Predicting Solar Power Output using Complex Fuzzy Logic

Omolbanin Yazdanbaksh
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Alberta, Canada
yazdanba@ulaberta.ca

Alix Krahn
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Alberta, Canada
akrahn@ualberta.ca

Scott Dick
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Alberta, Canada
sdick@ualberta.ca

***Abstract* – Photovoltaic (PV) power is one of the most promising renewable energy sources. However, it is also intermittent, and thus short-term forecasts of PV power generation are needed to integrate PV power into the electricity grid. This article compares two existing machine-learning approaches for forecasting (ANFIS and radial basis function networks) against a new approach based on complex fuzzy logic (ANCFIS). The proposed approach was more accurate in predicting power output one minute in advance on a simulated solar cell.**

INTRODUCTION

Fossil fuels are by their nature a finite and non-renewable resource. As they are also a principal energy source for the world, we will inevitably face major energy shortages in the future, unless the world transitions to other, renewable energy sources. Moreover, more than 90% of greenhouse gas emissions are associated with production and consumption of fossil fuels [1]. Renewable and clean energies such as solar, wind, biomass, and hydropower energy are promising alternative energy sources (and all but biomass have been widely deployed). Solar photovoltaic (PV) energy production in particular has grown by 58% annually in recent years [2].

PV power, however, is intermittent. The power produced by PV plants varies with the instantaneous solar irradiance and outside air temperature; power output will drop whenever a cloud passes across the sun [3]. Furthermore, PV power is zero until dawn, increases during the day until reaching a maximum in the afternoon, and then decreases and returns to zero at night. Temperature has an inverse effect on the PV power; as temperature increases, the power drops [4]. These characteristics are a poor fit for the power grid, in which the instantaneous power supply must equal the instantaneous demand. This is an advantage of fossil fuels – the output of fossil-fuel plants is steady. If a PV plant is feeding energy to the grid, and its production drops significantly, that deficit must be made up in real-time with only the briefest of delays – otherwise the grid's protection circuits will activate, and customers could be blacked out [5]. Thus, in order to dispatch PV power to the electrical grid, an accurate prediction of PV power output is needed.

In this paper, we explore the use of a novel time-series forecasting technique to predict PV power output. The Adaptive Neuro Complex Fuzzy Inferential System (ANCFIS) operationalizes the new idea of complex fuzzy sets in a machine-learning algorithm. Complex fuzzy sets have a membership function whose codomain is the unit disc of the complex plane [6]. In other words, each membership degree has a magnitude and phase. ANCFIS has shown promising results on time series forecasting [7] and also it has been suggested as a reasonable model for "approximately periodic" phenomena [8]. Thus, in this paper, we apply ANCFIS for prediction of PV power, comparing it against the adaptive neuro-fuzzy inference system (ANFIS) [9] and Radial Basis Function Networks (RBFNs) [10].

There are few public-domain datasets recording PV power for actual installations. This data is usually held as proprietary information by electric utility companies. In order to carry out this research, we therefore employed a public-domain of solar irradiance (the power delivered by the sun per unit area) and temperature. We converted this to PV power output using a model from [11]. The resulting dataset contains nearly one million observations, sampled at a rate of once per minute. This dataset is much larger, and more fine-grained, than the existing public-domain PV power datasets. Our results in this paper cover a subset of this data (7 days of observations).

The reminder of this paper is organized as follows: Section 2 reviews related work in PV power forecasting. Section 3 reviews the learning algorithms used in this paper. Section 4 presents our methodology; experimental results are presented in section 5 and section 6 offers a summary and discussion of future works.

II. LITERATURE REVIEW

PV systems are made up of panels containing PV materials such as silicon. The panels convert solar energy to electricity energy through the PV effect. Originally discovered in the 1800's, the PV effect occurs when photons strike an electron in a PV material, causing it to gain energy and become dissociated from its atom. In a practical PV system, these free electrons are pulled to one physical edge of a solar cell by an electric field created by doping the PV material (creating regions of positive and negative charge in the solar cell), thus creating an electrical current that can be tapped [12]. Large-scale grid-connected installations account for most current PV

978-1-4799-0348-1/13/$31.00 ©2013 IEEE

power production, although small-scale off-grid installations are more numerous [2].

As photovoltaic technologies developed, many researchers have investigated forecasting PV power. [13] used regression methods, while [14, 15] used time series approaches such as ARIMA and ARMA. [5] used adaptive time series model for on-line forecasting, and [16] used a fuzzy forecasting system which works based on class labels assigned to input patterns.

