International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:3, No:5, 2009

Performance Analysis of Evolutionary ANN for Output Prediction of a Grid-Connected Photovoltaic System

S.I Sulaiman, T.K Abdul Rahman, I. Musirin and S. Shaari

***Abstract*—**This paper presents performance analysis of the Evolutionary Programming-Artificial Neural Network (EPANN) based technique to optimize the architecture and training parameters of a one-hidden layer feedforward ANN model for the prediction of energy output from a grid connected photovoltaic system. The ANN utilizes solar radiation and ambient temperature as its inputs while the output is the total watt-hour energy produced from the grid-connected PV system. EP is used to optimize the regression performance of the ANN model by determining the optimum values for the number of nodes in the hidden layer as well as the optimal momentum rate and learning rate for the training. The EPANN model is tested using two types of transfer function for the hidden layer, namely the tangent sigmoid and logarithmic sigmoid. The best transfer function, neural topology and learning parameters were selected based on the highest regression performance obtained during the ANN training and testing process. It is observed that the best transfer function configuration for the prediction model is [logarithmic sigmoid, purely linear].

***Keywords*—**Artificial neural network (ANN), Correlation coefficient (*R*), Evolutionary programming-ANN (EPANN), Photovoltaic (PV), logarithmic sigmoid and tangent sigmoid.

I. Introduction

IN the grid-connected photovoltaic (PV) system, a major question among a user is how much energy output can be harvested from the grid-connected PV system throughout its operation. In many cases, the customer would want to know the performance of their installed grid-connected PV systems under different climatic conditions. Therefore, many studies have been conducted to predict the output of the systems. One of the popular techniques used for the prediction is the artificial neural network (ANN). A three-layer feedforward ANN was used to predict the energy output of a grid connected PV system by knowing the solar radiation, module temperature and clearness index [1]. Similarly, the output of a PV module was also predicted using the same architecture but with different types of inputs and outputs [2]. The ANN utilizes solar radiation, ambient temperature and module temperature as its inputs whereas voltage and current are used as outputs. Although these studies have produced many important discoveries in the prediction of PV system outputs using ANN, the design of ANN models using specific sets of design constraints relies so much on past experience with same applications and is subjected to trial and error processes [3]. For example, in a grid connected PV system, a study has been conducted to predict the total AC power output of a grid-connected PV system using manually-designed ANN [4]. However, this manual design of ANN is time consuming and vulnerable to inaccuracy issues due to the tedious heuristic process experienced by the ANN designers. As a result, evolution process has been introduced to provide faster training of the ANN [5].

Evolution in multilayer feedforward ANN design can be performed by evolving the connection weights, architectures or learning rules of the ANN. The evolution of the connection weights is done by online training of the ANN connections using a predetermined architecture whereas the evolution of ANN architectures is achieved by adjusting the ANN topologies for different learning tasks. The evolution of connection weights requires modification of ANN weights to learn a particular function. On the other hand, the evolution of architectures demonstrates the evolution of topological parameters such as number of nodes and number of hidden layers. Apart from that, the evolution of learning rules is related to the adaptive process of finding the best learning rules.

Despite having many types of evolution methods for evolving ANN, one of the most popular methods is the Evolutionary Programming (EP). This technique is an optimization tool based on natural evolution [6]. In this study, Evolutionary Programming (EP) is chosen to perform the evolution of ANN. It was initially introduced to simulate artificial intelligence through the evolution of finite state machines [7]. In addition, the usage of Gaussian mutation and self-adaptation has been proved to become a determining factor in improving the performance of EP [8].

S. I. Sulaiman is with the Universiti Teknologi MARA Malaysia, Shah Alam, 40450 Malaysia (phone: +603-55436031; fax: +603-55435077; e-mail: shahril@salam.uitm.edu.my).
T. K. Abdul Rahman, is with the Universiti Teknologi MARA Malaysia, Shah Alam, 40450 Malaysia (phone: +603-55435051; fax: +603-55435077; e-mail: khawa@salam.uitm.edu.my).
I. Musirin is with the Universiti Teknologi MARA Malaysia, Shah Alam, 40450 Malaysia (phone: +603-55435044; fax: +603-55435077; e-mail: ismailbm@salam.uitm.edu.my).
S. Shaari is with the Universiti Teknologi MARA Malaysia, Shah Alam, 40450 Malaysia (phone: +603-55444567; fax: +603-55444562; e-mail: solarman@salam.uitm.edu.my).

