Photovoltaic Plants Predictive Model by means of ANN trained by a Hybrid Evolutionary Algorithm

Davide Caputo, Francesco Grimaccia, *Member, IEEE*, Marco Mussetta, *Member, IEEE*
and Riccardo E. Zich, *Member, IEEE*

***Abstract*— This paper introduces a hybrid evolutionary optimization algorithm as a tool for training an Artificial Neural Network used for production forecasting of solar energy PV plants. This hybrid technique is developed in order to exploit in the most effective way the uniqueness and peculiarities of two classical optimization approaches, Particle Swarm Optimization (PSO) and Genetic Algorithms (GA). This procedure essentially represent a bio-inspired heuristic search technique, which can be used to solve combinatorial optimization problems, modeled on the concepts of natural selection and evolution (GA), but also based on cultural and social behaviours derived from the analysis of the swarm intelligence and interaction among particles (PSO). Some simulation results are reported to highlight advantages and drawbacks of the proposed technique in order to suitably apply this algorithm to neural network applications in engineering problems.**

I. Introduction

Computation techniques play an important role in most engineering problems in which optimization problems have to be solved. Usually the complex nature of many practical problems involves an effective use of Artificial Neural Networks (ANNs) to solve them. ANNs are useful tools when it is necessary to understand the complex and nonlinear relationships among data, without any a-priori assumption concerning the nature of these correlations.

One of the most critical phase in managing ANNs is the training one, when the weights of the neural connections have to be set properly [1]. The parameters of the network have to be optimized in order to reach a good and accurate output. Therefore the learning process should result in finding the weights configuration associated to the minimum output error, namely the optimized weights configuration. Usually problems are associated to an objective function to be optimized. This function, called also fitness, cost or energy function, provides the interface between the physical problems and the optimization algorithm itself. There are often many problems related to these algorithms. The huge number of variables is the first difficulty when dealing with one of these optimizing issues. Secondly, there are lots of configurations with different values of the objective function which are quite similar each other and very close to the global optimum case, even if these configurations are sub-optimal. Generally finding a solution in an optimization process means to reach a balance among different and often conflicting goals; as a consequence such a search could be extremely difficult. This article aims to achieve a straightforward and more effective training phase and the ANN is here used mainly for forecasting purposes but not from end-user side a [2], [3].

The authors are with Politecnico di Milano, Dipartimento di Energia, Via La Masa 34, I-20156 Milano, Italy. E-mail: davide.caputo@polimi.it, francesco.grimaccia@polimi.it, marco.mussetta@polimi.it

At a first sight, the approaches used to optimize and train an ANN may be divided into two main categories: the ones that need to calculate derivatives in order to optimize the objective function, namely classical methods, and those that, starting from an initial position in the solution space, explore iteratively this space without requiring any derivatives calculation (e.g. evolutionary methods). They are used to train the network by learning the nonlinear behavior between input and output data.

In this paper the authors propose a simple and fast method, the recently developed Genetical Swarm Optimization algorithm, in order to replace the traditional training processes of the ANN.

II. Classical Learning Algorithms

Error Back Propagation algorithm is a well known analytical algorithm used for neural networks training. In literature, there are several forms of back-propagation, all of them requiring different levels of computational efforts; the conventional back-propagation method is, however, the one based on the gradient descent algorithm.

One of the drawbacks of this method is the strong dependence upon the initial guess. A bad choice of the starting point may result in the possibility to get stuck in a local minimum and consequently to find a solution that is not the best one. Besides, most of the typical problems requiring optimization often have non-differentiable or/and discontinuous regions in the solution domain and this fact introduces difficulties in the application of these traditional methods based on derivatives calculations. These aspects are often overcome by evolutionary methods.

The most effective evolutionary algorithm developed until now is Genetic Algorithm (GA) that is now quite familiar to the engineering community and widely used [4]. Genetic algorithms are very efficient at exploring the entire search space, but are relatively poor in finding the precise local optimal solution in the covergence region. Some additional operators can be introduced for GA in order to get a better predictive power of ANNs selecting an optimal combination of input variables.

