Chapter 8
Time-Series Forecasting via Complex Fuzzy Logic

Omolbanin Yazdanbakhsh and Scott Dick

Abstract Adaptive neuro-complex-fuzzy inference system (ANCFIS) is a neuro-fuzzy system that employs complex fuzzy sets for time-series forecasting. One of the particular advantages of this architecture is that each input to the network is a windowed segment of the time series, rather than a single lag as in most other neural networks. This allows ANCFIS to predict even chaotic time series very accurately, using a small number of rules. Some recent findings, however, indicate that published results on ANCFIS are suboptimal; they could be improved by changing how the length of an input window is determined, and/or subsampling the input window.

We compare the performance of ANCFIS using three different approaches to defining an input window, across six time-series datasets. These include chaotic datasets and time series up to 20,000 observations in length. We found that the optimal choice of input formats was dataset dependent, and may be influenced by the size of the dataset. We finally develop a recommended approach to determining input windows that balances the twin concerns of accuracy and computation time.

8.1 Introduction

Time-series forecasting has emerged as the first major application of complex fuzzy sets and logic, which were first described by Ramot in [1]. Beginning in 2007, complex-valued neuro-fuzzy systems were developed to inductively learn forecasting models; these include the adaptive neuro-complex-fuzzy inference system (ANCFIS) architecture [2], and the family of complex neuro-fuzzy system (CNFS) architectures [3]. Both ANCFIS and CNFS are modifications of the well-known ANFIS architecture, in which complex fuzzy sets and complex-valued network signals are used. These architectures showed that complex fuzzy sets were naturally useful in creating very accurate forecasting models. ANCFIS in particular is also very parsimonious;

S. Dick (✉) · O. Yazdanbakhsh
Department of Electrical and Computer Engineering, University of Alberta, 2nd Floor ECERF Building, 9107–116 Street, Edmonton, AB T6G 2V4, Canada
e-mail: dick@ece.ualberta.ca

O. Yazdanbakhsh
e-mail: yazdanba@ualberta.ca

© Springer Science+Business Media, LLC 2015

A. Sadeghian, H. Tahayori (eds.), *Frontiers of Higher Order Fuzzy Sets,*
DOI 10.1007/978-1-4614-3442-9_8

experiments in [2] showed that this architecture could forecast even chaotic systems with no more than three complex fuzzy rules.

One of the key reasons why ANCFIS is so parsimonious is its input format. Most generic machine-learning algorithms must use lagged inputs in order to create a forecasting model. While this approach is mathematically sound, it means that the number of inputs to the learning algorithm has to be equal to the number of lags required to reconstruct the state space of the system that generated the time series (i.e., to form a delay reconstruction in the sense of Takens [4]). This directly leads to a combinatorial explosion in the complexity of the model. However, due to the nature of complex fuzzy sets, ANCFIS does not use lagged inputs; rather, an entire windowed segment of the time series is taken as a single input to the network, greatly reducing the curse of dimensionality. Recent experiments reported in [5] indicated that we might be able to further improve the accuracy of ANCFIS by subsampling the input windows. This is possible because in ANCFIS, we use sinusoidal membership functions for the complex fuzzy sets, which are sampled and convolved with the input window. Subsampling the input window simply implies that we also sample the complex fuzzy sets at a lower rate. Our goal in the current chapter is to determine if such subsampling generally leads to improved accuracy, or if this was a dataset-specific effect.

We compare the forecast accuracy of ANCFIS using three different approaches to identifying and sampling input windows on six time-series datasets. Two of these (a realization of the Mackey–Glass map and the Santa Fe Laser A dataset) are known to be chaotic; the remainder are observations of physical processes (sunspots, stellar brightness, waves, solar power production). All but the last one have been previously studied in the forecasting community and in [2]. The solar power dataset was developed in our laboratory and is discussed in depth in [5]. In [2], the length of the input windows for each of the five datasets was set at one "period" in the dataset, as determined by ad-hoc inspection. We explore the use of heuristics from [4] to construct two different delay embeddings of the time series: one with an "optimal" delay between each lag and one with unit delays between each lag. These input lags are concatenated together in chronological order to form our input windows.

The remainder of this chapter is organized as follows. Section 8.2 provides an essential background on complex fuzzy sets and logic, as well as the ANCFIS architecture. Section 8.3 describes our datasets and experimental methodology. We provide our experimental results in Sect. 8.4 and close with a summary and discussion of future work in Sect. 8.5.

8.2 Literature Review

8.2.1 Complex Fuzzy Sets and Logic

Ramot et al. in 2002 proposed the *complex fuzzy set* (CFS) as a fuzzy set whose membership function takes complex-valued grades, bounded by the unit circle [1]:

$$\mu_s(x) = r_s(x).\mathrm{e}^{(jw_s(x))} j = \sqrt{-1}, \tag{8.1}$$

where $r_s(x) \in [0,1]$ is the magnitude and $w_s(x)$ is the phase of the *complex fuzzy set* S. Ramot defined complex fuzzy intersection and union to act solely on the magnitude of complex fuzzy sets; the phase was treated as a means of adding application-dependent context to the CFS. In 2003, Ramot et al. proposed an isomorphic *complex fuzzy logic* (CFL) based on the generalized modus ponens inference rule. To implement complex fuzzy implication, he suggested the complex product [6]:

$$\mu_{A \to B}(x, y) = \mu_A(x) \cdot \mu_B(y), \tag{8.2}$$

where $\mu_{A \to B}(x, y)$ is the complex-valued membership function of the implication, and $\mu_A(x), \mu_B(y)$ are both CFS. To aggregate multiple rules, Ramot proposed a complex-valued weighted sum called vector aggregation as given below [6]:

$$v: \{a|a \in C, |a| \le 1\}^n \to \{b|b \in C, |b| \le 1\} \tag{8.3}$$

$$\mu_A(x) = v(\mu_{A_1}(x), \mu_{A_2}(x), \ldots, \mu_{A_n}(x)) = \sum_{i=1}^{n} w_i \mu_{A_i}(x), \tag{8.4}$$

where $w_i \in \{a|a \in C, |a| \le 1\}$ for all i, and $\sum\limits_{i=1}^{n} |w_i| = 1$. In vector aggregation, rules can interfere constructively or destructively with each other.