A study to find the optimum ANN hidden layer size using EP has been successfully performed [9]. Similarly, an algorithm to determine the ANN hidden layer size and weight coefficients has been developed [10]. Therefore, besides predicting the watt-hour energy output from grid-connected PV system, this paper also demonstrates the design of a hybrid prediction model using Evolutionary Programming-Artificial Neural Network (EPANN). Two types of transfer function configuration are tested in the implementation of the EPANN model.

II. THE SYSTEM AND ANN DATA COLLECTION

The site under study is the Malaysia Energy Center (PTM), Bangi, Malaysia located at latitude approximately 2°53′0″N and longitude 101°40′0″E. The site contains 6 grid-connected PV systems with a total amount of 92kWp installed capacity. However, only one system is investigated in this study. The system under study comprises 1.92kWp polycrystalline PV array and one unit of grid-tied inverter. The PV array is tilted at 7° with respect to the horizontal plane. The inverter is capable of recording the energy output from the inverter. In addition, two ambient temperature sensors and an irradiance sensor are strategically positioned near the PV array and connected to the inverter. Data logging was performed using a data logger connected to the inverter. In this study, data have been collected during the month of February 2008. The data comprises solar irradiance, *SI* (in kW/m^2) falling on horizontal plane, ambient temperature at sensor 1, *AT1* (in ºC), ambient temperature at sensor 1, *AT2* (in ºC) and total energy output (in Wh). These data has been utilized to simulate the general ANN model illustrated in Fig. 1.

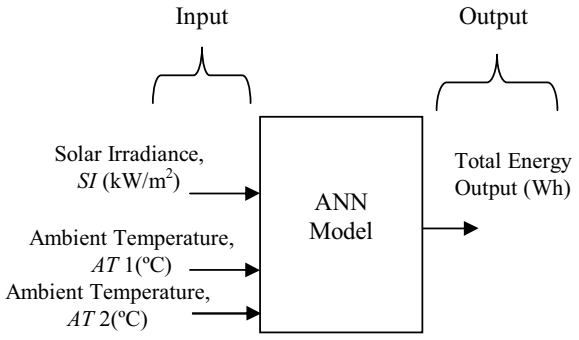

Fig. 1. ANN model for the prediction of energy output from the grid connected PV System located at Malaysia Energy Centre (PTM), Bangi, Malaysia

Since its introduction, ANN has been employed to solve various problems in power engineering and many other complex problems. In basic computational model of ANN, a node in an ANN collects input signals from other nodes and merges them. Relevant computation is performed before the result is mapped to an output node [11]. In this study, a three-layer feedforward neural network has been employed for investigation. In Fig. 1, a three-variate ANN model that uses *SI*, *AT1* and *AT2* as its input and total watt-hour energy as its output has been employed. This work is valid as the performance of PV modules is also influenced by ambient temperature besides the solar irradiance [12].

III. GENERAL DESIGN OF ANN

After selecting the type, inputs and output of the ANN, suitable ANN parameters for evolution process have to be identified as there are several methods available to evolve the ANN. The first method involves the evolution of connection weights during ANN training [13]-[14]. This method emulates the conventional technique of adjusting weights of ANN to learn an unknown function. The second method entails the evolution of ANN architecture [15]. Unlike the traditional method that requires the number of nodes of hidden layers to be determined heuristically by ANN designers, this more advanced method allows the size of nodes to expand or shrink automatically using simulated evolution. Besides that, the evolutionary training of ANN can also be achieved through the evolution of transfer functions [16]. Apart from that, the values of the ANN parameters to be fixed were also determined at this stage.

