American Journal of Intelligent Systems 2022, 12(1): 9-33
DOI: 10.5923/j.ajis.20221201.02

Hybrid Adaptive Neural-Fuzzy Algorithms Based on Adaptive Resonant Theory with Adaptive Clustering Algorithms for Classification, Prediction, Tracking and Adaptive Control Applications

Vincent A. Akpan[1,*], **Joshua B. Agbogun**[2]

[1]Department of Biomedical Technology, The Federal University of Technology, Akure, Ondo State, Nigeria
[2]Department of Computer Science and Mathematics, Godfrey Okoye University, Enugu, Enugu State, Nigeria

Abstract The development of a single compact algorithm for clustering, classification, prediction, tracking and adaptive control applications is currently of great challenge in the computational intelligence and adaptive control communities. This paper presents a new hybrid adaptive neural-fuzzy algorithm based on adaptive resonant theory with adaptive clustering algorithm (HANFA-ART with ACA) for clustering, classification, prediction, tracking and adaptive control applications. The HANFA-ART with ACA consist of nine major components, namely: (i) Mamdani fuzzy-type model; (ii) Takagi-Sugeno fuzzy-type model; (iii) ANN; (iv) FRBL; (v) adaptive resonant theory (ART); (vi) adaptive clustering algorithm (ACA) which is made up of (a) K-means clustering as the initialization algorithm, (b) adaptive mountain climbing clustering (AMCC) algorithm used in conjunction with Mamdani fuzzy-type model, and (c) adaptive Gustafson and Kessel clustering (AG-KC) algorithm used in conjunction with Takagi-Sugeno fuzzy-type model; (vii) Modified Leveberg-Marquardt algorithm (MLMA); (viii) adaptive recursive least squares (ARLS) algorithm; and (ix) several classes of membership function. The integration of these nine components results in the HANFA-ART with ACA which have been applied for the (i) classification of related diseases for the cause of meningitis using Mamdani fuzzy-type model and (ii) setpoint determination for activated-sludge waste water treatment plant (AS-WWTP) influent pump output predictions, tracking and adaptive control. Simulation results demonstrates the efficiency of the HANFA-ART with ACA when compared to standard adaptive neuro-fuzzy inference system trained with back-propagation with momentum (ANFIS with BPM) and proportional-integral-derivative (PID) control algorithms based on some evaluation criteria. The HANFA-ART with ACA can be adapted and deployed for extensive clustering, classification, prediction, tracking and adaptive control applications without explicit process model as justified in this work.

Keywords Adaptive clustering algorithms (ACA), Adaptive neuro-fuzzy inference system (ANFIS), Adaptive resonant theory (ART), Artificial neural network (ANN), Fuzzy rule-based logic (FRBL), Hybrid adaptive neural–fuzzy algorithms (HANFA), Proportional-integral-derivative (PID) controller

1. Introduction

Among the various combinations of methodologies in soft computing, the one that has highest visibility at this juncture is that of fuzzy rule-based logic (FRBL) and neurocomputing (NC), leading to neuro-fuzzy systems (NFS) based on experience [1–3]. Within FRBL, such systems play a particularly important role in the induction of rules from observations and based on experience. An effective method developed for this purpose is so-called adaptive neuro-fuzzy inference system (ANFIS) [4–8]. The nonlinear universal function approximation property of FRBL systems and artificial neural networks (ANNs) qualifies these algorithms to be powerful candidates for identification, prediction, classification and control of nonlinear dynamical systems.

As a result, these techniques have been successfully applied to solve various kinds of problems in control system design resulting in the emergence of intelligent control systems. However, each of these two intelligent techniques has its own drawbacks which limit its usefulness for certain situations and other applications [2,3]. For instance, fuzzy logic controllers suffer from some problems like the selection of appropriate membership functions, the selection of fuzzy if-then rules, and furthermore how to tune both of them to achieve the desired online performance. On the other

* Corresponding author:
vaakpan@futa.edu.ng (Vincent A. Akpan)
Received: Dec. 6, 2021; Accepted: Jan. 4, 2022; Published: Feb. 24, 2022
Published online at http://journal.sapub.org/ajis

hand, ANNs have some problems such as their black-box nature, the lack of knowledge representation power, and the selection of the proper structure and size to solve a specific problem [9–11].

In order to overcome these drawbacks, the integration of adaptive resonant theory (ART) [12,13], clustering algorithms [14–16], FRBL [15–19] and ANNs [20–22] in a unified system has received great attention in the literature which resulted in the appearance of a rapidly emerging field of neuro-fuzzy systems. One of the most widely used neuro-fuzzy systems is the ANFIS network [1,2,23–25].

The adaptive neuro-fuzzy inference system (ANFIS) is a sophisticated hybrid network that belongs to the category of neuro-fuzzy neural networks [18,25]. It is known that there is no sufficient MATLAB program for implementing neuro-fuzzy classifiers due to the complexity and accuracy of the classification problems [18,25]. Generally, ANFIS has been used as a classifier and as a function approximator. But, the usage of ANFIS for classification is unfavorable. For example, there are three classes labelled as 1, 2 and 3 [1,2,25]. The ANFIS outputs are not integer. For this reason, the ANFIS outputs are rounded to determine the class labels. But, sometimes, ANFIS can give 0 or 4 class labels. These situations are not acceptable and as a result, ANFIS is not suitable for classification problems except this problem is addressed with appropriate advanced algorithms which is one of the main focus of this paper [3,13].

To address the issue just mentioned above, a new hybrid adaptive neuro-fuzzy algorithm based on adaptive resonant theory (HANFA-ART) with adaptive clustering algorithm (ACA) is proposed in this paper. The differences between the proposed HANFA-ART with ACA lies in the architecture and the type of initialization, training and adaptation algorithms used to train the ANFIS, multiple-ANFIS (MAFIS), co-active ANFIS (CANFIS), etc [1,3,26,27]. The idea used to develop the proposed HANFA-ART with ACA closely follow from previous works [1–3,26,28]. Rather than using the so-called back-propagation algorithm or the modified versions of the back-propagation algorithm with momentum [25–31]; the proposed HANFA-ART with ACA is developed and trained with an adaptive recursive least squares (ARLS) algorithm and a modified Levenberg-Marquardt algorithm (MLMA) with an adaptive clustering schemes. Another challenging issue addressed in this paper is the integration of the ARLS and the MLMA algorithms, developed by Akpan and co-workers [31–33], into the proposed HANFA-ART with ACA architecture.

