LIBERA UNIVERSITÁ INTERNAZIONALE DEGLI STUDI SOCIALI

LUISS "GUIDO CARLI"

Adaptive Neuro-Complex Fuzzy Inference System for Financial Time Series Forecasting

Supervisor

Prof. Antonio Simeone

Co-supervisor

Prof. Giuseppe Italiano

Candidate

Valerio Schettini

Master's Degree in Data Science and Management

Data Driven Models for Investments

Academic Year 2022/2023

Contents

1 Introduction

This dissertation proposes to explore the predictive capabilities of a novel neural network architecture that leverages fuzzy reasoning systems for complex function approximation and chaotic time series forecasting: the Adaptive Neuro-Complex Fuzzy Inference System. The thesis is organized as follows. The first chapter proposes an overview of the evolution of traditional binary logic into fuzzy logic, and of the latter into complex fuzzy logic. The usefulness of such logical framework is analyzed in the context of Fuzzy Reasoning Systems. Chapter two describes two neural network frameworks developed on top of both real and complex fuzzy systems, namely Adaptive Neuro-Fuzzy Inference System (ANFIS) and Adaptive Neuro-Complex Fuzzy Inference System (ANCFIS). After a short introduction to Adaptive Networks, the network architecture and parameter learning mechanisms of ANFIS and ANCFIS are presented. The fourth chapter starts by exploring the research literature on the application of Neuro-Fuzzy Systems to financial time series forecasting. It then proceeds by introducing the financial time series chosen for testing ANCFIS predictive accuracy. A section is dedicated to delay embedding, a method borrowed from dynamical systems theory used in the data preprocessing with the objective to maximize the information content of past observations for prediction purposes. The last section of this chapter is dedicated to the presentation of results and their discussion. Finally, a summary of the research and possible improvements is the topic of Conclusions.

The whole analysis was performed in Python, from data retrieval, preprocessing, and visualization to model implementation. A python library specifically implementing the ANCFIS model is currently unavailable, so a custom python module was coded using PyTorch. This library was used to implement the model at a low level, from single node computations to layer structure and overall model architecture. The standard backpropagation algorithm widely used for traditional neural networks was modified to implement the hybrid learning method described in section 3.1.3, and the default backward computation was altered to allow optimization of model parameters at the first layer through a chaotic variant of simulated annealing. The code is available for inspection on GitHub.

2 From Classical to Fuzzy Logic

"Logic is but a small part of the human capacity to reason. Logic can be a means to compel us to infer correct answers, but it cannot by itself be responsible for our creativity or for our ability to remember. In other words, logic can assist us in organizing words to make clear sentences, but it cannot help us determine what sentences to use in various contexts" [1]. In other words, logic deals with syntax rather than semantics, with form rather than meaning. Moreover, traditional logic systems are bounded by Aristotle's excluded middle axiom. Allowing for a statement to be either true or false restricts a reasoning system to a black and white worldview, blinding it to some of the inherent complexities of real world phenomena.

2.1 Classical Logic

In classical logic, a proposition P is a declarative statement which evaluates to either true or false on a given input. Mathematical treatment of logic is very much facilitated by the analogies with set-theoretic and function-theoretic concepts. From a set-theoretic point of view, a logical proposition is identified by a subset of the available information with respect to which the proposition evaluates to true, i.e. its truth set. From a functional point of view, logic propositions are identified by characteristic functions associating the set of possible inputs, also known as Universe of Discourse, with the binary set $\{0,1\}$. Given this correspondence, in the following dissertation the word set and proposition will be used interchangeably, where with "set" one refers to the truth set of said proposition.
A range of logical operators, such as logical connectors and relations, allow for the construction of Logic Inference Systems. Table 1 shows a non-exhaustive list of them, together with their set-theoretic equivalents. The main objective of logic sets is to represent the state of knowledge about a domain, while logic operators use it to construct a system of rules to act upon domain-related events.

Another fundamental concept for devising Inference Systems is that of a relation. Given two sets A and B defined on different Universes of Discourse, a logic relation is a subset defined on the Cartesian product $A \times B$. The subset is associated with a characteristic function, taking a value of either 1 or 0 corresponding to the presence or absence of a relation between any two elements of the sets. Composition of logical relations is possible if their respective domains have at least one set in common. The definition of composition between relations is not unique and the choice of using one or another usually depends on the application domain. Given three sets A,B, and C, and two relations R and S, defined on $A \times B$ and $B \times C$, respectively, two well known definitions are:

- *Max-min composition*: $R \circ S = \bigvee_{b \in B} \; (\chi_R(a,b) \wedge \chi_S(b,c))$
- *Max-product composition*: $R \circ S = \bigvee_{b \in B} \; (\chi_R(a,b) \cdot \chi_S(b,c)$

where $\circ$ represents the composition operation, $a \in A$, $b \in B$, and $c \in C$ are elements of the initial sets, χ_R is the characteristic function of relation R, $\vee$ is the logic or, and $\wedge$ is the logic and.
Finally, generalized modus ponens is a tool for devising deductive inference systems. It leverages the concept of a relation to represent an if-then rule, and that of relation composition to infer new rules as new evidence is collected. As will be discussed in the next sections, generalized modus pones can be leveraged in a fuzzy logic setting to devise a Fuzzy Inference System.

Even though classical logic underpins a wide range of theories and applications, from proof theory to digital computers, its specific properties restrict the range of possible use cases. As mentioned in the introduction to this chapter, logic is very well known as a way to construct syntactically correct sentences, but it is not well suited for treatment of semantic information. The next section reviews the foundations of fuzzy logic, and describes the way in which classical logic concepts are naturally extended through the definition of fuzzy sets and operations.

2.2 Fuzzy Logic

Fuzzy logic is as an extension of boolean logic which allows for the construction of approximate reasoning systems. Its core components are called Fuzzy Sets and were first introduced by Zadeh [2] in 1965. Fuzzy sets are defined through characteristic functions associating an element from a universe of discourse to its corresponding membership degree on the real interval [0,1].
More formally, given a universal set U, a fuzzy set A is defined as

$$A := \{(x, \mu_A(x))\} \quad \forall x \in U$$

where x is an element of U, and $\mu_A(x) \in [0,1]$ is the degree of membership of x in set A.

Fuzzy sets constitute a useful framework to represent semantic information. To clarify with an example, think of a word in natural language and a set of classes to which it may belong. For instance, a glass of water and the sets "full" or "empty". These two classes only represent the extremes of the possible states the glass may be in. Would a half full glass be classified as full or empty? This is the choice imposed by boolean logic. The fuzzy answer would be that the glass is both full and empty with a degree of truth of

0.5. In other words, the glass belongs to the fuzzy set "full" with a degree of membership of 0.5, and to the fuzzy set "empty" with a degree of 0.5. Ambiguity, not admissible in a crisp set, finds a mathematical representation through fuzzy sets.

Leveraging axiomatic properties of logical operators, namely commutativity, associativity, monotonicity, and boundary conditions, a derivation of fuzzy logic equivalents is possible. This approach has led to the definition of more than one fuzzy analogous per each logical operator. Fuzzy conjunction, represented by fuzzy sets intersection, is defined in function theoretic form by functions known as Triangular Norms, or T-norms. Three widely known definitions are listed in Table 1. Similarly, fuzzy equivalents of classical disjunction are represented by Triangular co-norms, or T-conorms. Two examples are listed in Table 2. Such generalization of fundamental logic operators, which straightforwardly applies to relations as well, is sufficient for the construction of other relevant operators, such as implications, which are then leveraged by modus ponens to devise a fuzzy deductive inferene system. A review of core characteristics and types of Fuzzy Inference Systems is the topic of section 1.4. Before that, an introduction to a further extension to traditional logic involving complex numbers is in order.

Table 1: List of T-norm functions

Operation	**Expression**
Minimum T-norm	$A \cap B = \mu_{A \cap B}(x) = min(\mu_A(x), \mu_B(x))$
Product T-norm	$A \cap B = \mu_{A \cap B}(x) = \mu_A(x) \cdot \mu_B(x)$
Lukasiewicz T-norm	$A \cap B = \mu_{A \cap B}(x) = max(0, \mu_A(x) + \mu_B(x) - 1)$

Table 2: List of T-conorm functions

Operation	**Expression**
Maximum T-conorm	$A \cup B = \mu_{A \cup B}(x) = max(\mu_A(x), \mu_B(x))$
Lukasiewicz T-conorm	$A \cap B = \mu_{A \cap B}(x) = min(1, \mu_A(x) + \mu_B(x))$

2.3 Complex Fuzzy Logic

As fuzzy logic can be seen as an extension of classical logic, complex fuzzy logic may be similarly interpreted as an extension fuzzy logic. It consists of extending the domain of fuzzy membership function from the real interval [0,1] to the complex unit disk. [10] define complexy fuzzy logic as "a unique framework, designed to maintain the advantages of traditional fuzzy logic, while benefiting from the properties of complex fuzzy sets". Even though the authors were not the only ones to introduce the concepts of complex fuzzy sets and the logical systems that derive, the focus of this section will be on their definition since it is the one most widely combined with adaptive networks to form a complex version of ANFIS. This means that the following is a non-exhaustive review of the topic, which can instead be found in [11]. Following [12] a complex fuzzy set C follows the formal definition of a fuzzy set through its membership function

$C = \{(x, \mu_C(x) \mid x \in U\}$, with $\mu_C(x) = r_C(x) \cdot e^{j\omega_C(x)}$

where $r_C(X) \in [0, 1]$ and $\omega_C(x) \in \mathcal{R}$.

From this definition it is evident that the truly new component introduced by complex fuzzy sets is the phase term of the membership function. Indeed, if such term is considered equal to zero, the complex fuzzy set is found to be equal to its real fuzzy counterpart. Such relationship is extremely useful for extending fuzzy operators to the complex domain. In fact, [12] propose the amplitude term of complex membership to follow the traditional definition for fuzzy unions and intersections, i.e. T-conorm and T-norm functions, while proposing four possible functions for phase term derivation applicable to both union and intersection. Formally, given two complex fuzzy sets A and B, we have:

- Union $A \cup B := \mu_{A\cup B}(x) = r_{A\cup B}(x)e^{j\omega_{A\cup B}(x)} = r_A(x) \oplus r_B(x) \cdot e^{j\omega_{A\cup B}(x)}$
- Intersection $A \cap B := \mu_{A\cap B}(x) = r_{A\cap B}(x)e^{j\omega_{A\cap B}(x)} = r_A(x) \star r_B(x) \cdot e^{j\omega_{A\cap B}(x)}$

where $\oplus$ stands for any T-conorm operator, $\star$ for any T-norm function. The phase term $\omega_{A\cap B}(x)$ can be derived through the same formulas for both unions and intersections. [12] proposed the following four options:

- $\omega_A + \omega_b$
- $max(\omega_A, \omega_b)$
- $min(\omega_A, \omega_b)$
- ω_A if $r_A > r_b$ else ω_B

while [18] listed three other possibilities:

- $\frac{r_A\omega_A + r_B\omega_b}{r_A + r_B}$
- $\frac{\omega_A + \omega_b}{2}$
- $\omega_A - \omega_B$

To understand the practical benefits introduced by complex fuzzy sets [10] introduce the following example. Consider an individual pondering an investment in a stock. Two facts may be relevant for his decision: relative attractiveness of the stock, measured for example by its price/earnings ratio, and current phase of the overall stock market cycle. While traditional fuzzy logic would require two independent fuzzy sets to represent such information, complex fuzzy logic elegantly allow for a single complex number to encode it. The amplitude term could represent relative attractiveness of the stock, while the phase term the current phase of the stock market cycle. Even though the information could be represented using two distinct traditional fuzzy sets, reasons to choose a complex representation include "ease of representation and calculations, physical accuracy of the complex representation (e.g., one of the parameters does indeed represent phase) and utility of complex algebra in the specific application" [10]. Moreover, [13] suggest another advantage of complex fuzzy logic could be the ability of complex numbers "to model 'approximately periodic' phenomena: patterns that approximately repeat themselves but are never exactly the same twice" [14]. In this regard, particularly useful is the vector aggregation operation as introduced by [12]. Such operation is used to derive a final value by combining the results of membership evaluation on different sets. It consists of a weighted sum of membership values:

$$\sum_{i=1}^{n} \mu_{A_i}(x) \cdot w_i$$

where $\mu_{A_i}(x)$ is the membership function for set A_i, and $\sum_{i=0}^{n} |w_i| = 1$.

