatically for $a_0 \leq 0.5$.

Similarly it is found that the ANN gives better results using as a hyperbolic tangent activation function in both layers with parameters $a_1 = a_2$=0.25 and $b_1 = b_2$=0.0. It is mentioned that in Table 2 the results for different activation functions are registered.

The final calibration of the ANN model is realized for 40 to 50 neurons, $a_0 = 0.8 - 0.9$, T_a = 1800-2000-2200, $\eta_0 = 0.5 - 0.6$, T_η = 1000-1200-1400, activation functions in both layers: hyperbolic tangent with parameters $a_1 = a_2$=0.20-0.25-0.30, $b_1 = b_2$=0.

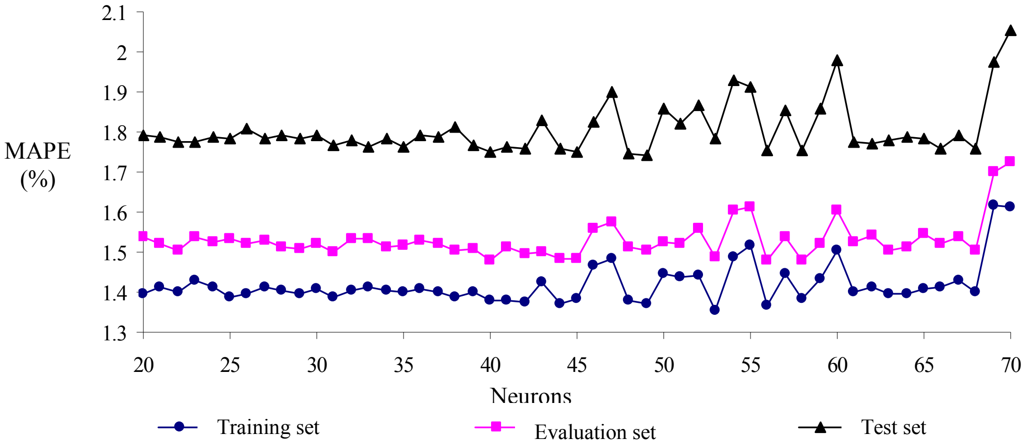

Fig. 2. *MAPE*(%) index for the all sets, neurons: 20-70, $a_0 = 0.4$, $T_a = 1800$, $\eta_0 = 0.5$, $T_\eta = 2000$, activation functions in both layers: hyperbolic tangent, a_1=a_2=0.25, b_1=b_2=0.0

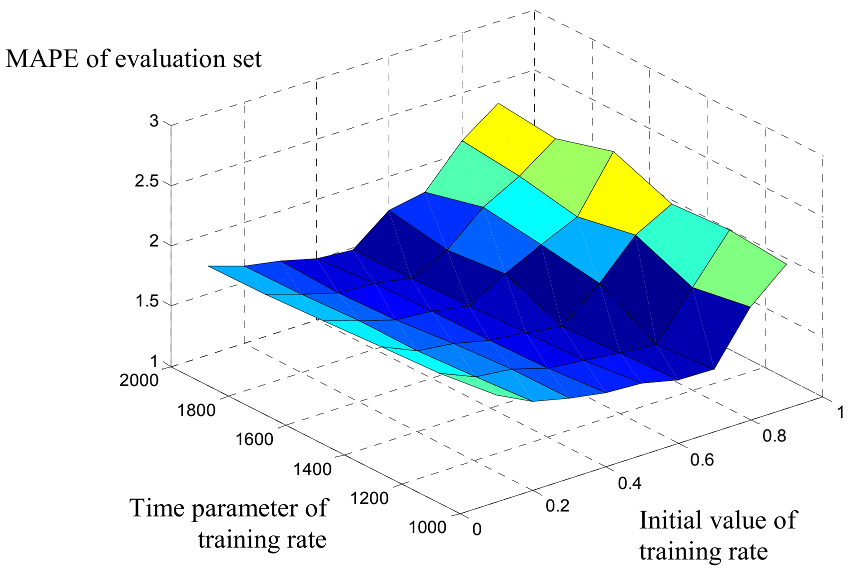

Fig. 3. *MAPE*(%) index for the evaluation set, $\eta_0 = \{0.1, 0.2, ..., 0.9\}$, $T_\eta = \{1000, 1200, ..., 2000\}$, neurons: 45, $a_0 = 0.4$, $T_a = 1800$, activation functions in both layers: hyperbolic tangent, a_1=a_2=0.25, b_1=b_2=0.0

ISSN: 1790-5125

ISBN: 978-960-474-036-9

The best result for the *MAPE* index of the evaluation set is 1.48% and is obtained for an ANN with 45 neurons in the hidden layer, $a_0 = 0.9$, $T_a = 2000$, $\eta_0 = 0.5$, $T_\eta = 2000$, $a_1 = a_2 = 0.25$ and $b_1 = b_2 = 0$ using hyperbolic tangent activation function in both layers.

