

Memetic Type-2 Fuzzy System Learning for Load Forecasting

Iván Castro León¹ Philip C. Taylor¹

¹Newcastle University

Abstract

This paper presents an automatic method to design interval type-2 fuzzy systems for load forecasting applications using a memetic algorithm. This hybridisation of a variable-length genetic algorithm and a gradient descent method allows for concurrent learning of the system's parameters and structure in a versatile fashion. Results are presented addressing chaotic system and market-level one-day-ahead load forecasting.

Keywords: Interval type-2 fuzzy systems, memetic learning, load forecasting.

1. Introduction

Accurate one-day-ahead one-hour-resolution on country/market level load forecasting is a key input for economic and secure power systems operation [1, 2]. Many are the tasks that require a precise determination of the future load demand (e.g. unit commitment or economic dispatch, among others). However, as short- and medium-term load behaviour are non-linear and influenced by many factors, forecasting still is a challenging task.

From the several methods which have been proposed in literature for load forecasting [3], two categories can be distinguished: traditional and artificial intelligence methods. The former are restricted by their linearity, although they are significantly more efficient in terms of computation costs. The latter, mainly neural networks and fuzzy systems. Their suitability to forecasting can be explained by the fact that both are universal approximators and can express non-linear relationships. However, despite the fact that there are no consistent comparative results which indicate their supremacy over traditional methods [4], AI-based approaches are considered a suitable approach to load prediction [5].

In [6] a electricity-domain comparative study was performed indicating that interval type-2 fuzzy systems [7] outperform type-1 fuzzy systems and neural networks in terms of generalisation power and forecasting error. Hence, from the AI-based options, interval type-2 fuzzy systems seem to be a suitable candidate. Interval type-2 fuzzy systems

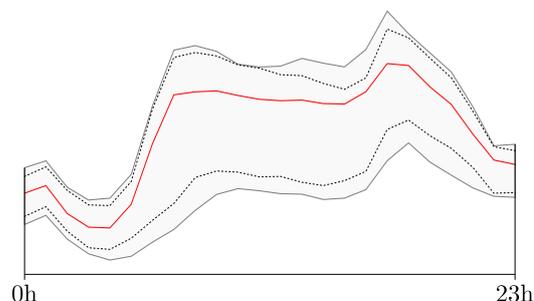


Figure 1: Variability of normalised load demand over 24 hours in the Australian electricity market (Victorian region) on May 2010. Maximum spread is shaded in grey, the median is depicted with a red solid line, whilst the black dotted ones represent the 10th and 90th percentiles. The sample has 1 hour resolution.

performance relies on their capability to handle imprecise information. This is achieved through the use of “*non-crisp*” fuzzy sets rather than “*crisp*” and totally precise as in type-1 fuzzy systems. Furthermore, due to the lower trade-off between modelling accuracy and interpretability in interval type-2 fuzzy systems, a smaller rule-base and fewer number of fuzzy sets to define the universe of discourse can be used [7].

Nevertheless, in interval type-2 fuzzy systems, as in type-1 fuzzy systems, model identification is an important yet challenging problem especially when the number of inputs increases [8, 9]. Structure identification and parameter determination are the main two interwoven tasks to be completed. The first involves variable selection, rule generation, including input space partitioning and membership function specification, and ruleset optimisation. The latter, mainly involves fine-tuning the membership functions which describe rule antecedents and consequents.

Genetic algorithms [10], stochastic search algorithms based on biological principles, are widely used for this purpose [11]. Although, these approaches mainly utilise a fixed-length coding scheme (i.e. a fixed number of rules with a fixed number of inputs and fuzzy sets per rule) which limits their applicability. By employing a variable-length approach inspired by messy coding [12], a flexible fuzzy system representation can be obtained which allows for a variable number of rules, inputs and

This research project is part of “The Autonomic Power System” project and it is supported by the UK Engineering and Physical Science Research Council (EP/I031650/1).

fuzzy sets [13]. However, despite the apparent suitability of genetic algorithms to effectively achieve accurate solutions, there is a gap for improvement. Parameter fine-tuning is still challenging given the unbounded size of the search space even if type-2 fuzzy systems allow for imprecision in their fuzzy set definition [7]. As consequence, a gradient descent method is here embedded to improve accuracy and speed up the learning process. Therefore, this hybridisation of global/local learning, denoted as memetic learning [14, 15], allows for finding a suitable predictor which accurately models the data in a simple-to-complex fashion independently from any kind of initial rule number assumption.

Hence, given that structure identification and parameter determination cannot be independently achieved without resorting to one-another, this paper proposes a method to successfully construct in an automatic fashion an interval type-2 fuzzy system for load forecasting applications. Therefore, being able to increase its generalisation power and reduce the forecasting error without being strongly penalised by the curse of dimensionality.