As explained in [10] the sum of complex numbers allows for constructive and destructive interference between membership values. As will be explained later, this characteristic results in rule interference in complex fuzzy inference systems and contributes to the system suitability to model periodic phenomena. The last aspect to discuss is the concept of complex fuzzy relations and their compositions. The definition of relation remains the same, with the only exception that the membership function defined on the product space is a complex one. Therefore

$$\{((x,y), \mu_R(x,y)) | x \in A, y \in B\}$$

where A and B are complex fuzzy sets that can be defined on different universe sets. From the above formula it is evident that a fuzzy relation is a subset of the Cartesian product A$\times$ B where $\mu_R(x,y)$ represents the strength of association between the pair (x,y). Again, following the same approach for extending traditional fuzzy operators, compositions can be extended by considering the amplitude and phase term separately. Given two relations R(U,V) and S(U,V) and their corresponding membership functions

$$\mu_R(x,y) = r_R(x,y) \cdot e^{j\omega_R(x,y)}$$
$$\mu_S(y,z) = r_S(y,z) \cdot e^{j\omega_S(y,z)}$$

the membership function of the composition of R and S is described by

$$\mu_{R\circ S}(x,z) = r_{R\circ S}(x,z) \cdot e^{j\omega_{R\circ S}(x,z)}$$

For the amplitude term $r_{R\circ S}(x,z)$, the sup-star composition of traditional fuzzy sets is applied

$$r_{R\circ S}(x,z) = sup_{y\in Y}[r_R(x,y) \star r_S(y,z)]$$

For the phase term we have

$$r_{R\circ S}(x,z) = f[g(r_R(x,y), r_S(y,z))]$$

where function g can be any of the formulas used to determine the phase term in complex fuzzy intersection and conjunction, while f can be substituted with either the supremum or infimum operators

$$sup_{y\in Y}[g(r_R(x,y), r_S(y,z))]$$

$$inf_{y\in Y}[g(r_R(x,y), r_S(y,z))]$$

To conclude the list of operations that allow Complex Fuzzy Sets to develop into a Complex Fuzzy Logic System analogous to Fuzzy Logic, one last operator needs to be defined. According to [12], given two complex fuzzy sets A and B, the complex fuzzy implication operation could be defined as either of the following:

- Min-implication: $\mu_{A\to B} = min(\mu_A, \mu_B)$
- Product-implication: $\mu_{A\to B} = \mu_A \cdot \mu_B$

[10] chose the product implication for the above mentioned properties of complex numbers. Indeed, multiplying μ_A with μ_B means multiplying their respective amplitude terms and summing their phases. Once again, the analogous of the amplitude term of complex implication with traditional fuzzy implication

is evident. Moreover, phase addition is directly related with rotations in the complex unit disk. In fact, the whole product implication operation could be seen as the starting set membership function μ_A being scaled by the amplitude term of μ_B and rotated by the latter's phase term. Even though the phase isn't much informative in itself "it becomes a significant parameter when several implication relations are considered together" [10] such as in a multi-rule fuzzy inference systems.

2.4 Fuzzy Inference Systems

As previously shown in the glass of water example, one of the advantages of fuzzy logic is its ability to give a mathematical representation to concepts expressed in natural language. A consequence of this is that Inference Systems built on Fuzzy Logic resemble rule-based systems expressed in natural language, as an example later in this section will help to show. Being that fuzzy sets and operations are proven to be accurate generalizations of their crisp equivalents, it is possible to transform a classical rule-based knowledge system into a combination of fuzzy sets and operations to obtain a Fuzzy Inference System (FIS).

Following the definition of Jang [3], a Fuzzy Inference System is composed of five modules:

- a rule base comprised of fuzzy if-then rules
- a database defining the membership functions of the fuzzy sets used in the rule base
- a fuzzification engine to convert fuzzy inputs into membership values according to the database
- a decision-making unit performing inference based on fuzzified inputs and rule base
- a defuzzification engine to convert the fuzzy outputs of the decision-making unit back into crisp values

The steps generally involved in a FIS can be summarizied as follows:

1. Fuzzification: compute membership degrees of the inputs
2. Premise evaluation: combine the membership values according to the fuzzy rules, usually involving some T-norm functions, obtaining *firing strengths* for each rule
3. Consequent Evaluation: derive the consequent by combining *firing strenghts* with corresponding rule consequents
4. Defuzzification: defuzzify the output through some type of disaggregation operation

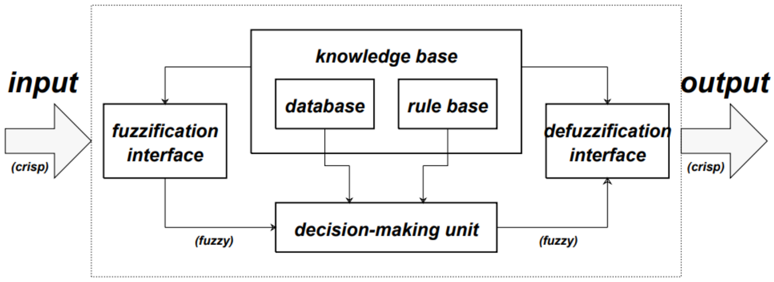

Figure 1: Diagram of a Fuzzy Inference System, image taken from [3]

Various authors have proposed different FIS frameworks by using different formulations of consequent evaluation and defuzzification.

Mamdami Inference Systems have found a wide range of applications in air conditioning automation [4], performance evaluation of manufacturing systems [5], and transportation [6], among others. Their distinctive characteristic is the usage of fuzzy sets in the consequent evaluation step. A range of disaggregation methods can be used as defuzzification engine, e.g. center of gravity, minimum of maximum, or bisector.

Tsukamoto Inference Systems also leverage fuzzy sets in consequent evaluation, but the corresponding memberships are required to be monotonically increasing as they are used to directly obtain crisp outputs. Defuzzification engine therefore consists of an average of such crisp values weighted by the rules' firing strenghts.

Another important framework for FIS is the Takagi-Sugeno Inference System. This is the framework that found the most applications in Neuro-Fuzzy Systems, as will be shown later. The consequent evaluation consits of a function of inputs, weighted by the firing strenghts obtained in the premise evaluation step. This process makes defuzzification much less computationally expensive, which is among the reasons why this framework has been preferred to be combined with nerual networks to form Neuro-Fuzzy Systems.

To have a better view of fuzzy sets, rules, and inference systems, consider a simple case of air conditioning automation. The objective of the system is to leverage expert knowledge and combine it with evidence in order to decide what actions to take. Let us say temperature and humidity are the two variables observed upon which the power of the air conditioning system has to be adjusted. Again, for simplicity consider a simple system of two rules. Each rule will be a combination of two fuzzy sets representing the state of the system at a given point in time. Three membership functions, named "High", "Medium", and "Low", are defined for each input variable. Figure 2 shows membership functions for each variable.

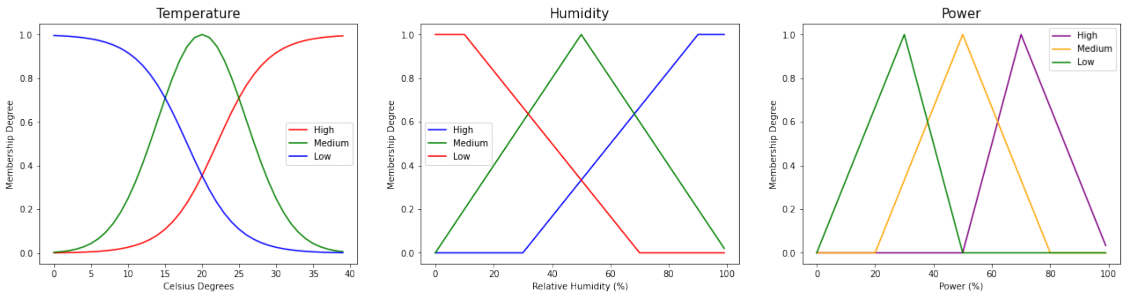

Figure 2: membership functions for temperature, humidity, and power

The two rules are the following:

1. IF Temperature is High AND Humidity is High, THEN Power is High
2. IF Temperature is High AND Humidity is Medium, THEN Power is Medium.

Suppose the observations for temperature and humidity are 25°C and 60%, respetively. The premise evaluation step, consists in computing the membership values for temperature and humidity and combining them according to the fuzzy conjunction operator. We obtain a membership of 0.7 for the set High of temperature variable and of 0.5 and 0.8 for set High and Medium of humidity variable. Choosing the min operator as fuzzy conjunction we obtain a firing strength of 0.5 for Rule 1 and 0.7 for Rule 2. For each rule, these values together with the consequent fuzzy set are given as inputs to a max-min composition, obtaining the fuzzy consequent. Figure 3 and 4 show a visualization of the steps described above for Rule 1 and 2.

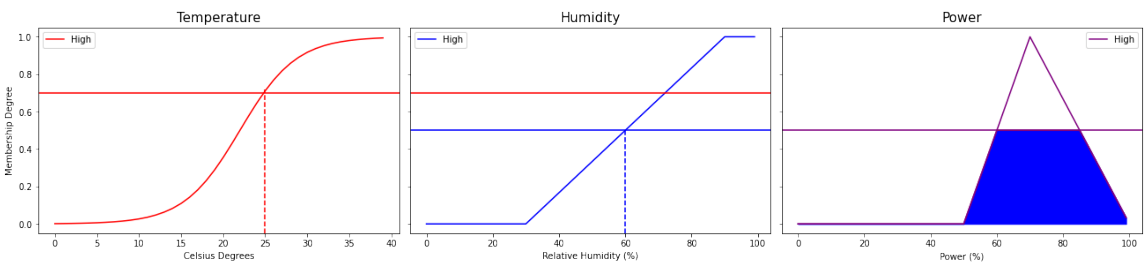

Figure 3: Result for Rule 1

The next step is aggregation: the max operator is applied to the consequent of each rule, obtaining

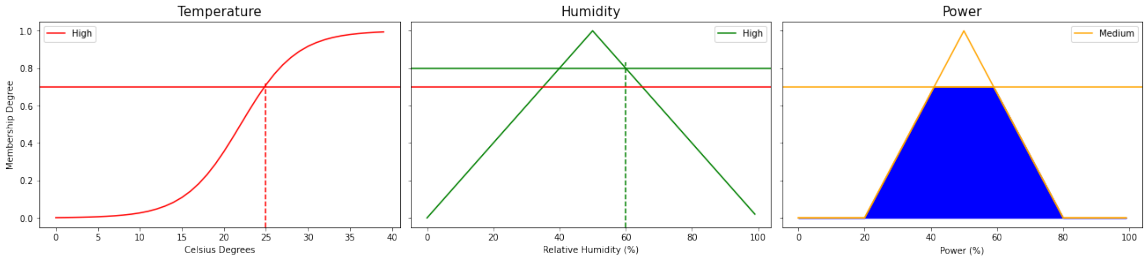

Figure 4: Result for Rule 2

the final fuzzy output shown in Figure 5.

