General Type-2 Fuzzy Logic Systems: A Data-Driven Fuzzy Model

Adrian Rubio-Solis¹, Patricia Melin², Uriel Martinez-Hernandez³ and George Panoutsos¹

Abstract—This paper proposes a new General Type-2 Radial Basis Function Neural Network (GT2-RBFNN) that is functionally equivalent to a GT2 Fuzzy Logic System (FLS) of either Takagi-Sugeno-Kang (TSK) or Mamdani type. The neural structure of the GT2-RBFNN is based on the α-planes representation, in which the antecedent and consequent part of each fuzzy rule uses GT2 Fuzzy Sets (FSs). To reduce the iterative nature of the Karnik-Mendel algorithm, the Enhanced-Karnik-Mendel (EKM) type-reduction and three popular direct-defuzzification methods, namely the 1) Nie-Tan approach (NT), the 2) Wu-Mendel uncertainty bounds method (WU) and the 3) Biglarbigian-Melek-Mendel algorithm (BMM) are employed. For that reason, this paper provides four different neural structures of the GT2-RBFNN and their structural and parametric optimisation. Such optimisation is a two-stage methodology that first implements an Iterative Information Granulation approach to estimate the antecedent parameters of each fuzzy rule. Secondly, each consequent part and the fuzzy rule base of the GT2-RBFNN is trained and optimised using an Adaptive Gradient Descent method (AGD) respectively. Several benchmark data sets, including a problem of identification of a nonlinear system and a chaotic time series are considered. The reported comparative analysis of experimental results is used to evaluate the performance of the suggested GT2-RBFNN with respect to other popular methodologies.

Index Terms—General Type-2 FLSs, Radial Basis Function Neural Networks, α-plane representation, fuzzy modelling.

I. INTRODUCTION

General Type-2 Fuzzy Logic is now well established and is gaining more and more in popularity [1]–[6]. This is mainly credited to the capability of General Type-2 Fuzzy Sets (GT2 FSs) to better handle and minimise the effect of high levels of uncertainty with respect to other high order FSs such as Interval Type-2 Fuzzy Sets (IT2 FSs) [7]–[14]. Compared to Type-1 Fuzzy Sets (T1 FSs) and IT2 FSs, a GT2 FS weights uncertainty nonuniformly and is described by a Membership Function (MF) that is characterised by more parameters, so using GT2 FSs allows for more design degrees of freedom [7], [14]. Furthermore, a GT2 FS is characterised by a Footprint of Uncertainty (FOU) and an MF (secondary MF), where uncertainty can be modelled with any degree between 0 and 1, whereas T1 and IT2 FSs associate uncertainty only to crisp values of 0 or 1 respectively [9].

As indicated in [7], a GT2 FLS can be thought as a high order fuzzy set uncertainty model with more flexibility. Therefore, a GT2 Fuzzy Logic System has the potential to outperform not only the use of FLSs of T1, but also to provide a performance than an FLS with IT2 FSs cannot achieve [7]. Although GT2 FLSs are still in their infancy, the number of applications of higher order fuzzy systems has experienced an important increase during the past five years [15], in particular in areas such as Pattern Recognition [12], [13], Automatic Control [2], [16], Image Processing [17] and Robotics [1], [3], [18]. In this applied context, the usage of GT2 FSs usually increases the computational complexity with respect to T1 and IT2 FLSs. This is clearly compensated not only by a higher model accuracy but also with a better treatment of uncertainty that can be obtained by using GT2 FSs as well as due to new computing technologies. For example in [19], a Mamdani fuzzy neural network with a hidden layer that employs GT2 FSs was proposed. In [19], a comparison about the prediction of noisy time series between the proposed GT2 neural network (NN), a monolithic network and an IT2 NN revealed the superiority of GT2 models to better manage uncertainty. In [17], the authors developed an edge detection system based on a morphological gradient technique and GT2 FSs.
According to [17], the proposed GT2 edge detection architecture showed a higher performance than IT2 and T1 FLSs for edge detection when image processing is under high levels of noise. Similar to T1 and IT2 FLSs, a GT2 FLS involves a similar architecture as illustrated in Fig. 1. Specifically, an FLS can be regarded of GT2 if only one of the associated FSs is of GT2 [7]. In this sense, several efforts have been made to represent GT2 FLSs (or T2 FLSs) [20]–[22]. Particularly, horizontal slice-representation allows using everything learned in IT2 FLSs theory [9]. According to the \( \alpha \)-cut decomposition theorem, \( \alpha \)-cuts decomposition offers a practical way to represent GT2 FLSs (including of IT2 and T1). This is because a GT2 FLS can be represented as the union of all its \( \alpha \)-planes raised to a level \( \alpha \), where each \( \alpha \)-plane is the union of its \( \alpha \)-cuts [9]. Thus, based on the \( \alpha \)-cuts decomposition theorem, at each input \( x' = \bar{x}_p \), a GT2 FLS simultaneously uses \( \alpha \)-cuts for each vertical slice over the secondary MF domain and the associated \( \alpha \)-planes [9].

Based on the \( \alpha \)-plane representation, in this paper a new General Type-2 Radial Basis Function Neural Network (GT2-RBFNN) that is functionally equivalent to a GT2 Mamdani (or TSK) FLS is suggested. To provide a high trade-off between accuracy and model simplicity, two different GT2 RBFNN structures are implemented. On the one hand, to reduce the iterative nature of the Karnik-Mendel method (KM), a GT2 RBFNN with an Enhanced KM algorithm is suggested. On the other hand, three different GT2 RBFNN structures based on direct-defuzzification methods are also presented, i.e. a GT2 RBFNN with a a) Nie-Tan approach, a b) Wu-Mendel Uncertainty bounds method and a c) Biglarbegian-Melek-Mendel procedure. A learning methodology based on an Iterative Information Granulation process (IIG) and an Adaptive Gradient Descent (AGD) approach is implemented to identify the parameters of each antecedent and consequent in the rule base of a GT2 RBFNN. The major contributions of the GT2 RBFNN are twofold. The first contribution is the proposal of a novel RBFNN based on GT2 FSs. Current applications only focus on novel learning methodologies and the implementation of metaheuristics to improve the generalisation properties of the RBFNN. The suggested GT2 RBFNN incorporates GT2 FSs not only to better model and minimise the effects of uncertainty, but also to provide a higher level of model accuracy than its counterparts the RBFNN and the IT2 RBFNN. Compared to ensemble of neural networks where uncertainty is viewed as a measure of disagreement among on some inputs, a GT2 RBFNN treats uncertainty as a deficiency that results not only from imprecise boundaries in the FSs of an RBFN and IT2 RBFNN, but also as consequence of information-based imprecision. The second contribution is the proposal of GT2 RBFNN structures based on direct-defuzzification methods and the implementation of an adaptive learning for model simplification and improvement of the convergence of a traditional gradient descent approach.

The rest of this paper is organised as follows: In Sections II and III, a brief review of T2 FSs and the functional equivalence between the RBFNN and GT2 FLSs is provided. Sections IV and V detail the architecture of a GT2-RBFNN with an EKM and three simplified neural structures respectively. In Section V, a parameter identification approach for the GT2 RBFNN models is described. A comparative analysis and a discussion of experiments results are presented in Sections VI and VII correspondingly. Finally, conclusions are drawn in section IX.

II. GENERAL TYPE-2 FUZZY LOGIC

This section provides a brief review of General Type-2 Fuzzy Sets (GT2 FSs) and theory of \( \alpha \)-plane representation.

A. Definition of a General Type-2 Fuzzy Set

A General Type-2 Fuzzy Set (GT2 FS) denoted by \( \hat{A} \) (also called T2 FS) is characterised by a bivariate MF \( \mu_{\hat{A}}(x,u) \subseteq [0,1] \) on the Cartesian product \( \mu_{\hat{A}} : X \times [0,1] \), where the primary variable is \( x \in X \). And the \( y-\)axis is called secondary variable or primary MF \( u \in J_x \subseteq [0,1] \) as illustrated in Fig. 2. Thus, \( \hat{A} \) is represented by:

\[
\hat{A} = \{(x,u), \mu_{\hat{A}}(x,u) \mid \forall x \in X, \forall u \in J_x \subseteq [0,1] \}
\]

\( \{\mu_{\hat{A}}(u) | u \in U \} \) is a vertical slice of \( \mu_{\hat{A}}(x,u) \) and it can also be represented by its \( \alpha \)-cut decomposition.

B. \( \alpha \)-plane Representation

An \( \alpha \)-plane for a GT2 FS \( \hat{A} \) is denoted by \( \hat{A}_\alpha \), is the union of the primary MFs of \( \hat{A} \) whose secondary grades are greater than or equal to \( \alpha \) (0 ≤ \( \alpha \) ≤ 1)

\[
\hat{A}_\alpha = \{(x,u), \mu_{\hat{A}}(x,u) \geq \alpha | x \in X, u \in [0,1] \}
\]

where the lower and upper limits for \( \hat{A}_\alpha \) are defined by

\[
\begin{align*}
LMF(\hat{A}_\alpha) &= a_\alpha \\
UMF(\hat{A}_\alpha) &= b_\alpha
\end{align*}
\]

That means when \( \hat{A}_\alpha \) is raised to level \( \alpha \), it is a plane at that level that can be obtained by connecting all the corresponding \( \alpha \)-cuts of the associated vertical slices of the secondary MFs of \( x \in X \) [7]. Hence, the horizontal-slice representation of a GT2 FS \( \hat{A} \) is defined by

\[
\hat{A} = \sup_{\alpha \in [0,1]} \alpha/ \left[ \int_{x \in X} [a_\alpha(x), b_\alpha(x)]/x \right] = \bigcup_{\alpha \in [0,1]} \alpha/\hat{A}_\alpha
\]
III. RADIAL BASIS FUNCTION NEURAL NETWORK AND GENERAL TYPE-2 FUZZY LOGIC SYSTEMS

It has been proven that under some mild conditions the RBFNN can be viewed as a Type-1 Fuzzy Logic System of either Mamdani or Takagi-Sugeno-Kang type (TSK) [23], [24]. This equivalence has been further extended in [24] in order to design an Interval Type-2 RBFNN (IT2 RBFNN) with a Karnik-Mendel type-reduction in which all the fuzzy sets are of Interval Type-2. An RBFNN can be regarded as an FLSs whose main inference engine is interpreted as an adaptive filter [7], [24]–[26]. It resembles an additive weighted combination of the MFs of the fired-rule output sets in the hidden layer of the RBFNN (See Fig. 3) [7]. Thereby, every hidden receptive unit in the RBFNN is functionally equivalent to a fuzzy rule \( R^i \) described by a multi-variable Gaussian MF \( \mu_{R^i}(\vec{x}_p, y_p) = \mu_{R^i}[x_1, \ldots, x_n, y] \), where the input vector \( \vec{x}_p \in X_1 \times \ldots X_n \) and the implication engine is defined as:

\[
\mu_{R^i}(\vec{x}_p, y_p) = \mu_{A^i \rightarrow G^i} = \left[ \frac{T_{00}^i}{T_{10}^i,\mu_{F_k}(x_k)} \right] \mu_{G^i}(y) \tag{5}
\]