Dick in 2005 showed that considering the phase as a relative quantity in Ramot et al.'s papers [1, 6] can be interpreted as rotational invariance, meaning that if two vectors undergo rotation by φ radians about the origin, their union, intersection, or complement will be rotated by the same amount [7]. It was shown that the algebraic product and the traditional complement, $f(x) = -x$, are not rotationally invariant. He then proposed a new formulation of membership degree by considering amplitude and phase simultaneously. The algebraic product was shown to be a conjunction operator, and the existence of a dual disjunction operator was proved. He then argued that capturing the behavior of approximately periodic phenomena was a possible application for CFL, and sinusoidal functions were suggested as appropriate complex fuzzy membership functions.

Tamir et al. in [8] proposed a new definition for complex fuzzy degrees using the Cartesian representation of complex numbers where both real and imaginary parts may vary from 0 to 1. These are called "pure" complex fuzzy sets and are defined as [8]:

$$\begin{aligned} \mu(V, z) &= \mu_r(V) + j\mu_i(z) \\ \mu_r, \mu_i &\in [0,1], \end{aligned} \tag{8.5}$$

where $\mu_r(V)$ and $\mu_i(Z)$ are the real and imaginary part of the pure complex fuzzy membership grade, $\mu(V, z)$. An interpretation of pure complex fuzzy grades was proposed based on the pure fuzzy class of order 1; "a fuzzy class is a finite or infinite collection of objects and fuzzy sets which can be unambiguously defined and complies with the class theory; a pure fuzzy class of order 1 can have only fuzzy sets" [8]. Consider the pure complex fuzzy membership function, $\mu_\Gamma(V, z)$:

$$\mu_\Gamma(V, z) = \mu_r(V) + j\mu_i(z). \tag{8.6}$$

Let Γ be a complex fuzzy class, V a fuzzy set, and z a variable in the universe of discourse U. $\mu_\Gamma(V, z)$ can be interpreted as the degree of membership of z in V and the degree of membership of Vin Γ. Complement, intersection, and union operations were also proposed for the complex fuzzy classes in the paper [8].

Tamir et al. proposed a first-order predicate CFL in [9]. In the CFL system, based on ŁΠ system (propositional logic system) by Běhounek et al. [10], a complex fuzzy proposition, $\Gamma = \Gamma_r + j\Gamma_i$, is considered as a composition of two propositions each with truth value in the interval [0, 1]. The proposed CFL was extended to generalized propositional CFL, applicable in multidimensional fuzzy propositional and predicate logic, through definitions in [10] based on fuzzy Łukasiewicz logical system [11]. Tamir et al. [12] proposed an extended complex post-logical system (ECPS) based on the extended Post system (EPS) of order $p > 2$ by DiZenzo [13]. One of the possible applications of the proposed system is in discrete processes such as digital signal processing (DSP), real-time applications, and embedded systems.

Salleh [14] defined complex Atanassov's intuitionistic fuzzy sets (CAIFS). Intuitionistic fuzzy sets were introduced by Atanasov [15]; they record the degree of membership and nonmembership of an element in a set, each indicated by a value in [0, 1]. In complex intuitionistic fuzzy sets, the degrees of membership and nonmembership are each drawn from the unit circle in the complex plane. Basic operations on CAIFS, including complement, union, and intersection, were also presented in the paper. Yager and Abbasov [16] defined Pythagorean membership grades as a subset of complex fuzzy grades, $\mu = r\mathrm{e}^{\theta}$ with the properties $r \in$ [0,1] and $\theta \in [0, \frac{\pi}{2}]$.

Zhang et al. [17] studied different operations and their properties on the complex fuzzy sets introduced by Ramot et al. [1] when phases are restricted to [0, 2π]. A new definition for distance of complex fuzzy sets was introduced in the paper:

$$d(A, B) = \max\left(sup_{X\in U} \left|r_A(X) - r_B(X)\right|, \frac{1}{2\pi} sup_{X\in U} \left|arg_A(X) - arg_B(X)\right| \right), \tag{8.7}$$

where $d(A, B)$ is the distance of two complex fuzzy sets $A = r_A(X)e^{j\cdot arg_A(X)}$ and $B = r_B(X)e^{j\cdot arg_B(X)}$. Then, based on this definition, δ-equalities of complex fuzzy sets were proposed; two complex fuzzy sets, A and B, are δ-equal if and only if $d(A, B) \leq 1 - \delta, 0 \leq \delta \leq 1$. Zhang et al. also defined δ-equalities for complex fuzzy relations [18]. Alkouri and Salleh [19] introduced a distance between two CAIFS, and proposed complex intuitionistic fuzzy relations by extending operations defined in [17, 18]. They also proposed projection and cylindrical extensions for CAIFS.

A few authors investigated complex-valued membership grades before Ramot's work. Moses et al. [20] proposed a CFS with memberships drawn from the unit square $U \rightarrow$ [0,1] × [0,1]. These complex fuzzy sets contain two fuzzy sets represented by the real and imaginary domains. Linguistic coordinate transformations (i.e., the relationship between linguistic variables when transforming from Cartesian to polar coordinates or vice versa) were also studied. Nguyen et al. [21] proposed an optimization approach to select the best representation of complex fuzzy sets for a particular application.

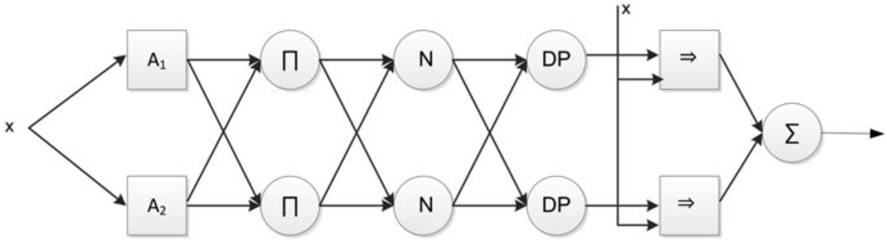

Fig. 8.1 Two-rule ANCFIS architecture for univariate time-series problems [2]

8.2.2 Adaptive Neuro-Complex-Fuzzy Inference System

Chen et al. [2, 22] proposed the first inductive machine-learning realization of the CFL proposed by Dick [7] and Ramot [6]. The ANCFIS architecture is a relative of Jang's well-known ANFIS. The main differences are (1) ANCFIS uses a sinusoidal membership function as follows:

$$r(\theta) = d\sin\left(a(\theta = x) + b\right) + c, \tag{8.8}$$

where $r(\theta)$ is the amplitude and θ is the phase of the membership grade of element x; (2) an additional layer implements rule interference, inspired by Ramot's vector aggregation in [6]; (3) the network signals are complex-valued up through this rule interference layer; and (4) the learning algorithm incorporates a derivative-free optimization component (Fig. 8.1).