Despite having an argument on the most suitable method of evolution, the type of method to be used in ANN evolution actually depends on the amount and quality of information obtained about the proposed ANN prediction model. Therefore, the ANN design could also be very subjective. For example, if more previous information on the learning algorithms is available compared to the previous information on the ANN architecture, the evolution of learning algorithms should be given the top priority before considering the evolution of ANN architecture. Therefore, not every ANN parameter is evolved. As the region of evolution is more restricted based on the prior information obtained, the convergence of the evolution process becomes faster. However, this informal guideline is invalid if there is a specific attention to the type of learning algorithm or the size of architecture to be investigated in a study.

In this study, a few ANN characteristics and parameters have been fixed for the training process. Firstly, the performance of the prediction is quantified using the correlation coefficient, *R*. Secondly, the Levenberg-Marquardt algorithm has been chosen as the learning algorithm for the ANN as it has been proven useful in many prediction studies. Besides that, the evolution of weights was not conducted as the back-propagation method could satisfactorily perform the optimal computation of the set of weights in the ANN. Apart from that, the number of epochs is set to be very large (1000 epochs) in order to allow accurate convergence of the ANN. The mean-square-error training goal is chosen to be sufficiently small (10^{-3}) to ensure satisfactory prediction accuracy. As these settings have been fixed at the beginning of training process, the evolutionary process can be considered semi-automatic. On the other hand, the transfer function configuration is set to be either [logarithmic sigmoid, purely linear] or [tangent sigmoid, purely linear] after preliminary investigation. These transfer functions are selected due to their

proven capability in various prediction tasks [17]-[18]. Both configurations were tested one at a time to determine the best transfer function for the hidden layer.

IV. DESIGN OF EPANN MODEL

After the ANN parameters and characteristics have been identified, the evolution technique has been specified for the ANN model. In this study, the EP is chosen to perform the evolution process. EP is a heuristic technique used to conduct a semi-random search in optimizing an objective function. It consists of a few important processes namely generation of random population, fitness evaluation, mutation, combination, selection and convergence test. It has been shown that EP perform better than Genetic Algorithm (GA) in optimization task [19]. In the proposed evolutionary ANN, the number of nodes in hidden layer, x_1, learning rate, x_2 and momentum rate, x_3 of the ANN are allowed to evolve to reach their optimum values using EP. The best solution is achieved by maximizing the correlation coefficient, R of the prediction during ANN training. Higher R would imply a higher accuracy of the prediction model. The proposed evolutionary training algorithm is summarized in the following procedure.

1) Generate random numbers to represent the evolving variables (x_1, x_2, and x_3) as described previously. x_1 is set to be random integers in the range from 1 to 100. In contrast, $x_2$and x_3 are random values from 0 to 1.
2) Run preliminary fitness evaluation. At this stage the ANN model is trained using the generated random numbers to determine the R value. Constraints are set during this stage. The resulting R must be greater than or equal to 0.99 and less than unity.
3) Decision: If constraints are not violated, proceed to step 4. Otherwise, return to step 1.
4) Load the set of random numbers into pool population known as parent. The maximum number of population allowed is 20.
5) Decision: If pool population is full, proceed to the next step. Otherwise, return to step 1.
6) Determine the maximum and minimum values for x_1, x_2 and x_3. These statistical values are required for mutation process in later stage.
7) Train ANN to determine the values of fitness function. The fitness function represents the value of R for the prediction.
8) Determine the maximum and minimum values for R. These statistical values are also required for mutation process.
9) Mutate parents to produce offspring using Gaussian mutation operator.
10) Train the ANN using the mutated set of random numbers (offspring).
11) Combine parents and offspring. The total population now becomes 40.
12) Perform priority selection strategy. The set of random numbers with highest R value will be of the top priority while the set of random numbers with lowest R value will be ranked as the lowest priority.
13) Transcribe the 20 sets of random numbers with highest priority into a new generation for the next evolution.
14) Test the new generation using stopping criteria.
15) Decision: If fitness function converges, the program is stopped. Otherwise, step 6 is repeated.