Artificial neural networks have shown promising results in the prediction of time series [17-19]; [20] applied RBF networks for predicting solar irradiation in order to forecast PV power. [21, 22] used ANFIS to model and simulate PV power supply systems. A review of research in this area may be found in [23].

III. BACKGROUND

A. ANFIS

ANFIS is a neural network designed to be equivalent to a Takagi-Sugeno-Kang (TSK) fuzzy inferential system [9]. It uses a hybrid of gradient descent and least-squares estimation to learn the network weights. Consider the following Takagi-Sugeno rules:

$$if\ x\ is\ A_1 and\ y\ is\ B_1 then\ f_1 = p_1x + q_1y + r_1$$

$$if\ x\ is\ A_2 and\ y\ is\ B_2 then\ f_2 = p_2x + q_2y + r_2$$

where x and y are input variables, A_i and B_i are fuzzy sets, and f_i is a linear function based on input variables. The corresponding ANFIS network structure is:

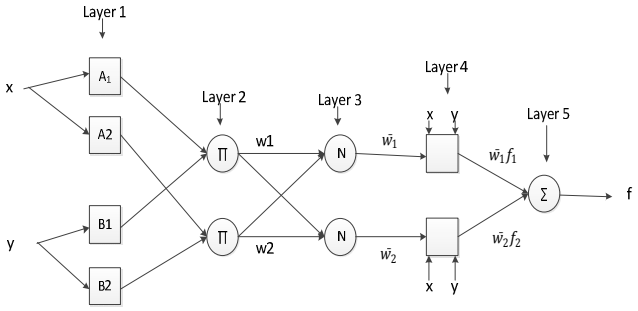

Fig 1. Equivalent ANFIS [9]

Each layer is described below:

- *Layer 1:*

$$O_{1,i} = \mu_{A_i} \quad (1)$$

where $O_{1,i}$ is the output of *i*-th node of layer 1, defined as the membership degree, μ_{A_i}, of an input value *x* for the fuzzy set A_i.

- *Layer 2:* This layer implements the product t-norm, giving us the firing strength of a rule.

$$w_k = \mu_{A_i}(x) \times \mu_{B_j}(y),\ \ i = 1,2 \quad (2)$$

where w_k is the firing strength of the *k*-th rule.

- *Layer 3:* Nodes in this layer normalize the rules' firing strengths.

$$\bar{w}_i = \frac{w_i}{w_1 + w_2}, \qquad i = 1,2. \quad (3)$$

- *Layer 4:* This layer computes the consequents of each rule.

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \quad (4)$$

Where p_i, q_i and *r* are constants, and $O_{4,i}$ indicates the output of i-th node of layer 4.

- *Layer 5:* Computes the overall output.

$$O_{5,i} = \frac{\sum_i \bar{w}_i f_i}{\sum_i \bar{w}_i} \quad (5)$$

where $O_{5,i}$ is the output of the *i*-th node of layer 5.

In the ANFIS network, there are two kinds of parameters that can be updated:

1. Antecedent parameters (parameters of membership functions). In a Gaussian membership function:

$$\mu(x) = \frac{-(x - c_i)^2}{2{\sigma_i}^2} \quad (6)$$

$\{c_i, \sigma_i\}$ are updatable parameters.

2. The consequent coefficients $\{p_i, q_i, r_i\}$.

Updating parameters using only gradient descent is slow and it is susceptible to being trapped in local minima. Instead, a hybrid learning procedure is used to train the network. In the forward pass, least squares optimization is applied to update the consequent parameters, while antecedent parameters are updated through gradient descent in the backward pass [9].

B. ANCFIS

ANCFIS is a neuro-fuzzy system based on complex fuzzy logic. Complex fuzzy systems were first proposed by Ramot et.al [6], which defined complex fuzzy memberships as being any value in the unit disc of the complex plane. In other words, each membership degree $\mu(x)$ is a vector:

$$\mu(x) = r(x).e^{(jw(x))} j = \sqrt{-1} \quad (7)$$

where $r(x) \in [0,1]$ is the magnitude and $w(x)$ is the phase. Other works have extended this concept; [24] introduced the isomorphic complex fuzzy logic, in which the magnitude is the truth value, and the phase is a modifier. The property of rule interference (constructive or destructive interference between the phases of rule firing strengths) was also introduced in this paper. [8] suggested another formation of membership degrees which studied the phase and magnitude simultaneously. It also suggested a sinusoidal membership function could capture periodic behaviours. [25] studied distances and orderings for complex fuzzy sets. [26] studied the mathematical-logic properties of complex fuzzy logic.