Moreover, in recent years also the Particle Swarm Optimization (PSO) algorithm is gaining increasing attention for the integration in the training phase of ANNs [5].

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

III. Genetical Swarm Optimization

The GSO algorithm was born as a hybrid evolutionary technique developed in order to combine the best properties of GA and PSO and overcome the problem of premature convergence.

Some comparisons of the performances of GA and PSO [6] emphasize the reliability and convergence speed of both methods, but still keep them separate. GA and PSO show a marked application driven characteristic for any respective technique: PSO seems to have faster convergence in the first runs, but often it is outperformed by GA for long simulations, when the last one finds better solutions.

Some attempts to exploit the qualities of the two algorithms have been done in the last ten years with a kind of integration of the two strategies [7], [8], but GSO aimed to reach a stronger co-operation of the two techniques stressing its hybrid nature and maintaining the GA and PSO integration for the entire run. In the last few years the authors have proved repeatedly such an updating technique can improve traditional evolutionary mechanism for a wide range of applicationsincreasing individuals' score by means of an effective combination of natural selection and knowledge sharing.

In particular, in [9], [10], we presented some comparisons of GSO and classical methods performances, emphasizing the reliability and convergence speed of the first one and applying it to different case studies. In these previous experiments, for example on large EM optimization problems, the algorithm proved to be a fast and robust technique, outperforming classical procedures.

A. The Genetical Swarm Optimization mechanism

The basic concepts of GSO have been presented in [9]: in every iteration of the population is randomly divided into two parts which are evolved with GA and PSO techniques respectively. Then the fitness of the newly generated individuals is evaluated and they are recombined in the updated population which is again divided into two parts in the next iteration for the next run of genetic or particle swarm operators.

The population update concept can be easily understood thinking that a part of the individuals is substituted by new generated ones by means of GA, while the remaining are the same of the previous generation but moved on the solution space by PSO.

The driving parameter of GSO algorithm is the *hybridization coefficient* (hc); it expresses the percentage of population that in each iteration is evolved with GA: so $hc = 0$ means the procedure is a pure PSO (the whole population is processed according to PSO operators), $hc = 1$ means pure GA (the whole population is optimized according to GA operators), while $0 < hc < 1$ means that the corresponding percentage of the population is developed by GA, while the rest with PSO technique.

In [9] GSO has been tested on problem of different dimensions: while for a small number of unknowns GSO performance is similar to GA and PSO ones, if the size of the problem increases, GSO behavior improves and outperforms GA and PSO during iterations. Moreover, the best hc value found in that preliminary study does not depend on the dimension of the problem, as it has been reported also in [11]. Furthermore, the obtained best hc value (0.2) means that, for a big-sized problem, the basic PSO can be strongly improved by adding a small percentage of genetic operators on the population.

In further studies a convenient value was found to be $0.2 \leq hc \leq 0.3$ for several fitness functions, but the authors extended the class of GSO algorithms by considering several variation rules for hc, in order to explore different hybridization strategies for the GSO algorithm and to compare new approaches with others already present in literature. The full set of hybridization rules considered by the authors is also reported in [9].

B. Recently developed enhancements

In [10] the authors introduced new rules for varying the hc value during the run, to combine more efficiently the properties of GA and PSO, in order to have a general procedure. In fact for engineering optimization problems the best mix of GA and PSO operators can be not always obvious. In particular there are situations where a fixed hc is the right choice, and others where a variable $hc(k)$ during the run is better. This means that also the "amount" of hybridization plays a role in affecting the performances of this procedure, but the correct value for hc is hardly known *a priori*. Therefore the authors chose to let the procedure adjust the $hc(k)$ value by itself during the iterations, according to a predefined set of rules.