The rest of the paper is organised as follows: Sections 2 and 3 outline the interval type-2 fuzzy sets and systems which are used in this work and Section 4 describes the proposed hybrid genetic learning approach for their definition and tuning. In Section 5 the proposed scheme is firstly tested and evaluated on a Mackey-Glass time-series ($\tau = 17$), a common benchmark for AI-based forecasters. Subsequently, assessment on real load datasets from an Australian electricity market is carried out. Finally, self-analysis, conclusions and future work are drawn in Section 6.

2. Type-2 Fuzzy Sets

A type-2 fuzzy set [7], denoted by \tilde{F} is characterised by a type-2 membership function $\mu_{\tilde{F}}(x, u)$, where $x \in X$, $u \in J_x \subseteq [0, 1]$ respectively, i.e.

$$\tilde{F} = \{(x, u), \mu_{\tilde{F}}(x, u) \mid \forall x \in X, \forall u \in J_x \subseteq [0, 1]\} \quad (1)$$

in which $0 \leq \mu_{\tilde{F}}(x, u) \leq 1$. \tilde{F} can also be expressed as [7]:

$$\tilde{F} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{F}}(x, u)/(x, u) J_x \subseteq [0, 1] \quad (2)$$

where $\int \int$ denotes union over all admissible x and u [7].

J_x is called primary membership of x , where $J_x \subseteq [0, 1]$ for $\forall x \in X$ [7]. The uncertainty in the primary memberships of a type-2 fuzzy set \tilde{F} , consists of a bounded region called footprint of uncertainty (FOU) defined by the union of all primary memberships [7].

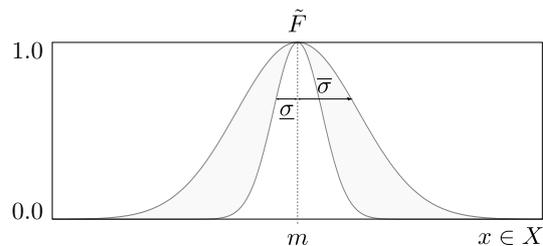


Figure 2: Graphical representation of an interval type-2 gaussian membership function with uncertain deviation (i.e. $\bar{\mu}_{\tilde{F}}(x) = N(m, \bar{\sigma})$, $\underline{\mu}_{\tilde{F}}(x) = N(m, \underline{\sigma})$) as the ones used in this paper. Note that the shaded areas represent its footprint of uncertainty $FOU(\tilde{F})$.

$$FOU(\tilde{F}) = \bigcup_{x \in X} J_x \quad (3)$$

At each value of x , say $x = x'$, the 2D plane whose axes are u and $\mu_{\tilde{F}}(x, u)$ is called a vertical slice of $\mu_{\tilde{F}}(x, u)$. A secondary membership function is a vertical slice of $\mu_{\tilde{F}}(x, u)$. It is $\mu_{\tilde{F}}(x = x', u)$ for $x' \in X$ and $\forall u \in J_x \subseteq [0, 1]$ [7] i.e.,

$$\mu_{\tilde{F}}(x = x', u) \equiv \mu_{\tilde{F}}(x) \quad (4)$$

$$\mu_{\tilde{F}}(x) = \int_{u \in J_{x'}} f_{x'}(u)/(u) J_{x'} \subseteq [0, 1]$$

in which $0 \leq f_{x'}(u) \leq 1$. Because $\forall x' \in X$, the prime notation on $\mu_{\tilde{F}}(x')$ is dropped and refer to as a secondary membership function; it is a type-1 fuzzy set which is also referred to as secondary set [7].

However, due to its complexity, here like in almost all real applications of type-2 fuzzy sets [16] a special case of type-2 fuzzy sets are used to avoid implementation obstacles referred to as interval type-2 fuzzy sets, where $f_x(u) = 1, \forall u \in J_x \subseteq [0, 1]$ [7] and where interval secondary membership functions reflect a uniform uncertainty at the primary memberships of x [7].

The use of interval type-2 fuzzy sets significantly simplifies the computation, an interval type-2 fuzzy set \tilde{F} is fully described by $\underline{\mu}_{\tilde{F}}(x) = FOU(\tilde{F})$ and $\bar{\mu}_{\tilde{F}}(x) = \overline{FOU(\tilde{F})}$ which are the lower and upper membership functions that bound the $FOU(\tilde{F})$ [7] as shown in Figure 2. The FOU provides additional degrees of freedom which make it possible to directly model and handle of uncertainties [7]. Hence, for each x the membership of an interval type-2 set is an interval rather than a crisp number as is the case for type-1 fuzzy sets. Note that if $\overline{FOU(\tilde{F})} = \underline{FOU(\tilde{F})}$ then it reduces to a type-1 fuzzy set.