4 Application for the Set of 14 ANN's Training Algorithms

Following, the proposed methodology of section 2 is applied to each training algorithm for the short-term load forecasting in Greek intercontinental power system properly selected ANN's parameters (see Table 3). The training, the evaluation and the test sets are the same with the respective ones of section 3. The results of all training algorithms are registered in Table 4.

The best results of *MAPE* for evaluation set are given by the stochastic training algorithm with the use of adaptive rules for the learning rate and the momentum term and by the scaled conjugate gradient algorithm. The respective results of *MAPE* for the test set are satisfied, even if the stochastic training algorithm with the use of decreasing functions for the learning rate and the momentum term presents slightly better results. But the scaled conjugate gradient algorithm also presents very good generalization of the results for the test set, because the respective *MAPE* has the second smallest value.

The analogy of the respective computational time (with the proper parameters calibration) for the first 11 training algorithms is: 3.2÷3÷1.2÷4.0÷3.5÷1.4÷10÷12÷12÷12÷1. Finally, the scaled conjugate gradient algorithm is proposed to be used.

In Fig. 4 the estimated and the measured chronological load curves on Thursday, June 8, 2000, are presented indicatively, where the respective MAPE equals to 1.173%.

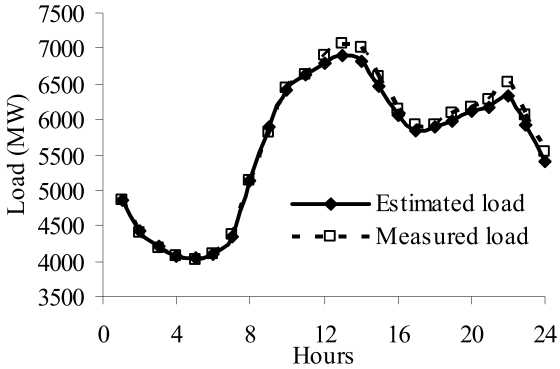

Fig. 4. Chronological load curve (measured and estimated values) for a day of the test set (Thursday, June 8, 2000)

5 Conclusions

This paper compares the performance of 14 different Artificial Neural Network (ANN) training algorithms (see Table 1) during the prediction of the hourly load demand of the next day in Greek intercontinental power system. The structure of the input and the output neurons (71 and 24 respectively) are determined by Kiarzis et al [9]. The rest parameters, such as number of neurons of the hidden layer, activation functions, weighting factors, learning rate, momentum term, etc, are determined by the proposed calibration methodology of section 2 through an extensive search. The performance of each algorithm is measured by the Mean Absolute Percentage Error (MAPE) for the evaluation set. Finally, the scaled conjugate gradient training algorithm is proposed, because of its small values of MAPE and the smallest computational time.

In future the proposed methodology can be improved (i) by using different kinds of input and outputs, (ii) by using compression techniques for inputs, (iii) by estimating the optimization process and (iv) by determining the confidence intervals of the under prediction chronological load curves.

TABLE 2

MAPE(%) OF (A) TRAINING SET, (B) EVALUATION SET, (C) TEST SET FOR DIFFERENT ACTIVATION FUNCTIONS FOR NEURONS: 45, $a_0 = 0.4$, $T_a = 1800$, $\eta_0 = 0.5$, $T_\eta = 2000$, $a_1 = a_2 = 0.25$, $b_1 = b_2 = 0.0$

	Activation function of hidden layer								
	Hyperbolic sigmoid			Hyperbolic tangent			Linear		
Activation function of output layer	(A)	(B)	(C)	(A)	(B)	(C)	(A)	(B)	(C)
Hyperbolic sigmoid	2.030	2.048	2.625	1.510	1.621	1.817	1.788	1.850	2.091
Hyperbolic tangent	1.671	1.737	2.042	1.383	*1.482*	1.749	1.900	1.987	2.200
Linear	1.603	1.691	1.903	1.390	1.522	1.747	1.936	2.023	2.194