As suggested by Dick [7], one possible application of CFL is capturing approximately periodic behavior of phenomena; Chen et al. [2] suggested that time-series forecasting is a good example of such behaviors. Thus, sinusoidal functions are candidate complex fuzzy membership functions since a periodic function can be represented by a Fourier series, i.e., a sum of sin and cosine functions. In Eq. (8.8), 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:

$$0 \le d + c \le 1,\ 1 \ge c \ge d \ge 0. \tag{8.9}$$

The use of a sinusoidal CFS in ANCFIS also implies an important operational difference between ANCFIS and ANFIS (and indeed most other machine-learning algorithms). In using ANFIS and other algorithms for time-series forecasting, input vectors containing lagged values of a variate are presented to the network to predict the next value of the variate. The components of the input vectors are considered orthogonal; thus, to predict $f(t)$, the components $f(t-1), f(t-2), \ldots, f(t-n)$ of an input vector are presented as separate inputs to the system. However, this cannot work in ANCFIS, because matching a sinusoidal membership function to an observation requires that we keep the phase information in our inputs. Orthogonal lagged inputs destroy this phase information by definition. Instead, in ANCFIS, we take a sliding window of the variate as a single input, $[f(t-1), f(t-2), \ldots, f(t-n)]$, and then match that window to the membership functions. This implies that ANCFIS requires only a single input for each variate of a time series, whereas systems using a lagged input require $\prod_{i=1}^{n} r_i$ inputs, where r_i is the number of lags for a given variate, and n is the number of variates. Thus, ANCFIS significantly reduces the combinatorial explosion inherent in time-series forecasting [2].

As the network signals in ANCFIS are complex-valued, the backward-pass computations in the network must also be different from ANFIS. Like ANFIS, ANCFIS uses a hybrid learning rule where consequent parameters are updated on the forward pass, and antecedent parameters on the backward pass. Indeed, as network signals are real-valued at the consequent layer (layer 5 in ANCFIS), we employ the same least-squares algorithm as ANFIS. However, the backward pass requires back-propagation of complex-valued signals; and ultimately, there is no closed-form expression for the partial derivative of network error with respect to the CFS parameters in Eq. (8.8). As described in [2], we use gradient descent to determine the back-propagating error signals until layer 1, and then use a derivative-free optimization technique (a variant of simulated annealing) to determine the update to the CFS parameters. This technique is the variable neighbourhood chaotic simulated annealing (VNCSA) algorithm.

In chaotic simulated annealing (CSA), the generation of new candidate solutions is governed by a chaotic map instead of a random number generator. This potentially makes the algorithm faster, as we only search a fractal subset of the total solution space. In VNCSA, we create an initial population of solutions by iterating the logistic map starting from a random point. We create new solutions by iterating the Ulam-von Neumann map and adding that value to the existing best solution, weighted by a neighborhood size factor. As with other simulated-annealing algorithms, we will allow a new solution with a worse objective function to replace the current solution with a probability that depends on the current temperature T. As the routine continues, T is gradually reduced by a constant factor at each iteration. When T is updated, the neighborhood size is also updated, depending in part on how much the objective function changed from the last iteration to the current one (thus limiting the neighborhood in which new candidate solutions are generated in the next iteration). For additional details, please see [2].

The ANCFIS architecture has six layers as follows [2]:

- *Layer 1*: In this layer, the input vector is convolved with the membership function. First, the membership function is sampled over one period by

$$r_k(\theta_k) = d\sin(a\theta_k + b) + c, \quad \theta_k = \frac{2\pi}{n}k, \qquad (8.10)$$
$$k = 1, 2, \ldots, n$$

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

$$conv = \sum_{k=1}^{2n-1} \sum_{j=\max(1,k+1-n)}^{\min(k,n)} f(j)g(k+1-j), \qquad (8.11)$$

where $f(.)$ is the input vector and $g(.)$ is the sampled membership function (in Cartesian coordinates). To ensure that the convolution sum remains within the unit disc, it is normalized using the Eliot function:

$$O_{1,i} = \frac{conv}{1 + |conv|}. \tag{8.12}$$

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

$$O_{2,i} = \prod_i O_{1,i}, \quad i = 1, 2, \ldots, |O_1|, \tag{8.13}$$

where $|O_1|$ is the number of nodes in layer 1. For univariate time series, neurons in this layer reduce to the identity function.
- *Layer 3*: The output of each node represents the normalized firing strength of a rule:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{\sum_{j=1}^{|O_2|} |w_j|}, \quad i = 1{,}2, \ldots, |O_2|, \tag{8.14}$$

where $|O_2|$ is the number of rules. This layer only normalizes the magnitude whereas phases are unchanged.
- *Layer 4*: This layer realizes the property of "rule interference" from [6], using the dot product:

$$O_{4,i} = w_i^{DP} = \overline{w_i} \cdot \sum_{i=1}^{|O_3|} \overline{w_i}, \tag{8.15}$$

where $|O_3|$ is the number of nodes in layer 3 and $\sum_{i=1}^{|O_3|} \overline{w_i}$ is the complex sum. Both constructive and destructive interference are possible.
- *Layer 5*: This layer implements the linear consequent function:

$$O_{5,i} = w_i^{DP} \left[\sum_{j=1}^{n} p_{i,j} x_j + r_i \right], \tag{8.16}$$

where w_i^{DP} is the output of layer 4, x_j is the *j*th data point if the input vector, n is the length of the input vector, and $p_{i,j},\ r_i$ are the parameters of a linear function of x_j. $\left\{p_{i,j},\ r_i\right\}$ are obtained in the forward pass by least-squares estimation.

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

The ANCFIS architecture is related to the more general area of complex-valued neural networks (CVNN) where inputs, outputs, biases, and weights can take on complex values [23–26]. In the specific domain of neuro-fuzzy systems, a few other studies have investigated complex-valued network signals in the ANFIS framework. Li and Jang [27] proposed CANFIS, which accepts the real and imaginary parts of a complex number as separate inputs in ANFIS. Malekzadeh-A and Akbarzadeh-T

[28] designed a different CANFIS, using CFS in the first layer—but preprocessing them through a CVNN first. This architecture is limited to complex-valued inputs and outputs. A complex steepest descent algorithm and a complex least-square estimator were used to update complex membership function parameters and complex bias and weight parameters, respectively.