3. Type-2 Fuzzy Systems

A type-2 fuzzy system describes a non-linear mapping through a set of if-then rules using the previ-

ously described type-2 fuzzy sets. Considering an interval type-2 fuzzy system with p inputs and M rules, let the i -th rule be denoted by R^i [7] as:

$$R^i : \mathbf{if} \ x_1 \text{ is } \tilde{F}_1^i \dots \text{ and } x_p \text{ is } \tilde{F}_p^i \quad (5)$$

$$\mathbf{then} \ y^i \text{ is } \tilde{G}^i = [y_l^i, y_r^i]$$

where \tilde{G}^i is a interval type-1 consequent which corresponds to the centroid of the interval type-2 fuzzy consequent that can be computed via a type-reduction algorithm (e.g. Karnik-Mendel (KM) algorithm [7, 17]).

Hence, as singleton fuzzification is used here and with meet under product t-norm, given a $x' = (x_1 \dots x_p)$, the result of the input and antecedent operations is an interval type-1 fuzzy set $F^i(x')$ [7], i.e.,

$$F^i(x') = [\underline{f}^i(x'), \bar{f}^i(x')] \quad (6)$$

$$\underline{f}^i(x') = \underline{\mu}_{\tilde{F}_1^i}(x'_1) \star \dots \star \underline{\mu}_{\tilde{F}_p^i}(x'_p)$$

$$\bar{f}^i(x') = \bar{\mu}_{\tilde{F}_1^i}(x'_1) \star \dots \star \bar{\mu}_{\tilde{F}_p^i}(x'_p)$$

where \star represents the product operation.

Thus, the interval type-1 fuzzy set output Y which is computed via a type-reduction from a collection of M rules can be expressed as (7) [7], by just finding its two endpoints y_l and y_r . This calculation can be performed for instance via the KM procedure, although there are several methods to perform this task [18] which can affect the system performance. Note that as result, a measure of the uncertainty as it flows through the system is obtained. This interval type-1 output is finally defuzzified into “*crisp*” outputs by computing its mean. As an alternative, type-reduction can be bypassed using the Nie-Tan (NT) algorithm [19] or a neural-net-based algorithm [18]. Although these methods do not provide information about the output uncertainty, both require less computational effort and have shown a higher degree of accuracy when tackling the load forecasting problem [18]. Following is the equation which describes how the final output is computed using the NT algorithm, which has been used in this paper as it does not require any previous knowledge. Note the use of singleton consequents denoted as w rather than interval type-1 fuzzy sets.

$$y = \frac{\sum_{i=1}^M \bar{f}^i w^i + \underline{f}^i w^i}{\sum_{i=1}^M \bar{f}^i + \underline{f}^i} \quad (8)$$

4. Memetic Type-2 Fuzzy System Learning

As introduced in Section 1, the goal is to propose a memetic learning algorithm which concurrently

$$Y = [y_l, y_r] = \int_{y^1 \in [y_l^1, y_r^1]} \int_{y^M \in [y_l^M, y_r^M]} \int_{f^1 \in [\underline{f}^1, \bar{f}^1]} \int_{f^M \in [\underline{f}^M, \bar{f}^M]} 1 \left/ \frac{\sum_{i=1}^M f^i y^i}{\sum_{i=1}^M f^i} \right. \quad (7)$$

Parameters: Population size, stop condition, fitness and selection thresholds, cut & splice and mutation probabilities, gradient descent learning rates and objective function weights.

Input: Normalised training data set.

Output: Interval type-2 fuzzy system.

Begin

Initialise population

Rank population & Local optimisation

while not stop condition do

Hierarchical parent selection

Cut & splice

Mutate

Rank population & Local optimisation

End

End

Table 1: Pseudocode.

designs type-2 fuzzy systems in a versatile and automatic fashion for forecasting applications which seek to handle high-dimensional scenarios. Pseudocode is presented in Table 1.

4.1. Variable-Length Coding Scheme

The need for a variable-coding scheme arises with genetic algorithms when the interwoven structure learning and parameter estimation are carried out concurrently. Not knowing upfront a near-optimal structure in terms of accuracy (i.e. how many fuzzy rules are required or which position or shape should the fuzzy clauses that compose those have), implies modifying fixed-coded chromosomes through trial-and-error. Thus, unlike conventional genetic algorithms, messy genetic algorithms do allow for under-/over-specified chromosomes with respect to the fuzzy system definition [13]. Following a similar approach, a population of interval type-2 fuzzy systems will increasingly evolve to more complex forms (i.e. the number of rules dynamically evolves). This represents a key attribute when one wishes to improve accuracy via increasing the number of inputs to be dealt with.