Where \( \star \) is the minimum \(-\) norm that represents the shortest Euclidean distance to the input vector \( \vec{x}_p \). And each receptive unit is the \( i \)th fuzzy rule:

\[
R^i : \text{IF } x_1 \text{ is } F_1^i \text{ and } \ldots \text{ IF } x_k \text{ is } F_k^i \text{ and } \ldots \\
\text{IF } x_n \text{ is } F_n^i \text{ THEN } y \text{ is } G^i; \ i = 1, \ldots, M \tag{6}
\]

So that, the firing strength \( f_i \) of each receptive unit is

\[
\mu_{A^i \rightarrow G^i}(\vec{x}_p, y) = \prod_{k=1}^{n} \mu_{F_k}(x_k)
\]

\[
= f_i \left( \exp \left( - \frac{\sum_{k=1}^{n} (x_k - m_k)^2}{\sigma_i^2} \right) \right) \tag{7}
\]

where \( A_i = F_1^i \times \ldots \times F_n^i - m_k \) and \( \sigma_i \) are the center and width of a multi-variable Gaussian MF respectively. By combining all the rules in the output layer, \( y_p \) is [27]

\[
y_p = \frac{\sum_{i=1}^{M} \mu_{A^i \rightarrow G^i}(\vec{x}_p, y)w_i}{\sum_{i=1}^{M} \mu_{A^i \rightarrow G^i}(\vec{x}_p, y)} \tag{8}
\]

Strictly speaking, any kind of FLS enhancement might be directly applicable to the RBFNN theory because the structure of its fuzzy rule base in going from T1 FSs to T2 FSs does not change; it is the way the associated antecedents and consequents are modelled [7]. Thus, an RBFNN can be functionally equivalent to a kind of GT2 FLS that is based on the horizontal-slice representation if an RBFNN consists of:

I. An input layer with a singleton fuzzification.
II. The T-norm operator used to compute each rule’s firing strength is multiplication (meet).
III. The secondary MF of each GT2 FS is convex.
IV. The \( \alpha \)-cut of each T1 secondary MF \( A, A_\alpha \) is given by a set of the lower and upper firing strengths \( [f_1^\alpha, f_1^\beta] \) as described in Fig. 4.

The structure of an RBFNN can be viewed as GT2 Takagi-Sugeno-Kang FLS if for each \( i \)th fuzzy rule

\[
\tilde{F}_\alpha^i : \text{IF } x_1 = \tilde{F}_1^i \text{ and } \ldots \text{IF } x_k = \tilde{F}_k^i \text{ and } \ldots \\
\text{IF } x_n = \tilde{F}_n^i \text{ THEN } y = \tilde{g}_i^\alpha(\vec{x}_p); \ i = 1, \ldots, M \tag{9}
\]

whereas for a Mamdani inference fuzzy system, the consequent part is defined as \( y = \tilde{g}^i \). For a GT2 RBFNN of Mamdani (TSK) type, when \( \vec{x}_p = x_i^\alpha \), a vertical slice in the \( i \)th receptive unit for the \( i \)th antecedent \( \tilde{F}_k^i \) is activated, and its \( \alpha \)-cut decomposition is given by:

\[
\tilde{F}_k^i(x_i^\alpha) \Leftrightarrow \mu_{A_k^i \rightarrow G_k^i}(\vec{x}_p, y) = \sup_{\alpha \in [0,1]} \frac{\alpha}{\left\lfloor f_1^\alpha, f_1^\beta \right\rfloor} \tag{10}
\]

For simplicity, it is used \( f_1^\alpha(\vec{x}_p) = f_1^\beta \). Hence, the level \( \alpha \) firing set in each receptive unit is defined by

\[
F_\alpha^i = \left\lfloor f_1^\alpha, f_1^\beta \right\rfloor , \alpha \in [0,1] \tag{11}
\]

Built upon a horizontal-slice representation, a GT2 RBFNN can be defined as [28]

1. A horizontal-slice Mamdani (TSK) FLS that is analogous to an IT2 FLS where a number of operations described for IT2 FSs theory occur for each horizontal slice [7].
2. A Wagn-Hagras (WH) GT2 RBFNN FLS that results from the union over \( \alpha \) of the horizontal-slice Mamdani (TSK) FLSs [14].
IV. GENERAL TYPE-2 RADIAL BASIS FUNCTION NEURAL NETWORK (GT2 RBFNN)

Based on Fig. 5, this section describes the GT2 RBFNN structure that can be viewed as a Mamdani (TSK) GT2 FLS with an Enhanced Karnik-Mendel (EKM) type-reduction layer, where all the FSs are of GT2. For demonstration purposes, here a GT2 RBFNN with an uncertain width $\sigma_i = [\sigma_i^1, \sigma_i^2]$ and a fixed mean $m_k$ is implemented. A horizontal-slice representation is used for the simplest Mamdani (TSK) GT2 RBFNN structure that consists of a singleton fuzzification with a secondary MF that is convex, a Center-Of-Sets (COS) type reduction that uses an EKM and an average of end-points defuzzification (AED) as described in Fig. 6. To avoid additional parameters, and as shown in 4, the secondary MFs are vertical slices, a triangle function is employed where its base is equal to $\bar{f}_i^0 - \overline{f}_i^0$ and its Apex location given by

\[ \text{Apex}(\bar{x}_p) = \overline{f}_i^0(\bar{x}_p) + \frac{1}{2} \alpha \left[ \overline{f}_i^0(\bar{x}_p) - f_i^0(\bar{x}_p) \right]; \alpha \in [0, 1] \]  \hspace{1cm} (12)

A. GT2RBFNN Input Layer

The proposed GT2 RBFNN is a Multi-Input-Single-Output FLS, in which the input data is a multidimensional crisp vector represented by $\bar{x} = [x_1, \ldots, x_n] \in \mathbb{R}^n$ where only the current state is fed into the layer and then forwarded to next layer.

B. General Type-2 RBF Layer

It is assumed a singleton fuzzification, i.e. for each value $x_k$ only a T1 vertical slice for an antecedent GT2 FS $\tilde{F}_k^i$ is activated. Compared to an IT2 RBFNN, to describe each horizontal slice in the GT2 layer of a GT2 RBFNN, a number of $S$ firing strengths $[\tilde{f}_i^\alpha_1, \tilde{f}_i^\alpha_2]$ is required (See Fig. 6) where $\alpha_2 > \alpha_1$. At input $\bar{x}_p$, for each fuzzy rule in the GT2-Mamdani (TSK) RBFNN, only one firing interval $\tilde{f}_i^\alpha$ is activated for level $\alpha$ in the GT2 RBF layer as (See Fig. 6, 7 and 8) [29]:

Note the term $\alpha$ is not a variable [7]. The subscript ‘s’ is used to denote each $\alpha$-level in the GT2 RBFNN.

C. Type-reduction Layer

In the type reduction layer, a Center Of Sets Type Reduction (COS TR) is used. This layer performs a mathematical operation that maps a GT2 FS into a T1 FS. Hence, the centroid of each consequent at the $\alpha_s$-plane is computed as:

\[ C_{\alpha_s} = \alpha_s / [w_{i,\alpha_s}, w_{r,\alpha_s}] \]  \hspace{1cm} (14)

According to [7], [29], for a Mamdani GT2 RBFNN, $[w^i_{1,\alpha_s}, w^i_{r,\alpha_s}]$ is an Interval Weighted Average (IWA) that is used along with the firing interval $\tilde{f}_i^\alpha_s$ to compute the reduced set $[y^i_{\alpha_s}(\bar{x}_p), y^e_{\alpha_s}(\bar{x}_p)]$ for $\alpha_s$-level as:

\[ y^i_{\alpha_s} = \frac{\sum_{i=1}^{L_s} w_{1,\alpha_s} \tilde{f}_i^\alpha_s + \sum_{i=L_s+1}^{M} w_{1,\alpha_s} \tilde{f}_i^\alpha_s}{\sum_{i=1}^{L_s} \tilde{f}_i^\alpha_s + \sum_{i=L_s+1}^{M} \tilde{f}_i^\alpha_s} \]  \hspace{1cm} (15)

\[ y^e_{\alpha_s} = \frac{\sum_{i=1}^{R_s} w_{r,\alpha_s} \tilde{f}_i^\alpha_s + \sum_{i=R_s+1}^{M} w_{r,\alpha_s} \tilde{f}_i^\alpha_s}{\sum_{i=1}^{R_s} \tilde{f}_i^\alpha_s + \sum_{i=R_s+1}^{M} \tilde{f}_i^\alpha_s} \]  \hspace{1cm} (16)

where $Y_{COS,\alpha_s} = 1 / [y^i_{\alpha_s}(\bar{x}_p), y^e_{\alpha_s}(\bar{x}_p)]$. For a TSK GT2 RBFNN, a normalised $A2 - C0$ GT2 FLS version is used in which the antecedents are GT2 FSs, and the associated consequent is $g_{\alpha,\alpha_s} = c_0 + c_1 x_0 + c_2 x_1 + \ldots + c_n x_n$, where $x_0 = 1$ and $c_m^{\alpha,\alpha_s}$, $(m = 0, \ldots, n)$ are crisp numbers.

\[ y^i_{\alpha_s} = \frac{\sum_{i=1}^{L_s} g_{\alpha,\alpha_s} \tilde{f}_i^\alpha_s + \sum_{i=L_s+1}^{M} g_{\alpha,\alpha_s} \tilde{f}_i^\alpha_s}{\sum_{i=1}^{L_s} \tilde{f}_i^\alpha_s + \sum_{i=L_s+1}^{M} \tilde{f}_i^\alpha_s} \]  \hspace{1cm} (17)

\[ y^e_{\alpha_s} = \frac{\sum_{i=1}^{R_s} g_{\alpha,\alpha_s} \tilde{f}_i^\alpha_s + \sum_{i=R_s+1}^{M} g_{\alpha,\alpha_s} \tilde{f}_i^\alpha_s}{\sum_{i=1}^{R_s} \tilde{f}_i^\alpha_s + \sum_{i=R_s+1}^{M} \tilde{f}_i^\alpha_s} \]  \hspace{1cm} (18)
D. Defuzzification Layer