Furthermore, two types of different adaptive neuro-fuzzy clustering algorithms (classifiers) have been proposed in this study, namely: *1)*. adaptive mountain climbing clustering (AMCC) and *2)*. Adaptive Gustafson-Kessel clustering (AG-KC). As will be seen in the next Section, the *K*-means clustering algorithm has been used to initialize the clustering process for the clustering algorithms. For this reason, the number of cluster for each class must be supplied. Also, the Gaussian membership function is only used for fuzzy set descriptions, because of its simple derivative expressions. Although, the proposed classifiers follows the work described in [1–3]; the differences between these classifier schemes are about how the rules and parameters optimization are implemented and updated. The rules are adapted by the number of rule samples. Then, the optimum values of nonlinear parameters are determined by efficient optimization algorithms using the ARLS and MLMA [31–33].

Nonetheless, the MLMA uses the least squares estimation method for gradient estimation without using all training samples [31–33]. Next, linguistic hedges are applied to the fuzzy set of rules and are adapted by MLMA algorithm. By this way, some distinctive features are emphasized by higher power values, and some irrelevant features are damped with lower power values. The power effects in any feature are generally different for different classes. The use of linguistic hedges increases the recognition rates. Lastly, the powers of fuzzy sets are used for feature selection. If linguistic hedge values of classes in any feature are bigger than 0.5 and close to 1, this feature is relevant, otherwise it is irrelevant. The program creates a feature selection and a rejection criterion by using power values of features.

Subsequent to the development of the proposed HANFA-ART with ACA algorithms, a number of methods have been proposed for learning rules and for obtaining an optimal set of rules [26,34–37]. However, the four most widely used methods to update the parameters of ANFIS, MANFIS and the CANFIS structures have been reported and listed according to their computational complexities [1–3,26,27,34–37]: *1)*. Gradient decent only: all parameters are updated by the gradient descent; *2)*. Gradient decent only and one pass of least squares error (LSE): the LSE is applied only once at the very beginning to get the initial values of the consequent parameters and then the gradient decent takes over to update all parameters; *3)*. Gradient decent only and LSE: this is the hybrid learning; *4)*. Sequential LSE: using extended Kalman filter to update all parameters.

The above listed methods update antecedent parameters by using gradient descent or Kalman filtering [14,18,36,37]. These methods have high computational complexities. In this paper, a method is introduced which has less computational complexity and guaranteed stability with fast convergence properties. The HANFA-ART with ACA algorithms proposed in this work maps: *1)*. input characteristics to input membership functions; *2)*. input membership function to rules; *3)*. rules to a set of output characteristics; *4)*. output characteristics to output membership functions; *5)*. the output membership function to a single-valued output; and/or *6)*. a decision associated with the output.

It should be noted that only considered membership functions that have been fixed, and somewhat arbitrarily chosen. Also, we have only applied fuzzy inference to modeling systems whose rule structure is essentially predetermined by the interpretation of the characteristics of the variables in the model. In general, the shape of the membership functions depends on parameters that can be adjusted to change the shape of the membership function. The parameters can be automatically adjusted depending on the data of the system to be modelled. The list of the variables and their definition used in this work are listed in Table 1 while other variables are introduced and defined where used.

Table 1. List of variables and their definitions

Variables	Definitions	Variables	Definitions
i	i-th input linguistic variable	COD	Chemical oxygen demand ($g \cdot m^{-3}$)
j	j-th input-tem node from i-th input linguistic variable	BOD	Biochemical oxygen demand ($g \cdot m^{-3}$)
k	Corresponding index of j-th input-term node	FMR	Food-to-microorganisms ratio (mg·BOD/mg·MLVSS)
m	Number of system inputs	SNH	Ammonia and ammonium nitrogen ($g \cdot m^{-3}$)
n	Number of system outputs	TSS	Total soluble solids ($kg \cdot d^{-1}$)
A	First coded input normalization	IQ	Influent quality ($kg \cdot d^{-1}$)
B	Second coded input normalization	TKN	Total Kjeldahl nitrogen ($g \cdot m^{-3}$)
a	Lower input distribution interval	Ntotal	Total nitrogen ($g \cdot m^{-3}$)
b	Upper input distribution interval	MLVSS	Mixed liquor volatile suspended solids ($kg \cdot d^{-1}$)
l	l^{th} rule of Mamdani and Sugeno fuzzy systems	c	A cluster point
μ	Antecedent term	e	Approximation error variable
y	Output variable	f	Fuzzy model
y	Output vector	$\bar{f}$	Rule consequences variable
$\mathbf{y'}$	Complement coded output vector	w	T-norm operator on fuzzy-AND operation
$\bar{\mathbf{y}}$	Mean value of output vector	$\bar{w}$	Output of normalization strength
$\hat{y}$	Predicted node output variable	v_i	Cluster centers
u	Inputs variable	γ	Membership function fuzziness regulator
u	Input vector	κ	Right-flat points of membership functions
$\mathbf{u'}$	2n-dimensional complement coded input vector	ε	Fuzzy model output prediction error
$\bar{\mathbf{u}}$	Mean value of complement coded input vector	υ	Left-flat points of membership functions
C	Third coded input normalization	S_1	Set of premise (nonlinear) parameters
D	Fourth coded input normalization	S_2	Set of consequence (linear) parameters
E	Approximation error vector	$J(U,\theta)$	Cost function to minimize θ
N	Normalization strength	Π	Fuzzy-AND operation
P	Fuzzy covariance matrix	⏋, ⎾, ⎾⏋	Input fuzzification membership functions
S	Set of total parameters	u_k	End-point from cluster center
U	Set of rules	U_k	Cluster point vector
W	Regularization factor	U_c	Cluster sum of squares
Z^N	Number of input variables	$d(v_i,u_k)$	Distance measured from v_i to u_k
Z^T	Number of output variables	$U^{(j)}$	A priori control signal
M	Mountain function for AMCC	$U^{(j-1)}$	Posteriori control signal
V	Set of future vectors	θ	All adjustable parameters of the fuzzy system
Z^R	Number of fuzzy-based rules	$\in$	Optimization termination quantity

2. Formulation of the Adaptive HANFA-ART with ACA Training Algorithms

2.1. The Mamdani and Takagi-Sugeno Fuzzy Ruled-Based Logic: Systems and Models

Generally, fuzzy inference systems (FIS) can be systematically constructed from "pure" input–output data. All methods are based on the optimization of a cost function to minimize the "distance" between the predictions of the FIS and the output data. The main differences among the methods are the initialization and the adjusted parameters.