[7, 27] proposed the first machine-learning realization of complex fuzzy logic (ANCFIS). ANCFIS uses a sinusoidal membership function as follows:

$$r(\theta) = dsin(a(\theta = x) + b) + c \quad (8)$$

where $r(\theta)$ is amplitude and θ is the phase of the membership grade of element *x*. The four parameters $\{a, b, c, d\}$ act as follows: a changes the frequency of the sine wave, b gives a phase shift whereas c shifts the wave vertically, and d changes the amplitude of the sine wave. Since the amplitude of complex fuzzy memberships is limited to [0,1], the parameters must satisfy the following conditions:

$$0 \;\; d + c \;\; 1, \qquad 1 \;\; c \;\; d \;\; 0 \quad (9)$$

ANCFIS is based on ANFIS, although there are major differences between them. In the forward pass, ANCFIS employs convolution to obtain the membership degree input; this can be interpreted as the degree of matching between the inputs and each of the membership functions. In the backward pass, there is no closed-form solution for the partial derivative of error w.r.t the parameters $\{a, b, c, d\}$. Therefore, gradient descent learning is combined with a derivative free optimization technique. This is the Variable Neighbourhood Chaotic Simulated Annealing (VNCSA) algorithm.

VNCSA is a variation of the simulated annealing (SA) algorithm, which emulates formation of a crystal. SA begins at a high temperature which allows the objective function to accept new solutions of lower utility than the current one. As the temperature lowers, lower-utility solutions are no longer accepted. In SA, the entire search space is explored, which makes the procedure slow. However, in chaotic annealing, the search space will be limited to fractal subsets of the search space, defined by chaotic maps. VNCSA employs the Ulam-von Neumann map defined as [7]

$$y_{n+1} = 1 - r{y_n}^2 \quad y_n \in [-1,1] \quad (10)$$

where the function is chaotic and covers the codomain of $[-1,1]$ for parameter values $r \;\; 2$.

VNCSA uses three steps to find a solution for the objective function:

1. Generating new solutions by iterating the Ulam-von Neumann map in a neighbourhood.
2. Checking if the new solutions satisfy constraints of the objective function.
3. Comparing the new value of the objective function with the current one and the temperature.

The size of the neighbourhood and the temperature are updated after reaching a maximum number of iterations. VNCSA is controlled by several parameters: T_{max} and T_{min} are the initial and final temperatures, L_{max} is the number of iterations in a given temperature, M is the number of solutions generated per iteration, $\beta$is the temperature change factor, and α and ω control the neighbourhood size.

The ANCFIS architecture has six layers [7]:

- Layer 1: In this layer, each input vector is convolved with the complex fuzzy sets. At first, the membership function is sampled by:

$$r_k(\theta_k) = dsin(a\theta_k + b) + c, \;\; \theta_k = \frac{2\pi}{n}k, \quad k = 1,2,\dots,n \quad (11)$$

where *n* is the length of the input vector. Then the sampled membership functions are convolved with the input vector:

$$conv = \sum_{k=1}^{2n-1} \sum_{j=\max(1,k+1-n)}^{\min(k,n)} f(j)g(k+1-j) \quad (12)$$

where $g(.)$ is the sampled membership function (in rectangular coordinates), $f(.)$ is the input vector and the summations are complex-valued. Since the convolution must be restricted to the unit disc of the complex plane, the output is normalized by the Eliot function:

$$O_{1,i} = \frac{conv}{1 + |conv|} \quad (13)$$

- *Layer 2:* In this layer, the firing strength of a fuzzy rule is calculated.

$$O_{2,i} = \prod\nolimits_i O_{1.i}\,, i = 1,2,\dots,|O_1| \quad (14)$$

where O_1 is the number of nodes in layer 1. Currently, only the complex product (identified as a complex fuzzy convolution in [8] is supported. For univariate time series, neurons in this layer reduce to identity functions.

- *Layer 3*: The output of each node represents the normalized firing strength of a rule.

$$O_{3,i} = \bar{w}_\iota = \frac{w_i}{\sum_{j=1}^{|O_2|}|w_j|}, i = 1,2,\dots,|O_2| \quad (15)$$

where O_2 is the number of rules. Only the magnitude is normalized; phases are unchanged.