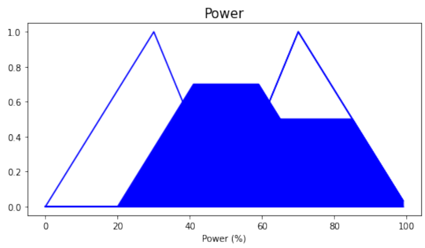

Figure 5: Final fuzzy output

The final step is defuzzification. As explained before, it is possible to choose among many methods. For instance, applying the minimum of maximum in the example would result in a final output of 40% power. The same premise structure could be used for a Takagi-Sugeno System. The main difference stands in the structure of the consequent and the defuzzification method. A rule base inhereting from the example structure is the following:

1. IF Temperature is High AND Humidity is High, THEN $p_1 = a_1 + b_1 t + c_1 h$.

2. IF Temperature is High AND Humidity is Medium, THEN $p_2 = a_2 + b_2 t + c_2 h$.

where p, t, and h are the values for the variables power, temperature and humidity respectively, p_1 and p_2 are the values of the outcome variable for Rules 1 and 2, and a,b,c are the consequent parameters of each rule. The final output is the result of a weighted average of single rule outputs where the weights are given by each rule's firing strength.

2.5 Limitations of Fuzzy Inference Systems

This Chapter outlined a brief review of the conceptual steps that lead to the construction of Approximate Reasoning Systems, i.e. Fuzzy Inference Systems. Starting from classical logic and its extension to fuzzy logic through the concept of fuzzy sets, the concept of a Fuzzy Inference System was introduced, together with an example of two of the most well-known Fuzzy Systems.

Even though Fuzzy Infernce System offer a straightforward way of translating expert knowledge and decision making into clear computational steps, also enhancing the ability of the system to deal with nonlinear phenomena [3], it has a few drawbacks. The first is the large design space. As described in section 1.2, each classical logic operator finds more than one equivalent in fuzzy logic. Also, there are many types of fuzzy systems to choose from. The result is a large number of possible implementations of fuzzy systems, leading to a time intensive process of optimization. Secondly, fuzzy systems as treated in this section are mere translations of an expert knowledge base into actionable insights. This means that faults in human interpretation of the phenomena at hand are automatically transferred to the fuzzy system. Lastly, the types and parameters of membership functions play a crucial role in the system. Even though expert knowledge and intuitions are useful in determining the best shape for a membership function, a clear structure for optimal parameter inference is lacking. For these reasons, fuzzy systems and neural networks were combined in a novel model structure to allow for the formulation of a clear optimization process to match the model's framework with input-output data pairs. Such new model category is known as Neuro-Fuzzy Systems and is the topic of next chapter.

3 Neuro-Fuzzy Inference Systems

A Neuro-Fuzzy system, or Fuzzy Inference Neural Network, is an artificial neural network structure inspired by Fuzzy Inference Systems.

3.1 Adaptive Neuro-Fuzzy Inference System

The first Neuro-Fuzzy System was introduced by [3] in 1993, namely Adaptive Neuro Fuzzy Inference System. It was proposed to overcome two limitations in fuzzy systems: the absence of a standard method to determine the rule base and database for a FIS, and the lack of an algorithm for tuning membership function parameters. The novelty in Jang's approach is to reconfigure a FIS into a neural network to benefit from the extensive research literature in the field. The model has been tested for both nonlinear function modeling and chaotic time series prediction, showing results that inspired many researchers in the following years to refine [7] and apply ANFIS to various areas, from medicine [8] to finance [9].

3.1.1 Adaptive Networks and Backpropagation

"An adaptive network is a multi-layer feed-forward network in which each node performs a particular function (node function) on incoming signals as well as a set of parameters pertaining to this node" [3]. The presence of node parameters is what makes the network "adaptive". Each node's output is a function of previous layer outputs and the node's own parameters. The set of node parameters is referred to as the model's parameter set. Well-known training algorithms, e.g. backpropagation, can be applied to such neural net to optimize the parameter set with respect to a given loss function.
To briefly explain backpropagation logic, consider a multiple input-multiple output feed-forward neural network as the one shown in Figure 6.

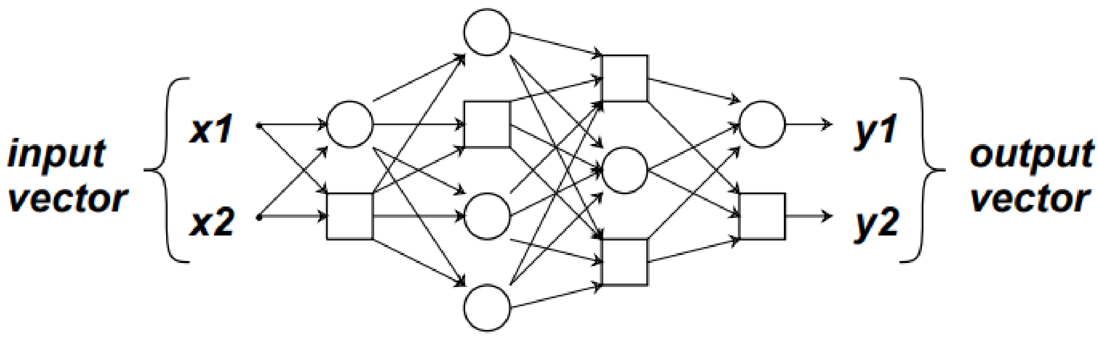

Figure 6: Adaptive network diagram taken from [3]

Let L denote the last layer of the network, k a general network layer where $1 \leq k \leq N$, and #k the number of nodes in layer k. Denote the output of node i in layer k as O_i^k, where O_i^k represents both the output and the corresponding node function.
Also, let the loss (or energy) function be:

$$E_p = \sum_{j=1}^{\#k} (T_{j,p} - O_{j,p})^2$$

where p denotes the p-th observation in the dataset, and $T_{j,p}$ is the j-th component of our target observation for input p. The loss function for the whole dataset will then be:

$$E = \sum_{p=1}^{P} E_p$$

where P is the total number of observations in the dataset. Backpropagation consists of computation of partial derivatives and repeated applications of the chain rule. Given the standard Euclidean norm as our loss function, the computation of its derivative with respect to node i of layer L is straightforward:

$$\frac{\partial E_p}{\partial O_i^L} = -2(T_{i,p} - O_{i,p})$$

For a general node i at layer k its derivative can be computed applying the formula:

$$\frac{\partial E_P}{\partial O_i^k} = \sum_{n=1}^{\#(k+1)} \frac{\partial E_p}{\partial O_{n,p}^{(k+1)}} \cdot \frac{\partial O_{n,p}^{(k+1)}}{\partial O_{i,p}^k}$$

Following Jang's notation, the above can be applied to a given node's parameter α:

$$\frac{\partial E_P}{\partial \alpha} = \sum_{O^\star \in S} \frac{\partial E_p}{\partial O^\star} \cdot \frac{\partial O^\star}{\partial \alpha}$$

The parameter can in turn be updated by:

$$\Delta\alpha = \eta \cdot \frac{\partial E_P}{\partial \alpha}$$

3.1.2 ANFIS Structure

In this section an overview of ANFIS structure is given, explaining the role of each layer in output generation. As detailed in [3], the model is composed of five layers:

1. *Fuzzification Layer*: each node function consists of the membership function relating an input to a corresponding fuzzy set:

   $$O_i^1 = \mu_{A_i}(x)$$

   . where x is a component of the input vector and A_i the fuzzy set the input is being mapped to by μ_{A_i}.
   Even though any piece-wise differentiable membership function can be used, bell-shaped functions, especially Gaussian functions, are particularly well suited since changing parameters of these two functions will almost always produce functions of the same class.

2. *Product Layer*: previously computed membership degrees are multiplied by each other in this layer. The result is analogous to performing the premise evaluation step assuming the product T-norm as fuzzy conjunction.

3. *Normalization Layer*: firing strengths obtained from previously layer are normalized according to the following formula:

   $$\overline{w_i} = \frac{w_i}{\sum_{j=1}^{k} w_j}$$

   where k is the number of nodes of this layer, and w_i is the i-th output of previous layer.

4. *Consequent Layer*: i-th output of this layer corresponds to the consequent evaluation of a Takagi-Sugeno system for i-th Rule:

   $$O_i^4 = \overline{w_i} \cdot f_i = \overline{w_i} \cdot (r_i + \sum_{j=1}^{n} q_{i,j} \cdot x_j)$$

   where x_j is the j-th component of input vector x, n is the dimension of x, and $\{r_i, q_{i,j}\}$ with $j = 1, ..., n$ is the parameter set for node i.

5. *Weighted-Sum Layer*: this layer is the sum of previous node outputs:

   $$O_1^5 = \sum_{i=1}^{k} O_i^4$$

   where k is the number of nodes in previous layer.

3.1.3 Hybrid Learning

An hybrid learning rule has been proposed by Jang for parameter estimation of ANFIS. It is important to note that such hybrid is not generally applicable. In fact, it leverages the linearity of the loss function derivative with respect to the consequent parameters. This means that unless a suitable invertible function can be applied to the whole model so as to satisfy such linearity condition for the consequent parameter set, one needs to resort to traditional learning methods, i.e., gradient-descent, genetic algorithms, etc. This means that using a traditional Euclidean norm as loss function allows for said hybrid learning. Given the structure of consequent layer outlined in the previous section, the invertible function just mentioned is quite conveniently the identity function[1].

The hybrid learning rule consists of separating the whole parameter set into a set of linear parameters to be estimated with least squares methods such as Ordinary Least Squares (OLS) or Recursive Least Squares Estimation (RLSE), and a set of model parameters to be trained by traditional learning methods. Again, linearity of loss function derivative with respect to the set of linear parameters is paramount to ensure least squares methods find the optimum values for this linear subset of the parameter space.

According to the hybrid learning rule, model training proceeds as follows. A first pass is performed with fixed premise parameters to obtain the firing strengths to be used in the consequent layer. Once rule weights have been computed for each input in the training dataset, the consequent parameters are fitted with respect to the Mean Squared Error function using either OLS or RLSE. Such consequent parameters are fixed for the rest of the training loop. A second forward pass is then performed to compute the loss function. Finally, the backward pass takes care of optimizing premise parameters through backpropagation.

As will be shown in later section, this same hybrid learning rule can and will be applied to model training of the complex version of ANFIS, which is discussed in the next section right after a short introduction to complex fuzzy logic.

[1]For further details, see [3].

3.2 Complex Neuro-Fuzzy Systems

One of the first attempts to extend ANFIS to the complex domain can be found in [15]. The proposal was to adapt a traditional ANFIS to deal with complex input-output combinations. Even though the proposal doesn't involve complex fuzzy logic, it can still be regarded as one of the earliest intuitions about the potential usefulness of involving complex numbers in adaptive fuzzy inference systems. On the other hand, other implementations leveraged complex fuzzy logic to extend ANFIS capabilities, such as [16], [17], and [14].