The basic FRBL scheme was original proposed by Wang [37]. In this paper, some simple modifications are proposed and these modifications are related to the consequence calculations. In the proposed modifications: the position, the shape and the distribution of the membership functions are choices for the designer rather than being fixed. Further, the rule-based logic is initialized through clustering based on experimentally acquired data and the proposed method finds only the consequences of the rules.

Supposed that a sequence of input–output $\{u^i, y^i\}$ for $i = 1, 2, \ldots, n$ data is collected, the inputs $u^i \in V \subset \Re^p$ and the output $y^i \in V \subset \Re^q$. The subset $\wp$ is a portion of the space $\Re^p$ and is defined as $\wp = [a_1, b_1] \times \cdots \times [a_p, b_p]$. The procedure to construct the model is laid out in the following [12,13,16–21]:

1). For each of the *p* inputs of the system distribute over the interval $[a_i, b_i]$ and n_i membership functions. The shape, position and distribution can be arbitrarily specified. The only condition is that the full interval must be covered and at least two membership functions should be placed on each point of the input domain. However, note that the shape and the distribution affects the smoothness and the accuracy of the approximation.

2). Generate the rules using all possible combinations among the antecedents and the AND-operator (choosing in advance "*min*" or "*product*" operator).

(i). The rule *l* of the rules for Mamdani fuzzy systems is IF u_1^i is A_1^l AND … AND u_p^i is A_p^l THEN y is $y = \bar{y}^l$

(ii). The rule l of the rules for Takagi–Sugeno fuzzy systems is IF u_1^i is A_1^l AND … AND u_p^i is A_p^l THEN $y = a_1^l u_1^i + a_2^l u_2^i + \cdots + a_p^l u_p^i + b^l$

3). Calculate the inference of each rule. For rule *l* of the form

$$\mu_l(u^i) = \min\left\{\mu_l^1(u_1^i),\ \mu_l^2(u_2^i),\ \ldots, \mu_l^p(u_p^i)\right\} \tag{1}$$

$$\text{or}\quad \mu_l(u^i) = \mu_l^1(u_1^i) \bullet \mu_l^2(u_2^i) \bullet \cdots \bullet \mu_l^p(u_p^i) \tag{2}$$

The general expressions for these fuzzy systems with *L* rules:

(i) For the Mamdani fuzzy models are given by

$$f(u^i) = \frac{\sum_{l=1}^{L} \bar{y}^l \mu_l(u^i)}{\sum_{l=1}^{L} \mu_l(u^i)} \tag{3}$$

(ii) For the Takagi–Sugeno fuzzy models are given by

$$f(u^i) = \frac{\sum_{l=1}^{L}\left(a_1^l u_1^i + a_2^l u_2^i + \cdots + a_p^l u_p^i + b^l\right)\mu_l(u^i)}{\sum_{l=1}^{L} \mu_l(u^i)} \tag{4}$$

4). Calculate the consequence parameters.

(i). In the Mamdani fuzzy model, the parameter to be calculated is $\bar{y}^l$ for $l = 1, 2, \ldots, L$ such that $f(u^i) \approx y^i$. Observe that (3) can be written as

$$f(u^i) = \sum_{l=1}^{L} \bar{y}^l \omega_l(u^i) \tag{5}$$

where
$$\omega_l(u^i) = \frac{\mu_l(u^i)}{\sum_{l=1}^{L} \mu_l(u^i)} = \omega_l^i \tag{6}$$

The *n* outputs values can be represented as the vector Y in terms of the inference process as:

$$\underbrace{\begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^n \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} \omega_1^1 & \omega_2^1 & \cdots & \omega_L^1 \\ \omega_1^2 & \omega_2^2 & \cdots & \omega_L^2 \\ \vdots & \vdots & \ddots & \vdots \\ \omega_1^n & \omega_2^n & \cdots & \omega_L^n \end{bmatrix}}_{W} \underbrace{\begin{bmatrix} \bar{y}^1 \\ \bar{y}^2 \\ \vdots \\ \bar{y}^n \end{bmatrix}}_{\theta} + \underbrace{\begin{bmatrix} e^1 \\ e^2 \\ \vdots \\ e^n \end{bmatrix}}_{E} \tag{7}$$

(ii) In the Takagi–Sugeno fuzzy model, the parameters to be calculated are $a_1^l, a_2^l, \ldots, a_p^l$ and b^l for $l = 1, 2, \ldots, L$ such that $f(u^i) \approx y^i$. Using the reasoning applied for the Mamdani fuzzy models, Equation (4) can be written as:

$$f(u^i) = \sum_{l=1}^{L}\left(a_1^l u_1^i + a_2^l u_2^i + \cdots + a_p^l u_p^i + b^l\right)\omega_l(u^i) \tag{8}$$

where $\omega_l(u^i)$ has the same form shown in Equation (6). Again, the *n* output values can be represented as the vector *Y* in terms of the inference process as

$$\underbrace{\begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^N \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} \omega_1^1 u_1^1 & \cdots & \omega_1^1 u_p^1 & \omega_1^1 & \omega_2^1 u_1^1 & \cdots & \omega_L^1 u_p^1 & \omega_L^1 \\ \omega_1^2 u_1^1 & \cdots & \omega_1^2 u_p^1 & \omega_1^2 & \omega_2^2 u_1^1 & \cdots & \omega_L^2 u_p^1 & \omega_L^2 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \omega_1^N u_1^1 & \cdots & \omega_1^N u_p^1 & \omega_1^N & \omega_1^N u_1^1 & \cdots & \omega_L^N u_p^1 & \omega_L^N \end{bmatrix}}_{W} \underbrace{\begin{bmatrix} a_1^1 \\ a_2^1 \\ \vdots \\ a_p^1 \\ b^1 \\ a_1^2 \\ \vdots \\ a_p^L \\ b^L \end{bmatrix}}_{\theta} + \underbrace{\begin{bmatrix} e^1 \\ e^2 \\ \vdots \\ e^n \end{bmatrix}}_{E} \tag{9}$$

In both cases (i.e. Equations (7) and (9)), the vector E is the approximation error and the aim is to reduce the norm of this vector as much as possible. Using the quadratic norm to measure the approximation error, we obtain

$$\min_{\theta} \|E\|_2 = \min_{\theta} \|Y - W\theta\|_2 \tag{10}$$

It is a least-squares problem and the consequences can be calculated using least squares. The basic solution to this least-squares problem is

$$\theta = \arg\min_{\theta} \|E\|_2 = \left(W^T W\right)^{-1} Y^T W \tag{11}$$

This solution is applicable as far as the *rank* $(W^T W)$ = *dim* (θ); otherwise other methods must be applied to guarantee a reliable set of consequences for the rules. In this paper, an efficient algorithm called the adaptive recursive least squares (ARLS) has been adapted for estimating and updating the consequences parameters [31–33].