- *Layer 4*: This layer implements the idea of rule interference [24] using the dot product:

$$O_{4,i} = w_i^{DP} = \bar{w}_\iota \cdot \sum\nolimits_{i=1}^{|O_3|} \bar{w}_\iota \quad (16)$$

where '·' represents the dot product. If phases are aligned, the amplitude of the output (rule firing strength) will increase, otherwise it will decrease.

- *Layer 5:* This layer implements the linear consequent function

$$O_{5,i} = w_i^{DP}\left[\sum\nolimits_{j=1}^{n} p_{i,j}x_j + r_i\right] \quad (17)$$

where $x_j$is the j-th value of the input vector and $\{p_\iota, \bar{r}_\iota\}$ are consequent parameters, as in ANFIS.

- *Layer 6:* This layer sums all incoming signals.

Fig. 2 is an example of ANCFIS with two rules.

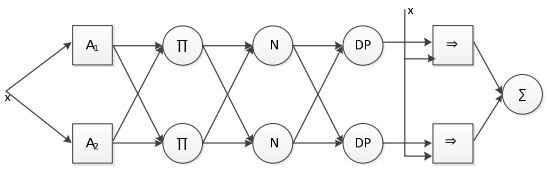

Fig 2. Two-rule ANCFIS network [7],

One of the main advantages of ANCFIS in time series studies is that it treats an entire window of the time series as one input, which decreases the number of rules substantially. This is an essential feature of the convolution function in Layer 1; convolution only makes sense if the time-ordering of observations is preserved. Machine learning algorithms that use delay embeddings of a time series (like ANFIS, or RBFN, see Section III) destroy this time ordering, since each input value is treated as belonging to an orthogonal dimension. For a univariate time series, ANCFIS creates *m* rules where *m* is the number of membership functions defined for the input, irrespective of the number of observations in the input window. On the other hand, the delay embedding in ANFIS generates m^r rules where *r* is the number of data points in the input vector.

Other complex fuzzy machine-learning architectures have also been proposed; the CNFS architecture [28-31] is also a complex-valued variation on ANFIS, but uses complex Gaussian fuzzy sets (which are not periodic). [32] construct complex fuzzy sets directly from time-series data.

C. RBFN

The radial basis function network (RBFN, see Figure 3) is a two layer neural network consisting of a hidden layer with nonlinear transformation and a linear layer output [10].

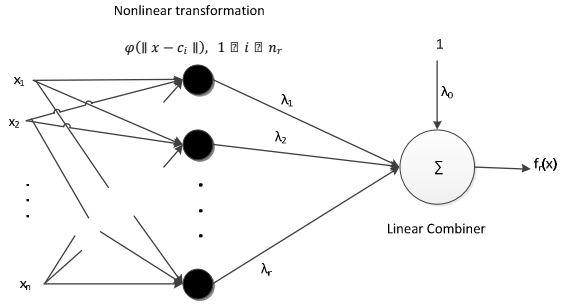

Fig 3: Schematic of RBF network[10].

The network follows a mapping from inputs to output according to [10]:

$$f_r(x) = \lambda_r + \sum_{i=1}^{n_r} \lambda_i \varphi(\| x - c_i \|) \quad (18)$$

where $x \in \mathbb{R}^n$ is the input vector, $\varphi(.)$ is a given function from $\mathbb{R}^+$ to $\mathbb{R}$, $\|.\|$ denotes the Euclidean norm, λ_i are the weights $0 \ i \ n_r$, $c_i \in \mathbb{R}^n$ are the RBF centers, $1 \ i \ n_r$, n_r is the number of centers, and $\varphi(.)$ can be a Gaussian function [10]:

$$\varphi(x) = exp\left(\frac{-\| x - c_i \|^2}{\beta^2}\right) \quad (19)$$

where β is the spread parameter. We used the MATLAB implementation of RBFN (newrb.m) in which centers are iteratively selected from the existing dataset, and then the weights are determined via ordinary least-squares [10].

IV. METHODOLOGY

In this paper, we use a dataset containing solar irradiance $(\frac{W}{m^2})$ and temperature(℃)measured every one minute at the Lowry Range Solar Station, obtained from [33]. The corresponding power is obtained by using a model proposed in [11]. Nighttime measurements were deleted as they had a constant value of zero. A chronological single-split design was used in all experiments, with the first four days of data used for training and the latter three days reserved for testing. This is a subset of the full solar-power dataset we have created from the NREL data, which contains over one million observations.