In the *dynamical* approach the hc parameter is updated during iterations according to the following rule:

$$hc(k') = \begin{cases} hc(k) + \nu \frac{e^{-\xi \frac{k'}{K}}}{N} & if \quad \Delta\hat{f}(k') < \Delta\hat{f}(k) \\ hc(k) & if \quad \Delta\hat{f}(k') \geq \Delta\hat{f}(k) \end{cases} \tag{1}$$

where: $k' = k + \Delta k$, $\nu = \pm 1$ (*versus*), $\xi = 2$ (*damping*), and $\Delta\hat{f}(k') = \hat{f}(k') - \hat{f}(k)$; here $\hat{f}(k')$ is the best fitness value obtained after k' iterations and $\Delta k = 5$.

In the so-called *self-adapting* approach, the rule implemented for the self-adaptation comes, in part, from the very simple and reliable PSO technique: in fact, if we consider the value of $hc(k')$ in the k'-th iteration, then we can call $hc_v(k')$ the variation between $hc(k')$ and $hc(k)$ and so we can write:

$$hc(k') = hc(k) + hc_v(k') \tag{2}$$

Therefore, the problem is simply to find the right *velocity update* to properly change hc during the run. According to the PSO similarity, we can define a personal best hc_p value that has been obtained during the run and therefore write:

$$hc_v(k') = \omega \cdot hc_v(k) + \phi \cdot \eta \cdot (hc_p - hc(k)) \tag{3}$$

where hc_p is chosen analyzing the slope of the fitness score increase, during the iterations, i.e. if in iteration $\bar{k}$ the increment of fitness is higher than in the previous history, then $hc_p = hc(\bar{k})$.

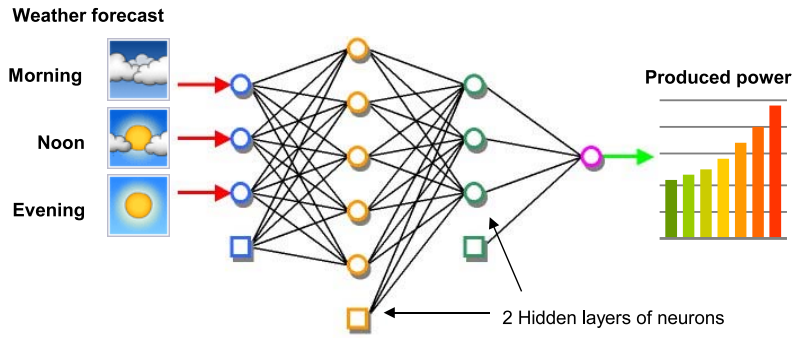

Fig. 1. Simplified view of the implemented feed-forward ANN with details on input, output, and hidden layers.

The overall results reported by the authors in cited papers show that, although the static GSO is generally the more fast and robust strategy in order to optimize multi-modal functions, a self-adaptive approach is a suitable and reliable solution especially when the proper hc value is not known *a priori* for a specific problem.

IV. Photovoltaic Energy Production Forecast

Energy management needs real-time optimization of energy production, transfer, storage and consumption in smart cities, villages and also rural areas [12], [13]. In fact, it is possible to conceive an energy hub as a micro-grid where electrical loads and small generation systems (such as renewables in the range of 25-100kW) are integrated into a Low Voltage (LV) distribution network with micro-storing systems (composed i.e. by the integration of electric vehicle into the grid infrastructure). By this point of view, an energy hub appears as a micro-marketplace since it should be composed by generation units, storing devices and a small number of consumers and it can operate interconnected to the main distribution grid or in autonomous way in case of external fault. By integrating instant access to energy forecasting into the architecture of energy hubs, it is possible to have economic scheduling of micro-generators and to trade energy and information with local providers [14].

Load and energy production forecasting has always been crucial for the effectiveness of power system planning and operation [5]. Lots of electric power companies are now forecasting power load based on traditional prediction methods [15]. However, since the relationship between power load and factors influencing power production is nonlinear, it is difficult to identify its nonlinearity by using traditional prediction methods.