2.2. The HANFA–ART with ACA Network Architecture

The proposed HANFA-ART with ACA network consists of six layers as shown in Figure 1. The proposed network architecture employs two major algorithms for parameters adjustment and learning (i.e. the ARLS and MLMA) and one algorithm for automatic structure learning (i.e. the fuzzy-ART). In the following, we briefly describe the operation of each layer. A descriptive representation of a *m*-inputs and *n*-outputs HANFA-ART with ACA network architecture with 4-fuzzy rules is shown in Figure 1. The code segment that executes the Fuzzy-ART algorithm is adopted (with the necessary modifications) from the well-structured, reliable and tractable Fuzzy-ART [13].

Unlike the common and most widely used ANFIS architectures including the one implemented in MATLAB / Simulink® [25], the version adopted in this work was first proposed in Lin and Lin [12] and later modified by Garrrett [13]. It is called the adaptive neuro-fuzzy inference system (ANFIS) based on adaptive resonant theory (ART). Instead of the conventional five layers, the proposed HANFA-ART with ACA has six layers as shown in Figure 1. For completeness, consistency and simplicity, each of these six layers is described below with their accompanying mathematical descriptions.

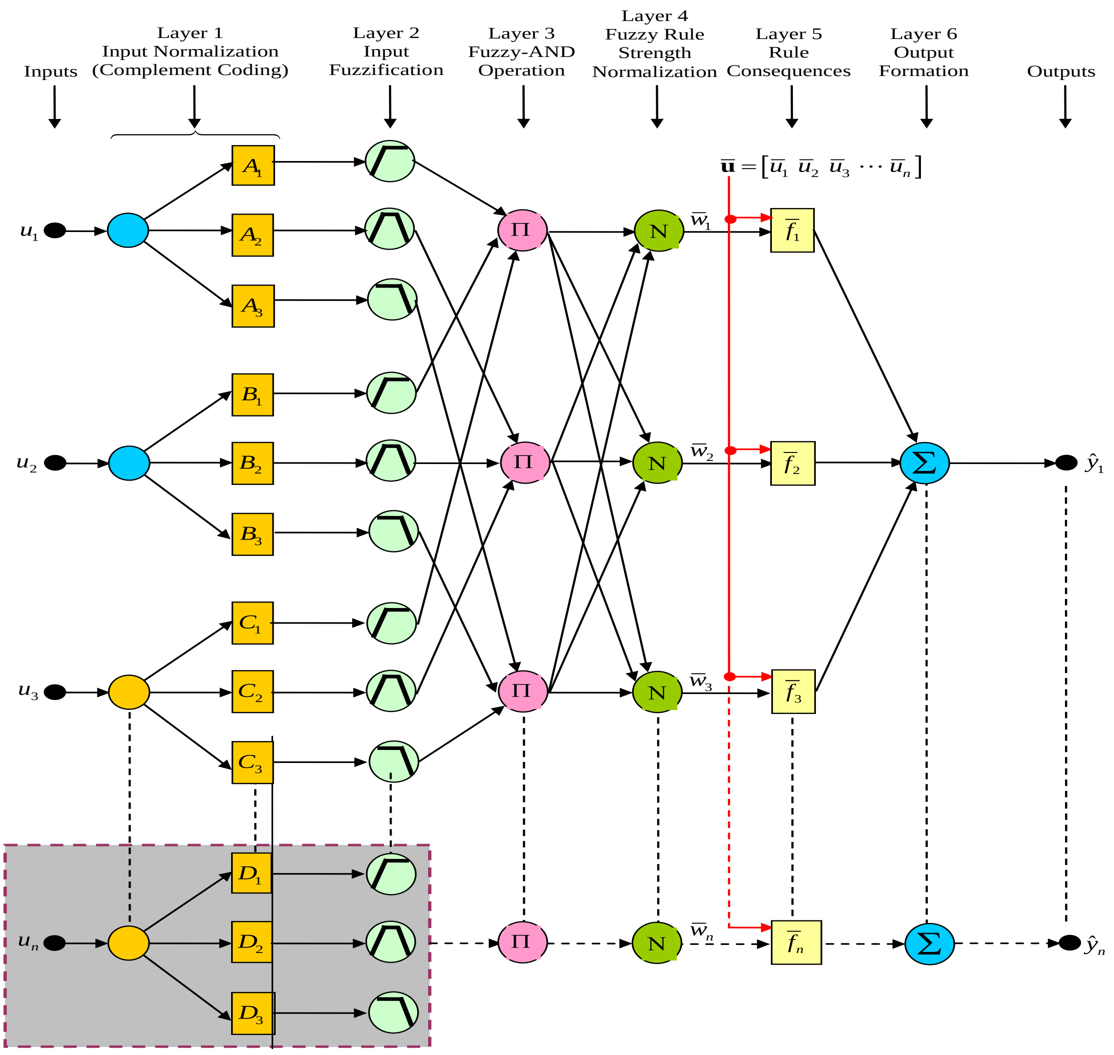

Inputs
Layer 1 Input Normalization (Complement Coding)
Layer 2 Input Fuzzification
Layer 3 Fuzzy-AND Operation
Layer 4 Fuzzy Rule Strength Normalization
Layer 5 Rule Consequences
Layer 6 Output Formation
Outputs
$\bar{\mathbf{u}} = [\bar{u}_1\ \bar{u}_2\ \bar{u}_3 \cdots \bar{u}_n]$
u_1
u_2
u_3
u_n
A_1
A_2
A_3
B_1
B_2
B_3
C_1
C_2
C_3
D_1
D_2
D_3
Π
N
$\bar{w}_1$
$\bar{w}_2$
$\bar{w}_3$
$\bar{w}_n$
f_1
f_2
f_3
f_n
Σ
$\hat{y}_1$
$\hat{y}_n$

Figure 1. The HANFA-ART with ACA network architecture

Layer No.1 - Inputs Normalization Layer (Complement Coding)