The program starts with the initialization of three random numbers that represent the number of nodes in the hidden layer (x_1), learning rate (x_2) and momentum rate (x_3). These numbers are generated randomly based on normal distribution function. The search space of this program is restricted to the results from prior studies in predicting PV systems output using ANN. The randomly generated values for x_1 are specified to be within a range from positive integer 1 to 100. The upper limit is set to be sufficiently large (100) such that the optimal number of neurons can be reached. Thus, problem of trapping at a local optima can be avoided. Nevertheless, x_2 and x_3 are selected to have random values between 0 and 1 as these numbers indicate the learning rate and momentum rate respectively. The maximum range possible is set for these numbers as inadequate previous information were obtained about the most suitable values for the learning rate and the momentum rate in predicting the output of a PV system. In fact, these numbers are usually unique according to the nature of a problem. The R values for the constraints are set to be from 0.99 to 1.00 so that only good prediction results are transcribed into the population. This constraint was tested by training the ANN using the set of random numbers generated previously. After the training process, if the set of random numbers generated produces R value outside the range defined by the predetermined constraints, the random number generation process is repeated. If not, the set of random numbers are loaded into the pool population. This set of random numbers, known as parent, will experience a mutation process to produce an offspring at a later stage. After a parent is successfully created, the random number generation is repeated until the pool population is completely filled. The maximum number of parents in the population is set to be 20 based on a previous study [20]-[21]. This value is repeatedly found to be sufficient for an EP to converge. After the pool population has been filled, the statistical parameters are identified by determining the minimum and maximum values for each variable of the sets of random numbers obtained in the pool population. These values will be also useful for the mutation process later. Next, the first fitness function calculation is conducted by training the ANN repeatedly using the 20 sets of random numbers obtained from the parent population. Thus, twenty R values are produced from the twenty sets of parents. Similarly, maximum and minimum values of R from these results are determined for mutation process. Later, the evolution process proceeds with the mutation of each parent using Gaussian mutation. This process generates 20 sets of random numbers and each set is called as an offspring. Using the sets of random numbers generated from each offspring, the second fitness function calculation is conducted by training the ANN. Consequently, both parents and offspring were combined to form a new population of 40

sets of random numbers with each of them having their own fitness value. Next, these parents and offspring undergo the selection process using priority selection strategy. In this strategy, the population is arranged in descending order according to the individual fitness value of each parent or offspring [22]. Thus, the set of random numbers with highest R value will be ranked on the top while the set of random numbers with lowest R value will be ranked at the bottom. The top 20 sets of random numbers are later transcribed as the new generation of sets of random numbers. Before the evolution process continues, convergence test is performed to decide whether the search for the best set of random numbers should be continued or stopped. In this study, the stopping criteria are defined by the following equations.

$$R_{\max} - R_{\min} \leq 0.1 \quad (1)$$

$$x_{1,\max} - x_{1,\min} = 0 \quad (2)$$

The first equation halts the program if the difference between the maximum fitness value and the minimum fitness value in the new generation is less than or equal 0.1. Nevertheless, the second equation should also be satisfied for the whole process to stop. The second equation is added to the set of constraint equations because the final number of nodes recommended by the search should be an integer value. If these stopping criteria are not satisfied, the search for optimum set of random numbers is continued. However, the new search is performed by repeating the mutation process based on the statistical values obtained from the new generation of random numbers. At the end, the actual prediction performance during training is quantified using the average R value. The average R value is a better indicator of the overall training performance since the EPANN would not show an absolute convergence ($R_{max} = R_{min}$).

After the EPANN training has been completed, testing process is performed to validate the training process. The EPANN is implemented twice using logarithmic-sigmoid (logsig) and hyperbolic tangent-sigmoid (tansig) as the transfer function at the hidden layer. However, the transfer function at the output layer is fixed to purely linear.

V. RESULTS AND DISCUSSION

The EPANN model was trained using logsig and tansig transfer function at the hidden layer. The results for the prediction of total energy output of the grid-connected PV system using the EPANN model are illustrated in Table I.