Other complex fuzzy machine-learning architectures have also been proposed; Aghakhani and Dick [29] proposed an online learning algorithm for ANCFIS. The architecture uses recursive least squares instead of least-square algorithm in the forward pass, and applies the downhill simplex algorithm instead of VNCSA in the backward pass. Li and Chiang [3] proposed a different variation of ANFIS called the CNFS. Their learning algorithm is a hybrid of particle swarm optimization (PSO) and recursive least-square estimator (RLSE) algorithm. Li and Chiang [30] extended CNFS by using a Gaussian-type CFS as:

$$c\text{Gaussian }(h, m, \sigma) = Re(c\text{Gaussian }(h, m, \sigma)) + jIm(c\text{Gaussian }(h, m, \sigma)) \tag{8.17}$$

$$Re(c\text{Gaussian }(h, m, \sigma)) = \exp\left[-0.5\left(\frac{h-m}{\sigma}\right)^2\right] \tag{8.18}$$

$$Im(c\text{Gaussian }(h, m, \sigma)) = -\exp\left[-0.5\left(\frac{h-m}{\sigma}\right)^2\right] \times \left(\frac{h-m}{\sigma^2}\right), \tag{8.19}$$

where $\{m, \sigma\}$ are the mean and spread of the Gaussian function, and h is the input. This paper also proposed the "dual-output property," which refers to treating the real and imaginary components of the output as separate variates (and thus, the network can naturally handle bivariate time series). Several other papers from this research group explore other variations on the CNFS architecture; Li and Chan [31] used the artificial bee colony (ABC) algorithm instead of PSO, and applied CNFS to image restoration (Li et al. [33] used the CNFS from [30] for image restoration as well). Li and Chan [32] used CNFS with the ABC-learning algorithm for knowledge discovery. Li and Chiang [35] replaced PSO in the CNFS proposed by [30] with a multiple-swarm variation called hierarchical multi-swarm PSO, while also proposing a new Gaussian-type complex fuzzy set:

$$c\text{Gaussian }(h, m, \sigma, \lambda) = r_s(h, m, \sigma) \exp\left(jw_s(h, m, \sigma, \lambda)\right) \tag{8.20}$$

$$r_s(h, m, \sigma) = \text{Gaussian }(h, m, \sigma) = \exp\left[-0.5\left(\frac{h-m}{\sigma}\right)^2\right] \tag{8.21}$$

$$w_s(h, m, \sigma, \lambda) = -\exp\left[-0.5\left(\frac{h-m}{\sigma}\right)^2\right] \times \left(\frac{h-m}{\sigma^2}\right) \times \lambda, \tag{8.22}$$

where $\{m, \sigma, \lambda\}$ are the mean, spread, and phase frequency factor for the complex fuzzy set, and h is the input. Li and Chiang [36] developed a model based on CNFS proposed in [35] where the consequent layer is an ARIMA model, CNFS-ARIMA. Li and Chiang [34] applied the system proposed in [35] for financial time-series forecasting. Li et al. [37] proposed a CNFS using the Gaussian membership function introduced in [35], which updates the premise and consequent parameters based on PSO-RLSE-learning algorithm. To minimize the rule base of CNFS, a clustering algorithm called FCM-based splitting algorithm (FBSA) is employed [38].

Ma et al. [39] proposed a product–sum aggregation operator for the complex fuzzy sets introduced by Ramot et al. [1]. Based on this operator, a prediction method was developed to solve multiple periodic factor prediction problems in multisensory data fusion applications containing semantic uncertainty and periodicity. We can view this as the first CFL inferential system whose rules are declaratively specified rather than inductively learned. Alkouri et al. [40] defined linguistic variables of complex fuzzy sets based on the ideas in [39]. Linguistic hedges (as introduced by Zadeh [41]) were also extended to complex fuzzy sets. Hamming, Euclidean, normalized Hamming, and normalized Euclidean distances and their boundaries were also obtained for complex fuzzy sets. Deshmukh et al. [42] designed a hardware implementation for the CFL proposed in [6].

8.2.3 Delay Embedding of a Time Series

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; a delay vector is a chronologically ordered sequence of lags [43]:

$$S_n = \left(s_{n-(m-1)\tau},\ \ s_{n-(m-2)\tau},\ \ldots\ ,\ s_n\right), \tag{8.23}$$

where S_n is the delay vector with dimension m, and τ specifies the delay between successive observations. Each delay vector can be considered as a point in a state space, whose dimensions are the chosen lags. According to Takens' embedding theorem [4], if a sufficient number of lags are taken (i.e., a sufficiently large delay vector constructed), the resulting state 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.

Takens' embedding theorem does not, however, provide a constructive method for determining the parameters m and τ; instead, we need to use heuristics to determine *adequate* values for both parameters. Mathematically, embeddings with different τ are equivalent to each other; however, in real-world data, the choice of the delay parameter has a significant influence on the utility of an embedding. Small values of τ generally lead to higher correlations between observations in each delay vector; and thus the distribution of delay vectors (and hence the apparent state-space trajectory) tend to be concentrated in a small region of the embedding space, potentially obscuring important features of the trajectory. On the other hand, large values of τ tend to make observations in a delay vector poorly correlated. This tends to result in the delay vectors becoming a weakly differentiated cloud of points, with little apparent structure. Heuristics for determining a "best" value for τ include taking the first zero of the autocorrelation function, or the first minimum of the time-delayed mutual information. These are given in Eqs. (8.24) and (8.25), respectively [4]:

$$c_{\tau} = \frac{1}{\sigma^2} \left\langle \left(s_n - \langle s \rangle\right) \left(s_{n-\tau} - \langle s \rangle\right) \right\rangle , \tag{8.24}$$

where c_τ is the autocorrelation between values of s_n and $s_{n-\tau}$ where there is a time lag of τ between them. $\langle . \rangle$ indicates average over time, and σ^2 denotes the variance.

$$I(\tau) = \sum_{i,j} p_{ij}(\tau) \ln p_{ij}(\tau) - 2 \sum_{i} p_i \ln p_i, \tag{8.25}$$

where $I(\tau)$ is the mutual information between s_n and $s_{n-\tau}$. To compute this value, consider a histogram of s_n. p_i is the probability that s_n lies in the i-th interval, and p_{ij} is the joint probability that s_n has values in the i-th interval and $s_{n-\tau}$ has values in the j-th interval.