Here as in [13], fuzzy clauses are the key elements of the coding scheme. However, contrary to [13] only antecedents follow the very same approach. Interval type-2 fuzzy antecedents are represented by 4-tuples $(n, m, \bar{\sigma}, \underline{\sigma})$, in which n is the input index, and $m, \bar{\sigma}, \underline{\sigma}$ the mean, upper and lower deviation of an interval type-2 gaussian membership function. The following text provides an example is provided where a set of unsorted 4-tuples corresponds to a

rule antecedent. Note that due to the NT type-reducer, rule consequents are singletons.

$$\begin{aligned}
& (w^i, \{(1, m_1^i, \bar{\sigma}_1^i, \underline{\sigma}_1^i), (3, m_3^i, \bar{\sigma}_3^i, \underline{\sigma}_3^i)\}) \\
& \quad \downarrow \\
& \tilde{F}_1^i = [N(m_1^i, \bar{\sigma}_1^i), N(m_1^i, \underline{\sigma}_1^i)] \\
& \tilde{F}_3^i = [N(m_3^i, \bar{\sigma}_3^i), N(m_3^i, \underline{\sigma}_3^i)] \\
& \quad \downarrow \\
& R^i : \mathbf{if} \ x_1 \text{ is } \tilde{F}_1^i \text{ and } x_3 \text{ is } \tilde{F}_3^i \ \mathbf{then} \ y^i \text{ is } w_i
\end{aligned}$$

Thus, chromosomes representing interval type-2 fuzzy systems can be simply defined by an unordered and variable number of rules as follows,

$$\{R^0, R^1, \dots, R^i, \dots, R^M\}$$

4.2. Initialisation

Each individual is generated by shuffling, iteratively assessing each example of the input-output training pairs and creating rules where a lack of covering arises (i.e. less than 1% maximum rule firing strength). This random selection of examples maximises the diversity of the initial population whilst also ensures that each example fires at least one fuzzy rule. This mechanism ensures a proper covering over all the training sample [20] and also specifies a variable number of rules in the initial individuals.

Hence, each fuzzy rule is initialised with an unordered set of fuzzy antecedents, one per input (i.e. each initial rule has all the pre-specified inputs). Please note that input selection is beyond the scope of this paper.

Singleton consequents are initially established as random numbers from the interval $[-2, +2]$ as in [7].

4.3. Global Optimisation

As the learning problem can be easily formulated as an optimisation one where the main objective is to minimise the approximation error whilst avoiding over-fitting issues (i.e. lack of generality) and curse of dimensionality problems (i.e. exponential growth of ruleset size as the number of inputs increases).

– *Accuracy:*

$$\mathcal{P}_1 = \frac{1}{N} \sum_{k=1}^N (\hat{y}_k - y_k)^2 \quad (9)$$

where N is the size of the the training sample, \hat{y}_k is the prediction and y_k the actual value.

– *Complexity:*

$$\mathcal{P}_2 = M \quad (10)$$

$$h(S_1, S_2) = \begin{cases} \min(M_{S_1}, M_{S_2}) & \text{if } \min(f(S_1), f(S_2)) \geq \xi_{min} \text{ and } |f(S_1) - f(S_2)| \leq \xi_{sim} \\ \max(f(S_1), f(S_2)) & \text{otherwise} \end{cases} \quad (12)$$

with M being the number of rules. Note that alternative complexity metrics can also be suitable.

Thus, the performance index is defined as follows,

$$f(\mathcal{P}_1, \mathcal{P}_2) = \begin{cases} \frac{\lambda_1}{1+\mathcal{P}_1} & \text{if } \lambda_1 \mathcal{P}_1 \leq \xi_{min} \\ \frac{\lambda_1}{1+\mathcal{P}_1} + \frac{\lambda_2}{1+\mathcal{P}_2} & \text{otherwise} \end{cases} \quad (11)$$

where the step function $f(\mathcal{P}_1, \mathcal{P}_2)$ is used here to ensure ξ_{min} accurate-enough solutions are available before any complexity-pressure is established. This allows for finding building blocks before any pruning is carried out. Weighting factors λ_1, λ_2 simply guide the maximisation towards a specific direction.