In Table I, the optimal number of nodes in the hidden layer of the EPANN model with the transfer function [logarithmic-sigmoid, purely linear] is 2 whereas the optimal number of nodes in the hidden layer of the EPANN model with the transfer function [hyperbolic tangent-sigmoid, purely linear] is only 1. Hence, the EPANN model can be realized using a smaller neural topology using the tansig transfer function. This feature is very important if the EPANN is going to be implemented in hardware. Besides that, the learning rate for the EPANN with logsig and the EPANN with tansig is 0.05796 and 0.02831 respectively. On the other hand, the momentum rate for the EPANN with logsig and the EPANN with tansig is 0.06831 and 0.00799 respectively. Therefore, it can be observed that the EPANN with tansig transfer function requires lower learning rate and momentum rate compared to the EPANN with logsig transfer function. Apart from that, the average correlation coefficient, R during training for EPANN with tansig is found to be 0.99388. This value is slightly higher than 0.99319 achieved by EPANN with logsig. However, the testing process shows that the EPANN with logsig performs better than the EPANN with tansig. The EPANN with logsig has an R value of 0.99519 while the EPANN with tansig has an R value of 0.99360. The prediction performance of the EPANN with logsig during testing is better possibly due to higher number of nodes in the hidden layer which is found during the evolutionary training. The higher number of nodes might indicate a better generalization capability of the neural network model. As a result, higher R value during testing is obtained using the EPANN with logsig transfer function. Nevertheless, both EPANN models consume almost equal computation time during the prediction task.

TABLE I
TRAINING PARAMETERS AND RESULTS FOR EPANN USING DIFFERENT TRANSFER FUNCTION CONFIGURATION.

Parameters / Results	EPANN with logsig	EPANN with tansig
Number of training patterns	300	300
Number of testing patterns	200	200
Number of nodes in hidden layer, x_1	2	1
Learning rate, x_2	0.05796	0.02831
Momentum rate, x_3	0.06831	0.00799
Number of nodes in output layer	1	1
Training algorithm	trainlm	trainlm
Type of transfer function	logsig-purelin	tansig-purelin
Average correlation coefficient, R during training	0.99319	0.99388
Correlation coefficient , R during testing	0.99519	0.99360
Training duration (seconds)	225.35	233.53

The detailed prediction performance of the EPANN models during training is illustrated in Fig. 2 and Fig. 3. The fitness values of both models converge after the fifth evolution. In addition, the average fitness curve is consistently close to the maximum fitness curve in both cases. Therefore, it can be safely concluded that the EPANN models shows good performance in trying to maximize the *R* value during the training process. Nevertheless, the EPANN with tansig transfer function shows a slight inconsistency towards achieving the maximum fitness value during the evolution process.

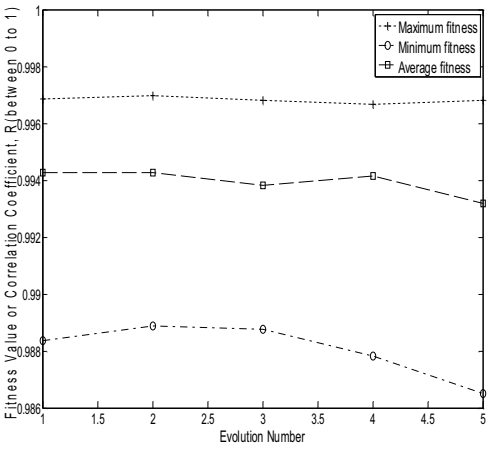

Fig. 2: Plot of Different Types of Fitness Values at Different Evolution Number for EPANN with logsig transfer function

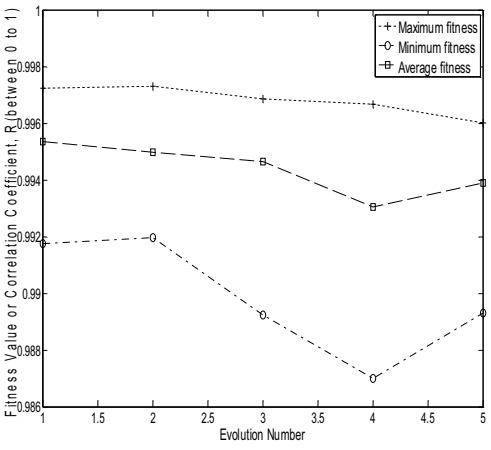

Fig. 3: Plot of Different Types of Fitness Values at Different Evolution Number for EPANN with tansig transfer function

The search of the optimal number of nodes in both EPANN models is illustrated in Fig. 4 and Fig. 5. In general, both models show an approximately similar declining trend of number of nodes towards the final evolution number. The only difference is the final number of nodes in the hidden layer for each model. The EPANN with logsig yields 2 as the optimal number of nodes whereas the EPANN with tansig produces 1 node as the optimal value.