To determine the dimensionality of the embedding space, Kennel et al. [44] proposed applying the false nearest neighbors technique. The Euclidean distance between one delay vector and its rth nearest neighbor in the embedding space of dimension m is given by [44]:

$$R_d^2(n,r) = \sum_{k=0}^{m-1} \left[s_{n-k\tau} - s_{n-k\tau}^r \right]^2, \tag{8.26}$$

where R_d is the Euclidean distance and $s_{n-k\tau}$ are elements of the delay vector in the embedding space. When the dimension of the embedding space increases to $m+1$, the delay vectors have one more coordinate which is $s_{n-m\tau}$. The Euclidean distance between the delay vectors in the new embedding space is calculated as [44]:

$$R_{d+1}^2(n,r) = R_d^2(n,r) + \left[s_{n-m\tau} - s_{n-m\tau}^r \right]^2. \tag{8.27}$$

Thus, the false nearest neighbor method can be stated as the following criterion [44]:

$$\frac{\left| s_{n-m\tau} - s_{n-m\tau}^r \right|}{R_d(n,r)} > R_{\text{tol}}, \tag{8.28}$$

where R_{tol} is a threshold. That means increasing the embedding dimensionality must not increase the distance between two neighbors more than the given threshold. The estimated number of dimensions is determined by plotting the fraction of false nearest neighbors in the dataset against the number of dimensions, for several different values of the threshold R_{tol}. When all of the curves saturate at a low value, we consider that to be the best estimate of the necessary embedding dimensionality for the dataset. All three heuristics can be computed using the TISEAN software package [43].

8.3 Methodology

8.3.1 Experimental Design

The goal of the current chapter is to evaluate alternative time-series representations in forecasting with ANCFIS. We explore three different approaches for setting the length of the input windows and subsampling them. The first is the approach used in [2], the second is the delay embedding technique from Sect. 8.2.3, and the third is a hybrid of the two. Specifically:

- Method 1 is to make an ad hoc determination of the length of one "period" in the dataset. The input window is set to this length and is not subsampled. As this method was used in [2] for five of the six datasets, we will use the same period lengths as in that paper. For the sixth dataset (solar power forecasting), the length of a period is clearly 1 day (see our discussion in Sect. 8.3.2.1 for further details).
- Method 2 is to form a delay embedding, relying on the heuristics from Sect. 8.2.3 to guide our selection of the embedding dimensionality and delay. We will use the mutual-information heuristic to select the delay, and the false nearest neighbors technique to select the dimensionality. We can consider this a subsampling of an input window; for dimensionality m and delay τ, we select every τ-th sample from an input window of length $((m+1)\cdot\tau)+1$.
- Method 3 is to assume that the delay is always equal to 1 and to employ the false nearest neighbors technique for selecting the embedding dimensionality under that assumption. This will mean that the input window is again not subsampled.

Our experiments follow a common design in the time-series forecasting literature. We use a single-split design, with all elements of the training set chronologically earlier than elements of the testing set. The embedding dimension and delay are determined from the training set only, and are then applied to both the training and testing sets. The results of the three input representations are compared in terms of root mean square error (RMSE):

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{\sum_{i=1}^{n} \left(y_i - \overset{\wedge}{y_1}\right)^2}{n}}, \tag{8.29}$$

where y_i is the expected value, $\hat{y}_1$ is the predicted value, and n is the number of inputs.

8.3.2 Datasets

8.3.2.1 Solar Power Dataset

This dataset was created in [5], as very few high-resolution solar power datasets are publicly available. It was developed from a public dataset recording air temperature

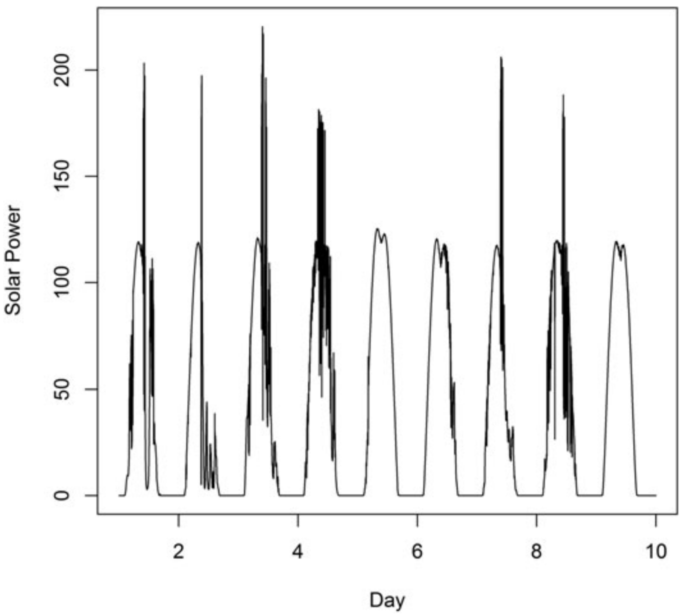

Fig. 8.2 Training set for the solar power dataset

(°C) and total solar radiation (W/m^2) measured every 1 min from May 30, 2008, to August 12, 2012, at the Lowry Range Solar Station. Total solar radiation is the sum of direct irradiance, diffuse irradiance, and ground-reflected radiation, and is measured by a LICOR LI-200 Pyranometer mounted 7 ft above ground level on a Rotating Shadow Band Radiometer (RSR). Air temperature is measured by a thermometer mounted 5 ft above ground level inside a naturally aspirated radiation shield [45]. These two measurements are the principal variables affecting power output from a photovoltaic cell; we convert them to an estimated power output using the model proposed in [46], following the specifications of a Photowatt PW 2650-24V panel. The result is a new time series recording solar power production at 1-min intervals over a period of 5 years, giving 2,212,520 observations.[1] For our experiments in the current chapter, we use data from July 31, 2012, to August 13, 2012, giving us 20,000 measurements. The dataset is split into 2/3 and 1/3 for the training and testing sets, respectively. A plot of the training set is given in Fig. 8.2.

In this dataset, the length of one period is clearly 1 day, or 1440 observations. When we attempted to take this entire period as an input window following Method 1, we found that the computation time is infeasibly long on our computer system (Intel(R) Core$^{\text{TM}}$ 2 Duo CPU E8400 @ 3.00 GHz, 4 GB of memory). We were thus

[1] Available online at http://www.ualberta.ca/yazdanba/SolarData.txt.