Machine learning algorithms are usually applied to a delay embedding of a time series dataset (also called the *lagged* representation), rather than the raw dataset itself. Each lag is a previous observation of the time series, with a collection of *k* lags considered a delay vector. According to Takens' embedding theorem, if a sufficient number of lags are taken (a sufficiently large delay vector constructed), the resulting space is equivalent to the original state space of the system that gave rise to the time series. Thus, the evolution of the time series can be predicted from its trajectory through the embedding space, because this trajectory is equivalent to the original system's trajectory in its state space. We can sample each successive value in the time series, every second value, etc. All of these choices are mathematically equivalent to one another (although a good choice of this "delay" parameter is helpful in modeling). Thus, the critical parameter that has to be determined is the number of lags (dimensions of the embedding space) [34].

To determine the delay embedding of this dataset for ANFIS and RBFN, we turn to the false nearest neighbors technique [34], implemented in the Tisean software package [35]. The idea is that if a delay vector is mapped to a neighborhood in the embedding space, its one-step-ahead evolution is also mapped to the one-step-ahead evolution of that neighborhood. If not, then that point only seemed to fall in the neighborhood due to unresolved projections of the true embedding space [34]. Fig. 4 shows the fraction of nearest false neighbors as a function of embedding dimension for different distance threshold ratios which indicates that an input vector with length of 45 is sufficient for the time series.

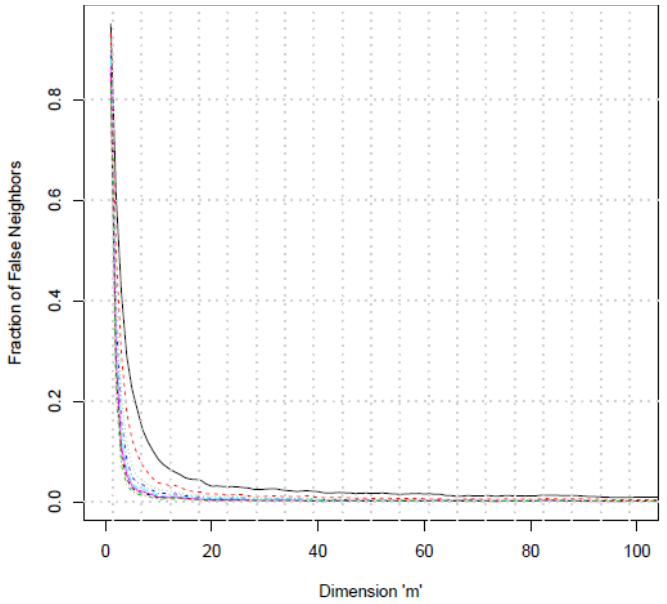

Fig 4: Fraction of nearest false neighbor as a function of embedding dimension

For our ANCFIS experiments, we assign two complex fuzzy sets of the form in Eq. (8) to the single input vector (of length 45). As previously noted, this will mean only 2 rules. A trial-and-error parameter exploration finds that the best parameter values are:

$\eta(StepSize) = 0.01, Increaserate = 1.2, Decreaserate = 0.9$
$T_{max} = 100,\ L_{max} = 2, \alpha = 0.99, \omega = 0.95, T_{min} = 0.01,$
$\beta = 0.98, M = 400$

In ANFIS, on the other hand, an input vector length of 45 lags with 2 fuzzy sets per input yields an infeasible number of rules (2^{45}, more than 17 trillion), a classic case of the curse of dimensionality. To deal with the problem, we effectively subsample the dataset, by accepting only 5 lags, with a delay of 9 between each. This yields a more tractable $2^5 = 32$ rules. The gradient parameters in ANFIS are also obtained by parameter exploration:

$\eta(StepSize) = 0.1, Increaserate = 1.2,$
$Decreaserate = 0.9$

In RBFN, we use the same delay vectors as ANFIS. Parameters explored in the RBFN are the number of neurons in the hidden layer and the spread of each RBF. The best parameters were:

$Width = 90, Neuron = 20$

The results of the three approaches are compared in terms of root mean square error (RMSE):

$$RMSE = \sqrt{MSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \quad (20)$$

where y_i is the expected value, $\hat{y}_i$ is the observed value, and n is the number of inputs.