The HANFA-ART uses the technique of *complement coding* from fuzzy-ART to normalize the input training data. Complement coding is a normalization process that replaces an *n*-dimensional input vector $\mathbf{u} = [u_1, u_2, \ldots, u_n]$ with its 2*n*-dimensional complement coded form u' such that:

$$\mathbf{u}' \equiv [\bar{u}_1, 1-\bar{u}_1, \bar{u}_2, 1-\bar{u}_2, \ldots, \bar{u}_n, 1-\bar{u}_n] \quad (12)$$

where $[\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_n] = \bar{\mathbf{u}} = \mathbf{u}/\|\mathbf{u}\|$. Complement coding helps avoiding the problem of category proliferation when using fuzzy-ART for data clustering. Having this in mind, we can write the I/O function of the first layer as follows:

$$\hat{y}_i^{(1)} = \left(\overline{u_i^{(1)}}, 1-\overline{u_i^{(1)}}\right), \qquad i = 1, 2, \ldots, Z^N \quad (13)$$

where Z^N is the number of input variables.

Layer No.2 - Input Fuzzification Layer

The nodes belonging to this layer are called input-term nodes and each represents a term of an input-linguistic variable and functions as a 1-D membership function. Here we use the following trapezoidal membership function:

$$\left.\begin{array}{c} \hat{y}_{ij}^{(2)} = 1 - g\left(u_{ij}^{(2)} - \upsilon_{ij}^{(2)}, \gamma\right) - g\left(\kappa_{ij}^{(2)} - u_{ij}^{(2)}, \gamma\right), \\ i = 1, 2, \ldots, Z^N \end{array}\right\} \quad (14)$$

where Z^N is the number of input variables; and $\upsilon_{ij}^{(2)}$ and $\kappa_{ij}^{(2)}$ are respectively the left-flat and right-flat points of the trapezoidal membership function of the *j*-th input-term node of the *i*-th input linguistic variable. $u_{ij}^{(2)}$ is the input to the *j*-th input-term node from the *i*-th input linguistic variable (i.e. $u_{ij}^{(2)} = \hat{y}_i^{(1)}$). Also, the function $g(\bullet)$ is defined as:

$$g(s,\gamma) = \begin{cases} 1, & if \quad s\gamma > 1 \\ s\gamma, & if \quad 0 \le s\gamma \le 1 \\ 0, & if \quad s\gamma < 0 \end{cases} \quad (15)$$

The parameter γ regulates the fuzziness of the trapezoidal membership function. More details on how this membership function works on the real *n*-dimensional space combining *n* inputs, can be found in [18].

Layer No.3 - Fuzzy-AND Operation

Each node in this layer performs a fuzzy-AND operation. Similar to the other ANFIS networks [1–3,7–16], the *T-norm operator* of the algebraic product is selected. These results make each node's output to be the product of all its inputs:

$$\hat{y}_{k-j}^{(3)} = w_{k=j} = \prod_{i=1}^{Z^T} \hat{y}_{ij}^{(3)}, \qquad k(=j) = 1, 2, \ldots, Z^T \quad (16)$$

where Z^T is the number of output variables.

The output of each node in this layer represents the firing strength (or activation value) of the corresponding fuzzy rule. Note that the number of the fuzzy rules equals the number of input term nodes. The latter is common for all the input variables. Therefore, each fuzzy rule may be assigned an index *k* equal to the corresponding index *j* of the input term node, which is common for each input linguistic variable.

Layer No.4 - Normalization of Each Rule Firing Strength (Fuzzy Rules Strength Normalization Layer)

The output of the *k*-th node in this layer, is the firing strength of each rule divided by the total sum of the activation values of all the fuzzy rules. This results in the normalization of the activation values of all fuzzy rules:

$$\hat{y}_k^{(4)} = \overline{w_k} = \frac{\hat{y}_k^{(3)}}{\sum_{m=1}^{Z^R} \hat{y}_m^{(3)}}, \qquad k(=j) = 1, 2, \ldots, Z^R \quad (17)$$

where Z^R is the number of fuzzy-based rules.

Layer No. 5 – Rule Consequent Layer

Each node *k* in this layer is accompanied by a set of adjustable parameters $a_{1k}, a_{2k}, \ldots, a_{N_{Inputs}k}, a_{0k}$ and implements the linear function:

$$\left.\begin{array}{c} \hat{y}_k^{(5)} = \overline{w_k f_k} = \overline{w_k}\left(a_{1k}\overline{u_1^{(1)}} + a_{2k}\overline{u_2^{(1)}} + + a_{Z^N}\overline{u_{Z^N}^{(1)}} + a_{0k}\right), \\ k(=j) = 1, 2, \ldots, Z^R \end{array}\right\} (18)$$

The weight $\overline{w_k}$ is the normalized activation value of the *k*-th rule calculated with the aid of Equation (18). Those parameters are called the *consequent parameters* or *linear parameters* of the HANFA-ART system and are regulated by the proposed ARLS algorithm [31–33].

Layer No.6 - Output Layer

For the proposed HANFA-ART system, this layer consists of one and only node that creates the network's output as the algebraic sum of the node's inputs:

$$\hat{y}^{(6)} = \sum_{k=1}^{N_{Rules}} \hat{y}_k^{(5)} = \sum_{k=1}^{Z^R} \overline{w_k}\,\overline{f_k} = \frac{\sum_{k=1}^{Z^R} w_k \overline{f_k}}{\sum_{k=1}^{Z^R} w_k} \quad (19)$$

Equations (12) through (19) shows how the input vector is fed through the network layer by layer to the predicted, estimated or classified output for a particular given problem.

2.3. The Adaptive Clustering Algorithms (ACA)

Until now, the fuzzy sets of the input domains have been placed on their initial positions which is usually equally distributed from the experimental data. Two choices are available for re-positioning these input domains, namely: *1)* the number of membership functions and *2)* their initial distribution. However, the methods based on clustering aim to obtain both parameters at the same time, the number of fuzzy sets needed to make the function approximation and their distribution along the input domains.

Clustering methods are a set of techniques to reduce groups of information U represented as p-dimensional vectors into characteristic sets A_i characterized by feature vectors $v_i \in \Re^p$ and membership functions μ_A.

The methods based on clustering are considered as data-driven methods [14,16–21]. The main idea of these methods is to find structures (clusters) among the data according to their distribution in the space of the function and assimilate each cluster as a multidimensional fuzzy set representing a rule. The cluster prototypes can be either a point (to construct Mamdani models) or a hyperplane (to construct Takagi–Sugeno models) [16–21].