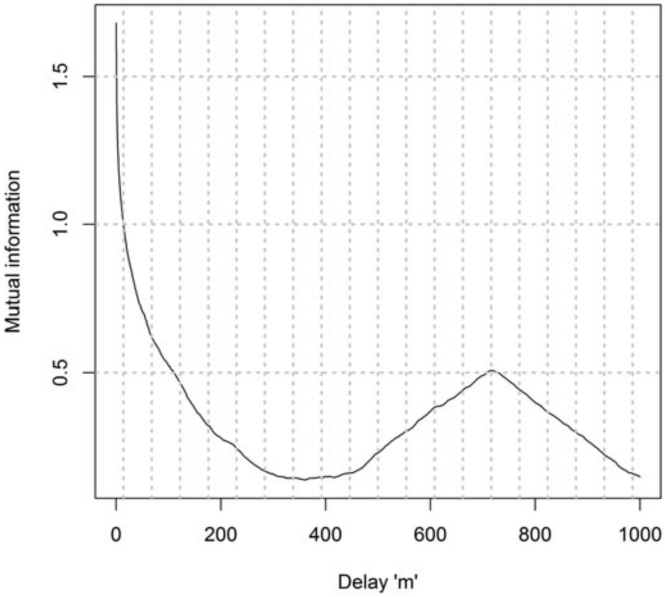

Fig. 8.3 Mutual information versus delay

forced to subsample this window; we take every tenth measurement, giving us 145 observations, which we concatenate together in chronological order. For Method 2, we plot the mutual information statistic versus delay in Fig. 8.3. The first minimum of the curve occurs at $\tau = 370$, and so, we adopt this value as our delay parameter. With this delay, the false nearest neighbor plot is given in Fig. 8.4. For all values of R_{tol} examined, the curves saturate at $m = 12$, and so, we adopt this value as our embedding dimension. We can also view this input as a subsampled window of length $(m-1)\tau = 4070$ data points. For Method 3, we set the delay $\tau = 1$, and rerun the false nearest neighbor procedure. This time, the apparent minimum embedding dimension is 15, and so, we adopt this as our window length.

8.3.2.2 Mackey–Glass Dataset

This dataset is a realization of the Mackey–Glass differential equation, a frequently used benchmark for testing time-series forecasting algorithms. The equation is given by [47]:

$$\dot{x}(t) = \frac{0.2x(t-\delta)}{1 + x^{10}(t-\delta)} - 0.1x(t). \tag{8.30}$$

This equation is useful because it exhibits chaotic behavior for appropriate choices of the parameter δ. In particular, we follow the design in [47], where $124 < t < 1123$, $x(0) = 1.2$, $\tau = 17$, and the time step is 0.1. The first 500 data points are used as the

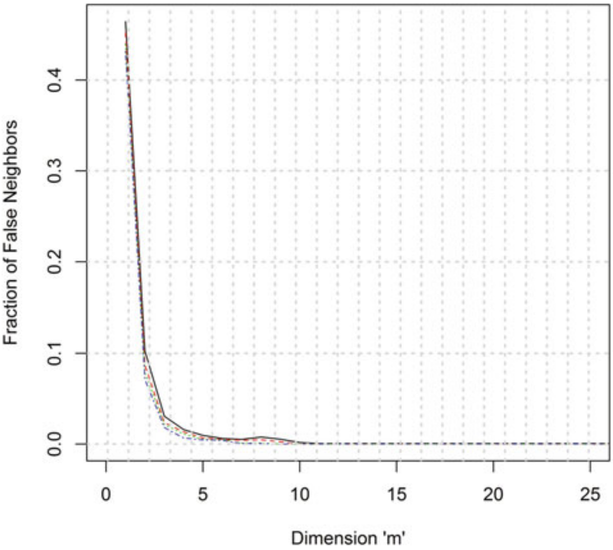

Fig. 8.4 Fraction of false nearest neighbors versus dimensionality

training set and the remaining data points form the testing set; this is also the design that was used in [2]. We omit the details of how the embedding dimension and delay are determined in the interest of brevity; we simply note that they followed the same process as in Sect. 8.3.2.1.

8.3.2.3 Santa Fe Laser A

The Santa Fe time-series forecasting competition held in 1991 has left us six datasets for use as benchmarks. The "Laser A" dataset is frequently used, as it also exhibits chaotic behavior. This dataset records the amplitude of an 81.5-micron 14NH3 cw (FIR) laser being controlled by the Lorenz system of equations for modeling turbulent flow; with appropriate choices of parameters, the Lorenz system is also chaotic. We normalized the data to the range [0, 1] (matching [2]). The dataset has 1000 data points; the first 900 data points are the training set, and the last 100 are the testing set (again, matching [2]).

8.3.2.4 Sunspot Dataset

This dataset consists of observations of a physical system: it is the average number of sunspots observed for each day in a calendar year, recorded from 1700–1979 (280 observations) [48]. The first 220-year measurements, years 1700–1920, are used as the training set and the remaining points form the testing set; this again matches [2].

Table 8.1 Results for the solar power dataset

Method	Input vector design		RMSE
	Dimension	Delay	
1	145	10	5.293
2	12	370	3.1057
3	15	1	4.846

8.3.2.5 Stellar Brightness

This dataset is also made up of observations of a physical system. The time-series records the daily observed brightness of a variable star on 600 successive midnights. The first 480 night measurements are used as the training set, and the remainder form the testing set. This is again the same split used in [2].

8.3.2.6 Waves

This dataset also records observations of a physical system, but this time in a laboratory setting. The time-series measures the oscillation of a cylinder suspended in a tank of water every 0.15 s. There are a total of 320 data points of which the first 256 data points form the training set and the last 64 points are the testing set. This also matches the split in [2].

8.4 Experimental Results

The following tables record the input window parameters and out-of-sample error for each of our input representations in Sect. 8.3.1, over each of the datasets described in Sect. 8.3.2. Table 8.1 presents our results for the solar power dataset.

Quite plainly, in this dataset, the traditional delay embedding was superior to the down-sampled "one period" input window, even though the smaller dimensionality provides far fewer tunable parameters in the consequent layer. It was also superior to the unit-delay input window created from Method 3, even though Method 2 again resulted in fewer dimensions.