This layer performs defuzzification that consists of a process of aggregation of all horizontal slices. Here, the Average of End-Points Defuzzification (AEFD) is used [14]:

\[ y_p(\bar{x}_p) = \frac{\sum_{s=1}^{S} \alpha_s [(y^{\alpha_s}_r(\bar{x}_p) + y^{\alpha_s}_2(\bar{x}_p))/2]}{\sum_{s=1}^{S} \alpha_s} \tag{19} \]

V. SIMPLIFIED GENERAL TYPE-2 RADIAL BASIS FUNCTION NEURAL NETWORK

In this paper, a GT2 RBFNN that employs a direct-defuzzification algorithm as an output layer is called Simplified General Type-2 Radial Basis Function Neural Network (SGT2 RBFNN). For practical reasons, particularly for real world T2 FLSs, the need to bypass the iterative nature of KM algorithms that results from the number of permutations that are needed to calculate the reduced set has become a priority. Type-reduction is usually used as going from a T2 FS to a T1 FS [30]. In this paper, the term direct-defuzzification and closed-form type reduction are used indistinctly to refer to the mapping that goes from a GT2 FS to a crisp number (type-0). Due to their simplicity and accuracy with respect to KM algorithms, in this paper three popular direct-defuzzification approaches are selected, i.e. a) Nie-Tan closed-form (NT) [31], b) Wu-Mendel Uncertain Bounds approach (WU) [32] and c) the Biglarbegian-Melek-Mendel method (BMM) [33].

A. Simplified Wu-Mendel GT2-RBFNN

The second simplified structure is a Mamdani GT2-RBFNN that employs the Wu-Mendel Uncertain Bounds method and that is called WM GT2-RBFNN for short. For each \( \alpha \)-level in the GT2-RBFNN, the WM method replaces the type reduction with an approach that calculates the inner and outer-bound sets for the type reduced of IT2 FLSs [32]. As shown in Fig. 7, for each input vector \( \bar{x}_p \), the WM GT2RBFNN output is

\[ y_p(\bar{x}_p) = \sum_{s=1}^{S} \alpha_s y_{W,M,\alpha_s} / \sum_{s=1}^{S} \alpha_s \tag{20} \]

For each \( \alpha \)-level, \( y_{W,M,\alpha_s} \) is computed as:

\[ y_{W,M,\alpha_s} = \frac{1}{4} (y^{\alpha_s}_r(\bar{x}_p) + \overline{y}^{\alpha_s}_r(\bar{x}_p) + y^{\alpha_s}_2(\bar{x}_p) + \overline{y}^{\alpha_s}_2(\bar{x}_p)) \tag{21} \]

where

\[
\overline{y}^{\alpha_s}_i = y_i^{\alpha_s} - 
\left[ \frac{\sum_{i=1}^{M} f_i^{\alpha_s} (w_{i,\alpha_s} - w_{i,\alpha_s}) \sum_{i=1}^{M} f_i^{\alpha_s} (w_{i,M} - w_{i,\alpha_s})}{\sum_{i=1}^{M} f_i^{\alpha_s} (w_{i,\alpha_s} - w_{i,\alpha_s}) + \sum_{i=1}^{M} f_i^{\alpha_s} (w_{i,M} - w_{i,\alpha_s})} \right] \tag{22} 
\]

\[
\overline{y}^{\alpha_s}_r = y_r^{\alpha_s} + 
\left[ \frac{\sum_{i=1}^{M} f_i^{\alpha_s} (w_{r,\alpha_s} - w_{r,\alpha_s}) \sum_{i=1}^{M} f_i^{\alpha_s} (w_{r,M} - w_{r,\alpha_s})}{\sum_{i=1}^{M} f_i^{\alpha_s} (w_{r,\alpha_s} - w_{r,\alpha_s}) + \sum_{i=1}^{M} f_i^{\alpha_s} (w_{r,M} - w_{r,\alpha_s})} \right] \tag{23} 
\]
Where \( F_p = \sum_{i=1}^{M} (f^i_\alpha - f^{\alpha}_\alpha) / \sum_{i=1}^{M} f^i_\alpha \), in which, consequents \( w_{i,\alpha} \) are different for each \( \alpha \)-level. The terms \( \bar{y}^i_\alpha \) and \( \bar{y}^{\alpha}_r \) are given by

\[
\bar{y}^i_\alpha = \min \left\{ \sum_{i=1}^{M} f^i_\alpha w_{i,\alpha} / \sum_{i=1}^{M} f^i_\alpha , \sum_{i=1}^{M} f^i_\alpha w_{i,\alpha} / \sum_{i=1}^{M} f^i_\alpha \right\}
\]
(24)

\[
y^{\alpha}_r = \max \left\{ \sum_{i=1}^{M} f^i_\alpha w_{i,\alpha} / \sum_{i=1}^{M} f^i_\alpha , \sum_{i=1}^{M} f^i_\alpha w_{i,\alpha} / \sum_{i=1}^{M} f^i_\alpha \right\}
\]
(25)

### B. Simplified Nie-Tan GT2-RBFNN

The second structure is a GT2-RBFNN that uses the Nie-Tan method as a direct-defuzzification layer as illustrated in Fig. 8. The Nie-Tan is a direct-defuzzification method initially developed for IT2 FLSs. Such method uses the vertical representation of the Footprint of Uncertainty (FOU) [31] before the process of defuzzification to finally compute the centroid of the IT2 FS. The NT layer can be considered a zero order Taylor series approximation of Karnik-Mendel-defuzzification methods. It has been proved the Nie-Tan operator is equivalent to an exhaustive and accurate type-reduction for both discrete and continuous IT2 FSs [31]. Although there has been improvements on the Nie-Tan operator, in this paper, the centroid \( y_{NT,\alpha} \) at each \( \alpha \)-level is calculated as:

\[
y_{NT,\alpha} = \frac{\sum_{i=1}^{M} w_{i,\alpha} (f^i_\alpha + \bar{f}^i_\alpha)}{\sum_{i=1}^{M} \bar{f}^i_\alpha + \sum_{i=1}^{M} f^i_\alpha}
\]
(26)

For each input vector \( \bar{x}_p \), the NT GT2RBFNN output \( y_p(\bar{x}_p) \) is calculated as:

\[
y_p(\bar{x}_p) = \frac{\sum_{s=1}^{S} \alpha_s y_{NT,\alpha}}{\sum_{s=1}^{S} \alpha_s}
\]
(27)

### C. Simplified Biglarbegian-Melek-Mendel GT2-RBFNN

As an alternative to computing the output of a TSK GT2-RBFNN is the Biglarbegian-Melek-Mendel closed form equation [33]. The last simplified neural structure that is called TSK BMM GT2-RBFNN for short. According to Fig. 9, \( y_p \) is:

\[
y_p(\bar{x}_p) = \frac{\sum_{s=1}^{S} \alpha_s y_{BMM,\alpha}}{\sum_{s=1}^{S} \alpha_s}
\]
(28)

for each \( \alpha \)-plane:

\[
y_{BMM,\alpha} = m_{\alpha} y_{m} + n_{\alpha} y^{\alpha}_n
\]
(29)

In which \( y_{m} = \sum_{i=1}^{M} g_{i,\alpha} f_{i}^\alpha / \sum_{i=1}^{M} f_{i}^\alpha \) and \( y_{n} = \sum_{i=1}^{M} g_{i,\alpha} f_{i}^\alpha / \sum_{i=1}^{M} f_{i}^\alpha \). The terms \( f_{i}^\alpha \) and \( \bar{f}_{i}^\alpha \) are calculated using (13), and consequents \( g_{i,\alpha} \) and adaptation parameters \( m_{\alpha} \) and \( n_{\alpha} \) are different for each \( \alpha \)-level, where \( g_{i,\alpha} = c_{i,0}^{\alpha} x_0 + \ldots + c_{i,m}^{\alpha} x_m \) for \( x_0 = 1 \).

### VI. LEARNING METHODOLOGY OF THE GT2 RBFNN

To identify the optimal parameters of the GT2 RBFNN, and its neural structure, a two-stage learning methodology based on the concept of Iterative Information Granulation (IIG) and an Adaptive Gradient Descent (AGD) approach is implemented. IIG is a clustering technique whose main essence is to discover a structure in data while producing representatives called granules [34]. Such granules are formed based on a data compatibility measure, and their geometrical properties are used to estimate the initial values of each antecedent in the GT2 RBFNN (See flow diagram, Fig. 10). Similarly to [26], the number of fuzzy rules or hidden units in the GT2 RBFNN is initially approximated by using the gradient of the compatibility curve that is obtained by the IIG.
In a second stage, AGD is applied to optimise the parameters \(\{\sigma_1^i, \sigma_2^i\}\) and \(m_k^i\) and to determine the optimal number of fuzzy rules according to cross-validation results.

### A. Iterative Information Granulation

In this work, IIG is used not only to granulate/cluster data (Fig. 11), but also used as an approximation to the optimal number of fuzzy rules (hidden nodes) in the GT2 RBFNN as well as the initial values for \(\{\sigma_1^i, \sigma_2^i\}\) and \(m_k^i\) of each MF [26].