The fuzzy inference system is constructed by means of projecting the clusters into the input space and approximating the projected cluster with a one-dimensional fuzzy set. The advantage of these methods is that the membership functions can be automatically generated such that only the parameters of the clustering algorithms (number of clusters and distance function) can be arbitrarily chosen. According to the type of model to be constructed, the clustering method can be slightly different. In the next few sub-sections, the clustering algorithms as applied to the Mamdani and the Takagi–Sugeno fuzzy models are presented.

2.3.1. The *K*-Means Clustering Algorithms

The hard-partitioning methods are simple and popular, though its results are not always reliable and these algorithms have numerical problems as well [14,25]. From an $N \in n$ dimensional data set, K-means and K-medoid algorithms allocates each data point to one of c clusters to minimize the within-cluster sum of squares [14]:

$$U_c = \sum_{i=1}^{c} \sum_{k \in A_i} \left\| \mathbf{U}_k - \mathbf{v}_i \right\|_2 \tag{20}$$

where A_i is a set of objects (data points) in the i-th cluster and v_i is the mean for that points over cluster i. Equation (20) denotes actually a distance norm. In K-means clustering v_i is called the cluster prototypes, i.e. *the cluster centers*:

$$v_i = \frac{\sum_{k=1}^{N_i} \mathbf{u}_k}{N_i}, \qquad \mathbf{u}_k \in A_i \tag{21}$$

where N_i is the number of objects in A_i.

However, in *K*-medoid clustering the cluster centers are the nearest objects to the mean of data in one cluster $V = \{\mathbf{v}_i \in U \mid 1 \le i \le c\}$. The K-means clustering algorithm is used to initialize the data clustering process. This initial clustering by the *K*-means algorithm sets the initial partitioning of the given data set from the initial distribution.

2.3.2. Adaptive Mountain Climbing Clustering (AMCC) Algorithm

In the AMCC algorithm, a super set of the feature vectors V is proposed in advance, then some vectors are selected according to the value of the mountain function calculated for the given vector v_i. The mountain function is defined as [17,18]:

$$M(v_i) = \sum_{k=1}^{N} e^{-\left(\alpha d(v_i, u_k)\right)} \tag{22}$$

where α is a positive constant and $d(v_i, u_k)$ is a distance measure from v_i to u_k and it is typically but not necessarily:

$$d(v_i, u_k) = \left\| v_i - u_k \right\|_2 \tag{23}$$

Two of the most popular elections of the feature vector candidates are:

Table 2. The AMCC algorithm

```
Given the data set X with N vectors, select the parameter $\alpha$ and $\beta$, the
termination tolerance $\varepsilon$, and the set of feature vector candidates $V$ with
$c$ elements.
1). step 1: Calculate the initial mountain function
    $M^{(0)}(v_i) = \sum_{k=1}^{N} e^{-(\alpha d(v_i,u_k))}$   for $1 \le i \le c$      (24)
2). Repeat for $j = 1, 2, \ldots$
3). Step 2: Find the largest mountain values:
    $M^{(j-1)*} = \max_{v_i} M^{(j-1)}(v_i)$      (25)
4). Step 3: Define the location of the maximum of the mountain value
    as the center of the j-1 cluster:
    $v^{(j-1)*} = \arg\max_{v_i} M^{(J-1)}(v_i)$      (26)
5). Step 4: calculate the revised mountain function $M^{(}j)(v_i)$ :
    $M^{(j)}(v_i) = M^{(j-1)}(v_i) - M^{(j-1)*} \sum_{k=1}^{N} e^{-\left(\beta d\left(v^{(j-1)*}, u_k\right)\right)}$   for $1 \le i \le c$      (27)
6). $M^{(j-1)*} < \in$
```

1). Take the set of feature candidates equal to the set of data point $V = \mathbf{u}$. This has also been shown not to be very efficient for large data sets [1–3,12–21]; and

2). Take a grid (arbitrary) defined in the interval where the points of $\mathbf{u}$ are defined. This has also been shown not to be very efficient for vectors defined on a large-dimensional space [1–3,12–21].

Compared with other clustering methods [14–16], the mountain clustering method has as its main advantage the fact that the number of clusters does not need to be defined in advance. The implementation of the adaptive mountain clustering algorithm is summarized in Table 2.

2.3.3. Adaptive Gustafson and Kessel Clustering (AG-KC) Algorithm

The adaptive Gustafson and Kessel clustering algorithm is based on the minimization of the cost function [18]:

$$\left.\begin{array}{l}\min\limits_{(U,V,A)}\left\{J_m(U,V,A;\mathbf{u})=\right.\\ \left.\sum\limits_{k=1}^{n}\sum\limits_{i=1}^{c}(\mu_{ik})^m (u_k-v_i)^T A_i (u_k-v_i)\right\}\end{array}\right\} \tag{28}$$

where $\mathbf{u}$ is the set of vectors, $u_k \in \Re^p$ with the information, $V=[v_1,...,v_c]$ is the set of feature vectors, $A=[A_1,...,A_c]$ is a set of c norm-inducing matrices and $U \in M_{fc}$ is the fuzzy partition matrix, defined as an element of the set:

$$M_{fc}=\left\{\begin{array}{ll} U\in\Re^{c\times N}\,\big|\mu_{ik}\in[0,1], & \forall i,k;\\ \sum\limits_{i=1}^{c}\mu_{ik}=1, & \forall k;\\ 0<\sum\limits_{k=1}^{N}\mu_{ik}<N, & \forall i\end{array}\right\} \tag{29}$$

The ith row of the fuzzy partition matrix contains the membership values of the vectors u to the A_i fuzzy set. Observe that the cost function can be arbitrarily small by reducing the norm of each A_i. For this reason a constraint is introduced to preserve the norm of A_i: $|A_i|=\rho i,\ \rho>0$.