Table 8.2 presents our results on the Mackey–Glass dataset; the results for Method 1 are taken from [2]. Interestingly, this dataset presents a completely different picture than the solar power dataset. The traditional delay embedding gave—by two orders of magnitude—the least accurate predictions, even though the most accurate approach (Method 3) differed only in the delay length (reflecting what was stated in Sect. 8.2.3; all values of the delay parameter are theoretically equivalent, but in practice a good choice of the delay parameter can significantly impact the performance of a forecasting algorithm).

Table 8.3 presents our results for the Santa Fe Laser A dataset. This time, Method 1 appears to be the best, while Method 2 yields the worst results. Table 8.4 presents

Table 8.2 Results for the Mackey-Glass dataset

Method	Input vector design		RMSE
	Dimension	Delay	
1 [2]	44	1	5.57e-4
2	9	11	0.015
3	9	1	5.29e-4

Table 8.3 Results for Santa Fe Laser A

Method	Input vector design		RMSE
	# Lags	Delay	
1 [2]	8	1	0.033
2	9	2	0.114
3	9	1	0.067

Table 8.4 Results for sunspot

Method	Input vector design		RMSE
	# Lags	Delay	
1 [2]	12	1	0.091
2	5	4	0.103
3	6	1	0.089

Table 8.5 Results for stellar brightness

Method	Input vector design		RMSE
	# Lags	Delay	
1 [2]	27	1	7.49e-3
2	6	7	1.4e-2
3	6	1	1.3e-2

Table 8.6 Results for Waves

Method	Input vector design		RMSE
	# Lags	Delay	
1 [2]	12	1	0.0032
2	4	4	0.00866
3	4	1	0.0104

our results for the sunspot dataset. This time, the traditional delay embedding is somewhat worse than Methods 1 and 3; however, the difference is not very large. In addition, Methods 1 and 3 are nearly indistinguishable from one another.

Table 8.5 presents our results for the Stellar Brightness dataset. This time, Method 1 is substantially better than Methods 2 or 3; furthermore, Method 2 is slightly less accurate than Method 3.

Table 8.6 presents our results for the Waves dataset. Once again, Method 1 proved to be the most accurate. However, this time Method 3 was less accurate than Method 2.

8.4.1 Discussion

As with many other experiments in pattern recognition, our general finding is that the "best" input representation for ANCFIS is dataset dependent. In five of our six datasets, the traditional delay embedding was clearly outperformed by the window-based approaches, but in our single largest dataset, Method 2 was clearly the best. Four times, Method 1 was either the best approach or virtually identical to Method 3 and superior to Method 2.

Our findings do, however, suggest which methods seem more likely to succeed in future experiments. Method 3 was the better approach once, and essentially tied with Method 1 on a second dataset. This method also seems to lead to lower embedding dimensionalities than Method 1. This still matters in ANCFIS, even though we have reduced the combinatorial explosion of rules seen in other machine-learning methods using orthogonal lags. A complexity analysis carried out in [2] indicates that the running time of both the least-squares estimate of the consequent parameters, and the VNCSA optimization of the CFS parameters, depend linearly on the length of the input vector (this explains why running ANCFIS on the full input window for the solar power dataset took an infeasibly long time). Thus, with Method 3 often providing strong results, and usually resulting in a significantly smaller input window, this seems to be the most effective initial approach to modeling a time series with ANCFIS. We would recommend that Method 1 be tried next, and finally, the traditional delay embedding.

8.5 Conclusions

In this study, we have explored three different approaches for representing time-series inputs for the ANCFIS machine-learning algorithm. We compared input windows based on an ad hoc determination of what constitutes one "period" in the dataset; the traditional delay embedding, guided by the mutual-information and false-nearest-neighbor heuristics; and the use of only the false-nearest-neighbor heuristic, across six time-series datasets. While the "best" method appears to be dataset dependent, we found enough evidence that we recommend Method 3 as the best combination of accuracy and expected computation time.

In future work, we intend to explore the use of ANCFIS as a stream data mining algorithm. An online-learning version of ANCFIS was developed in [29], meaning that incremental learning is possible in the ANCFIS framework. This characteristic, along with the demonstrated success of ANCFIS as a time-series forecasting algorithm, indicates that it may be an appropriate algorithm for modeling data streams. One of the datasets we will use in evaluating this possibility is the full solar power dataset.

Acknowledgments This research was supported in part by the Natural Science and Engineering Research Council of Canada under grant no. RGPIN 262151, and in part by Transport Canada under grant no. RES0017834.