4.4. Hierarchical Parent Selection

One of the major issues when variable-coding is employed in genetic algorithms, is the possibility of an uncontrolled growth of the individuals size [21]. This *bloat* phenomenon can directly result in generational augmentation of the fitness computational effort. As a way to minimise its effects a hierarchical selection similar to [21] is here proposed. The selection h occurs according to (12), where given to individuals S_1, S_2 are two individuals with $f(S_1), f(S_2)$ fitnesses, M_{S_1}, M_{S_2} number of rules and ξ_{min}, ξ_{sim} a pair of thresholds. On one hand, ξ_{min} guarantees a minimum level of performance similarly to the objective step function. On the other hand, ξ_{sim} ensures that size prioritisation is carried out only if $f(S_1), f(S_2)$ are similar enough in terms of fitness.

4.5. Genetic Operators

4.5.1. Cut & Splice

The cut & splice operation as in [13] acts on two levels. On the ruleset level, child rulesets are originated from two parents not necessarily using the same crossover points. Following, is an example of this procedure.

$$\begin{aligned}
P_1 &= \{R_1^0, R_1^1, \mid R_1^2\}, P_2 = \{R_2^0, \mid R_2^1, R_2^2\} \\
& \quad \downarrow \\
P_{11} &= \{R_1^0, R_1^1\}, P_{12} = \{R_1^2\} \\
P_{21} &= \{R_2^0\}, P_{22} = \{R_2^1, R_2^2\} \\
& \quad \downarrow \\
C &= \text{choice}(P_{11}, P_{12}) \cup \text{choice}(P_{21}, P_{22})
\end{aligned}$$

Although this makes flexible the number of rules per child, it may occur that meaningful rules from either on or both of the parents could be lost. Hence, a pool of rules with the non-present ones is created to complete the child ruleset ensuring that no lack

of covering arises. A greedy approach which prioritises rules covering larger number of examples is here used for their selection.

Alike [13], on the rule level, a rule from the first parent is firstly selected. The second rule must not be triggered by similar conditions. Hence, each candidate rule is assessed using the Jaccard similarity measure [22] over the training sample, i.e.,

$$S_J(R^a, R^b) = \frac{\sum_{k=1}^N \min(\bar{f}^a, \bar{f}^b) + \min(\underline{f}^a, \underline{f}^b)}{\sum_{k=1}^N \max(\bar{f}^a, \bar{f}^b) + \max(\underline{f}^a, \underline{f}^b)} \quad (13)$$

where R^a, R^b are the compared rules, N the size of the training sample and \underline{f}, \bar{f} their firing interval.

Following is an example of this rule-level cut & splice mechanism.

$$\begin{aligned} R^i &: (w^i, \{(1, m_1^i, \bar{\sigma}_1^i, \underline{\sigma}_1^i), (3, m_3^i, \bar{\sigma}_3^i, \underline{\sigma}_3^i)\}) \\ R^j &: (w^j, \{(1, m_1^j, \bar{\sigma}_1^j, \underline{\sigma}_1^j), (2, m_2^j, \bar{\sigma}_2^j, \underline{\sigma}_2^j)\}) \\ &\downarrow \\ A_1^i &= \{(1, m_1^i, \bar{\sigma}_1^i, \underline{\sigma}_1^i)\} \\ A_2^i &= \{(3, m_3^i, \bar{\sigma}_3^i, \underline{\sigma}_3^i)\} \\ A_1^j &= \{(1, m_1^j, \bar{\sigma}_1^j, \underline{\sigma}_1^j)\} \\ A_2^j &= \{(2, m_2^j, \bar{\sigma}_2^j, \underline{\sigma}_2^j)\} \\ &\downarrow \\ R^c &= (\text{choice}(w^i, w^j), \\ &\quad \text{choice}(A_1^i, A_2^i) \cup \text{choice}(A_1^j, A_2^j)) \end{aligned}$$

Note that as is the previous case, it may occur that the rule ends over- or under-specified. For the former, a first-come first-served basis for the antecedents is established, whilst for the latter, a random completion from a pool of non-present antecedents is followed. Finally, the child's singleton consequent is randomly selected from the cut & spliced rules. Note that these operations occur according to pre-defined probabilities p_{c_s}, p_{c_r} .

4.5.2. Mutation

In this messy-inspired coding scheme mutations also occur in different levels and are also problem-specific; ruleset and rule. In the first one, according to a given p_{m_s} probability, a rule is either generated based on the example with worst covering for that particular system or deleted from the individual's ruleset. Selection for removal occurs in three different ways; the rule with the smallest number of examples covered, the rule with the largest degree of overlapping or a randomly selected one as in [23]. Similarly, in the rule level, the parameters of a randomly selected antecedent or consequent are slightly modified according to a probability p_{m_r} .