On the other hand, the search for the optimal training parameter values is illustrated in Fig. 6 and Fig. 7. In Fig. 6, the learning rate and momentum rate of the EPANN with logsig shows a different trend of convergence during the evolution. In contrast, the learning rate and momentum rate of the EPANN with tansig shows a similar trend towards convergence after the second evolution.

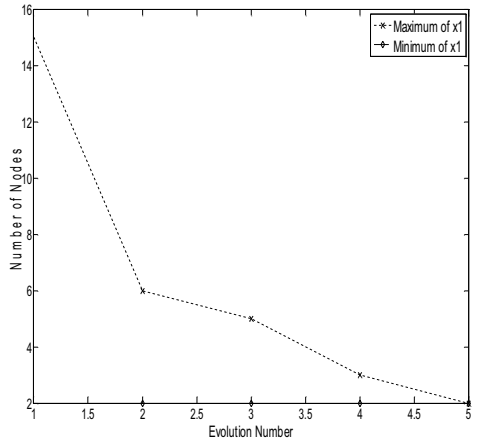

Fig. 4: Plot of Number of Nodes at Different Evolution Number for EPANN with logsig transfer function

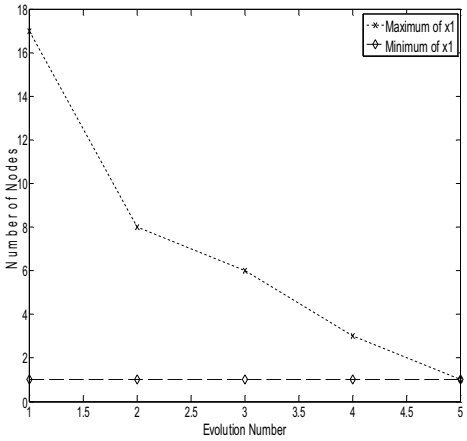

Fig. 5: Plot of Number of Nodes at Different Evolution Number for EPANN with tansig transfer function

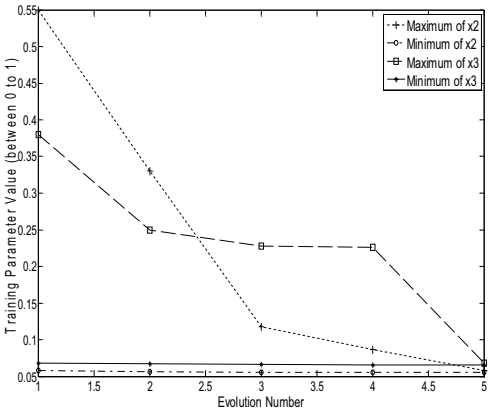

Fig. 6: Plot of Different Types of Training Parameter Values at Different Evolution Number for EPANN with logsig transfer function

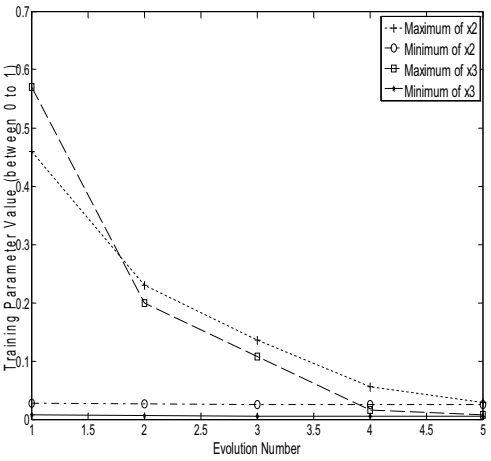

Fig. 7: Plot of Different Types of Training Parameter Values at Different Evolution Number with tansig transfer function