Being Adaptive Neuro-Complex Fuzzy Inference System (ANCFIS) based on complex fuzzy logic, the extension of real valued backpropagation to the complex domain by [19] played a crucial role. Even though [14] is not the only implementation of Complex Fuzzy Inference System, it will be the focus of this section as it has already been demonstrated a good function approximation technique [16] and chaotic time series predictor [14]. Therefore, this model will be the backbone on top of which slight changes will be applied to evaluate the performance of different ANCFIS versions on financial time series forecasting.

It has been argued [14] that such potential for time series prediction stems from the "coupling" of magnitude and phase already proposed by [13] in his review of [10]. As anticipated in the previous section, Ramot et al. proposed to separately consider amplitude and phase of complex membership function as traditional fuzzy linguistic value and its changing "context", respectively. The consequence of such interpretation is that complex fuzzy logic can be viewed as a straightforward extension of type-1 fuzzy logic with an amplitude term different from zero. [13] referred to such complex fuzzy logic implementations as characterized by a "rotational invariance" property. Namely, shifting the phase of two complex fuzzy sets results in the same shift for their complex fuzzy union. [13] went on in his dissertation to show that a product between memberships results in a conjunction operator not satisfying rotational invariance, though it would allow complex fuzzy memberships to "interfere" in the sense proposed by [12] also through intersections. Such interference would therefore happen at the antecedent level in a Complex Fuzzy Inference System using the product between memberships as conjunction operator. This extra layer of interference is leveraged by [14] in their proposed complex version of ANFIS.

Before delving into ANCFIS application to financial forecasting, a review of the network structure is in order. The model is composed of six layers:

1. **Fuzzification Layer:** the first layer is the Fuzzification Layer where input vectors are mapped to a complex membership value. One of the advantages of ANCFIS over ANFIS is the former being rather

parsimonious. In fact, instead of assigning a membership grade to each individual observation, ANCFIS assigns such membership to an entire vector of observations. This is done by sampling a phase and using it to derive a complex-valued vector of membership degrees. Specifically, the sampled phase is inputted into either a sine or a Gaussian function.

The sine function was proposed by [17], and is defined as follows:

$$rk(\theta_k) = d \cdot sin(a\theta_k + b) + c, \quad \theta_k = \frac{2\pi}{n}k$$

where n is the length of the input vector and k = 1,2,...,n is the index of the complex samples.

The obtain amplitude and phase are then transformed in polar coordinates through the usual transformations:

$$x_k = r_k cos(\theta_k)$$
$$y_k = r_k sin(\theta_k)$$

Having obtained the above vector of membership, the next step is to compute its convolution with the input vector as follows:

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

where n is the length of input vector f, $f(j)$ represents the j-th component of the input vector, and $g(k)$ the k-th component of the sampled membership vector.

Once the convolution is computed, the last step consists of normalizing the result applying the Elliot Function given by

$$f(z) = \frac{z}{1+|z|}$$

This ensures the final output of the layer for input i belongs to the complex unit disk:

$$O_{1,i} = \frac{\sum_{k=1}^{2n-1} h(k)}{1 + |\sum_{k=1}^{2n-1} h(k)|}$$

2. **Firing Strength Layer:** this layer is the first step in antecedent evaluation. It consists of multiplying previous node outputs with each other. By interpreting such product as a conjunction between fuzzy memberships a la [13], the node can be seen as applying a conjunction on individual fuzzified inputs. It is important to notice that the effect of conjunction is leveraged only on the memberships of different input vectors, meaning that for univariate time series this layer just passes along the previous node outputs.

   $$O_{2,i} = \prod_j O_{1,j}, \quad j = 1, 2, ..., |O_1|$$

   where $|O_1|$ is the number of nodes in layer 1.

3. **Normalization Layer:** Here the previous output are normalized to obtained the final firing strengths of each rule. Normalization is performed as

   $$O_{3,i} = \overline{w_i} = \frac{w_i}{\sum_{j=1}^{|O_2|} w_j}$$

   where $|O_2|$ is the number of nodes in layer 2, and w_i is the firing strength as obtained from the previous layer.

4. **Dot Product Layer**: this layer transforms the complex normalized firing strengths into real values by a dot product between the previously obtained firing strengths and their sum:

   $$O_{4,i} = w_i^{DP} = \overline{w_i} \cdot \sum_{j=1}^{|O_3|} \overline{w_j}, \quad i = 1, 2, ..., |O_3|$$

   Constructive or destructive interference may happen at this stage according to the phase terms of the arguments being aligned or not.

5. **Consequent Layer:** The output of this layer is the result consequent of a typical Takagi-Sugeno system:

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

where n is the length of the input vector, x_j is the j-th component of such vector, and $\{\vec{p_i}, r_i\}$ is the parameter set for rule i.

6. **Weighted Sum Layer:** this layer performs the sum of previous node outputs

$$\sum_{i=1}^{|O_5|} O_{5,i}$$

In much the same way as Adaptive Fuzzy Inference Systems, hybrid learning can also be applied to ANCFIS as long as the conditions discussed in section 2.1.3 hold. Nevertheless, an additional layer of complexity is added through the complex membership function. It is not possible to find a closed form formula for the differentiation of a sinusoidal membership function with respect to its parameters. Being such differentiation crucial in backpropagation, another way to optimize these parameters needs to be found. [14] investigated three possible search algorithms: one particle swarm optimization (PSO), and two chaotic simulated annealing. According to their findings a specific version of CSA is the one that performed best, i.e. Variable Neighbour Chaotic Simulated Annealing (VNCSA) [14].

Simulated annealing is a search algorithm belonging to the family of local search algorithms. It is a combination of classical local search with a stochastic component inspired by statistical mechanics. Such component is useful in giving the algorithm a probabilistic structure allowing "theoretical analysis of their asymptotic convergence" [20]. While traditional simulated annealing relies on the random initialization of initial solutions, VNCSA uses chaotic maps together with an adaptive component to determine the dimension of local search space (neighborhood) at each temperature. Chaotic maps induce a restriction of "the search to a fractal subset of a real-valued interval and can thus be faster" [14].
The two chaotic maps used in VNCSA are the Logistic map and the Ulam-von Neumann map. They are both iterative maps expressed by deterministic functions showing chaotic behavior in the sense of high sensitivity to initial conditions. The formula for the logistic map is

$$x_{i+1} = \mu \cdot x_i \cdot (1 - x_i)$$

where μ is a real valued parameter. The logistic map shows chaotic behavior for parameter values $\mu = 4.0$ and $\mu = 3.57$, if the initial condition are not in the set of fixed points $\{0, 0.25, 0.5, 0.75, 1.0, (5 + \quad 5)/8, (5 - \quad 5)/8, (2 + \quad 3)/4, (2 - \quad 3)/4\}$.
The Ulam-von Neumann map is expressed as

$$y_{i+1} = 1 - ry_i^2$$

which is chaotic and spans the real interval $[-1, 1]$ for $r \geq 2$.
The objective of a local search algorithm is to minimize the value of a cost function $f : \mathcal{R}^n \to \mathcal{R}$, which may be subject to a set of constraints, by searching the space of possible function parameters. As mentioned above, the characteristic of simulated annealing is to introduce a stochastic component in the search. This is done by introducing a temperature which determines the probability to select a sub-optimal solution as current best. The multiplicative inverse of an exponential function on the positive real line is chosen as probability distribution.

$$e^{[-f(new)-f(current)]/T}$$

The magnitude of the argument of the exponential determines the probability of selecting said sub-optimal solution. Such magnitude is directly proportional to the difference between cost function values for the new and current solutions, and inversely proportional to the temperature parameter. In other words, the higher the increase in cost by selecting the sub-optimal solution, the lower the probability it will be selected. The converse is true for temperature. The search is performed recursively, starting with an initial temperature value and decreasing it at each step. Selecting a high enough initial value for temperature leads to a high probability of selecting worse solutions, even for large errors. As temperature cools down, the error term becomes more and more relevant until only the best solutions are accepted.

The pseudocode in figure 7 shows the pseudocode of VNCSA. Below is a description of the three procedures:

1. *GenerateInitialSolutionPopulation():* The initial solutions are generated through the logistic map. Each variable of the cost function is generated separately. A random number is generated and used

Algorithm 1 VNCSA pseudocode, taken from [14]

$S_{current} \leftarrow GenerateInitialPopulation()$
while $T_k > T_{min}$ **do**
  while $l < L_{max}$ **do**
    while $m < M$ **do**
      $S_{new} \leftarrow NeighborSelection()$
      if $f(S_{new} < f(S_{current})$ **then**
        $S_{new} \leftarrow S_{current}$
      else
        $accept\ S_{new}\ as\ new\ solution\ with\ probability\ exp(-(f(S_{new} - f(S_{current}))/T_k)$
      end if
    end while
  end while
  $UpdateNeighborhood(D)$
  $T_k \leftarrow \beta \cdot T_k$
end while
$S_{best} \leftarrow S_{current}$
$output : S_{best}$

as initial solution for the map. Letting x_i, j represent the j-th iteration of the logistic map for the i-th variable:

$$x_{i,j+1} = 4.0 \cdot x_{i,j} \cdot (1 - x_i, j)$$

A linear transformation is then applied to the obtained variables to scale in the range of the variable of interest:

$$s_{i,j+1} = l_i + (u_i - l_i) \cdot x_{i,j+1}$$

where $s_{i,j+1}$ is the i-th variable of the generated initial solution, and l_i, u_i are the lower and upper bound for variable $s_{i,j+1}$, respectively.
Once all the variables have been determined they are checked against eventual constraints. The step is repeated in case these are not satisfied.

2. *NeighborSelection():* the Ulam-von Neumann map is used to generate new candidate solutions from the current one as follows:

$$S_{i,j+1}^{new} = S_{i,j+1}^{current} + D_i \cdot y_{i,j+1}$$

$$y_{i,j+1} = 1 - 2y_{i,j}^2$$

where $y_{i,j}$ is the i-th chaotic variable of the map at iteration j, $i = 1, 2, ..., N$, $j = 0, 1, ..., M$, N is the number of control variables and M is the total number of solutions generated. D_i determines the range of variation from current solution $S^{current}$ and is initialized as:

$$D_i = 0.1 \times (u_i - l_i)$$

.

3. *UpdateNeighborhood(D):* After a whole iteration for a given temperature is terminated the neighborhood range is updated by:

$$D_i^{new} = (1 - \alpha)D_i^{current} + \alpha\omega R_i$$

where R_i is "the magnitude of the successful change made to the i-th control variable, and α is the damping constant controlling the rate at which information from R_i is folded into D_i with weighting ω" [14].

4 Forecasting NASDAQ with ANCFIS

Many authors have explored the predictive capabilities of ANFIS and ANCFIS for financial time series forecasting. The large design space, especially given the large number of optimization algorithms that can be chosen for both premise and consequent optimization, has led to a wealth of different implementations of real and complex fuzzy logic through adaptive networks.

Financial time series are particularly difficult to predict given their non-stationary and heteroskedastic properties. Various models have been proposed in the finance and statistics literature, from ARIMA [21] to GARCH [26], to deal with these difficulties.
An interesting development that opened up the application of chaos theory to financial markets analysis came with the work of Benoit Mandelbrot. In his famous 1963 paper, Mandelbrot criticized the ubiquity of linear models in economic theory acknowledging that "everybody's preference for such models is of course based upon the unhappy but unquestionable fact that mathematics offers few work-able non-linear tools to the scientist" [28]. In his subsequent works he confuted the Gaussian hypothesis on which financial and economics theories rested. It is safe to say that his work has greatly contributed to increase the awareness of researchers towards non-linear methods for finance. Section 4.2.2. will present a preprocessing method typical of chaotic time series analysis. The method was explored given its potential to increase the information content of model inputs, as suggested in [38]. Moreover, as described in previous chapters ANFIS and ANCFIS belong to the category of non linear models thanks to the non linear activation functions of premise nodes and the role of rule interference in output generation. Therefore, successful application of ANCFIS to financial forecasting adds to the evidence of non-linear methods efficiency in the financial world.