4.6. Local Optimisation

Regardless, interval type-2 fuzzy sets are forgiving in respect to membership function parameters. At

each iteration a one-epoch fine-tuning via a gradient descent method is carried out to unveil all individuals accuracy potential and speed up the learning process.

\mathcal{P}_1 the objective function used for this procedure can be found in (9). It should be borne in mind that interval type-2 fuzzy sets with uncertain deviation (Figure 2), Nie-Tan type-reduction (8) and product as t-norm are used. Therefore, firing intervals, updating rules and partial derivatives for a given input-output training pair $(x^{(k)} : y^{(k)})$ can be defined as,

$$\bar{f}^i(x^{(k)}) = \prod_{n=1}^p \bar{\mu}_{\bar{F}_n^i}(x_n^{(k)}) \quad (14)$$

$$\underline{f}^i(x^{(k)}) = \prod_{n=1}^p \underline{\mu}_{\bar{F}_n^i}(x_n^{(k)})$$

$$\theta_j^{i\tau}(k+1) = \theta_j^{i\tau}(k) - \alpha_{\theta\tau} \left. \frac{\partial \mathcal{P}_1}{\partial \theta_j^{i\tau}} \right|_k = \quad (15)$$

$$\begin{aligned} &= \theta_j^{i\tau}(k) - \alpha_{\theta\tau} \times \\ &\times \frac{(\hat{y}(x^{(k)}) - y^{(k)})(w^i(k) - \hat{y}(x^{(k)}))}{\sum_{m=1}^M \bar{f}^m(x^{(k)}) + \underline{f}^m(x^{(k)})} \times \\ &\times \left[\left. \frac{\partial \bar{\mu}_{\bar{F}_j^i}}{\partial \theta_j^{i\tau}} \right|_k \prod_{\substack{n=1 \\ n \neq j}}^p \bar{\mu}_{\bar{F}_n^i}(x_n^{(k)}) + \right. \\ &\left. + \frac{\partial \underline{\mu}_{\bar{F}_j^i}}{\partial \theta_j^{i\tau}} \right|_k \prod_{\substack{n=1 \\ n \neq j}}^p \underline{\mu}_{\bar{F}_n^i}(x_n^{(k)}) \left. \right] \end{aligned}$$

$$w^i(k+1) = w^i(k) - \alpha_w \left. \frac{\partial \mathcal{P}_1}{\partial w^i} \right|_k = \quad (16)$$

$$\begin{aligned} &= w^i(k) - \alpha_w \times \\ &\times \frac{(\hat{y}(x^{(k)}) - y^{(k)})(\bar{f}^i(x^{(k)}) + \underline{f}^i(x^{(k)}))}{\sum_{m=1}^M \bar{f}^m(x^{(k)}) + \underline{f}^m(x^{(k)})} \end{aligned}$$

where $\theta_j^{i\tau}$ is the τ th parameter (i.e. $m_j^i, \bar{\sigma}_j^i, \underline{\sigma}_j^i$) of the j th antecedent and w^i the singleton consequent of the i th rule.

5. Experimental Results

5.1. Chaotic System Forecasting

Chaotic behaviour can be described as bounded fluctuations of the output of a non-linear system with high degree of sensitivity to initial conditions

Population size	30
Iterations	100
ξ_{sim}, ξ_{min}	0.001, 0.8
λ_1, λ_2	0.9, 0.1
p_{c_s}, p_{c_r}	0.6, 0.1
p_{m_s}, p_{m_r}	0.05, 0.05
$\alpha_{\theta_j^\tau}, \alpha_w$	0.01, 0.01

Table 2: Design parameters.

Method	RMSE _{mean}
[25]	0.0285
Proposed	0.0031

Table 3: Mackey-Glass time-series forecasting results and comparison.

[24]. That is to say, trajectories with nearly identical conditions can differ a lot from each other. The Mackey-Glass equation (for $\tau > 17$ exhibits chaos) is a non-linear delay differential equation that has become one of the main benchmarks for AI-based time-series prediction [7].

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x(t-\tau)^c} - bx(t) \quad (17)$$

Chosen constants are $\tau = 17$, $a = 0.2$, $b = 0.1$ and $c = 10$ as in [25]. Hence, the objective is to assess if the proposed learning method is able to produce an interval type-2 fuzzy system which accurately model x_{k+6} from a sample of 1000 input-output data pairs, the first 500 pairs for training and the remaining ones for testing. Selected set of inputs are $\{x_{k-18}, x_{k-12}, x_{k-6}, x_k\}$. Table 2 show the designed parameters chosen for both cases of study. The key for their selection is achieving a balance between the selection threshold, weighting factors and cut & splices rates to obtain a slow growth of the population complexity. A comparison with [25] can be found in Table 3 where the mean the root mean squared error over 10 experiments is used as performance index. These results depict the competence of the proposed scheme to carry out time-series forecasting. This improvement is mainly due to the use of interval type-2 fuzzy logic and a more suited model structure with a larger ruleset.