The process of IIG is based on a compatibility index \(C(A, B)\) that defines how good is the merging operation of any two granules \(A\) and \(B\). IIG consists of two main steps: [34, 35]:

- Find the two most 'compatible' information granules \(A\) and \(B\) by using Eq. (30) and merge them together as a new information granule \(g_i\) [26]. Where \(g_i\) is defined by its lower and upper corners \((l_{ki}, u_{ki})\) for the dimension \(k'\) and \(i = 1, \ldots, M\).
- Repeat the process of finding the two most compatible granules until a satisfactory data abstraction level is achieved. Where the compatibility ‘C’ is defined as [34]:

\[
C(A, B) = D_{MAX} - d_{A,B} \cdot e^{-\left(\frac{Cardinality_{MAX}}{Cardinality_{MAX}}\right)_{\alpha_A'y_{A,B}/Length_{MAX}}}
\]

(30)

Such as \(D_{MAX}\), \(Length_{MAX}\) and the term \(Cardinality_{MAX}\) is the maximum possible distance and length of a granule and the total number of granules in the data set respectively. \(d_{A,B}\) is the weighted multidimensional average distance of the resulting granule with \(w_k\) playing the importance weight for the dimension \(k\).

\[
d_{A,B} = \frac{1}{n} \sum_{k=1}^{n} w_k (max(u_{Ak}, u_{Bk}) - min(l_{Ak}, l_{Bk}))
\]

(31)

\(g_i\) is used as a fuzzy constraint to extract the initial parameters of LMF and UMF \((m_k^i, \sigma_i)\) which are calculated as:

\[
m_k^i = \frac{1}{2} (l_{ik} - u_{ik}); \ m_k^i = [m_1^i, \ldots, m_n^i]; \ i = 1, \ldots, M
\]

(32)

### B. Adaptive Gradient Descent Approach (AGD)

After structure identification, the common parameters \(m_k^i\) and \(\{\sigma_1^i, \sigma_2^i\}\) of the antecedent GT2 MFs as well as the weighting factors \([w_{1,\alpha}, w_{2,\alpha}]\) at each \(\alpha\)-level of a Mamdani (TSK) GT2 RBFNN should be optimised. Here, an Adaptive Gradient Descent (AGD) approach that evaluates the Root-Mean-Square Error (RMSE) and uses a performance index \(P(t) = \frac{1}{P} \sum_{p=1}^{P} e_p^2\) is applied [37]. For each \(p\) input-output training data \((x_p, y_p)\), \(p = 1, \ldots, P\), a cost function \(E_p = \frac{1}{2} (e_p)^2\) is also defined, where the error \(e_p = (y_p(x_p) - d_p)\), and \(d_p\) is the desired pattern.

To increase the convergence performance of a typical Gradient Descent approach and avoid getting trapped in a local minimum, a momentum term \(\gamma\) is introduced. A self-tuning learning rate \(\beta\) is defined to enhance the learning performance of the GT2-RBFNN. As feedback information, at each current and previous learning iteration ‘t’, the change trend of \(P(t)\) is evaluated and used to adjust the value of \(\gamma\) and \(\beta\) as follows:

- if \(P(t+1) \geq P(t)\) Then
  \[ \beta(t+1) = h_d \alpha(t), \gamma(t+1) = 0 \]
- if \(P(t+1) < P(t)\) and \(\frac{\Delta P(t)}{P(t)} < \delta\) Then
  \[ \beta(t+1) = h_t \alpha(t), \gamma(t+1) = \gamma_0 \]
- if \(P(t+1) < P(t)\) and \(\frac{\Delta P(t)}{P(t)} \geq \delta\) Then
  \[ \beta(t+1) = \beta(t), \gamma(t+1) = \gamma(t) \]
where \(h_d, (0 < h_d < 1)\) and \(h_i, (1 < h_i)\) are the decreasing and increasing factors respectively and \(\delta\) is a threshold rate for the RMSE. Thus, by using an EKM type reduction, at each \(\alpha\)−level of a GT2 RBFNN, the AGD must be able to track the corresponding parameters \(\sigma_i\) and \(m_{ik}\) in the antecedent active branch in which the value of \(L_{\alpha}\) and \(R_{\alpha}\) may change [24]. As pointed out in section III, a GT2 RBFNN is analogous to an IT2 FLS where all the IT2 FS computations occur for each horizontal slice and their aggregation is carried out by a defuzzification process [7]. Hence, for each \(\alpha\)−level, the final AGD equations for the consequents \([w_{l,\alpha}, w_{r,\alpha}]\) of a Mamdani GT2-RBFNN are updated as:

\[
\Delta w^{i}_{l,\alpha}(p + 1) = -\beta \frac{\partial E_p(\bar{x}_p)}{\partial w^{i}_{l,\alpha}} + \gamma \Delta w^{i}_{l,\alpha}(p)
\]

\[
\Delta w^{i}_{r,\alpha}(p + 1) = -\beta \frac{\partial E_p(\bar{x}_p)}{\partial w^{i}_{r,\alpha}} + \gamma \Delta w^{i}_{r,\alpha}(p)
\]

For a TSK GT2-RBFNN, the consequent coefficients \(c^{\alpha}_{m,i}\) are updated according to

\[
\Delta c^{\alpha}_{m,i}(p + 1) = -\beta \frac{\partial E_p(\bar{x}_p)}{\partial c^{\alpha}_{m,i}} + \gamma \Delta c^{\alpha}_{m,i}(p)
\]

To update the common parameters \(m_{ik}\) and \([\sigma^1_i, \sigma^2_i]\)

\[
\Delta m_{ik}(p + 1) = -\beta \frac{\partial E_p(\bar{x}_p)}{\partial m_{ik}} + \gamma \Delta m_{ik}(p)
\]

\[
\Delta \sigma^1_i(p + 1) = -\beta \frac{\partial E_p(\bar{x}_p)}{\partial \sigma^1_i} + \gamma \Delta \sigma^1_i(p)
\]

\[
\Delta \sigma^2_i(p + 1) = -\beta \frac{\partial E_p(\bar{x}_p)}{\partial \sigma^2_i} + \gamma \Delta \sigma^2_i(p)
\]

Therefore, the derivatives \(\partial E_p(\bar{x}_p)/\partial m_{ik}\), \(\partial E_p(\bar{x}_p)/\partial \sigma^1_i\), and \(\partial E_p(\bar{x}_p)/\partial \sigma^2_i\) should equal the addition of their updates for each \(\alpha\)−level as follows

\[
\frac{\partial E_p}{\partial m_{ik}} \bigg|_{\alpha_s} = \sum_{s=1}^{S} \frac{\partial E_p}{\partial m_{ik}} \bigg|_{\alpha_s}
\]

\[
\frac{\partial E_p}{\partial \sigma^1_i} \bigg|_{\alpha_s} = \sum_{s=1}^{S} \frac{\partial E_p}{\partial \sigma^1_i} \bigg|_{\alpha_s}
\]

\[
\frac{\partial E_p}{\partial \sigma^2_i} \bigg|_{\alpha_s} = \sum_{s=1}^{S} \frac{\partial E_p}{\partial \sigma^2_i} \bigg|_{\alpha_s}
\]

where \(\frac{\partial E_p}{\partial m_{ik}} \bigg|_{\alpha_s}\) is the partial derivative of \(E_p\) with respect to the parameter \(m_{ik}\) at the \(s\)th \(\alpha\)−level.

\[
\frac{\partial E_p}{\partial m_{ik}} \bigg|_{\alpha_s} = 2d_{\alpha} e_p \left[ \left( \frac{\partial y^{\alpha_s}_l}{\partial f^{\alpha_s}_l} + \frac{\partial y^{\alpha_s}_r}{\partial f^{\alpha_s}_r} \right) \frac{\partial f^{\alpha_s}_l}{\partial m_{ik}} \right] + \left( \frac{\partial y^{\alpha_s}_l}{\partial f^{\alpha_s}_l} + \frac{\partial y^{\alpha_s}_r}{\partial f^{\alpha_s}_r} \right) \frac{\partial f^{\alpha_s}_l}{\partial m_{ik}}
\]

\[
\frac{\partial E_p}{\partial \sigma^1_i} \bigg|_{\alpha_s} = 2d_{\alpha} e_p \left[ \left( \frac{\partial y^{\alpha_s}_l}{\partial f^{\alpha_s}_l} + \frac{\partial y^{\alpha_s}_r}{\partial f^{\alpha_s}_r} \right) \frac{\partial f^{\alpha_s}_l}{\partial \sigma^1_i} \right]
\]

\[
\frac{\partial E_p}{\partial \sigma^2_i} \bigg|_{\alpha_s} = 2d_{\alpha} e_p \left[ \left( \frac{\partial y^{\alpha_s}_l}{\partial f^{\alpha_s}_l} + \frac{\partial y^{\alpha_s}_r}{\partial f^{\alpha_s}_r} \right) \frac{\partial f^{\alpha_s}_l}{\partial \sigma^2_i} \right]
\]

where the derivatives \(\partial f^{\alpha_s}_l/\partial \sigma^2_i\) and \(\partial f^{\alpha_s}_r/\partial \sigma^2_i\) are zero. At each \(\alpha\)−plane, the AGD approach tracks each permutation that results for the implementation of an EKM method in order to calculate the derivatives with respect to each weight \(\frac{\partial E_p(\bar{x}_p)}{\partial w^{i}_{l,\alpha_s}}\) and \(\frac{\partial E_p(\bar{x}_p)}{\partial w^{i}_{r,\alpha_s}}\) and the coefficients \(c^{\alpha}_{m,i}\) of a GT2 RBFNN of Mamdani and TSK type respectively. In [24–26], a detailed description of this calculation for an IT2 fuzzy NN and an IT2 RBFNN with a Mamdani inference and using a KM method is provided.

### C. AGD for simplified GT2-RBFNN

Compared to a GT2-RBFNN that utilizes an EKM algorithm, direct-defuzzification-based structures do not need a sorting process. Thereby, the implementation of the AGD to identify the parameters of a GT2-RBFNN of Mamdani (TSK) type results much simpler.

1) Wu-Mendel GT2-RBFNN: To update the parameters of a WM GT2RBFNN of Mamdani type, the derivatives with respect to the weighting factors \(w^{i}_{l,\alpha_s}\) and \(w^{i}_{r,\alpha_s}\), and common parameters \(\sigma^1_i, \sigma^2_i, \) and \(m_{ik}\) are:

\[
\frac{\partial E_p}{\partial w^{i}_{l,\alpha_s}} = d_{\alpha} e_p \frac{\partial y^{\alpha_s}_l}{\partial w^{i}_{l,\alpha_s}}, \quad \frac{\partial E_p}{\partial w^{i}_{r,\alpha_s}} = d_{\alpha} e_p \frac{\partial y^{\alpha_s}_r}{\partial w^{i}_{r,\alpha_s}}
\]

\[
\frac{\partial E_p}{\partial m_{ik}} = \sum_{s=1}^{S} \frac{\partial E_p}{\partial m_{ik}} \bigg|_{\alpha_s}
\]

so that:

\[
\frac{\partial E_p}{\partial \sigma^1_i} \bigg|_{\alpha_s} = d_{\alpha} e_p \left[ \left( \frac{\partial y^{\alpha_s}_l}{\partial f^{\alpha_s}_l} + \frac{\partial y^{\alpha_s}_r}{\partial f^{\alpha_s}_r} \right) \frac{\partial f^{\alpha_s}_l}{\partial \sigma^1_i} \right]
\]

\[
\frac{\partial E_p}{\partial \sigma^2_i} \bigg|_{\alpha_s} = d_{\alpha} e_p \left[ \left( \frac{\partial y^{\alpha_s}_l}{\partial f^{\alpha_s}_l} + \frac{\partial y^{\alpha_s}_r}{\partial f^{\alpha_s}_r} \right) \frac{\partial f^{\alpha_s}_l}{\partial \sigma^2_i} \right]
\]