TABLE 3
VALUES INTERVAL DURING THE OPTIMIZATION PROCESS OF EACH PARAMETER OF EVERY ANN TRAINING ALGORITHM

No.	Values intervals of each parameter of the respective ANN training algorithm (see Table 1)
1-2	α_0 = 0.1,0.2,…,0.9, T_α = 1000,1200,…,2000, η_0 = 0.1,0.2,…,0.9, T_η = 1000,1200,…,2000
3	η_0 = 0.01,0.02,…,0.1,0.2,…,3
4	η_0 = 0.1,0.2,…,3
5-6	α_0 = 0.1,0.2,…,0.9, T_α = 1200,1500,…,6000, η_0 = 0.1,0.2,…,0.9, T_η = 1200,1500,…,6000
7, 9	s=0.04,0.1,0.2, T_{bv}=20, 40, T_{trix}=50, 100, e_{trix}=10^{-6}, 10^{-5}
8, 10	s=0.04,0.1,0.2, T_{bv}=20, 40, T_{trix}=50, 100, e_{trix}=10^{-6}, 10^{-5}, $\lim_{orthogonality}$ =0.1,0.5,0.9
11	σ=10^{-3}, 10^{-4}, 10^{-5}, λ_0=10^{-6}, 10^{-7}, $5\cdot10^{-8}$
12	δ_1 =0.1,0.2,…,0.5, δ_2 =0,1,0.2,…, 2
13	-
14	$\lambda(0) = 0.1, 0.2, ..., 1, 2, ..., 5$, $\beta = 2, 3, ..., 9, 10, 20, ..., 50$
Common	$N_n = 20,21,...,70$, activation function for hidden and output layer = hyperbolic tangent, linear, logistic, a_1=0.1,0.2,…, 0.5, a_2 =0.1,0.2,…, 0.5, b_1=0.0, ±0.1,±0.2, b_2 =0.0, ±0.1,±0.2

TABLE 4
MAPE(%) OF TRAINING, EVALUATION & TEST SETS FOR 14 DIFFERENT ANN TRAINING ALGORITHMS WITH THE RESPECTIVE PROPERLY CALIBRATED PARAMETERS

No. Of training algorithm	MAPE(%) of training set	MAPE(%) of evaluation set	MAPE(%) of test set	Neurons	Activation functions	Rest Parameters
1	1.383	1.482	1.749	45	f_1=tanh(*0.25*x), f_o=tanh(*0.25*x)	α_0=0.4, T_α=1800, η_0=0.5, T_η=2000, e= 10^{-5} max_epochs=10000
2	1.311	*1.475*	1.829	30	f_1=tanh(*0.40*x), f_o=1/(1+exp(*-0.25*x))	α_0=0.7, T_α=1800, η_0=0.5, T_η=1300, e= 10^{-5} , max_epochs=12000
3	1.372	1.489	1.833	48	f_1=tanh(*0.50*x), f_o=tanh(*0.25*x)	η_0=0.1, e= 10^{-5}, max_epochs=10000
4	2.356	2.296	2.602	48	f_1=tanh(*0.40*x), f_o=*0.40*x	η_0=2.0, e= 10^{-5} , max_epochs=12000
5	2.300	2.294	2.783	25	f_1=tanh(*0.50*x), f_o=*0.25*x	α_0=0.9, T_α=6000, η_0=0.9, T_η=6000, e= 10^{-5}, max_epochs=12000
6	2.019	2.026	2.475	22	f_1=tanh(*0.50*x), f_o=*0.25*x	α_0=0.9, T_α=6000, η_0=0.9, T_η=6000, e= 10^{-5}, max_epochs=12000
7	1.798	1.831	2.147	43	f_1=tanh(*0.40*x), f_o=*0.20*x	s=0.04, T_{bv} =20, T_{trix} =50, e= 10^{-5}, max_epochs=5000
8	2.544	2.595	3.039	43	f_1=tanh(*0.40*x), f_o=*0.20*x	s=0.04, T_{bv} =20, T_{trix} =50, e= 10^{-5}, max_epochs=5000, $\lim_{orthogonality}$=0.9
9	2.545	2.600	3.035	43	f_1=tanh(*0.40*x), f_o=*0.20*x	s=0.04, T_{bv} =20, T_{trix} =50, e= 10^{-5}, max_epochs=5000
10	2.544	2.600	3.035	43	f_1=tanh(*0.40*x), f_o=*0.20*x	s=0.04, T_{bv} =20, T_{trix} =50, e= 10^{-5}, max_epochs=5000, $\lim_{orthogonality}$=0.9
11	1.294	*1.487*	1.781	52	f_1=tanh(*0.50*x), f_o=tanh(*0.25*x)	σ=10^{-5}, λ_0=5x10^{-8}, e=10^{-5}, max_epochs= 10000
12-14	#	#	#	#	#	No convergence

Acknowledgements
The authors want to express their sincere gratitude to C. Anastasopoulos, D. Voumboulakis and P. Eustathiou from PPC for the supply of all the necessary data for this application.

References:

[1] B.F. Hobbs, S. Jitprapaikulsarn, D.J. Maratukulam. Analysis of the value for unit commitment of improved load forecasts. *IEEE Tran. on Power Systems*, Vol. 14, No.4, 1999, pp. 1342-48.

[2] H.B. Gooi, D.P. Mendes, K.R.W. Bell, D.S. Kirschen. Optimal scheduling of spinning reserve. *IEEE Tran. on Power Systems*, Vol. 14, No.4, 1999, pp. 1485-1492.