5.2. One-Day-Ahead Load Forecasting

The objective is to obtain a one-day-ahead one-hour-resolution load forecaster merely relying on historical data, similar to [18]. The historical used dataset corresponds to the Australian electricity market (Victorian region) electric demand for May 2010. This dataset is publicly available at www.aemo.com.au. The selected inputs as in [18] are $\{x_{k-24}, x_{k-25}, x_{k-26}, x_{k-27}\}$ and the output x_{k+24} respectively. Note that 80% of the examples were used for training whilst the remaining 20% for testing. Mean of the root mean squared error over 10 experiments is used again as performance index. The inclusion of more inputs can substantially increase prediction accuracy, although it can increase the model's complexity. Therefore another test was carried out with the following additional inputs $\{x_{k-144}, x_{k-145}, x_{k-168}, x_{k-169}\}$ derived from an auto-/partial-correlation analysis. Despite the training could be considered more challenging, accuracy was improved in similar conditions as previously. Results for both cases are shown in 4 and

Method	Inputs	RMSE _{mean}
[18]	4	405.521
Proposed	4	349.013
	8	278.996

Table 4: One-day-ahead load forecasting results and comparison.

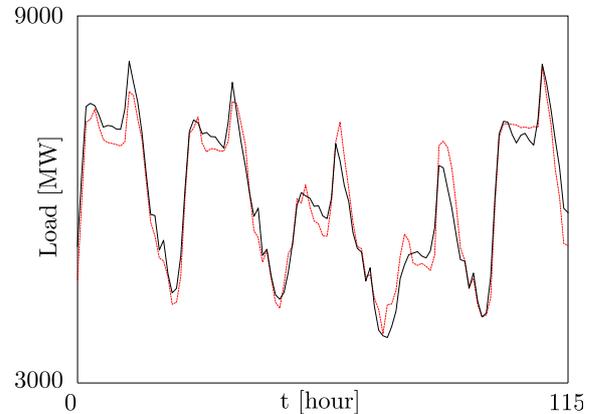


Figure 3: One-day-ahead load forecasting case. The 8-input interval type-2 fuzzy system output is depicted via a red dashed line whilst the actual value via a black solid one.

indicate a significant improvement on accuracy derived from a more data-suited structure. Finally, Figure 3 displays the error between the actual output and the obtained with the 8-input interval type-2 fuzzy system.

6. Conclusions, Self-Analysis & Future Work

This paper introduces a method to successfully and concurrently develop an interval type-2 fuzzy system for load forecasting applications which is able to identify its constituent parameters and structure without being strongly penalised by the curse of dimensionality. Hence, it has shown that by employing a global/local optimisation in conjunction with a variable-length coding scheme, accuracy improvements can be obtained. Consequently, the Table 5 honestly summarises the strengths, weaknesses, opportunities and threats which represent advantages, drawbacks, future work and potential future work risks.

References

- [1] D. Bunn and E. Farmer. *Comparative Models for Electrical Load Forecasting*.
- [2] D.K. Ranaweera, G.G. Karady, and R.G. Farmer. Economic impact analysis of load forecasting. *Power Systems, IEEE Transactions on*, 12(3):1388–1392, Aug 1997.
- [3] H. K. Alfares and M. Nazeeruddin. Electric load forecasting: literature survey and classi-

Strengths
<ul style="list-style-type: none"> – Concurrent design of membership functions and rule sets for interval type-2 fuzzy systems performing a global/local optimisation. – Flexible use of design parameters which suits the problem needs. – Its versatility and the use of type-2 sets allows for good initial solutions.
Weaknesses
<ul style="list-style-type: none"> – Slow convergence due to a notably large search-space. – Higher computational cost when compared to ‘crisp’ fuzzy approaches.
Opportunities
<ul style="list-style-type: none"> – Use of interval consequents that can provide useful information for power systems optimisation. – Improvements via alternative type-reduction algorithms or the use of TSK rather than Mamdani systems. – Thorough statistical assessment (e.g. Diebold-Mariano testing) including more load profiles at different voltage levels and time-scales. Robust experimentation on high-dimensional systems, including computational and scalability assessment.
Threats
<ul style="list-style-type: none"> – Lack of predictability due to an excessive randomness in load profiles as voltage level decreases. – Excessive need for tailoring the algorithm’s parameters when dealing with different real-world datasets.