Where the term \(\partial y^{\alpha_s}_l/\partial w^{i}_{l,\alpha_s}\) and \(\partial y^{\alpha_s}_r/\partial w^{i}_{r,\alpha_s}\) is

\[
\frac{\partial y^{\alpha_s}_l}{\partial w^{i}_{l,\alpha_s}} = \begin{cases} f_{l, \alpha_s} \sum_{i=1}^{M} f_{i, \alpha_s} \min \left\{ y_{l, \alpha_s}^{(0)}, y_{l, \alpha_s}^{(M)} \right\} = y_{l, \alpha_s}^{(0)} \\ \end{cases}
\]

\[
\frac{\partial y^{\alpha_s}_r}{\partial w^{i}_{r,\alpha_s}} = \begin{cases} f_{r, \alpha_s} \sum_{i=1}^{M} f_{i, \alpha_s} \min \left\{ y_{r, \alpha_s}^{(0)}, y_{r, \alpha_s}^{(M)} \right\} = y_{r, \alpha_s}^{(M)} \\ \end{cases}
\]
A similar procedure is used to calculate the terms $\frac{\partial y_i^{\alpha \gamma}}{\partial w_i^{\alpha \gamma}}$ and $\frac{\partial y_i^{\alpha \gamma}}{\partial w_i^{\alpha \gamma}}$. To exemplify the computation of the derivatives $\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial m_k^i}$, $\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial \gamma_k}$, $\frac{\partial \vec{f}_i^{\alpha \gamma}}{\partial \vec{x}_i^{\alpha \gamma}}$, $\frac{\partial \vec{f}_i^{\alpha \gamma}}{\partial \vec{f}_i^{\alpha \gamma}}$, $\frac{\partial \vec{f}_i^{\alpha \gamma}}{\partial \vec{f}_i^{\alpha \gamma}}$ and $\frac{\partial \vec{f}_i^{\alpha \gamma}}{\partial \vec{f}_i^{\alpha \gamma}}$, the calculation of $\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial \vec{x}_i^{\alpha \gamma}}$ and $\frac{\partial \vec{f}_i^{\alpha \gamma}}{\partial \vec{f}_i^{\alpha \gamma}}$ is shown below:

$$
\frac{\partial y_i^{\alpha \gamma}}{\partial \vec{x}_i^{\alpha \gamma}} = \left\{ \begin{array}{l}
\frac{w_i^{\alpha \gamma}}{\left( \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} + \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \right)^2} - \left( \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} - \vec{f}_i^{\alpha \gamma} \right) \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \\
0,
\end{array} \right.
$$

$$
\frac{\partial y_i^{\alpha \gamma}}{\partial \vec{f}_i^{\alpha \gamma}} = \vec{y}_i^{\alpha \gamma} + \left( -v_q \frac{\sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( \vec{f}_i^{\alpha \gamma} - \vec{f}_i^{\alpha \gamma} \right) M}{\left( \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \right)^2} + v_r \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \right) \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma}
$$

in which, the terms $v_q$ and $v_r$ are:

$$
v_q = \left( \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) + \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \right)
$$

$$
v_r = \left( \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) + \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \right)
$$

In which $v_l$ is used to simplify notation as $v_l = \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right) + \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} \left( w_i^{\alpha \gamma} - w_i^{\alpha \gamma} \right)$.

2) Nie-Tan GT2-RBFNN: For a NT GT2-RBFNN of Mamdani type, the AGD equations are defined for the consequent weight of each $\alpha-$level as an spike $w_i^{\alpha \gamma}$ and updated as:

$$
\Delta w_i^{\alpha \gamma} (p + 1) = -\frac{\partial E_p(\vec{x}_p)}{\partial w_i^{\alpha \gamma}} + \gamma \Delta w_i^{\alpha \gamma} (p) \quad (56)
$$

To implement the AGD for a Mamdani NT GT2-RBFNN, the derivative $\frac{\partial E_p(\vec{x}_p)}{\partial w_i^{\alpha \gamma}}$ in (56) is updated as:

$$
\frac{\partial E_p(\vec{x}_p)}{\partial w_i^{\alpha \gamma}} = \left[ \frac{\partial E_p(\vec{x}_p)}{\partial \vec{x}_i^{\alpha \gamma}} \frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial w_i^{\alpha \gamma}} \right] + \left[ \frac{\partial E_p(\vec{x}_p)}{\partial \vec{x}_i^{\alpha \gamma}} \frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial w_i^{\alpha \gamma}} \right] \quad \text{(57)}
$$

where: $\frac{\partial E_p(\vec{x}_p)}{\partial \vec{x}_i^{\alpha \gamma}} = \alpha_s / \sum_{s=1}^{S} \alpha_s$ and

$$
\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial w_i^{\alpha \gamma}} = \frac{\vec{f}_i^{\alpha \gamma} + \vec{f}_i^{\alpha \gamma}}{\sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma} + \sum_{i=1}^{M} \vec{f}_i^{\alpha \gamma}} \quad \text{(58)}
$$

Consequently, $\sigma_1^2$, $\sigma_2^2$ and $m_k^i$ are adjusted as

$$
\frac{\partial E_p(\vec{x}_p)}{\partial \sigma_i^1} = \frac{\partial E_p(\vec{x}_p)}{\partial \vec{x}_i^{\alpha \gamma}} \frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial \sigma_i^1} \quad \text{(59)}
$$

$$
\frac{\partial E_p(\vec{x}_p)}{\partial \sigma_i^2} = \frac{\partial E_p(\vec{x}_p)}{\partial \vec{x}_i^{\alpha \gamma}} \frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial \sigma_i^2} \quad \text{(60)}
$$

where

$$
\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial \sigma_i^1} = \vec{f}_i^{\alpha \gamma} \left( x_k - m_k^i \right)^2 / \left( \sigma_i^1 \right)^3 \quad \text{(61)}
$$

in which

$$
\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial m_k^i} = 2 \vec{f}_i^{\alpha \gamma} \left( x_k - m_k^i \right)^2 / \left( \sigma_i^1 \right)^2 \quad \text{(62)}
$$

$$
\frac{\partial \vec{x}_i^{\alpha \gamma}}{\partial \sigma_i^2} = 2 \vec{f}_i^{\alpha \gamma} \left( x_k - m_k^i \right)^2 / \left( \sigma_i^2 \right) \quad \text{(63)}
$$
For a TSK GT2-RBFNN, the AGD approach follows a similar procedure described in Eq. (59-63). However, the consequent coefficients $c_{m,\alpha}^i$ for each $\alpha$-level are updated as:

$$\frac{\partial E_p}{\partial c_{m,\alpha}^i} = \frac{\partial E_p}{\partial y_{NT,\alpha}} \left( \frac{x_m (f_i^{\alpha}_{m} + f_i^{\alpha}_{s}) + \sum_{m=1}^{M} f_i^{\alpha}_{s}}{\sum_{m=1}^{M} f_i^{\alpha}_{m} + \sum_{m=1}^{M} f_i^{\alpha}_{s}} \right)$$

(64)

where $m = 0, \ldots, n$, such that $x_0 = 0$ and $c_{m,\alpha}^i = 1$.

3) Biglarbegian-Melek-Mendel GT2-RBFNN: Based on the AGD approach, in order to update the parameters of a TSK BMM GT2-RBFNN with an uncertain $[\sigma_1^2, \sigma_2^2]$, fixed mean $m_k$ and consequents $g_{i,\alpha}$, for each $\alpha$-level

$$\frac{\partial E_p}{\partial c_{m,\alpha}^i} = 4d_{\alpha} \epsilon_{pm} \left( \frac{f_m^{\alpha}_{i} + f_s^{\alpha}_{i}}{\sum_{s=1}^{S} f_m^{\alpha}_{i} + \sum_{s=1}^{S} f_s^{\alpha}_{i}} \right)$$

(65)

To update $\sigma_1^2$, $\sigma_2^2$ and $m_k$

$$\frac{\partial E_p}{\partial \sigma_1^2} = 8d_{\alpha} \epsilon_{pm} \left( g_{i,\alpha} - y_m^{\alpha}_{i} \right) \left( f_m^{\alpha}_{i} \sum_{k=1}^{N} (x_k - m_k)^2 \right)$$

(66)

$$\frac{\partial E_p}{\partial \sigma_2^2} = 8d_{\alpha} \epsilon_{pm} \left( g_{i,\alpha} - y_m^{\alpha}_{i} \right) \left( f_m^{\alpha}_{i} \sum_{k=1}^{N} (x_k - m_k)^2 \right)$$

(67)

And

$$\frac{\partial E_p}{\partial m_k} = 8d_{\alpha} \epsilon_{pm} \left( g_{i,\alpha} - y_m^{\alpha}_{i} \right) \left( \sum_{i=1}^{M} f_m^{\alpha}_{i} + \sum_{i=1}^{M} f_s^{\alpha}_{i} \right)$$

$$\times \left[ f_m^{\alpha}_{i} \sum_{k=1}^{N} (x_k - m_k)^2 \right]$$

(68)

Please note for each $\alpha$-level, a different value for the coefficients $c_{m,\alpha}^i$ is employed.

VII. PERFORMANCE VERIFICATION

In this section, three different examples are used to compare the performance of the GT2 RBFNN structures with some well known algorithms such as the ANFIS, a Sequential Adaptive Fuzzy Inference System (SAFIS) [38], a network of Functionally Weighted Single-Input-Rule-Modules connected to a Fuzzy Inference System (FWSIRM-FIS) [38], Support Vector Regression (SVR), RBFNN of T1 and IT2, an ensemble of T1 RBFNNs based on a Negative Correlation Learning (E-RBFNN) [39], Support Vector Machine (SVM) [40], Least Square SVM (LS-SVM) [40] and an IT2 Fuzzy Neural Network with support vector regression (IT2-FNN-SVR) [41].

While the first example involves the modelling of 10 real-world benchmark data sets for multiclass classification and regression problems, the last two examples are used for nonlinear plant identification and chaotic time series prediction in the precene of randomness and Gaussian noise respectively. For the ANFIS, RBFNN, IT2-RBFNN and GT2 RBFNN models and E-RBFNN, all the simulations are carried out in MATLAB 2014 environment in an Intel Core i7, 2.7 GHZ CPU. Similar to a GT2 RBFNN, an AGD version is implemented to train the RBFNN and the IT2 RBFNN [24], [42].