V. RESULTS

In this section, results for the three algorithms are compared. Table 1 shows the RMSE for both training and testing datasets.

Table 1: Comparison of error for the approaches

RMSE	Train	Test
ANCFIS	6.509	7.713
ANFIS	4.91482	19.649
RBFN	12.573	13.869

Table 2 shows a more detailed comparison between ANFIS and ANCFIS.

Table 2: Comparison of ANFIS and ANCFIS

	# Rules	#Nodes	Length of input vector
ANCFIS	2	11	45
ANFIS	32	92	5

Clearly, the number of data points in ANCFIS is not the same as RBFN and ANFIS. ANCFIS considers 45 data points per input, whereas ANFIS and RBFN consider only 5 data points. We did another experiment to consider what happens when ANCFIS is given only the same five lags as in ANFIS and RBFN. This necessarily involves a new parameter exploration, in which we found the best settings were:

$\eta(StepSize) = 0.01, Increaserate = 1.2, Decreaserate = 0.7$
$T_{max} = 100,\ L_{max} = 2, \alpha = 0.99, \omega = 0.95, T_{min} = 0.01,$
$\beta = 0.98, M = 400$

which are the same as the previous experiment, except that the decrease rate changes to 0.7. Table 3 gives the result. Plainly, there is an improvement in ANCFIS' accuracy when the lagged data is used rather than the entire input window.

Table 3: ANCFIS Result

RMSE	Train	Test
ANCFIS	7.163640	6.978

VI. CONCLUSION

We have applied ANCFIS to PV power prediction, which is the first implementation of ANCFIS in the solar energy area. Our proposed approach is more accurate in a one step ahead prediction on the given data set compared to ANFIS and RBFN either with the lagged input, or an entire input window. In future work we will further examine the use of lagged inputs in ANCFIS, as it seems that this may improve accuracy.

REFERENCES

[1] Staff, "Inventory of U.S. greenhouse gas emission and sinks: 1990-2010," United States Environmental Protection Agency, Washington, DC, 2012.

[2] J. L. Sawin, *et al.*, "Renewables 2012 global status report," REN21 Secretariat, Paris, France2012.

[3] Y. Huang, *et al.*, "Comparative study of power forecasting methods for PV stations," in *2010 International Conference on Power System Technology*, 2010, pp. 24-28.

[4] V. Omubo-Pepple, C. Israel-Cookey, and G. Alaminokuma, "Effects of temperature, solar flux and relative humidity on the efficient conversion of solar energy to electricity," *European Journal of Scientific Research,* vol. 35, pp. 173-180, 2009.

[5] P. Bacher, H. Madsen, and H. A. Nielsen, "Online short-term solar power forecasting," *Solar Energy,* vol. 83, pp. 1772-1783, 2009.

[6] D. Ramot, R. Milo, M. Friedman, and A. Kandel, "Complex fuzzy sets," *IEEE T. Fuzzy Syst.,* vol. 10, pp. 171-186, 2002.

[7] Z. Chen, S. Aghakhani, J. Man, and S. Dick, "ANCFIS: A Neuro-Fuzzy Architecture Employing Complex Fuzzy Sets," *IEEE Trans Fuzzy Syst,* vol. 19, pp. 305-322, 2011.

[8] S. Dick, "Toward complex fuzzy logic," *Fuzzy Systems, IEEE Transactions on,* vol. 13, pp. 405-414, 2005.

[9] J.-S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE T. Syst., Man, Cybern.,* vol. 23, pp. 665 -685, 1993.

[10] S. Chen, C. Cowan, and P. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *Neural Networks, IEEE Transactions on,* vol. 2, pp. 302-309, 1991.

[11] A. Bellini, S. Bifaretti, V. Iacovone, and C. Cornaro, "Simplified model of a photovoltaic module," presented at the Applied Electronics, 2009.

[12] Staff. (2013, April 29). *Semiconductors and the Built-In Electric Field for Crystalline Silicon Photovoltaic Cells*. Available: http://www.eere.energy.gov/basics/renewable_energy/semiconductors.html

[13] M. Kudo, A. Takeuchi, Y. Nozaki, H. Endo, and J. Sumita, "Forecasting electric power generation in a photovoltaic power system for an energy network," *Electrical Engineering in Japan,* vol. 167, pp. 16-23, 2009.