VI. Conclusion

The EPANN has been successfully used to predict the watt-hour energy output from the grid-connected PV system. During the evolutionary training, the optimal number of nodes in the hidden layer, the learning rate and the momentum rate have been determined for each EPANN model. Nevertheless, due to the random nature of the ANN process, the fitness of the EPANN i.e. the correlation coefficient of the prediction may not show complete convergence. Therefore, average fitness has been introduced as a measure of the EPANN prediction performance. Apart from that, the EPANN with logsig as the transfer function at the hidden layer shows better overall prediction performance despite having slightly lower *R* value compared to *R* value of the EPANN model with tansig during training process. Since the *R* value of the EPANN model with logsig is higher than the *R* value of EPANN model with tansig, the EPANN model with logsig could be considered the better prediction model. Nevertheless, if the hardware implementation is considered, the EPANN model with tansig is more feasible as the optimal number of nodes in the hidden layer is found to be lower than the optimal number of nodes found for the EPANN model with logsig. In terms of overall computation time, both models could be implemented within approximately the same duration.

Acknowledgment

The authors would like to thank the Photovoltaic Monitoring Centre of Universiti Teknologi MARA and Malaysia Energy Centre (PTM) for making the training data and testing data available for study.

References

[1] I. Ashraf and A. Chandra, "Artificial neural network based models for forecasting electricity generation of grid connected solar PV power plant", *Int. Journal of Global Energy Issues*, vol. 21, no. 1/2, pp. 119-130, 2004.

[2] M. Balzani and A. Reatti, "Neural network based model of a PV array for the optimum performance of PV system", in *Proc. PhD Research in Microelectronics and Electronics Conf.*, vol. 2, 2005, pp. 123-126.

[3] X. Yao, "Evolving artificial neural networks", in *Proc. Of the IEEE*, vol. 87, no. 9, 1999.

[4] S.I. Sulaiman, T.K. Abdul Rahman, and I. Musirin, "ANN-based technique with embedded data filtering capability for predicting total AC power from grid-connected photovoltaic system", in *Proc*. 2nd *International Power Engineering and Optimization Conference (PEOCO2008)*, 2008, pp. 272-277.

[5] X. Yao and Y. Liu, "Towards designing artificial neural networks by evolution", *Applied Mathematics and Computation*, vol. 91, pp. 83-90, 1998.

[6] D.B. Fogel, "An introduction to simulated evolutionary optimization", *IEEE Transactions on Neural Networks*, vol. 5, pp. 3-14, 1994.

[7] L.J. Fogel, "Autonomous automata", *Industrial Research*, vol. 4, no. 1, pp.14-19, 1962.

[8] T. Back and H.-P. Schwefel, "Evolutionary computation: an overview", in *Proc. IEEE International Conference on Evolutionary Computation (ICEC'96)*, 1996, pp. 20-29.

[9] M. Sarkar and B. Yegnanarayana, "Feedforward neural networks configuration using evolutionary programming", in *Proc. International Conference on Neural Networks*, vol. 1, 1997, pp. 438-443.

[10] K. Peng, S.S. Ge, and C. Wen, "An algorithm to determine neural network hidden layer size and weight coefficients", in *Proc. 15th IEEE International Symposium on Intelligent Control (ISIC 2000)*, 2000, pp. 261-266.

[11] B. Yegnanarayana, *Artificial Neural Networks*, New Delhi: Prentice Hall of India, 2006, ch. 1.

[12] S.R. Wenham, M.A. Green, and M.E. Watt. *Applied Photovoltaics*. Centre for Photovoltaic Devices and Systems, Sydney: The University of New South Wales, 1995, ch. 1.

[13] W.M. Jenkins, "Neural network weight training by mutation", *Computers & Structures*, vol. 84, pp. 2107-2112, 2006.

[14] M. Annunziato, I. Bertini, A. Pannicelli and S. Pizzuti, "Evolutionary feed-forward neural networks for traffic prediction", in *Proc. International Congress on Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems (EUROGEN 2003)*, 2003, pp. 1-8.

[15] J. Fang and Y. Xi, "Neural network design based on evolutionary programming", *Artificial Intelligence in Engineering*, vol. 11, pp. 155-161, 1997.