Since the complexity of this scenario requires the capability to predict the dynamics of the system and to offer an optimal management it has been proposed the application of an Artificial Neural Network (ANN) integrated to an optimization algorithm in order to create a predictive model of the physical system and to provide an efficient control of resources and information [16]. To implement this approach, evolutionary optimization procedures adaptively and dynamically analyzing consumer profiles have been defined.

In recent years, renewable energy sources have increased the complexity of this scenario: solar power is getting more and more important as an alternative and renewable energy source, especially for small autonomous electrical power systems, villages and also rural areas. PV plants can also be connected to the traditional grid for energy distribution, but variations in solar power can cause, in general, voltage and frequency fluctuations.

Advanced forecasting through evolutionary computation techniques provide utilities with reliable production predictions and the opportunity to plan for additional power supply and to make proactive actions. This aspect can have an impact on the economic balance of the systems especially in an integrated smart grid solution perspective.

On the one hand these tools provide the ability to use stored energy or electric vehicle load to firms and renewable productions, increasing their intrinsic value. On the other hand, in this way the system-plant management is capable to plan appropriate preventive maintenance strategies in order to minimize energy losses due to unproductive suspensions. It can be estimated savings up to 0.5 M$/MW per year adopting these predictive algorithms in the renewable energy sector and just the 10% of these are due to an optimized operating efficiency [17].

Increased solar power penetration is possible if measures are taken concerning solar radiation forecasting. This procedure may also affect the energy efficiency of the conventional power stations, since it affects the operation point of the power units. Prediction of solar power is therefore of importance for efficient load management and operation of the system. It has been suggested in literature [18] that solar power forecast could be based on the use of ANNs. They are particularly appealing because of their ability to model an unspecified nonlinear relationship between production and weather variables. In fact, usually the complex nature of many engineering problems may involve to consider Soft Computing computational techniques in order to solve optimization tasks. ANN is an effective computing method, in order to easily process the dynamic behavior of time-varying variable, as previously presented by the authors in [5]. It is a computational model that simulates the features and behavior of the human brain neural networks. It consists

TABLE I

STRUCTURE OF INPUT LAYER

Neuron	Variable name	Range
i_1	Day	(1–366)
i_2	Time	(0:00–23:59)
i_3	Cloud cover index	(0–8)
i_4	Air temperature	(0–30) °C
i_5	Wind speed	(0–20) km/h
i_6	Air humidity	(0–100) %
i_7	UV index	(0–3)
i_8	Precipitation	(0–50) mm
i_9	Air pressure	(1000–1040) mbar

of an interconnected group of artificial neurons that suitably processes information according to the strength of connections among them. In more practical terms neural networks are non-linear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data.

As shown in Figure 1, weather forecasts are a key input when the ANN is used for forecasting. But, in case of rapid changes in solar radiation or temperature at the forecast day, produced power changes greatly and forecast error would increase. In traditional prediction methods the ANN uses all similar days data to learn the trend of similarity. However, learning all similar days' data is quite complex, and it does not help if weather conditions change suddenly. Therefore, it is necessary to integrate the neural network structure with real time information coming from local meteorological stations and, in particular, from surrounding regions and cities, where the weather change has already occurred.

V. SIMULATION RESULTS

In order to compare the different optimization approaches for training the ANN, a production forecasting problem has been considered with a different perspective compared to [15]. The focus is to highlight how fast and accurate the methods are in determining and optimizing the neural networks weights associated to the considered problem.

After a trial campaign, the network architecture which have provided better results present two hidden layers, as shown in Fig. 1. In particular, the number of neurons in the input layer is 22, as reported in Table I, which describes the meteorological parameter provided from the weather forecast service.