[3] M.H. Gravener, C. Nwankpa. Available transfer capability and first order sensitivity. *IEEE Tran. on Power Systems*, Vol. 14, No.2, 1999, pp. 512-518.

[4] H.T. Yang, C.M. Huang, C.L. Huang. Identification of ARMAX model for short term load forecasting: an evolutionary programming approach. *IEEE Trans. on Power Systems,* Vol. 11, No. 1, 1996, pp. 403 – 408.

[5] T. Haida, S. Muto. Regression based peak load forecasting using a transformation technique, *IEEE Trans. on Power Systems,* Vol. 9, No. 4, 1994, pp. 1788 – 1794.

[6] H.S. Hippert, C.E. Pedreira, R.C. Souza. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. on Power Systems,* Vol. 16, No. 1, 2001, pp. 44-55.

[7] P.A.Mastorocostas, J.B.Theocharis, A.G. Bakirtzis. Fuzzy modeling for short-term load forecasting using the orthogonal least squares method. *IEEE Trans. on Power Systems,* Vol. 14, No. 1, 1999, pp. 29–36.

[8] S.J.Kiartzis. *Artificial Intelligence Applications on Short-term Load Forecasting*. Dissertation, Thessaloniki, A.U.Th., 1998 (in greek).

[9] A.G. Bakirtzis, V. Petridis, S.J. Kiartzis, M.C. Alexiadis, A.H. Maissis. A neural network short term load forecasting model for the Greek power system. *IEEE Trans. on Power Systems*, Vol. 11, No.2, 1996, pp. 858-863.

[10] S.J. Kiartzis, C.E. Zournas, J.M. Theocharis, A.G. Bakirtzis, V. Petridis. Short term load forecasting in an autonomous power system using artificial neural networks. *IEEE Trans. on Power Systems*, Vol. 12, No.4, 1997, pp. 1591-1596.

[11] K. Kalaitzakis, G.S. Stavrakakis, E.M. Anagnostakis. Short-term load forecasting based on artificial neural networks parallel implementation. *Electric Power Systems Research*, Vol. 63, 2002, pp. 185-196.

[12] A.G. Bakirtzis, J.B. Theocharis, S.J.Kiartzis, K.J.Satsios. Short term load forecasting using fuzzy neural networks. *IEEE Trans. on Power Systems*, Vol. 10, No. 3, 1995, pp. 1518-1524.

[13] G.H. Asimakopoulou, V.T. Kontargyri, G. J. Tsekouras, F.H. Asimakopoulou, I.F. Gonos, I.A. Stathopulos. Artificial Neural Network Optimization Methodology for the Estimation of the Critical Flashover Voltage on Insulators. *IET Science, Measurement & Technology*, to be printed.

[14] P.S. Ghosh, S. Chakravorti, N. Chatterjee. Estimation of Time-to-flashover Characteristics of Contaminated Electrolytic Surfaces using a Neural Network. *IEEE Trans. on Dielectrics and Electrical Insulation*, Vol. 2, No. 6, 1995, pp. 1064-1074.

[15] L.M. Saini, M.K. Soni. Artificial neural network-based peak load forecasting using conjugate gradient methods. *IEEE Trans. on Power Systems*, Vol. 17, No 3, 2002, pp.907-912.

[16] R. Fletcher, C.M. Reeves. Function minimization by conjugate gradients, *Computer Journal*, Vol. 7, 1964, pp. 149-154.

[17] M.J. Powell. Restart procedures for the conjugate gradient method. *Mathematical Programming*, Vol. 12, 1977, pp. 241-254.

[18] E. Polak. *Computational methods in optimization: a unified approach*. Academic Press, New York, 1971

[19] M.F. Moller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, Vol. 6, 1993, pp. 525-533.

[20] M. Riedmiller, H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. *Proceedings of the IEEE International Conference on Neural Networks*, San Francisco, Vol. 1, March 1993, pp. 586-591.

[21] R. Battiti. First and second order methods for learning: Between steepest descent and Newton's method. *Neural Computation*, Vol. 4, No. 2, 1992, pp. 141-166.

[22] K. Levenberg. A method for the solution of certain problems in least squares. *The Quarterly of Applied Mathematics*, Vol. 2, 1944, pp. 164-168.

[23] D. Marquardt. An algorithm for least squares estimation of nonlinear parameters. *SIAM Journal Application Mathematics*, Vol. 11, 1963, pp. 431-441.

[24] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1994.

[25] G. J. Tsekouras, J. Koukoulis, M.-A. Nikolinakou, N.E. Mastorakis. Prediction of face settlement during tunneling excavation using artificial neural network. *WSEAS International Conference on Engineering Mechanics, Structures, Engineering Geology* (EMESEG '08), Heraklion, Crete Island, Greece, July 22-25, 2008.