The next section will present a literature review regarding the application of traditional and complex neuro-fuzzy systems to financial forecasting. Section 3.2 will delve into the specific model implemented together with results and discussion.

4.1 Review of ANCFIS for financial time series forecasting

P. Cheng and C. Quek applied ANFIS to predict investor reactions to stock market-related events [27]. K. K. Ang and C. Quek analysed the performance of an automated stock trading system based on forecasts produced by "pseudo outer-product based fuzzy neural network using the compositional rule of inference" [30]. Boyacioglu M. A. and Avci D. applied ANFIS to Instanbul Stock Exchange predictions. [31]. Y.K.

Cheng and C. Quek proposed a combination of ANCFIS and machine learning for financial predictions [32].

Applications of adaptive complex neuro-fuzzy systems are divided between two different implementations. On one hand, we have Complex Neuro-Fuzzy Systems (CNFSs) leveraging the so-called dual-output property of complex fuzzy inference systems. Complex fuzzy memberships obtained in the fuzzification layer are kept throught the whole network, effectively producing a complex output. Real and imaginary parts of the output are used to weight the impact of autoregressive models used in the consequent layer for the production of a final output. Li C. and Chiang T. W. explore the forecasting ability of a CNFS with Particle Swarm Optimization (PSO) for fuzzy antecedent parameters [33], while C. Li and T.W. Chiang investigate the prediction accuracy on three stock indices of a CNFS with an ARIMA model as consequent [34]. On the other hand, we have the previously discussed ANCFIS architecture. O. Yazdanbakhsh and S. Dick proposes a multivariate extension of ANCFIS, which they apply to Nasdaq among other time series [35]. The same authors also proposed "the Fast Adaptive Neuro-Complex Fuzzy Inference System" using "the Fast Fourier Transform algorithm to identify the dominant frequencies in a time series, and then create complex fuzzy sets to match them as the antecedents of complex fuzzy rules" [36].

4.2 Model implementation

The model implementation is based on the work of [14] who proposed the ANCFIS architecture described in section 3.2. Data used are the NASDAQ index weekly and daily closing prices collected from June 1970 to March 2023.

The following sections will give an overview of the feature engineering and specific implementation of ANCFIS. Next, different configurations of models' hyperparameters used for testing are described, together with a presentation and discussion of results.

4.2.1 Data Normalization

There are various reasons why data normalization improves performance of machine learning models. Usually, the advantages of normalization depends on the type of data and the framework chosen to model them. In the case of ANCFIS, the main advantage offered by data normalization is better convergence properties of the gradient-based portion of model training. This is because large values in the loss function caused by the scale of data may lead to wide rates of change of model parameters during learning, which in turn leads to erratic and unstable optimal parameter search.

In most cases properties of time series data hinder the application of well known normalization methods

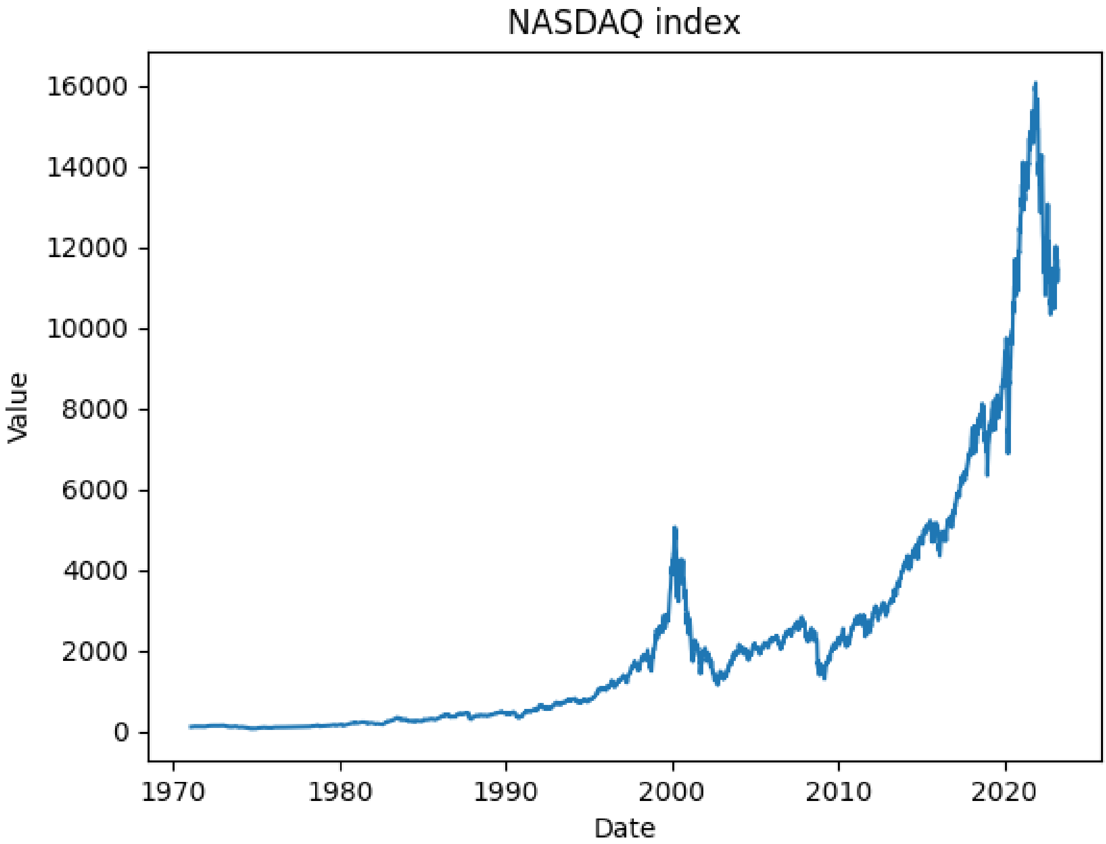

Figure 7: NASDAQ index, weekly data

such as min-max scaling or mean-variance standardization. In fact, time series data may exhibit explosive behavior, meaning that the data generating process may grow indefinitely in size through time. Therefore, establishing a maximum value as a normalization constant means loosing information whenever the process grows large enough to establish a consistently higher range for future observations. Moreover, characteristics of the random process underlying the data, i.e. mean and variance, may change over time. Mean-variance normalization would then produce different normalized values on different time intervals, causing this normalization method to be inconsistent.

Most financial time series, especially stock prices, exhibit both of the above characteristics. For this reason, normalization usually becomes a time-series specific task with a few general methods available to obtain a good balance between constraining data within a range of values and minimizing the information loss in the process of doing so. For example, trend and seasonality analysis are usually applied as first steps in time series normalization.

The data normalization process chosen to evaluate ANCFIS predictive accuracy is a combination of trend analysis and transformation functions. While various other possible techniques were available, this one showed good enough results for normalization purposes. Also, it's simplicity allows for efficient inverse transformation of model outputs.

Figure 8 depicts the logarithmic values of the original time series together with a simple regression over time. A simple detrending of log values using such regression line as a trend estimate yields the normalized data shown in figure 9.

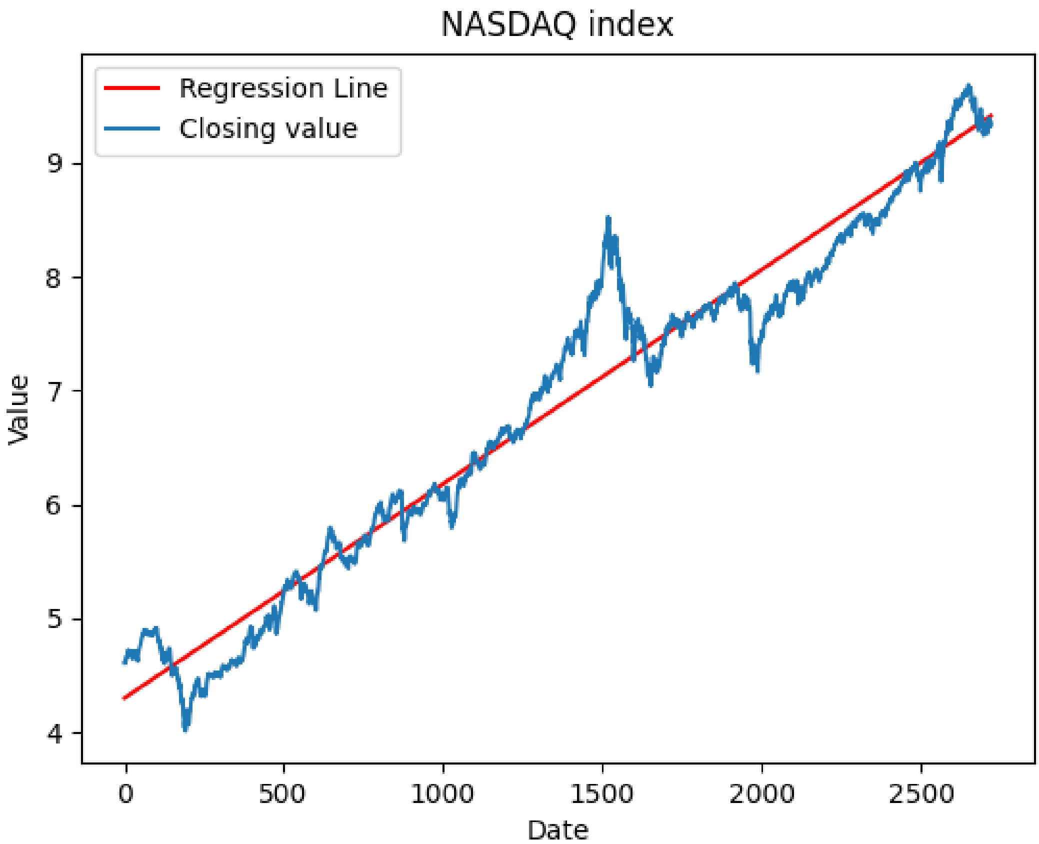

Figure 8: NASDAQ index, weekly log values

4.2.2 Delay Embedding

As briefly mentioned in the introduction to this chapter, the work of Mandelbrot has pioneered research on non-linear methods for financial forecasting. In a related work Hsieh [22] reviewed the state of the art of chaos and nonlinear dynamics applied to financial market. According to him "after the stock market crash of October 19, 1987, interest in nonlinear dynamics, especially deterministic chaotic dynamics,