TABLE II

STRUCTURE OF OUTPUT LAYER

Neuron	Variable name	Range
o_1	SM 4200S-2 Produced power	(0–4) kW
o_2	SM 4200S-3 Produced power	(0–4) kW
o_3	SM 35C-4 Produced power	(0–35) kW

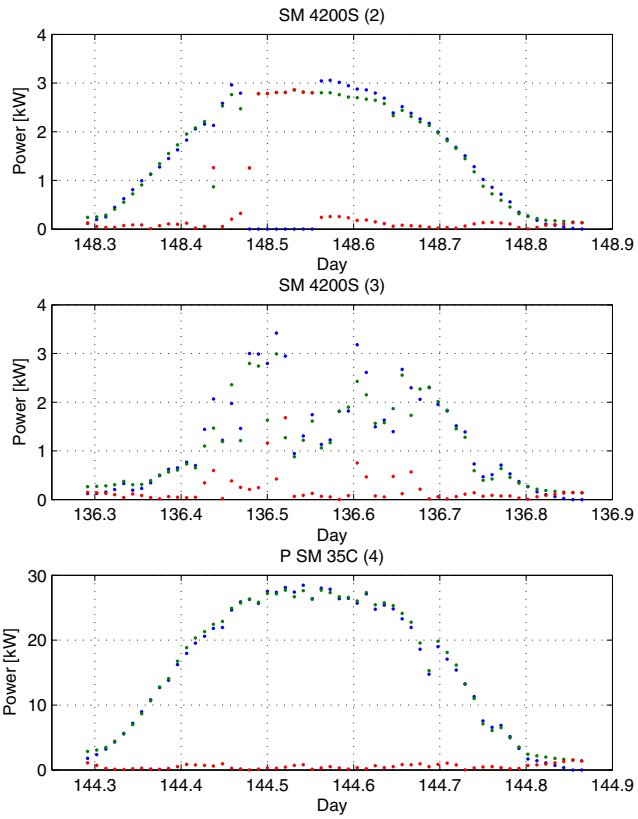

Fig. 2. Single-day detailed view for the three plants.

The neurons in the two hidden layers are, respectively, 17 and 9, while the output layer is composed of 3 neurons, since 3 different produced-power forecasts are considered: in fact, the authors performed simulations for three different photovoltaic power plants in order to test their method at various scale, since they have different productive capacity, as shown in Table II. To compare the numerical results it is important to underline that the three plant locations present exactly the same GPS reference points. Therefore the weather conditions can be assumed identical for the three PV plants.

Different time horizons can be considered in load forecasting: short term (day and several days), medium term (week and several weeks), long term (year and several years), and forecasts for different lead time can be used for different aims. We chose a short term time base for this specific application. In particular Fig. 2 shows some details for three days with different weather conditions. The neural network optimized by GSO exhibit good predictive performances in all the operative conditions, in a complete sunny day, a partly cloudy one and even a plant maintenance day.

Figure 3 shows the complete data set simulations around two months and the NN behaviour doesn't change with respect to the plant power size.

GSO here was used for the learning process of the artificial neural network and weights values of the ANN were changed to reach the minimum error in the network output in a faster