[16] M.F. Augusteijn and T.P. Harrington, "Evolving transfer functions for artificial neural networks", *Neural Computing and Applications*, vol. 13, pp. 38-46, 2004.

[17] D. Johari, T.K. Abdul Rahman and I. Musirin, "Artificial Neural Network Based Technique for Lightning Prediction", in *Proc. 5th Student Conference on Research and Development (SCOReD 2007)*, 2007.

[18] O.A. Dombaycr and M. Golcu, "Daily means ambient temperature prediction using artificial neural network method: a case study of Turkey", *Renewable Energy*, vol. 34, no. 4, pp. 1158-1161, 2009.

[19] W. Gao, "Study of new evolutionary neural network", *in Proc. 2nd International Conference on Machine Learning and Cybernetics*, 2003, pp. 1287-1292.

[20] I. Musirin and T.K. Abdul Rahman, "Evolutionary programming based optimization technique for maximum loadability estimation in electric power system", in *Proc. National Power and Energy Conference (PECon)*, 2003, pp. 205-210.

[21] S.I. Sulaiman, T.K. Abdul Rahman and I. Musirin, "Semi automatic design of two-hidden layer feedforward ANN for grid-photovoltaic system out prediction", in *Proc. International Graduate Conference of Engineering and Science*, 2008, pp. 91-96.

[22] F.I Hassim, I. Musirin and T.K. Abdul Rahman, "Voltage stability margin enhancement using Evolutionary Programming (EP)", in Proc. 4th *Student Conference on Research and Development (SCOReD 2006)*, 2006, pp. 235-240.

Shahril Irwan Sulaiman became a Member (M) of **IEEE** in 2008. He was born in Kelantan, Malaysia. He received his B.Eng in Electrical & Electronics Engineering from Universiti Tenaga Nasional (UNITEN), Malaysia in 2002 and M.Eng.Sc in Photovoltaic Engineering from University of New South Wales, Australia in 2003. Since 2004, he has been a lecturer in Department of Electronics, Faculty of Electrical Engineering, Universiti Teknologi MARA (UiTM), Malaysia. His research interests are in the areas of photovoltaics (PV), renewable energy technologies and artificial intelligence. He is currently working towards his PhD degree at Universiti Teknologi MARA (UiTM), Malaysia**.**

Titik Khawa Abdul Rahman received BSc E.E. (Hons) and PhD on 1988 from Loughborough University of Technology and University of Malaya, Malaysia on 1996 respectively. She is a Professor at the Faculty of Electrical Engineering and currently the Dean of Faculty of Electrical Engineering, Universiti Teknologi MARA Malaysia. She has written more than 80 technical papers and has been also supervising post-graduate students. Her research interest includes voltage profile studies, artificial neural network, Evolutionary Computation such as Evolutionary Programming and Genetic Algorithm, Artifical Immune System (AIS), Economic Dispatch and Loss Minimization. She is currently the secretary for the Power and Energy Society (PES) Chapter, IEEE Malaysia Section.

Dr. Ismail Musirin obtained Diploma of Electrical Power Engineering in 1987, Bachelor of Electrical Engineering (Hons) in 1990; both from Universiti Teknologi Malaysia, MSc in Pulsed Power Technology in 1992 from University of Strathclyde, United Kingdom and PhD in Electrical Engineering from Universiti Teknologi MARA (UiTM), Malaysia in 2005. He is currently an Associate Professor and Chair, Centre of Electrical Power Engineering Studies (CEPES), Faculty of Electrical Engineering, UiTM. He has published 2 books and more than 100 technical papers in the international and national, conferences and journals. He is reviewer for IEEE, IET and WSEAS journals and conferences. His research interest includes power system stability, optimization techniques, distributed generator and artificial intelligence.

Sulaiman Shaari received his bachelor's degree, master's degree and doctorate's degree from DeMontfort University, UK, Univ of Missouri, USA and Kansas State Univ, USA, respectively. He teaches at Universiti Teknologi MARA in Faculty of Applied Science as an Associate Professor. He also heads the Solar Energy Applications Research Laboratory at the university. His main research interests are in the design, sizing, monitoring and evaluation of PV systems.