A. Example 1: Modelling of Benchmark Data sets for Multiclass Classification and Regression

This example compares the performance of a GT2 RBFNN, RBFNN, IT2 RBFNN, E-RBFNN, ANFIS, SVM and LS-SVM on five real-world benchmark data sets for regression and five data sets for multiclass classification. In Table II and III, the specifications of the data sets are listed. The associated distributions of the data sets are unknown and most of them noisy-free. As indicated in Tables II and III, for cross-validation purposes the number of samples for training (column train) and testing (column test) are randomly selected. By increasing/decreasing by one the number of hidden units initially estimated by the IIG algorithm, the optimal number of hidden units (fuzzy rules) in the GT2 RBFNN are selected based on cross-validation results. In Tables II-III, column fuzzy rules is used to indicate the optimal value for the number of hidden units for the RBFNN, IT2 RBFNN and GT2 RBFNN models. It is selected a granulation factor of $\alpha_g = 0.3$, an initial value for $\Delta \sigma_i = 0.1$, $\sigma_i^1 = 1$ and $\sigma_i^2 = 0.1$ and $w_{i,\alpha} = 1$. For a TSK GT2 RBFNN with a BMM method, it was determined that the best value for $n_{m} = 0.9$ and $n_{e} = 0.1$. For all GT2 RBFNN models and for the E-RBFNN, it was found the best trade-off between accuracy and model simplicity is achieved by using three horizontal slices and 4 units in the hidden layer. In Tables IV and V, the generalisation performance of SVM, and LS-SVM presented in [41] is compared to the average performance results of 20 trials for the ANFIS, RBFNN, IT2 RBFNN, E-RBFNN and GT2 RBFNN. As indicated in [40], SVM and LS-SVM usually achieves a good generalisation performance. This heavily depends on the combination of values for the cost parameter $C$ and kernel parameter $\gamma$. Therefore, for each data set a large number of combinations to find the appropriate the $C$ and $\gamma$ is required. Opposite to this, from Tables II-V, it can be observed that a GT2 RBFNN needs a small number of hidden units to obtain a higher generalisation performance with respect to SVM and LS-SVM. This model simplification compensates the associated learning time that in most cases is similar to the time used to train an IT2 RBFNN, an ANFIS and less to an E-RBFNN, in particular, the simplified GT2 RBFNNs.

<table>
<thead>
<tr>
<th>TABLE II: SPECIFICATION OF MULTICLASS CLASSIFICATION.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasets</td>
</tr>
<tr>
<td>Iris</td>
</tr>
<tr>
<td>Wine</td>
</tr>
<tr>
<td>Glass</td>
</tr>
<tr>
<td>Segment</td>
</tr>
<tr>
<td>Shuttle</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE III: SPECIFICATION OF REGRESSION DATA SETS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasets</td>
</tr>
<tr>
<td>Pyrim</td>
</tr>
<tr>
<td>Housing</td>
</tr>
<tr>
<td>Space-ga</td>
</tr>
<tr>
<td>Abalone</td>
</tr>
<tr>
<td>HPC</td>
</tr>
</tbody>
</table>
### TABLE IV: AVERAGE PERFORMANCE OF 20 TRIALS OF GT2-RBFNN, IT2-RBFNN, RBFNN, E-RBFNN, ANFIS, SVM AND LS-SVM.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Testing (%)</th>
<th>Training Time (s)</th>
<th>Testing (%)</th>
<th>Training Time (s)</th>
<th>Testing (%)</th>
<th>Training Time (s)</th>
<th>Testing (%)</th>
<th>Training Time (s)</th>
<th>Testing (%)</th>
<th>Training Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>Mamdani (EKM)</td>
<td>99.23</td>
<td>1.57</td>
<td>100.0</td>
<td>1.88</td>
<td>70.22</td>
<td>2.01</td>
<td>99.28</td>
<td>98.12</td>
<td>100.0</td>
<td>10911.9</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>99.62</td>
<td>1.40</td>
<td>99.80</td>
<td>1.70</td>
<td>70.53</td>
<td>1.80</td>
<td>98.01</td>
<td>59.25</td>
<td>99.98</td>
<td>6329.1</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>99.82</td>
<td>1.78</td>
<td>100.0</td>
<td>1.94</td>
<td>70.01</td>
<td>2.41</td>
<td>99.07</td>
<td>84.91</td>
<td>100.0</td>
<td>9873.1</td>
</tr>
<tr>
<td>Wine</td>
<td>Mamdani (EKM)</td>
<td>100.0</td>
<td>1.75</td>
<td>100.0</td>
<td>1.89</td>
<td>69.23</td>
<td>3.84</td>
<td>99.41</td>
<td>93.10</td>
<td>13301.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>99.11</td>
<td>1.62</td>
<td>100.0</td>
<td>1.89</td>
<td>69.25</td>
<td>4.76</td>
<td>98.09</td>
<td>79.82</td>
<td>100.0</td>
<td>8802.1</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>98.39</td>
<td>1.39</td>
<td>100.0</td>
<td>2.18</td>
<td>70.30</td>
<td>3.78</td>
<td>99.10</td>
<td>88.12</td>
<td>100.0</td>
<td>14119.8</td>
</tr>
<tr>
<td>Glass</td>
<td>Mamdani (EKM)</td>
<td>99.82</td>
<td>1.78</td>
<td>100.0</td>
<td>1.94</td>
<td>70.01</td>
<td>2.41</td>
<td>99.07</td>
<td>84.91</td>
<td>100.0</td>
<td>9873.1</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>99.11</td>
<td>1.62</td>
<td>100.0</td>
<td>1.89</td>
<td>69.25</td>
<td>4.76</td>
<td>98.09</td>
<td>79.82</td>
<td>100.0</td>
<td>8802.1</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>98.39</td>
<td>1.39</td>
<td>100.0</td>
<td>2.18</td>
<td>70.30</td>
<td>3.78</td>
<td>99.10</td>
<td>88.12</td>
<td>100.0</td>
<td>14119.8</td>
</tr>
<tr>
<td>Segment</td>
<td>Mamdani (EKM)</td>
<td>99.23</td>
<td>1.57</td>
<td>100.0</td>
<td>1.88</td>
<td>70.22</td>
<td>2.01</td>
<td>99.28</td>
<td>98.12</td>
<td>100.0</td>
<td>10911.9</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>99.62</td>
<td>1.40</td>
<td>99.80</td>
<td>1.70</td>
<td>70.53</td>
<td>1.80</td>
<td>98.01</td>
<td>59.25</td>
<td>99.98</td>
<td>6329.1</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>99.82</td>
<td>1.78</td>
<td>100.0</td>
<td>1.94</td>
<td>70.01</td>
<td>2.41</td>
<td>99.07</td>
<td>84.91</td>
<td>100.0</td>
<td>9873.1</td>
</tr>
<tr>
<td>Shuttle</td>
<td>Mamdani (EKM)</td>
<td>100.0</td>
<td>1.75</td>
<td>100.0</td>
<td>1.89</td>
<td>69.23</td>
<td>3.84</td>
<td>99.41</td>
<td>93.10</td>
<td>13301.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>99.11</td>
<td>1.62</td>
<td>100.0</td>
<td>1.89</td>
<td>69.25</td>
<td>4.76</td>
<td>98.09</td>
<td>79.82</td>
<td>100.0</td>
<td>8802.1</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>98.39</td>
<td>1.39</td>
<td>100.0</td>
<td>2.18</td>
<td>70.30</td>
<td>3.78</td>
<td>99.10</td>
<td>88.12</td>
<td>100.0</td>
<td>14119.8</td>
</tr>
</tbody>
</table>

### TABLE V: AVERAGE RMSE OF 20 TRIALS OF GT2-RBFNN, IT2-RBFNN, RBFNN, E-RBFNN, ANFIS, SVM, LS-SVM.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Testing (RMSE)</th>
<th>Training Time (s)</th>
<th>Testing (RMSE)</th>
<th>Training Time (s)</th>
<th>Testing (RMSE)</th>
<th>Training Time (s)</th>
<th>Testing (RMSE)</th>
<th>Training Time (s)</th>
<th>Testing (RMSE)</th>
<th>Training Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pyrim</td>
<td>Mamdani (EKM)</td>
<td>0.0411</td>
<td>17.88</td>
<td>0.0845</td>
<td>2.78</td>
<td>0.0219</td>
<td>67.12</td>
<td>0.0299</td>
<td>60.11</td>
<td>5.170</td>
<td>173.1</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>0.0397</td>
<td>15.61</td>
<td>0.0801</td>
<td>1.89</td>
<td>0.0205</td>
<td>68.10</td>
<td>0.0301</td>
<td>63.03</td>
<td>5.310</td>
<td>177.9</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>0.0388</td>
<td>18.03</td>
<td>0.0811</td>
<td>2.72</td>
<td>0.0307</td>
<td>63.03</td>
<td>5.310</td>
<td>177.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Space-ga</td>
<td>Mamdani (EKM)</td>
<td>0.0478</td>
<td>19.41</td>
<td>0.0816</td>
<td>2.83</td>
<td>0.0198</td>
<td>71.06</td>
<td>0.0255</td>
<td>66.16</td>
<td>5.392</td>
<td>189.0</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>0.0401</td>
<td>14.99</td>
<td>0.0802</td>
<td>2.39</td>
<td>0.0109</td>
<td>56.19</td>
<td>0.0301</td>
<td>50.24</td>
<td>5.210</td>
<td>169.2</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>0.0428</td>
<td>18.77</td>
<td>0.0833</td>
<td>2.91</td>
<td>0.0165</td>
<td>75.89</td>
<td>0.0270</td>
<td>68.09</td>
<td>5.504</td>
<td>157.2</td>
</tr>
<tr>
<td>Abalone</td>
<td>Mamdani (EKM)</td>
<td>0.0604</td>
<td>14.08</td>
<td>0.0919</td>
<td>2.30</td>
<td>0.0469</td>
<td>47.19</td>
<td>0.0579</td>
<td>42.10</td>
<td>5.870</td>
<td>158.1</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>0.0678</td>
<td>13.02</td>
<td>0.0973</td>
<td>1.88</td>
<td>0.0466</td>
<td>44.06</td>
<td>0.0609</td>
<td>38.04</td>
<td>5.723</td>
<td>140.1</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>0.0699</td>
<td>14.58</td>
<td>0.1104</td>
<td>1.93</td>
<td>0.0679</td>
<td>51.02</td>
<td>0.0599</td>
<td>47.26</td>
<td>5.640</td>
<td>158.3</td>
</tr>
<tr>
<td>HPC (RMSE)</td>
<td>Mamdani (EKM)</td>
<td>0.0645</td>
<td>14.90</td>
<td>0.1095</td>
<td>2.52</td>
<td>0.0397</td>
<td>56.69</td>
<td>0.0601</td>
<td>50.30</td>
<td>5.560</td>
<td>166.2</td>
</tr>
<tr>
<td></td>
<td>NT</td>
<td>0.0609</td>
<td>14.72</td>
<td>0.1020</td>
<td>1.79</td>
<td>0.0487</td>
<td>47.63</td>
<td>0.0544</td>
<td>42.51</td>
<td>5.504</td>
<td>157.2</td>
</tr>
<tr>
<td></td>
<td>WM</td>
<td>0.0689</td>
<td>15.26</td>
<td>0.1118</td>
<td>2.33</td>
<td>0.0481</td>
<td>61.19</td>
<td>0.0520</td>
<td>54.47</td>
<td>5.670</td>
<td>175.7</td>
</tr>
</tbody>
</table>