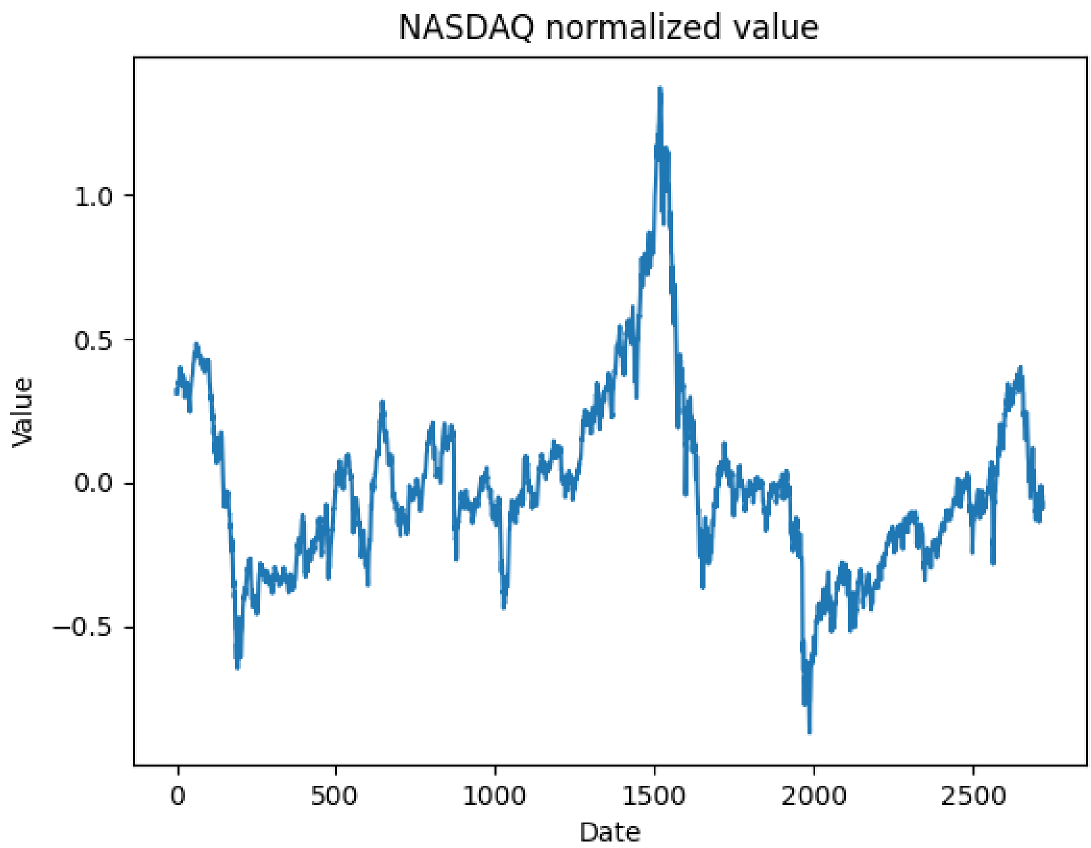

Figure 9: NASDAQ normalized values

has increased in both the financial press and the academic literature. This has come about because the frequency of large moves in stock markets is greater than would be expected under a normal distribution". An early method for deterministic chaos detection through delay time in financial markets was proposed by [23]. A more statistically grounded approach, has been developed in [24]. As a result of this increased interest in nonlinear dynamics, the financial literature has increasingly explored the application of analytic methods typical of dynamical systems theory. An example of this is phase space trajectory analysis.

In complex systems theory it is quite common to represent a system in terms of state vectors. Such state vectors contain all the information describing a system at a given time. This information can be represented in an n-dimensional Cartesian space, where n is the number of variables describing the state of the system recorded at successive time periods, i.e. the phase space of the system. The idea behind phase space analysis is that given enough data about past trajectories it is possible to predict future evolution of the system.

Unfortunately, the variables influencing a dynamical system may not be readily available. A common technique to overcome this issue is delay embedding. It is a method used to construct variates from a time series by sampling past observations at a chosen lag τ. The result of such a method is a number of delay vectors of certain lag and dimension. For example, let $\{x_i : 1 \leq i \leq N\}$ be a sequence of N time-indexed observations. A delay vector of lag τ and dimension d is constructed as follows

$$(x_{m-\tau\cdot d}, x_{m-\tau\cdot(d-1)}, ..., x_{m-\tau\cdot 2}, x_{m-\tau}, x_m)$$

where m is a time index up to N and such that $m \geq \tau \cdot (d-1)+1$. Such delay vector is then used as the new predictor variable instead of the vector of previous d observations. A direct consequence of using a delay vector as the new predictor is that given x_m as the last observation and a lag τ different from 1, the outcome to be predicted is not x_{m+1}, but $x_{m+\tau}$. In other words, the outcome variable to be predicted is not the next one, but the one a time distance τ from the last element of the delay vector.

The objective of delay embedding techniques is to overcome lack of information related to a dynamical system by substituting the state vector with delay vectors. Taken's theorem states that there exist an upper boundary for the minimum dimension of a delay vector to accurately reconstruct the phase space of a dynamical system [37]. Nevertheless, the theorem doesn't state any specific method to determine such delay vector. Therefore, various authors have proposed different delay embedding techniques.

As [38] suggest, "commonly-used heuristics for determining the delay τ include the first zero of the autocorrelation function, or the first minimum of the time-delayed mutual information function". The implementation described in this chapter uses the latter method to find optimal delay. The same authors also suggest the False Nearest Neighbors, as proposed by [39], to determine optimal dimension. This is used in combination with the mutual information criterion to construct delay vectors.

Time-delayed mutual information consists of computing the value of the mutual information function and picking the argument corresponding to the first minima of the function. The formula for time-delayed mutual information function is as follows:

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

where $I(\tau)$ represents the mutual information between the vector v_n and its delayed version $v_{n-\tau}$, $p_{i,j}$ represents the probability that v_n and $v_{n-\tau}$ assume values in the i-th and j-th intervals, respectively, while p_i is the probability that v_n assumes values in the i-th interval. Figure 10 and 11 show the mutual information function for both the weekly and daily timeframe normalized NASDAQ series.

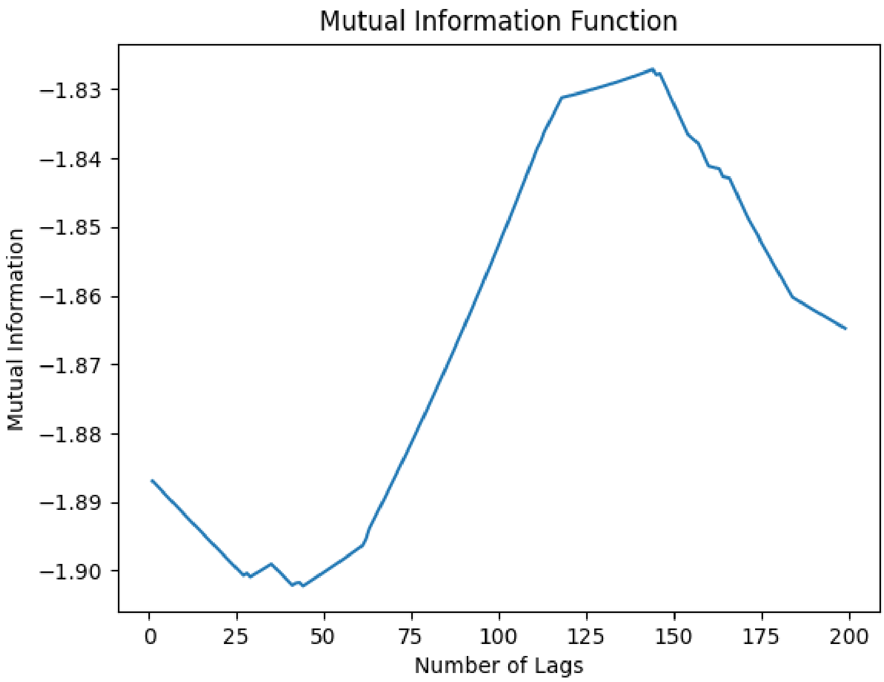

Figure 10: Mutual Information Function for NASDAQ weekly normalized values

According to [25] the optimal number of lags should correspond to the first minimum of the mutual information function. In this case, the weekly data the optimal value is 28, while for daily data it is 124. Nevertheless, as will be explained in the next section, these will not be the only lags used to construct delay vectors.

False Nearest Neighbor is a method for choosing optimal dimension in delay vector construction. It builds upon a simple idea: if delay vectors are to produce a good approximation of the phase space trajectory of a system, then neighbors in state space should not vary as dimension of delay vectors increases. If that would be the case, then two points in phase space would be considered neighbors only because they lay in a lower dimensional subspace onto which the original trajectory has been projected.

The method consists of choosing an initial dimension of 2 and computing the nearest neighbor to each delay vector in such dimension. A neighbor is defined as the pair of points (state vectors) in state space closest to each other, with distance metric:

$$d(x_i^n, x_j^n)^2 = \sum_{k=1}^{n} (x_{i,k} - x_{j,k})^2 \qquad \forall (i,j) \in \mathcal{P}$$

where n is the initial chosen dimension, (x_i, x_j) is a pair of neighbors and $\mathcal{P}$ is the set of all neighbor index pairs. The next step consists of increasing delay vectors dimension by one. The impact of such

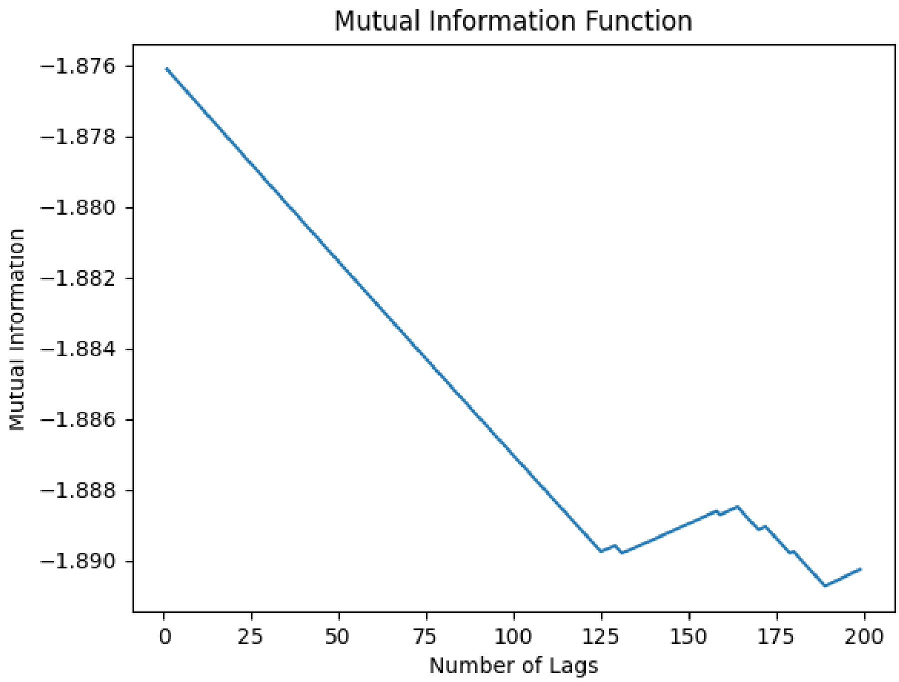

Figure 11: Mutual Information Function for NASDAQ daily normalized values

increase on two neighbors distance is:

$$d(x_i^{n+1}, x_j^{n+1})^2 = d(x_i^n, x_j^n)^2 + (x_{i,n+1} - x_{j,n+1})^2.$$

By setting a threshold R_{tol} for the impact the added dimension can have on the previously computed distance between neighbors, one can identify false neighbors as the percentage of pairs for which the new distance exceeds the previous one by more than such threshold. This procedure is iterated over successively increasing dimensions until a predetermined maximum is reached. For each dimension increase, a percentage of false nearest neighbors with respect to total neighbors is computed and plotted on a line graph. Even though there is no precise cutoff to choose the best dimension, a graphical inspection is usually employed.

The dimension value beyond which a decrease in false neighbors becomes less significant is to be chosen as delay vector dimension. In this case, a dimension of 40 was considered adequate candidates for testing. Results for both daily and weekly data are very similar to each other, which is the reason why only the result on weekly data was reported in Figure 12.