Table 5: SWOT analysis.

fication of methods. *International Journal of Systems Science*, 33(1), 2002.

[4] K. Metaxiotis, A. Kagiannas, D. Askounis, and J. Psarras. Artificial intelligence in short term electric load forecasting: a state-of-the-art survey for the researcher. *Energy Conversion and Management*, 44(9):1525 – 1534, 2003.

[5] S. Jurado, J. Peralta, A. Nebot, F. Mugica, and P. Cortez. Short-term electric load forecasting using computational intelligence methods. In *Fuzzy Systems (FUZZ), 2013 IEEE International Conference on*, pages 1–8, July 2013.

[6] A. Khosravi, S. Nahavandi, D. Creighton, and D. Srinivasan. Interval type-2 fuzzy logic systems for load forecasting: A comparative study. *Power Systems, IEEE Transactions on*, 27(3):1274–1282, Aug 2012.

[7] Jerry M. Mendel. *Uncertain Rule-Based Fuzzy*

Logic System: Introduction and New Directions. Prentice Hall, 2001.

[8] J. Yen. Fuzzy logic-a modern perspective. *Knowledge and Data Engineering, IEEE Transactions on*, 11(1):153–165, Jan 1999.

[9] Yaochu Jin. Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. *Fuzzy Systems, IEEE Transactions on*, 8(2):212–221, Apr 2000.

[10] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[11] O. Cordon, F. Herrera, F. Gomide, F. Hoffmann, and L. Magdalena. Ten years of genetic fuzzy systems: current framework and new trends. In *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, volume 3, pages 1241–1246 vol.3, July 2001.

[12] D. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3(5):493–530, 1989.

[13] Frank Hoffmann and Gerd Pfister. Evolutionary design of a fuzzy knowledge base for a mobile robot. *International Journal of Approximate Reasoning*, 17(4):447–469, 1997.

[14] Pablo Moscato. On evolution, search, optimization, genetic algorithms and martial arts - towards memetic algorithms, 1989.

[15] Yew-Soon Ong, Meng Hiot Lim, and Xianshun Chen. Memetic computation x2014;past, present x00026; future [research frontier]. *Computational Intelligence Magazine, IEEE*, 5(2):24–31, May 2010.

[16] H. Hagrass. Type-2 flcs: A new generation of fuzzy controllers. *Comp. Intell. Mag.*, 2(1):30–43, February 2007.

[17] Nilesh N. Karnik and Jerry M. Mendel. Centroid of a type-2 fuzzy set. *Information Sciences*, 132(1&4):195 – 220, 2001.

[18] A. Khosravi and S. Nahavandi. Load forecasting using interval type-2 fuzzy logic systems: Optimal type reduction. *Industrial Informatics, IEEE Transactions on*, 10(2):1055–1063, May 2014.

[19] Maowen Nie and Woei Wan Tan. Towards an efficient type-reduction method for interval type-2 fuzzy logic systems. In *Fuzzy Systems, 2008. FUZZ-IEEE 2008. (IEEE World Congress on Computational Intelligence). IEEE International Conference on*, pages 1425–1432, June 2008.

[20] Jorge Casillas, Pedro Martinez, and Alicia D. Benitez. Learning consistent, complete and compact sets of fuzzy rules in conjunctive normal form for regression problems. *Soft Comput.*, 13(5):451–465, 2009.

- [21] Jaume Bacardit and Josep Maria Garrell. Bloat control and generalization pressure using the minimum description length principle for a pittsburgh approach learning classifier system. In *Learning Classifier Systems*, volume 4399 of *Lecture Notes in Computer Science*, pages 59–79. Springer Berlin Heidelberg, 2007.
- [22] Dongrui Wu and Jerry M. Mendel. A comparative study of ranking methods, similarity measures and uncertainty measures for interval type-2 fuzzy sets. *Information Sciences*, 179(8):1169 – 1192, 2009.
- [23] S.E. Papadakis and J.B. Theocharis. A ga-based fuzzy modeling approach for generating {TSK} models. *Fuzzy Sets and Systems*, 131(2):121 – 152, 2002.
- [24] Martin Casdagli. A dynamical systems approach to modeling input-output systems. In *SFI Studies in the Sciences of Complexity*, volume 12, pages 265–281. Addison-Wesley, 1992.
- [25] Ho Jae Lee, Jin Bae Park, and Young Hoon Joo. Fuzzy model identification using a hybrid mgsa scheme with application to chaotic system modeling. In *Integration of Fuzzy Logic and Chaos Theory*, volume 187 of *Studies in Fuzziness and Soft Computing*, pages 81–97. Springer Berlin Heidelberg, 2006.