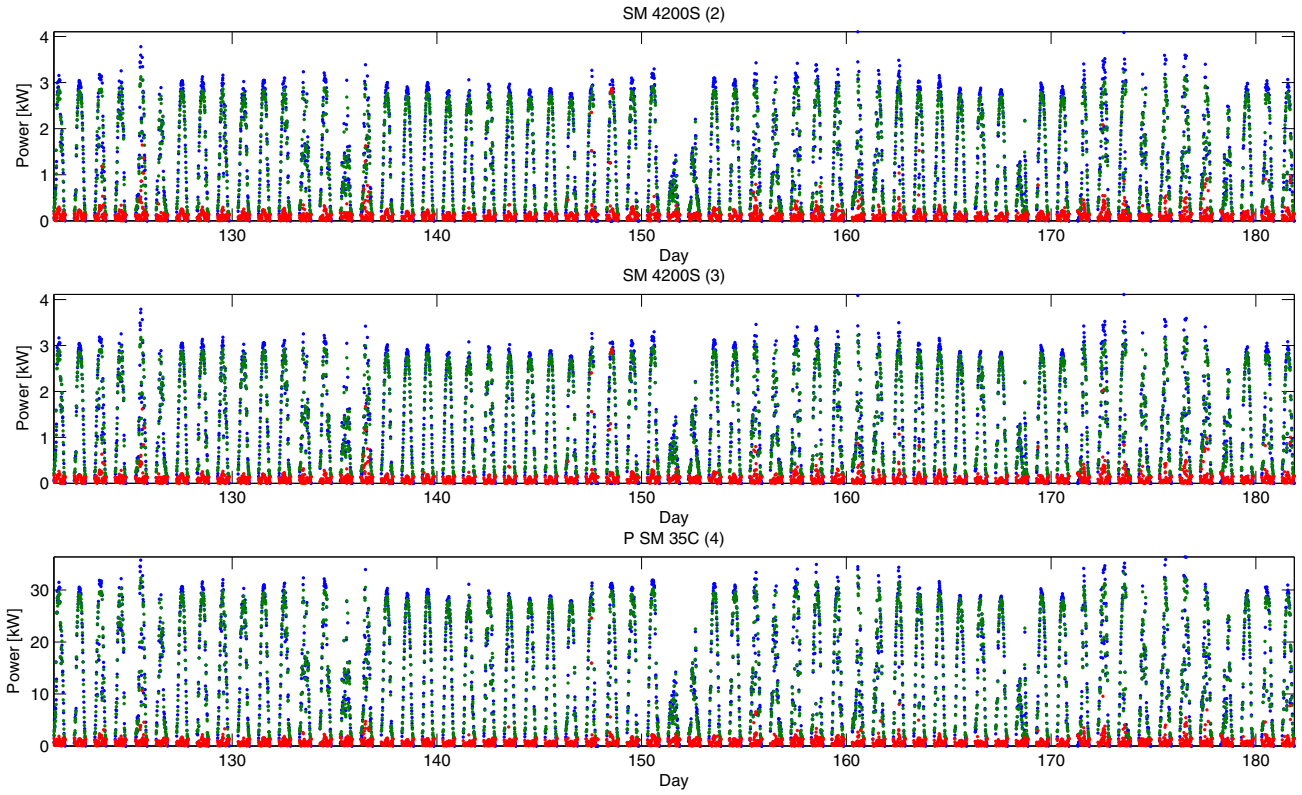

Fig. 3. The complete data-set from the three plants.

and easier way compared with EBP. Also other evolutionary procedures as PSO and standard GA were compared with the classical EBP in order to perform a comparison in terms of convergence rate and final obtained result (e.g. [19]), but GSO has already proven to outperform both GA and PSO.

VI. Conclusions

In this work the authors propose an effective and robust evolutionary procedure, called GSO, in order to replace classical training processes for Artificial Neural Networks. Energy plant prediction is very important in industrial applications. An accurate prediction enables to mitigate the price risk associated with the power exchange, for instance signing more bilateral contracts. Besides, a good estimate of the load profile results in avoiding economic penalties on the balancing market.

GSO emerges as a fast and simple method for ANNs training phase. Its ability to find the global best configuration of network weights makes it suitable for wide use in neural network applications for a wide spectrum of engineering problems.

Acknowledgment

This work was supported by Lombardy Regional Government, ICT-Metadristretti Action, under grant ID5109 - DGR N.6735.

References

[1] L. Hamm, B. Wade Brorsen, and M. T. Hagan, "Global optimization of neural network weights", *Proceedings of the 2002 International Joint Conference on Neural Networks, IJCNN02*, Vol. 2, May 2002, pp. 1228–1233.

[2] H. S. Hippert, C. E. Pedreira, R. C. Souza, "Neural networks for short-term load forecasting: a review and evaluation," *IEEE Transactions on Power Systems*, Vol. 16, No. 4, November 2001, pp. 798–805.