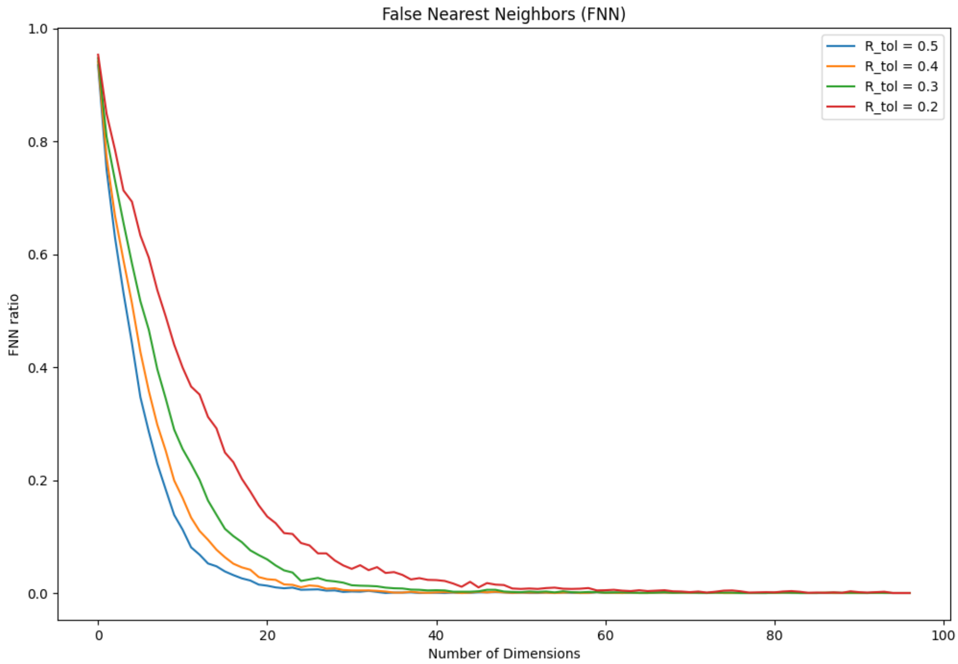

Figure 12: False Nearest Neighbor for normalized data

4.2.3 Results and Discussion

As discussed in section 3.2 the main hyper parameter for ANCFIS is the number of membership functions in the antecedent layer. Therefore, prediction results of different configurations of these were compared using delay vectors with various lag and dimensions as inputs. Specifically, lags of 1, 12, and 28 were chosen as possible candidates for delay vector construction with weekly data. The length of input vectors was fixed at 40 according to the results of False Nearest Neighbor method. Batch learning was implemented, and the first 80% of observations was used for training, while the remaining formed the test set. The training loop was implemented as follows:

1. The model is initialized with the given membership function and some random initial parameters.
2. A forward pass is performed to obtain firing strengths per each input. Such firing strengths are then used for optimizing consequent parameters through Ordinary Least Squares as described in section 2.2.3.
3. A second forward pass is performed to obtain model outputs and update membership function values according to gradient descent.

4. The membership values obtained in step 3 are given as input to the VNCSA algorithm which to update the corresponding membership function parameters. The loss function to be minimized by VNCSA is

$$sup\{|\mu^*(x) - \mu(x)| : x \in X\}$$

where X is the set of membership function values and μ^* is the optimal membership function values determined in step 3.

The training loop iterated over the whole train set per each epoch. A total of 50 epochs were used for training. Table 3 shows the results in terms of RMSE.

Table 3: Results on Weekly data

Lag	**Window**	**NMFS**	**RMSE**
1	40	2	0.02789
1	40	3	0.01351
1	40	4	0.02456
1	40	5	0.03728
12	40	3	0.04761
12	40	4	0.05353
12	40	5	0.06529
28	40	3	0.07024
28	40	4	0.03495
28	40	5	0.05649

Table 4: Results on Daily data

Lag	**Window**	**NMFS**	**RMSE**
1	40	3	0.02514
1	40	4	0.03348
1	40	5	0.04846
124	40	3	0.1205
124	40	4	0.1507
124	40	5	0.13567

According to the RMSE metric, the model performs best with delay vectors of dimension 40 and lag 1, while the optimal number of membership functions is 3 for all lags. Figure 12 and 13 show comparisons between predictions and actual closing price for a portion of both train and test set obtained from the model with lowest RMSE.

Table 4 shows results on daily data. The performance of the best model are inferior with respect to the one trained on weekly data. This can be a result of the greater noise introduced by daily observations.

The impact of noise evidently outweights the information content introduced by finer temporal resolution of data points. Therefore, the following considerations regarding the best model will refer to the one trained on the weekly timeframe.

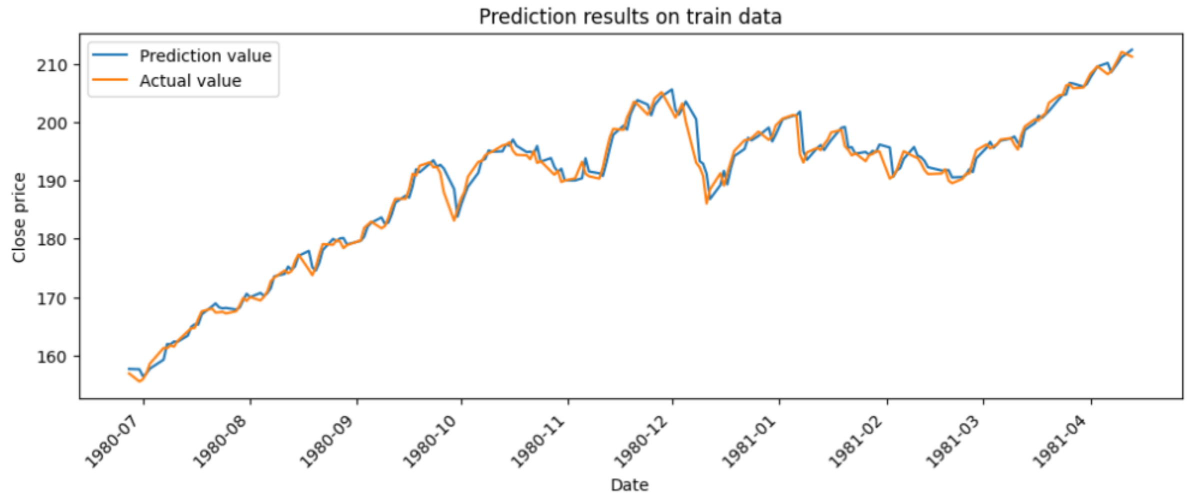

Figure 13: Model predictions with lag 1 delay vector on train data

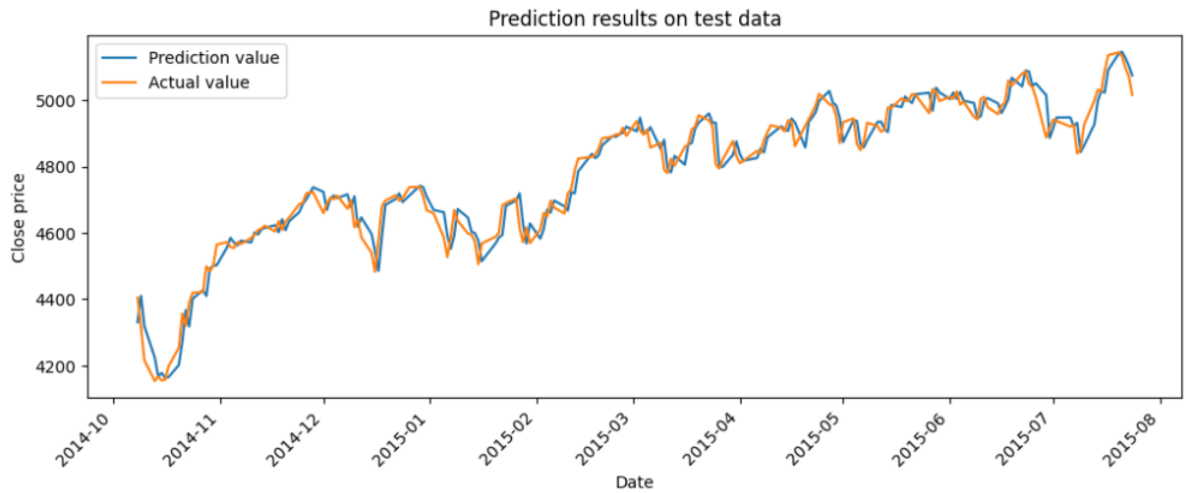

Figure 14: Model predictions with lag 1 delay vector on test data

The results are in line with findings in [38] and [33]. The under performance of delay vectors can be explained by the consequent reduction in data available for training. In fact, the optimal lag of 28 with dimension 40 for weekly data makes the first 1120 (40 times 28) observations unavailable for training. This represents a decrease of around 35% data points for the whole dataset. Similarly, optimal delay vector on daily data reduces by around 32% the number of available data points. When such loss of information is factored in, the decrease in performance seems less surprising. Another interesting aspect is the increase in RMSE caused by a number of membership functions greater than 3. As in many machine learning models, it is almost impossible to define in advance the optimal structure for a specific model, so the only way for the analyst to find the best model configuration is to test the performance impact of different combinations of layers, nodes per layer and corresponding activation functions. In the case of ANCFIS the space of possible configurations for the model is greatly reduced by the fuzzy-system inspired structure. The core idea behind such network is to reproduce the inner workings of a reasoning system, with the difference that fuzzy rules are induced from data rather than deduced from expert knowledge. The result is a fixed model architecture with the number of rules as the only true hyperparameter. Therefore, increasing the number of rules in the model has the same impact of increasing the number of layers or nodes per layer in a regular Artificial Neural Network. As past observations, and their rearrangement through delay vectors, is the only input variable for the model, a parsimonious structure reflected in a somewhat low number of rules is a direct consequence of effectively using a single variable to generate predictions. Notwithstanding this, it is also important to note that parsimony is a characteristic of ANCFIS when compared to its real valued counterpart. In fact, in ANCFIS complex membership functions are evaluated over input vectors. This is not possible in ANFIS, as each membership function can take as input a single real value. Therefore, the three rules of ANCFIS translate to 120 rules in ANFIS (3 memberships per each input times an input length of 40). This comparison shows why ANCFIS has been regarded in the literature as a model particularly suited to deal with complex systems modeling.

Another interesting feature of ANCFIS is the path the loss function follows as the model goes through its training epochs, shown in Figure 14 and observable in [14] and [33]. This highly irregular path differs from most of other machine learning model trained through pure gradient descent. The reason for this stands in the specific optimization routine applied to adjust complex membership parameters. As reported in [14], being gradient descent not applicable to initial model parameters, other search algorithms need to substitute this last step of parameter learning. Genetic algorithms and simulated annealing are good candidates for the task. Such algorithms operate using optimal membership function values obtained through backpropagation as the target values with respect to which complex membership parameters are to be found. While full gradient descent ensures the optimal parameters exactly match the desired output as they are the result of mathematical derivatives computation, other search algorithms necessarily introduce approximation error.