Applying the Lagrange multipliers to the above optimization problem of Equation (29) generates the following expression for A_i:

$$A_i=\left[\rho_i \det(P_i)\right]^{1/n} P_i^{-1} \tag{30}$$

where P_i is the fuzzy covariance matrix:

$$P_i=\frac{\sum_{k=1}^{n}(\mu_{ik})^m (u_k-v_i)(u_k-v_i)^T}{\sum\limits_{k=1}^{n}(\mu_{ik})^m} \tag{31}$$

The element of U are calculated as

$$\mu_{ik}=\left[\sum_{j=1}^{c}\left(\frac{(u_k-v_i)^T A_i (u_k-v_i)}{\left(u_k-v_j\right)^T A_i \left(u_k-v_j\right)}\right)^{\frac{2}{m-1}}\right]^{-1} \quad \forall i,k \tag{32}$$

The prototypes v_i are calculated as

$$v_i=\frac{\sum_{k=1}^{n}(\mu_{ik})^m u_k}{\sum_{k=1}^{n}(\mu_{ik})^m},\quad \forall i \tag{33}$$

The implementation of the adaptive Gustafson and Kessel clustering algorithm is summarized in Table 3.

Table 3. The Gustafson and Kessel clustering algorithm

Given the data set *Z* with *N* vectors, select the number of clusters *1<c<N*, the exponent m, the termination tolerance *ε>0* and the volumes ρ_i of the matrices A_i to calculate the induced norm, and initialize the matrix *U* randomly such that $U^{(0)} \in M_{fc}$.

1). repeat for $j=1,2,....$

2). Step 1: calculate the prototypes:

$$\left.\begin{array}{l} v_i^{(j)}=\dfrac{\sum_{k=1}^{n}\left(\mu_{ik}^{(j-1)}\right)^m u_k}{\sum_{k=1}^{n}(\mu_{ik})^m}\\ for\ \ 1\le i\le c\end{array}\right\} \tag{34}$$

3). Step 2: calculate fuzzy partition matrix:

$$P_i^{(j)}=\frac{\sum_{k=1}^{n}\left(\mu_{ik}^{(j-1)}\right)^m\left(u_k-v_i^{(j)}\right)\left(u_k-v_i^{(j)}\right)^T}{\sum_{k=1}^{n}\left(\mu_{ik}^{(j-1)}\right)^m} \tag{35}$$

4). step 3: Calculate the induced-norm matrices:

$$\left.\begin{array}{l} A_i=\left[\rho_i \det(P_i)\right]^{1/n} P_i^{-1}\\ for\ \ 1\le i\le c\end{array}\right\} \tag{36}$$

5). step 4: calculate the fuzzy partition matrix:

$$\left.\begin{array}{l} \mu_{ik}^{(j)}=\left[\sum\limits_{l=1}^{c}\left(\dfrac{\left(u_k-v_i^{(j)}\right)^T A_i^{(j)}\left(u_k-v_i^{(j)}\right)}{\left(u_k-v_l^{(j)}\right)^T A_i^{(j)}\left(u_k-v_l^{(j)}\right)}\right)^{\frac{2}{m-1}}\right]^{-1}\\ for\quad 1\le i\le c, 1\le k\le N\end{array}\right\} \tag{37}$$

6). Until $\left\|U^{(j)}-U^{(j-1)}\right\|<\in$

It is important to remark that when the vector u_k is equal to one of the prototypes v_i the Equation (37) becomes singular. For this case the membership value μ_{ik} for this vector is equal to one and zero for all the other entries in the kth row of U. The parameter m is a very important parameter. As $m\to\infty$, the means of the clusters tend to the mean of the set U.

2.4. The Adjustable Parameters of the HANFA-ART with FRBL System

This method requires the definition of the number of membership functions and their shape. Normally the *AND* function is fixed to be the "*product*" because an analytical expression for the gradient of the cost function is needed. The initial position of the membership functions is another element that must be chosen. The method proceeds as follows:

1). For each of the p inputs of the system, distribute over the interval $[a_i, b_i]$ and N_i membership functions; the shape, the initial positions and the distribution can be chosen arbitrarily as just discussed in Section 2.3. The membership functions must cover the input interval, and at least two membership functions should be placed on each input domain.
2). Generate the rule base using all possible combinations among the antecedents and the *AND* operator using "*product*".
3). Initialize the value of the consequences using prior knowledge, least squares or recursive least squares.
4). Optimize the value of the consequences $\bar{y}^l$ and the parameters of the membership functions. The criteria will be to minimize the cost function described in the previous section, but now the optimization will also adjust the membership functions of the antecedents. The cost function can be described as

$$J(U,\theta) = \frac{1}{2}\sum_{i=1}^{Z^T}\left(y^i - \hat{y}^i\right)^2 \tag{38}$$

where $\hat{y}^i = f\left(U^i, \theta\right)$ and θ is a vector representing all the "adjustable" parameters (consequences, parameters of the membership functions) of the fuzzy system $f(\bullet,\bullet)$. The problem will be the minimization of the cost function J. This minimization is a nonlinear and non-convex optimization problem. The objective is to obtain an "acceptable" solution and not necessarily "the global minima" of this cost function of Equation (38).

Different schemes for optimization can be applied to find this solution but the probably simplest method is the gradient descent method [31–33,38–40]. This method consists of an iterative calculation of the parameters oriented to the negative direction of the gradient [38–40]. The explanation behind this method is that by taking the negative direction of the gradient, the steepest route toward the minimum will be taken. This descent direction does not guarantee convergence of the scheme and it is for this reason that the parameter α is introduced which can be modified to improve the convergence rate and properties [31–33,38–40]. Some choices of α are given by Newton and quasi-Newton methods [9,38–40].

In this paper, two algorithms have been adopted and adapted for estimating and updating the adjustable parameters (i.e. the consequences parameters and the parameters of the membership functions) of the proposed HANFA-ART with ACA system [31–33]. These two algorithms are: *1).* an efficient recursive least squares algorithm called the adaptive recursive least squares (ARLS) algorithm, and *2).* an efficient gradient descent algorithm called the modified Levenberg-Marquardt algorithm (MLMA) [31,32].

2.5. Training of the Proposed HANFA-ART with ACA

Next, how the HANFA-ART with ACA is trained to learn the premise and consequent parameters for the membership functions with the associated rules is considered. There are numbers of possible approaches but the learning algorithm proposed by Jang and co-workers which uses a combination of steepest descent and least squares estimation (LSE) has been widely used with several challenges [38–40]. However, these challenges have been shown to be very complicated but a systematic implementation to overcome these challenges is presented in the remaining of this paper for the proposed HANFA-ART with ACA.