[3] W. Charytoniuk, M. S. Chen, "Very short-term load forecasting using artificial neural networks", *IEEE Transactions on Power Systems*, Vol. 15, February 2000, pp. 263-268.

[4] J. Y. Rahmat-Samii, E. Michielssen, *Electromagnetic Optimization by Genetic Algorithms*, New York, Wiley, 1999.

[5] E. Grimaldi, F. Grimaccia, M. Mussetta, R.E. Zich, "Pso as an effective learning algorithm for neural network applications", *Proceedings of the 3rd International Conference on Computational Electromagnetics and Its Applications ICCEA 2004*, 2004, pp. 557–560.

[6] D. W. Boeringer, D. H. Werner, "Particle Swarm Optimization Versus Genetic Algorithms for Phased Array Synthesis", *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 3, March 2004, pp. 771–779.

[7] J. Robinson, S. Sinton, Y. Rahmat-Samii, "Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna", *IEEE Proceedings of the International Symposium on Antennas and Propagation*, Vol. 1, 16-21 June 2002, pp. 314–317.

[8] Chia-Feng Juang,"A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design", *IEEE Transactions On Systems, Man, And Cybernetics-Part B: Cybernetics*, Vol. 34, No. 2, April 2004, pp. 997–1006.

[9] A. Gandelli, F. Grimaccia, M. Mussetta, P. Pirinoli, R.E. Zich, "Development and Validation of Different Hybridization Strategies between GA and PSO", *Proc. of the 2007 IEEE Congress on Evolutionary Computation*, Sept. 2007, Singapore, pp. 2782–2787.

[10] F. Grimaccia, M. Mussetta, R.E. Zich, "Genetical Swarm Optimization: Self-Adaptive Hybrid Evolutionary Algorithm for Electromagnet-

ics", *IEEE Transactions on Antennas and Propagation*, Vol.55, No.3, March 2007, pp. 781–785.

[11] A. Gandelli, F. Grimaccia, M. Mussetta, P. Pirinoli, R. E. Zich, "Genetical Swarm Optimization: an Evolutionary Algorithm for Antenna Design", *Journal of Automatika*, Vol. 47, No. 3–4, 2006, pp. 105-112.

[12] R.B. Chedid, S.H. Karaki, C.El-Chamali, "Adaptive fuzzy control for wind-diesel weak power systems", *IEEE Trans. Energy Conversion*, Vol. 15, 2000, pp. 71–78.

[13] A.N. Celik, A simplified model for estimating the monthly performance of autonomous wind energy systems with battery storage, *Renewable Energy*, Vol. 28, 2003, pp. 561–572.

[14] J.A. Pecas Lopes, Advanced Microgrids as a Component for Active Management of Distribution Networks, *Proc. Int. Conf. on Power Engineering, Energy and Electrical Drives, PowerEng '09*, 2009, pp. 7–8.

[15] T. Senjyu, H. Takara, K. Uezato, and T. Funabashi, "One-hour-ahead load forecasting using neural network,", *IEEE Transactions on Power Systems*, Vol. 17, No. 1, February 2002, pp. 113-118.

[16] M. Simonov, M. Mussetta, R.E. Zich, "Digital Energy: Clustering Micro Grids for Social Networking", *International Journal of Virtual Communities and Social Networking*, Vol. 1, No. 3, 2009, pp. 75–93.

[17] M. Simonov, M. Mussetta, A. Pirisi, F. Grimaccia, D. Caputo, R.E. Zich, "Real time energy management in smart cities by Future Internet", in: G. Tselentis et al. (Eds.), *Towards the Future Internet–Emerging Trends from European Research*, IOS Press, Amsterdam, NL, 2010, pp. 173–182.

[18] W. Taylor, and R. Buizza, Neural network load forecasting with weather ensemble predictions, *IEEE Transactions on Power Systems*, Vol. 17, No. 3, 2002, pp. 626–632.

[19] V.G. Gudise, G.K. Venayagamoorthy, "Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks", *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003. SIS '03*, 24-26 April 2003, p. 110–117.