Even though such error can be arbitrarily reduced increasing the amount of computational resources dedicated to this last optimization step, even tiny deviations from optimal membership values can lead to large gaps in the expected model outputs as a result of the complex interactions happening throughout the model. Therefore, the random increases in loss function are caused by the compounded effect of this

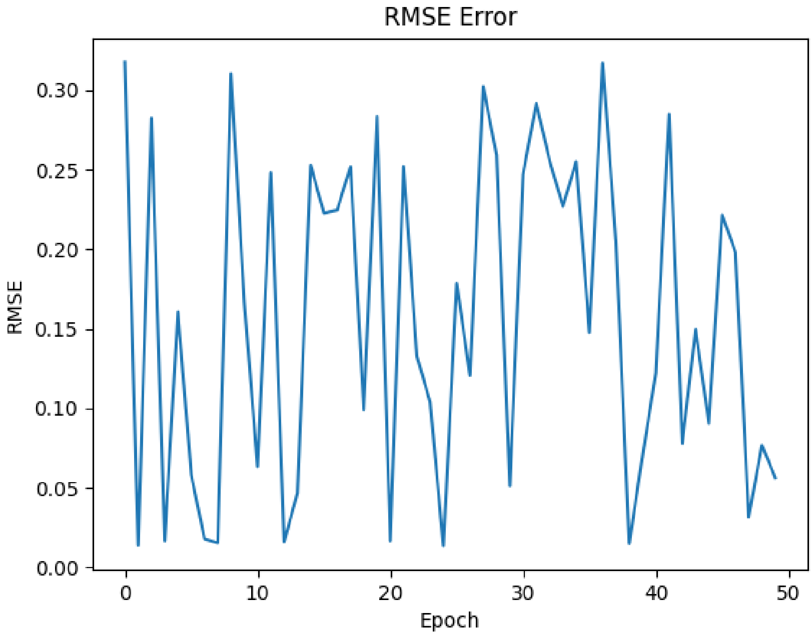

Figure 15: Model predictions with lag 1 delay vector on weekly train data

initial approximation error on model outputs. Nevertheless, this combination of gradient descent with other search algorithms is still efficient in finding satisfactory global minima that produce good overall results as RMSE values and Figures 13 and 14 show.

5 Conclusions

This thesis discussed the predictive capabilities of Adaptive Neuro Complex Fuzzy Inference Systems for financial time series. ANCFIS neural network architecture was presented as a good non linear function approximator and chaotic time series predictor. After introducing the core concepts of real and complex fuzzy logic together with their usage in logical reasoning frameworks, known as complex fuzzy systems, the dissertation delved into the application of neural networks to deduce the structure of the reasoning system from input-output data pairs. The third chapter reviewed the research literature on the application of both ANFIS and ANCFIS to financial time series forecasting. It then introduced the dataset on which predictive accuracy was tested: the NASDAQ index. An overview of the data preprocessing phase was also given. Delay embedding was then presented as a possible method to augment the information content of past observations with the hope to increase model performance. Given the resulting values for lag and dimension of optimal delay vectors, delay embedding was found to decrease prediction accuracy, an effect attributed to the lower number of data points left available for training.

The model performed best on the weekly time frame. This result was connected to the amount of noise introduced by daily timeframe data.

The optimal number of membership functions for the model was found to be three. Even though this value may be surprising given the complexity of the time series under analysis, such parsimony was presented as a feature of ANCFIS, being the model able to exponentially reduce the amount of fuzzy rules needed for an equivalent ANFIS model to deal with the same input data.

The results strengthen the findings of related works such as [14] and [33]. Findings are even more interesting when one considers the widely documented capabilities of ANCFIS for chaotic time series forecasting. In fact, the financial literature has shown increased interest in recent decades towards chaos theory, and more in general dynamical systems theory. Even though the combination of methods derived from chaos theory (delay embedding) failed to produce the expected results in this research, there is a wide range of other possible methods that may be combined with ANCFIS to increase the performance of the model. For example, fractal analysis and Lyapunov exponents may be combined as suggested in [29] to produce different results for delay embedding.

To conclude, ANCFIS was demonstrated to be a rather efficient and parsimonious model for financial forecasting. Furthermore, the results outlined in this research have added empirical evidence to the usefulness of nonlinear models in a financial setting. As a final consideration, it is worth noting that the model is developed assuming no specific knowledge of financial factors affecting the fundamental value of the NASDAQ index. A multivariate extension of the model is particularly straightforward. In fact, as described in [38], it consists of a simple combination of independent univariate models. Therefore, a possible further development of this research would be to combine expert knowledge about macroeconomic and financial factors affecting financial assets prices to increase the amount and quality of input data and evaluate their impact on model performance.

References

[1] Timothy J. Ross (1995). *Fuzzy Logic with Engineering Applications*. Wiley

[2] Zadeh, L. A. (1965). *Fuzzy sets*. Information and Control.

[3] Jang, R. J. (June 1993). *ANFIS Adaptive-Network-based Fuzzy Inference System*. IEEE Transactions on Systems Man and Cybernetics, pp. 665-685

[4] Kaur, A., Kaur, A. (2012). *Comparison of Mamdani-Type and Sugeno-Type Fuzzy Inference Systems for Air Conditioning System*. International Journal of Soft Computing and Engineering

[5] Pourjavad, E., Mayorga, V, R. (2017). *A comparative study and measuring performance of manufacturing systems with Mamdani fuzzy inference system*. Journal of Intelligent Manufacturing

[6] Iqbal, K. et al. (2018). *Intelligent Transportation System (ITS) for Smart-Cities using Mamdani Fuzzy Inference System*. International Journal of Advanced Computer Science and Applications

[7] Jang, J., S., E. Mizutani, (1996). *Levenberg-Marquardt method for ANFIS learning*. Proceedings of North American Fuzzy Information Processing

[8] Kiris, M., et al (2018). *An ANFIS perspective for the diagnosis of type II diabetes*. Research Institute for Basic Science, Wonkwang University

[9] Chang, J., R. et al. (2011). *A hybrid ANFIS model based on AR and volatility for TAIEX forecasting*. Applied Soft Computing

[10] Ramot, D., et al (2003). *Complex Fuzzy Logic*. IEEE Transactions on Fuzzy Systems, vol. 11, no. 4

[11] Yazdanbakhsh, O., Dick, S. (2017). *A Systematic Review of Complex Fuzzy Sets and Logic*. Fuzzy Sets and Systems

[12] Ramot, D. et al. (2002). *Complex fuzzy sets*. IEEE Transactions on Fuzzy Systems, vol. 10, pp. 171–186.

[13] S. Dick, (2005). *Towards complex fuzzy logic*. IEEE Transactions on Fuzzy Systems, vol. 13, no. 3, pp. 405–414

[14] Chen, Z. et al. (2011). *ANCFIS: A Neurofuzzy Architecture Employing Complex Fuzzy Sets*. IEEE Transactions on Fuzzy Systems, vol. 19, no. 2

[15] LI, Y., Jang, Y. (1996). *Complex Adaptive Fuzzy Inference Systems*. Soft Computing in Intelligent Systems and Information Processing, Proceedings of the 1996 Asian.

[16] Shoorangiz, R., Marhaban, M., H. (2013). *Complex Neuro Fuzzy System for Function Approximation*. International Journal of Applied Electronics in Physics and Robotics, Vol. 1 No. 2

[17] Man, J., Y., Chen, Z. and Dick, S. (2007). *Towards inductive learning of complex fuzzy inference systems*. Proc. NAFIPS, San Diego, CA, pp. 415–420

[18] Yazdanbakhsh, O., Dick, S. (2018). *A Systematic review of Complex Fuzzy Sets and Logic*. Fuzzy Sets and Systems 338.

[19] H. Leung and S. Haykin, (September 1991). *The complex back propagation algorithm*. IEEE Trans. Signal Process, Vol. 39, No. 9, pp. 2101–2104,

[20] van Laarhoven, P.J.M., Aarts, E.H.L. (1987). *Simulated annealing. In: Simulated Annealing: Theory and Applications*. Mathematics and Its Applications, Vol 37. Springer, Dordrecht

[21] Box, G, Jenkinks, G., M., Reinsel, G., C., Ljung, G., M. (1970). *Time Series Analysis: Forecasting and Control*. Wiley

[22] Hsieh, D., A. (1991). *Chaos and Nonlinear Dynamics: Application to Financial Markets*. The Journal of Finance, Vol. 46, Issue 5.

[23] Grassberger, P. & Procaccia, I. (1983). *Measuring the strangeness of strange attractors*. Physica 9D, 189–208.

[24] Brock, W., W., Dechert, Scheinkman, J. (1987). *A test for independence based on the correlation dimension*. Working Paper, University of Wisconsin at Madison, University of Houston, and University of Chicago.

[25] Fraser A.,M., H.,L., Swinney (1986). *Independent coordinates for strange attractors from mutual information*. Physical Review A, Vol. 33, No. 2.

[26] Bollerslev, T. (1986). *Generalized autoregressive conditional heteroskedasticity*. Journal of Econometrics, 31, 307–327.

[27] Cheng, P., Quek, C., & Mah, M. (2007). *Predicting the impact of anticipatory action on U.S. stock market – An event study using anfis (a neural fuzzy model)*. Computational Intelligence, 23, 117–141

[28] Mandelbrot, B. (Oct., 1963). *New Methods in Statistical Economics.* Journal of Political Economy, Vol. 71, No. 5 , pp. 421-440

[29] Peters, E. E. (1996). *Chaos and Order in the Capital Markets: a new view of cycles, prices and market volatility.* Wiley

[30] Ang, K., K., Quek, C. (Sept. 2006). *Stock Trading Using RSPOP: A Novel Rough Set-Based Neuro-Fuzzy Approach.* IEEE Transactions on Neural Networks, Vol. 17, No. 5

[31] Boyacioglu M. A. & Avci D. (Dec. 2010) *An Adaptive Network-Based Fuzzy Inference System (ANFIS) for the prediction of stock market return: The case of the Istanbul Stock Exchange.* Expert Systems with Applications , Vol. 37, Issue 12, Pages 7908-7912

[32] Cheng, Y., K., P., and Quek, C. (2011) *Stock trading with cycles: A financial application of ANFIS and reinforcement learning.* Expert Systems with Applications Vol. 38, Issue 5, Pages 4741-4755

[33] Li C. & Chiang T. W. (2012) .*Intelligent financial time series forecasting: A complex neuro-fuzzy approach with multi-swarm intelligence.* International Journal of Applied Mathematics and Computer Science, Vol. 22, No. 4

[34] Li, C. & Chiang, T., W. (2013).*Complex Neurofuzzy ARIMA Forecasting—A New Approach Using Complex Fuzzy Sets.* IEEE Transactions on Fuzzy Systems, Vol. 21, No. 3

[35] Yazdanbakhsh, O. & Dick, S. (2015). *Multi-variate timeseries forecasting using complex fuzzy logic.* 2015 Annual Conference of the North American Fuzzy Information Processing Society (NAFIPS) held jointly with 2015 5th World Conference on Soft Computing (WConSC), Redmond, WA, USA, 2015, pp. 1-6

[36] Yazdanbakhsh, O. & Dick, S. (Feb. 2019). *FANCFIS: Fast adaptive neuro-complex fuzzy inference system.* International Journal of Approximate Reasoning, Vol. 105, Pages 417-430

[37] H. Kantz and T. Schreiber (1997). *Nonlinear time-series analysis.* Cambridge ; New York: Cambridge University Press

[38] Yazdanbakhsh, O., Dick, S. (2018). *Applications of Complex Fuzzy Sets in Time-Series Prediction.* Department of Electrical and Computer Engineering, University of Alberta

[39] Kennel, M., B., Brown, R., Abarbanel, H., D., I., (1992). *Determining embedding dimension for phase-space reconstruction using a geometrical construction.* Physical review A, Vol. 45, No. 6