[14] A. Moreno-Munoz, J. de la Rosa, R. Posadillo, and V. Pallares, "Short term forecasting of solar radiation," in *Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on*, 2008, pp. 1537-1541.

[15] B. Chowdhury and S. Rahman, "Is central station photovoltaic power dispatchable?," *Energy Conversion, IEEE Transactions on,* vol. 3, pp. 747-754, 1988.

[16] E. D'Andrea and B. Lazzerini, "Fuzzy forecasting of energy production in solar photovoltaic installations," in *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, 2012, pp. 1-8.

[17] G. P. Zhang, "An investigation of neural networks for linear time-series forecasting," *Computers & Operations Research,* vol. 28, pp. 1183-1202, 2001.

[18] E. Cadenas and W. Rivera, "Short term wind speed forecasting in La Venta, Oaxaca, México, using artificial neural networks," *Renewable Energy,* vol. 34, pp. 274-278, 2009.

[19] S. A. Kalogirou, "Applications of artificial neural-networks for energy systems," *Applied Energy,* vol. 67, pp. 17-35, 2000.

[20] L. Ciabattoni, G. Ippoliti, S. Longhi, M. Cavalletti, and M. Rocchetti, "Solar irradiation forecasting using RBF networks for PV systems with storage," in *Industrial Technology (ICIT), 2012 IEEE International Conference on*, 2012, pp. 699-704.

[21] A. Mellit and S. A. Kalogirou, "Neuro-fuzzy based modeling for photovoltaic power supply system," in *Power and Energy Conference, 2006. PECon'06. IEEE International*, 2006, pp. 88-93.

[22] A. Mellit and S. A. Kalogirou, "ANFIS-based modelling for photovoltaic power supply system: A case study," *Renewable Energy,* vol. 36, pp. 250-258, 2011.

[23] A. Mellit and S. A. Kalogirou, "Artificial intelligence techniques for photovoltaic applications: A review," *Progress in Energy and Combustion Science,* vol. 34, pp. 574-632, 2008.

[24] D. Ramot, M. Friedman, G. Langholz, and A. Kandel, "Complex fuzzy logic," *IEEE T. Fuzzy Syst.,* vol. 11, pp. 450-461, 2003.

[25] G. Zhang, T. S. Dillon, K.-Y. Cai, J. Ma, and J. Lu, "Operation properties and delta-equalities of complex fuzzy sets," *Int. J. Approximate Reasoning,* vol. 50, pp. 1227-1249, 2009.

[26] D. E. Tamir and A. Kandel, "Axiomatic theory of complex fuzzy logic and complex fuzzy classes," *Int. J. Computers, Communication & Control,* vol. 4, pp. 562-576, 2011.

[27] J. Man, Z. Chen, and S. Dick, "Towards Inductive Learning of Complex Fuzzy Inference Systems," presented at the Proc. NAFIPS Int. C., San Diego, CA, USA, 2007.

[28] C. Li and T.-W. Chiang, "Function Approximation with Complex Neuro-Fuzzy System Using Complex Fuzzy Sets – A New Approach " *New Generation Computing,* vol. 29, pp. 261-276, 2011.

[29] C. Li and T.-W. Chiang, "Complex fuzzy model with PSO-RLSE hybrid learning approach to function approximation," *International Journal of Intelligent Information and Database Systems* vol. 5, pp. 409-430, 2011.

[30] C. Li, T. Wu, and F.-T. Chan, "Self-learning complex neuro-fuzzy system with complex fuzzy sets and its application to adaptive image noise canceling," *Neurocomputing,* vol. 94, pp. 121-139, 2012.

[31] C. Li, "Complex Neuro-Fuzzy ARIMA Forecasting — A New Approach Using Complex Fuzzy Sets," *IEEE Trans Fuzzy Syst,* In Press.

[32] J. Ma, G. Zhang, and J. Lu, "A method for multiple periodic factor prediction problems using complex fuzzy sets," *IEEE T. Fuzzy Syst.,* vol. 20, pp. 32-45, 2012.

[33] Staff. (2013, April 29, 2013). *Lowry Range Solar Station (LRSS)*. Available: http://www.nrel.gov/midc/lrss/

[34] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*. New York, NY: Cambridge University Press, 1997.

[35] R. Hegger, H. Kantz, and T. Schreiber, "Practical implementation of nonlinear time series methods: The TISEAN package," *Chaos,* vol. 9, pp. 413-435, Jun 1999.