To take full advantage of the equivalence between a GT2 RBFNN and GT2 FLSs, in this example a GT2 RBFNN with an EKM is used to provide some insights about the HPC data. High Performance Concrete (HPC) data is a collection of 1030 multi-dimensional samples where each set of points represents 8 inputs variables (cement, fly ash, water, superplasticiser, coarse aggregate, fine aggregate, age of testing and blast furnace slag, kg/m$^3$) and one output (Concrete Compressive Strength-MPa, CCS) [43], [44]. To illustrate model performance and physical interpretation, in Fig. 12-14, the data fit for CCS prediction for a Mamdani GT2 RBFNN with an EKM with 8 fuzzy rules and its variable effect surface for the ingredients cement and fly ash and final rule distribution for the input superplasticiser are presented respectively. A variable effect surface is created by keeping $N - 2$ input variables constant and plotting the remaining varying input variables. Here, the average of each input variable is used as a constant for the $N - 2$ variables. As indicated in [24], by using variable effect surfaces, expert’s opinion can confirm the behaviour of specific input variables with respect to a desired output.
The signal $u(t)$ has been randomly generated by a uniform distribution in the region $[-1.5, 1.5]$. For testing purposes, a data set of 200 observations has been generated where the input $u(t)$ is given by $u(t) = \sin(2\pi t/25)$. The experimental setup for the GT2 RBFNN models consists of a number of 3 horizontal slices, a granulation factor of $\alpha_g$ and three fuzzy rules. An initial value for $\Delta\alpha_1 = 0.05$, $\sigma^2 = 1.0$ and each factor $w_i^0 = w_i^1 = 1.0$. Based on simulation results, it was found for a TSK GT2 RBFNN with a BMM type reduction, the best value for $m_\alpha = 0.85$ and $n_\alpha = 0.15$. For an E-RBFNN, it was determined the optimal value to provide a high level of generalisation is with 4 hidden units, where each has 3 fuzzy rules. Table VI shows the average generalisation performance of 20 trials, the number of parameters per each model as well as the Average Training Time ATT of each GT2 RBFNN model with respect to an FWSIRM [38], SANFIS [38], RBFNN, IT2 RBFNN [24], E-RBFNN [39] and the ANFIS system. According to Table VI, the highest trade-off between accuracy and model simplicity is obtained by the RBFNN of GT2 using a NT algorithm. From Table VI, it is clear for most of GT2 RBFNN models the training time is comparable to that of some models such as the BPNN and RBFNN. It is worth noting, the generalisation performance of an E-RBFNN is higher than an IT2 RBFNN and similar to a GT2 RBFNN. Both, GT2 RBFNN and E-RBFNN treat uncertainty as measure for ambiguity. However, a GT2 RBFNN quantifies uncertainty as a deficiency that results not only from imprecise boundaries in the fuzzy sets (vagueness or fuzziness), but also as nonspecificity that refers to information-based imprecision, whereas an E-RBFNN defines ambiguity as a variation of the output of the ensemble members over unlabeled data. That means, uncertainty quantification is useful in an ensemble only if there is a disagreement among on some inputs [45].

### B. Example 2: Nonlinear Plant Identification

This example is to identify the nonlinear plant described by the equation below [38]:

$$y(t + 1) = f(y(t), y(t - 1), u(t)) = \frac{y(t)y(t - 1)(y(t) - 0.5)}{1 + y^2(t) + y^2(t - 1)} + 1$$  \(\text{(69)}\)

The equilibrium state of the unforced system given by Eq. (69) is $(0, 0)$. As [38], the training data consists of $5000 \times 3$ input vectors $[y(t) \ y(t - 1) \ u(t)]$ and one output $y(t + 1)$. The signal $u(t)$ has been randomly generated by a uniform distribution in the region $[-1.5, 1.5]$. For testing purposes, a data set of 200 observations has been generated where the input $u(t)$ is given by $u(t) = \sin(2\pi t/25)$. The experimental setup for the GT2 RBFNN models consists of a number of 3 horizontal slices, a granulation factor of $\alpha_g$ and three fuzzy rules. An initial value for $\Delta\alpha_1 = 0.05$, $\sigma^2 = 1.0$ and each factor $w_i^0 = w_i^1 = 1.0$. Based on simulation results, it was found for a TSK GT2 RBFNN with a BMM type reduction, the best value for $m_\alpha = 0.85$ and $n_\alpha = 0.15$. For an E-RBFNN, it was determined the optimal value to provide a high level of generalisation is with 4 hidden units, where each has 3 fuzzy rules. Table VI shows the average generalisation performance of 20 trials, the number of parameters per each model as well as the Average Training Time ATT of each GT2 RBFNN model with respect to an FWSIRM [38], SANFIS [38], RBFNN, IT2 RBFNN [24], E-RBFNN [39] and the ANFIS system. According to Table VI, the highest trade-off between accuracy and model simplicity is obtained by the RBFNN of GT2 using a NT algorithm. From Table VI, it is clear for most of GT2 RBFNN models the training time is comparable to that of some models such as the BPNN and RBFNN. It is worth noting, the generalisation performance of an E-RBFNN is higher than an IT2 RBFNN and similar to a GT2 RBFNN. Both, GT2 RBFNN and E-RBFNN treat uncertainty as measure for ambiguity. However, a GT2 RBFNN quantifies uncertainty as a deficiency that results not only from imprecise boundaries in the fuzzy sets (vagueness or fuzziness), but also as nonspecificity that refers to information-based imprecision, whereas an E-RBFNN defines ambiguity as a variation of the output of the ensemble members over unlabeled data. That means, uncertainty quantification is useful in an ensemble only if there is a disagreement among on some inputs [45].

### TABLE VI: COMPARISON OF THE AVERAGE PERFORMANCE OF 20 TRIALS OF DIFFERENT MODELS IN EXAMPLE 2.

<table>
<thead>
<tr>
<th>Model</th>
<th>Testing RMSE</th>
<th>Number of Parameters</th>
<th>ATT (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Best</td>
<td></td>
</tr>
<tr>
<td>GT2-RBFNN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mamdani</td>
<td>EKM 0.0266</td>
<td>0.0151</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>NT 0.0289</td>
<td>0.0134</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>WM 0.0288</td>
<td>0.0188</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>EKM 0.0267</td>
<td>0.0192</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>NT 0.0201</td>
<td>0.0177</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>WM 0.0256</td>
<td>0.0163</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>EKM 0.0445</td>
<td>0.0276</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>NT 0.0339</td>
<td>0.0194</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>WM 0.0439</td>
<td>0.0312</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>EKM 0.0458</td>
<td>0.0374</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>NT 0.0479</td>
<td>0.0348</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>WM 0.0481</td>
<td>0.0365</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>RBFNN 0.0501</td>
<td>0.0408</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>E-RBFNN 0.0470</td>
<td>0.0161</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>ANFIS 0.0580</td>
<td>0.0474</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>FWSIRM-FIS 0.0494</td>
<td>0.0274</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>SANFIS 0.0221</td>
<td></td>
<td>85</td>
</tr>
<tr>
<td></td>
<td>BPNN 0.0939</td>
<td>0.0611</td>
<td>151</td>
</tr>
</tbody>
</table>
The input data was divided into two subsets, i.e. a) and b) were introduced in [41]. The IT2-FNN-SVR is a six-layer interval type-2 fuzzy modelling methodologies, namely: a) an IT2FNN-SVR-(N), b) an IT2-FNN-SVR-(F) and an c) EKM IT2-RBFNN and d) E-RBFNN. The first two models a) and b) were introduced in [41]. The IT2-FNN-SVR is a six-layer interval type-2 fuzzy neural network with support vector machine regression that uses two different types of input nodes. For the first type, the input nodes in an IT2-FNN-SVR simply forwards each numerical data and is called IT2-FNN-SVR-(N) for short. Thus, the output of the IT2-FNN-SVR-(N) is a bounded interval which is described in terms the lower and upper limits of its Footprint Of Uncertainty (FOU). An IT2-FNN-SVR-(F) uses an input node layer that fuzzifies the input numerical data. The third IT2 methodology is an IT2-RBFNN with an EKM approach. And the last methodology is an ensemble of RBFNNs suggested in [39]. According to our experiments, it was determined a number of 3 horizontal slices for an IT2-RBFNN, and 3 hidden units with 3 fuzzy rules each for an E-RBFNN produce the highest balance between model performance and model simplicity. For statistical purposes, each experiment was repeated 10 times, and the RMSE average is used as a comparison performance index. Table VII and VIII show the training and testing results for the prediction of the Mackey-Glass time-series. From Table V, it can be viewed that in general GT2 neural structures outperform the IT2-FNN-SVR and its counterpart the IT2-RBFNN with an EKM. It is also worth noting, the superiority of the GT2-RBFNN is confirmed not only for validation purposes, but also in relation to the number of parameters.