References

1. D. Ramot et al., Complex fuzzy sets. IEEE Trans. Fuzzy Syst. **10**(2), 171–186 (2002)
2. Z. Chen et al., ANCFIS: A neurofuzzy architecture employing complex fuzzy sets. IEEE Trans. Fuzzy Syst. **19**(2), 305–322 (2011)
3. C. Li, T.-W. Chiang, Complex neuro-fuzzy self-learning approach to function approximation, in *Intelligent Information and Database Systems*. (Springer, Berlin, 2010), pp. 289–299
4. H. Kantz, T. Schreiber, *Nonlinear Time Series Analysis. Cambridge Nonlinear Science Series,* vol. xvi (Cambridge University Press, Cambridge, 1997), p. 304
5. O. Yazdanbaksh, A. Krahn, S. Dick, Predicting solar power output using complex fuzzy logic, in *IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS), 2013 Joint*, IEEE, 2013
6. D. Ramot et al., Complex fuzzy logic. IEEE Trans. Fuzzy Syst. **11**(4), 450–461 (2003)
7. S. Dick, Toward complex fuzzy logic. IEEE Trans. Fuzzy Syst. **13**(3), 405–414 (2005)
8. D.E. Tamir, L. Jin, A. Kandel, A new interpretation of complex membership grade. Int. J. Intelli. Syst. **26**(4), 285–312 (2011)
9. D.E. Tamir, A. Kandel, Axiomatic theory of complex fuzzy logic and complex fuzzy classes. Int. J. Comp. Commun. Control **VI**(3), (2011)
10. L. Běhounek, P. Cintula, Fuzzy class theory. Fuzzy Set. Syst. **154**(1), 34–55 (2005)
11. D.E. Tamir, M. Last, A. Kandel, The theory and applications of generalized complex fuzzy propositional logic, in *Soft Computing: State of the Art Theory and Novel Applications* (Springer, Berlin, 2013), pp. 177–192
12. D.E. Tamir et al., Discrete complex fuzzy logic, in Fuzzy Information Processing Society (NAFIPS), 2012 Annual Meeting of the North American, IEEE, 2012
13. S.D. Zenzo, A many-valued logic for approximate reasoning. IBM J. Res. Dev. **32**(4), 552–565 (1988)
14. A.R. Salleh, Complex intuitionistic fuzzy sets, in *AIP Conference Proceedings*, 2012
15. K.T. Atanassov, Intuitionistic fuzzy sets. Fuzzy Set. Syst. **20**(1), 87–96 (1986)
16. R.R. Yager, A.M. Abbasov, Pythagorean membership grades, complex numbers, and decision making. Int. J. intell. Syst. **28**, 436–452 (2013)
17. G. Zhang et al., Operation properties and δ-equalities of complex fuzzy sets. Int. J. Approx. Reason. **50**(8), 1227–1249 (2009)
18. G. Zhang et al., Delta-equalities of Complex Fuzzy Relations, in Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, IEEE, 2010
19. A.U.M. Alkouri, A.R. Salleh, Complex Atanassov's intuitionistic Fuzzy relation, in *Abstract and Applied Analysis* (Hindawi Publishing Corporation, Nasr City, 2013)
20. D. Moses et al., Linguistic coordinate transformations for complex fuzzy sets, in *Fuzzy Systems Conference Proceedings, 1999. FUZZ-IEEE'99. 1999 IEEE International*, IEEE, 1999
21. H.T. Nguyen, A. Kandel, V. Kreinovich, Complex fuzzy sets: towards new foundations, in *Fuzzy Systems, 2000. FUZZ IEEE 2000. The Ninth IEEE International Conference on*, IEEE, 2000
22. J. Man, Z. Chen, S. Dick, Towards inductive learning of complex fuzzy inference systems, in *Fuzzy Information Processing Society, 2007. NAFIPS'07. Annual Meeting of the North American*, IEEE, 2007
23. A. Hirose, *Complex-Valued Neural Networks*, vol. 400 (Springer, Berlin, 2012)
24. H.E. Michel, A.A.S. Awwal, Artificial neural networks using complex numbers and phase encoded weights. Appl. Opt. **49**(10), B71–B82 (2010)

25. A.J. Noest, Discrete-state phasor neural networks. Phys. Rev. A **38**(4), 2196 (1988)
26. I. Nishikawa, T. Iritani, K. Sakakibara, Improvements of the traffic signal control by complex-valued Hopfield networks, in *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, IEEE, 2006
27. Y. Li, Y.-T. Jang, Complex adaptive fuzzy inference systems, in Fuzzy Systems Symposium, 1996. 'Soft Computing in Intelligent Systems and Information Processing', Proceedings of the 1996 Asian, IEEE, 1996
28. A. Malekzadeh-A, M.-R. Akbarzadeh-T, Complex-valued adaptive neuro fuzzy inference system-CANFIS. Proc. World Aut. Congr. **17**, 477–482 (2004)
29. S. Aghakhani, S. Dick, An on-line learning algorithm for complex fuzzy logic, in *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, IEEE, 2010
30. C. Li, T.-W. Chiang, Function approximation with complex neuro-fuzzy system using complex fuzzy sets-a new approach. New Generat. Comput. **29**(3), 261–276 (2011)
31. C. Li, F. Chan, Complex-fuzzy adaptive image restoration-an artificial-bee-colony-based learning approach, in *Intelligent Information and Database Systems* (Springer, Berlin, 2011), pp. 90–99
32. Li, C. and F.-T. Chan, Knowledge discovery by an intelligent approach using complex fuzzy sets, in *Intelligent Information and Database Systems*. Springer, 320–329 (2012)
33. C. Li, T. Wu, F.-T. Chan, Self-learning complex neuro-fuzzy system with complex fuzzy sets and its application to adaptive image noise canceling. Neurocomputing **94**, 121–139 (2012)
34. C. Li, T.-W. Chiang, Intelligent financial time series forecasting: A complex neuro-fuzzy approach with multi-swarm intelligence. Int. J. Appl. Math. Comp. Sci. **22**(4), 787–800 (2012)
35. C. Li, T.-W. Chiang, Complex fuzzy computing to time series prediction a multi-swarm PSO learning approach, in *Intelligent Information and Database Systems* (Springer, Berlin, 2011), pp. 242–251
36. C. Li, T. Chiang, Complex Neurofuzzy ARIMA Forecasting—A New Approach Using Complex Fuzzy Sets, Fuzzy Systems, IEEE Transactions on **21**(3), 567–584 (2013)
37. C. Li, T.-W. Chiang, L.-C. Yeh, A novel self-organizing complex neuro-fuzzy approach to the problem of time series forecasting. Neurocomputing **99**, 467–476 (2013)
38. H. Sun, S. Wang, Q. Jiang, FCM-based model selection algorithms for determining the number of clusters. Pattern Recogn. **37**(10), 2027–2037 (2004)
39. J. Ma, G. Zhang, J. Lu, A method for multiple periodic factor prediction problems using complex fuzzy sets. IEEE Trans. Fuzzy Syst. **20**(1), 32–45 (2012)
40. A.U.M. Alkouri, A.R. Salleh, Linguistic variables, hedges and several distances on complex fuzzy sets. J. Intell. Fuzzy Syst. **26**(5), 2527–2535 (2014)
41. L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning-I. Inform. Sci. **8**(3), 199–249 (1975)
42. A. Deshmukh et al., Implementation of complex fuzzy logic modules with VLSI approach. Int. J. Comp. Sci. Netw. Security **8**, 172–178 (2008)
43. R. Hegger, H. Kantz, T. Schreiber, Practical implementation of nonlinear time series methods: The TISEAN package. Chaos **9**(2), 413–435 (1999)
44. M.B. Kennel, R. Brown, H.D. Abarbanel, Determining embedding dimension for phase-space reconstruction using a geometrical construction. Phys. Rev. A **45**(6), 3403 (1992)
45. NREL Staff, Lowry Range Solar Station (LRSS), Colorado State Land Board, http://www.nrel.gov/midc/lrss
46. A. Bellini et al., Simplified model of a photovoltaic module, in *Applied Electronics, 2009. AE 2009*, IEEE, 2009
47. J.S.R. Jang, ANFIS: Adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cyb. **23**(3), 665–685 (1993)
48. M. Li et al., Sunspot numbers forecasting using neural networks, in *Intelligent Control, 1990. Proceedings., 5th IEEE International Symposium on*, IEEE, 1990