### TABLE VII: PERFORMANCE OF THE MAMDANI (TSK) GT2-RBFNN AND OTHER MODELS WITH A TRAINING NOISE $\sigma = 0.2$ in example 3.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>EKM</th>
<th>NT</th>
<th>WM</th>
<th>EKM</th>
<th>NT</th>
<th>WM</th>
<th>SVR-(N)</th>
<th>SVR-(F)</th>
<th>E-RBFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Parameters</td>
<td>65</td>
<td>45</td>
<td>65</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>103</td>
<td>103</td>
<td>35</td>
</tr>
<tr>
<td>Number of Rules</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Training RMSE ($\sigma = 0.2$)</td>
<td>0.091</td>
<td>0.089</td>
<td>0.088</td>
<td>0.092</td>
<td>0.086</td>
<td>0.087</td>
<td>0.234</td>
<td>0.233</td>
<td>0.125</td>
</tr>
<tr>
<td>Clean</td>
<td>0.060</td>
<td>0.071</td>
<td>0.061</td>
<td>0.064</td>
<td>0.060</td>
<td>0.062</td>
<td>0.085</td>
<td>0.083</td>
<td>0.085</td>
</tr>
<tr>
<td>Test RMSE</td>
<td>$\sigma = 0.1$</td>
<td>0.067</td>
<td>0.068</td>
<td>0.070</td>
<td>0.073</td>
<td>0.075</td>
<td>0.071</td>
<td>0.105</td>
<td>0.103</td>
</tr>
<tr>
<td>$\sigma = 0.3$</td>
<td>0.097</td>
<td>0.107</td>
<td>0.095</td>
<td>0.108</td>
<td>0.096</td>
<td>0.102</td>
<td>0.186</td>
<td>0.180</td>
<td>0.122</td>
</tr>
</tbody>
</table>

### TABLE VIII: PERFORMANCE OF THE MAMDANI (TSK) GT2-RBFNN AND OTHER MODELS WITH A NOISE $\sigma = 0.3$ in example 3.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>EKM</th>
<th>NT</th>
<th>WM</th>
<th>EKM</th>
<th>NT</th>
<th>WM</th>
<th>SVR-(N)</th>
<th>SVR-(F)</th>
<th>E-RBFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Parameters</td>
<td>65</td>
<td>45</td>
<td>65</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>103</td>
<td>103</td>
<td>35</td>
</tr>
<tr>
<td>Number of Rules</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Training RMSE ($\sigma = 0.3$)</td>
<td>0.111</td>
<td>0.108</td>
<td>0.122</td>
<td>0.121</td>
<td>0.117</td>
<td>0.114</td>
<td>0.349</td>
<td>0.347</td>
<td>0.133</td>
</tr>
<tr>
<td>Clean</td>
<td>0.085</td>
<td>0.088</td>
<td>0.083</td>
<td>0.079</td>
<td>0.069</td>
<td>0.078</td>
<td>0.127</td>
<td>0.121</td>
<td>0.092</td>
</tr>
<tr>
<td>Test RMSE</td>
<td>$\sigma = 0.1$</td>
<td>0.109</td>
<td>0.096</td>
<td>0.091</td>
<td>0.081</td>
<td>0.083</td>
<td>0.105</td>
<td>0.138</td>
<td>0.131</td>
</tr>
<tr>
<td>$\sigma = 0.3$</td>
<td>0.125</td>
<td>0.118</td>
<td>0.131</td>
<td>0.133</td>
<td>0.127</td>
<td>0.129</td>
<td>0.188</td>
<td>0.184</td>
<td>0.144</td>
</tr>
</tbody>
</table>

In other words, a GT2 RBFNN can be viewed as an ensemble of interval Type-2 FLSs where all the IT2 FSs computations occur for each $\alpha$-level and ambiguity is nonuniformly weighted. In this example, a GT2 RBFNN results more practical than an ensemble, especially because it is a more compact model with less parameters and less expensive in terms of computational burden. To exemplify the performance of GT2 RBFNN models, in Fig. 15, the identification result for a GT2 RBFNN with a WM method is shown.

### Fig. 15: Testing data, and the output of the GT2-RBFNN with an WM direct defuzzification and 3 fuzzy rules.

C. Example 3: Noisy Chaotic Time-Series Prediction

As the last experiment, a time-series prediction problem to evaluate the performance of the GT2-RBFNN is employed. The Mackey-Glass chaotic time series is generated from the following differential equation [41]:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1 + 10x(t-\tau)} - 0.1x(t)$$  \(70\)

For comparison reasons with previous results, the parameters $\tau = 30$, $x(0) = 1.2$. Four past values were employed to predict $x(t)$ where the input data format is used as:

$$[x(t-24), x(t-18), x(t-12), x(t-6); x(t)]$$

A number of 1000 patterns were generated from the observation $t = 124$ to $t = 1123$. For cross-validation purposes, the input data was divided into two subsets, i.e. a) 50% for training and b) 50% for testing. For cross-validation purposes, two different types of training data were created by adding Gaussian noise with a standard deviation of $\sigma = 0.2$ $\sigma = 0.3$ and with a mean of 0 to the original data $x(t)$. This type of noise has been selected because it usually occurs in real situations and it is frequently employed to verify model robustness [25-29]. For testing data, three data sets were created from the original data set. The first consists of the original 500 values. The last two testing data sets were created by adding a Gaussian noise with a $\sigma = 0.2$ and $\sigma = 0.3$. To compare the performance of the GT2-RBFNN to other existing interval type-2 fuzzy modelling methodologies, namely: a) an IT2FNN-SVR-(N), b) an IT2-FNN-SVR-(F) and an c) EKM IT2-RBFNN and d) E-RBFNN. The first two models a) and b) were introduced in [41]. The IT2-FNN-SVR is a six-layer interval type-2 fuzzy neural network with support vector machine regression that uses two different types of input nodes. For the first type, the input nodes in an IT2-FNN-SVR simply forwards each numerical data and is called IT2-FNN-SVR-(N) for short. Thus, the output of the IT2-FNN-SVR-(N) is a bounded interval which is described in terms the lower and upper limits of its Footprint Of Uncertainty (FOU). An IT2-FNN-SVR-(F) uses an input node layer that fuzzifies the input numerical data. The third IT2 methodology is an IT2-RBFNN with an EKM approach. And the last methodology is an ensemble of RBFNNs suggested in [39]. According to our experiments, it was determined a number of 3 horizontal slices for an IT2 RBFNN, and 3 hidden units with 3 fuzzy rules each for an E-RBFNN produce the highest balance between model performance and model simplicity. For statistical purposes, each experiment was repeated 10 times, and the RMSE average is used as a comparison performance index. Table VII and VIII show the training and testing results for the prediction of the Mackey-Glass time-series. From Table V, it can be viewed that in general GT2 neural structures outperform the IT2-FNN-SVR and its counterpart the IT2-RBFNN with an EKM. It is also worth noting, the superiority of the GT2-RBFNN is confirmed not only for validation purposes, but also in relation to the number of parameters.
Hence, the highest accuracy is achieved by a GT2-RBFNN with a WM and NT method respectively. In relation to Table VI, the higher the noise level of the training and testing data, the better the performance of the GT2-RBFNN models with respect to the IT2 fuzzy models. Particularly those GT2 models with an EKM and NT direct defuzzification and of Mamdani type. Finally, in Fig. 16, the testing data-fit of a random experiment using a Mamdani GT2-RBFNN with an EKM and noise level of $\sigma = 0.3$ is illustrated.

**VIII. SUMMARY AND DISCUSSION**

From the comparative analysis presented in previous section, the following summarisation and discussion is provided:

a) By using GT2 FSs, model accuracy of an RBFNN can be improved importantly. Compared to its counterparts the RBFNN and IT2 RBFNN, a higher tradeoff between accuracy and model simplicity is provided. The term *model simplicity* is used because compared to other existing fuzzy models of T1 and T2, a reduced number of fuzzy rules, and hence of parameters is required to obtain similar or better results.

b) Two problems that involve the treatment of *randomness* for nonlinear plant identification, and for the prediction of noisy chaotic time series was provided. Compared to an RBFNN of T1 or IT2, a GT2 RBFNN weights uncertainty non uniformly. This allows an RBFNN to better model the effects of uncertainty. That means, an RBFNN with GT2 FSs quantifies uncertainty as a deficiency that results from imprecise boundaries of the associated FSs, so using GT2 FSs accounts to minimise information-based imprecision.

c) From tables IV-VI, column training time is the average time of training epochs spent by each model. As can be noted, the training speed of a GT2 RBFNN with simplified structures is similar to the RBFNN, faster to the ensemble of RBFNNs and similar to the ANFIS model when it comes to modeling large size data sets.

d) As illustrated in example 2, a GT2 RBFNN not only inherits the ability of NNs to approximate complex functions, but also the ability of fuzzy logic models to provide some insights about the system being modelled.

e) By using GT2 FSs usually increases the computational complexity, however this time can be compensated by an improvement in model performance and a model simplification that can be reached by fuzzy structures based on direct-defuzzification algorithms.

f) By using GT2 FSs usually increases the computational complexity, however this time can be compensated by an improvement in model performance and a model simplification that can be reached by fuzzy structures based on direct-defuzzification algorithms.

e) Further to point f), in terms of computation, the application of a Gradient Descent approach (GD) to identify the parameters of a GT2 FLS with KM methods ("or EKM") usually results more expensive than the parameter identification for an RBFNN of T1 or IT2. This is due to the number of iterations that are needed to calculate not only the associated derivatives but also to track the permutations created during the sorting process of any KM method [46]. A GD is usually not globally convergent. Thus, a number of optimisation methods based on metaheuristics have been proposed [47]. To make this less severe, in this paper an Adaptive version of a GD approach that includes a momentum term to avoid getting trapped in a local minimum and to speed up the GD convergence is suggested.

**IX. CONCLUSIONS**

This paper presents a General Type-2 Radial Basis Function Neural Network (GT2 RBFNN) that is functionally equivalent to a GT2 FLS based on the $\alpha$--plane representation, in which the main inference engine can be viewed as a TSK or Mamdani system. A detailed description of the neural structure and its corresponding parametric optimisation of a GT2-RBFNN with an EKM, and three simplified GT2-RBFNN models that employ's three different direct-defuzzification approaches is provided. To offer a comprehensive performance analysis, experimental results about the modelling of ten data sets for multiclass classification and regression problems is provided. Two problems for nonlinear identification and for the prediction of chaotic time series in the presence of randomness and Gaussian noise are considered. Based on experimental results, the suggested model is not only able to outperform its counterparts the RBFNN of type-1 and the Interval Type-2 Radial Basis Function Neural Network (IT2 RBFNN), but also to better treat and minimise the effects of uncertainty. It can be also observed from the simulation results that compared to other methodologies, including an ensemble of RBFNNs, the number of parameters of a GT2 RBFNN is usually smaller. Further developments of the GT2 RBFNN may be related with further advances of Type-2 Fuzzy Logic methodologies, Neural Networks and learning. Particularly to reduce the computational complexity and increase model performance. A future study will be also in terms of the evaluation of the GT2 RBFFN to formulate knowledge in a transparent way to interpretation and analysis of complex systems.

**REFERENCES**