It can be shown that for the network illustrated in Figure 1, if the premise parameters are fixed, the output is linear in the consequent parameters. Then, the total parameters set can be split into three sets: S = set of total parameters, S_1 = set of premise (nonlinear) parameters, and S_2 = set of consequent (linear) parameters. Thus, the HANFA-ART with ACA uses a two-pass learning algorithms:

1). Forward Pass: Here S_1 is unmodified and S_2 is computed using an adaptive recursive least square (ARLS) algorithm described in [31–33]; and
2). Backward Pass: Here S_2 is unmodified and S_1 is computed using a special gradient descent algorithm called the MLMA algorithm described in [31,32].

So, the proposed hybrid learning algorithm uses a combination of steepest descent algorithm (i.e. the proposed MLMA) and a recursive least squares algorithm (i.e. the proposed ARLS) to adapt the parameters in the adaptive network. The summary of the learning process is given below:

(1). The Forward Pass: This propagates the input vector through the network layer by layer as follows

i). Present the input vector;
ii). Calculate the node outputs layer by layer;
iii). Repeat for all data $\rightarrow$ A and y formed;
iv). Identify parameters in S_2 using the ARLS algorithm; and
v). Compute the error measure for each training pair.

(2). Backward Pass: In this case, the error is sent back through the network in a manner similar to the back-propagation but it is implemented here by the MLMA algorithm

i). Use the MLMA algorithm to update parameters in S_1; and
ii). For given fixed values of S_1, the parameters in S_2 found by this approach are guaranteed to be the

global optimum [31–33].

2.6. Summary of Algorithms for Mamdani and Takagi–Sugeno Fuzzy Models

The summary of the clustering algorithms incorporating the ARLS and the MLMA algorithms for calculating the consequences parameters that are not covered by the cluster using ARLS and then applying the MLMA to adjust the antecedents parameters appropriately as given in the following sub-sections.

2.6.1. Algorithm for Mamdani Fuzzy Models

Step 1: Collect the data and construct a set of vectors $Z^N = \{u^i, y^i\}$ where u and y are the inputs and the output of the function respectively. Observe that it is assumed here that $u^i \in \Re^n$ and $y^i \in \Re$;

Step 2: Search for clusters using the combination of the *K*-means and the AMCC algorithms for problems where the dimension of the input space is small;

Step 3: Project the membership functions from the partition matrix *U* into the input space;

Step 4: Approximate the appropriate projected membership function using convex membership functions such as triangular, Gaussian, polynomial, trapezoidal, etc. as described in Appendix A, B and C [18];

Step 5: Construct the rules with the appropriate projected membership functions;

Step 6: Calculate the consequences using recursive least squares using the ARLS algorithm; and

Step 7: Adjust the parameters of the antecedents using gradient descent-based MLMA.

2.6.2. Algorithm for Takagi–Sugeno Models

Step 1: Collect the data and construct a set of vectors $Z^N = \{u^i, y^i\}$ where u and y are the inputs and the output of the function respectively. Observe that it is assumed here that $u^i \in \Re^n$ and $y^i \in \Re$;

Step 2: Search for clusters using the combination of the *K*-means and the AG-KC algorithms;

Step 3: Check for similarities among the clusters. Do two clusters describe a similar hyperplane?;

Step 4: Project the membership functions from the partition matrix *U* into the input space;

Step 5: Approximate the appropriate projected membership function using convex membership functions such as triangular, Gaussian, polynomial, trapezoidal, etc. as described in Appendix A, B and C [18];

Step 6: Construct the rules with the projected membership functions;

Step 7: Generate the consequences using the covariance matrices of each cluster;

Step 8: Calculate the consequences that are not covered by the clusters using recursive least squares using the ARLS algorithm; and

Step 9: Adjust the parameters of the antecedents (if needed) using gradient descent-based MLMA.

3. Applications of HANFA-ART with ACA for Classification, Prediction and Control

3.1. Classification of Meningitis Using HANFA-ART with ACA Algorithm with Mamdani Fuzzy-Type Model

3.1.1. Problem Description

Meningitis is an acute inflammation of the protective membranes covering the brain and spinal cords, known collectively as meninges [41]. The inflammation may be caused by infection with bacterial, viral, fungal, parasitic, amebic, tuberculosis, non-infectious and other microorganisms (such as fungi, protozoa, etc) and less commonly by drugs [42]. The most common symptoms of meningitis are headache and neck stiffness associated with fever, confusion or altered consciousness, vomiting and inability to tolerate light (photphobia) or loudness (photophobia). Children usually show non-specific symptoms such as drowsiness and irritability [43]. Meningitis is a form of meningococcal disease which is very serious [41,42]. About 10 to 15% of people with meningococcal disease die even with appropriate antibiotic treatment. Of course those who recover have problems with their nervous system; up to 20% suffer from some serious after-effects, such as permanent hearing loss, limb loss, brain damage or suffer seizure or strokes [41–43].

Meningitis can be life threatening because of the inflammation proximity to the brain and spinal cord; therefore, the condition is classified as medical emergency [41–43]. This leads to the requirement of developing a new technique which can detect and classify the occurrence of meningitis giving some symptoms. This will help in better diagnosis in order to reduce the number of meningitis patients.

3.1.2. Formulation of the Meningitis Classification Problem

The proposed HANFA-ART with ACA algorithms is implemented here for the classification of the 18 attributes that could cause meningitis classified into six (6) classes namely: abscess which corresponds to "*Class 1*", bacteria which corresponds to "*Class 2*", bacteria E which corresponds to "*Class 3*", virus which corresponds to "*Class 4*", virus E which corresponds to "*Class 5*" and tuberculosis which corresponds to "*Class 6*". The 18 attributes include: *1)* cold, *2)* headache, *3)* fever, *4)* nausea, *5)* loc, *6)* seizure, *7)* BT, *8)* stiff, *9)* GCS, *10)* WBC, *11)* CRP, *12)* ESR, *13)* CSF cell, *14)* cell poly, *15)* cell mono, *16)* CSF pro, *17)* CSF pro, and *18)* age. A total of 140 data were collected for the 18 attributes just mentioned and their respective plots are shown in Figure 2 (a) – (f); while their six respective classes for the 18 attributes over the 140 experimental data set is shown in

Figure 3. The two algorithms are: *1)* adaptive neuro-fuzzy inference system trained with back-propagation (ANFIS with BPM) and *2)* the proposed HANFA-ART with ACA.

Three cases are randomly considered to demonstrate the efficiency of the HANFA-ART with ACA presented in this paper. The first case is on the classification of bacterial disease, the second case is concerned with the classification of a virus related disease while the third case demonstrates the classification of a tuberculosis related disease all of which could be major symptoms in addition to other factors that could result in the probability of one to have meningitis.

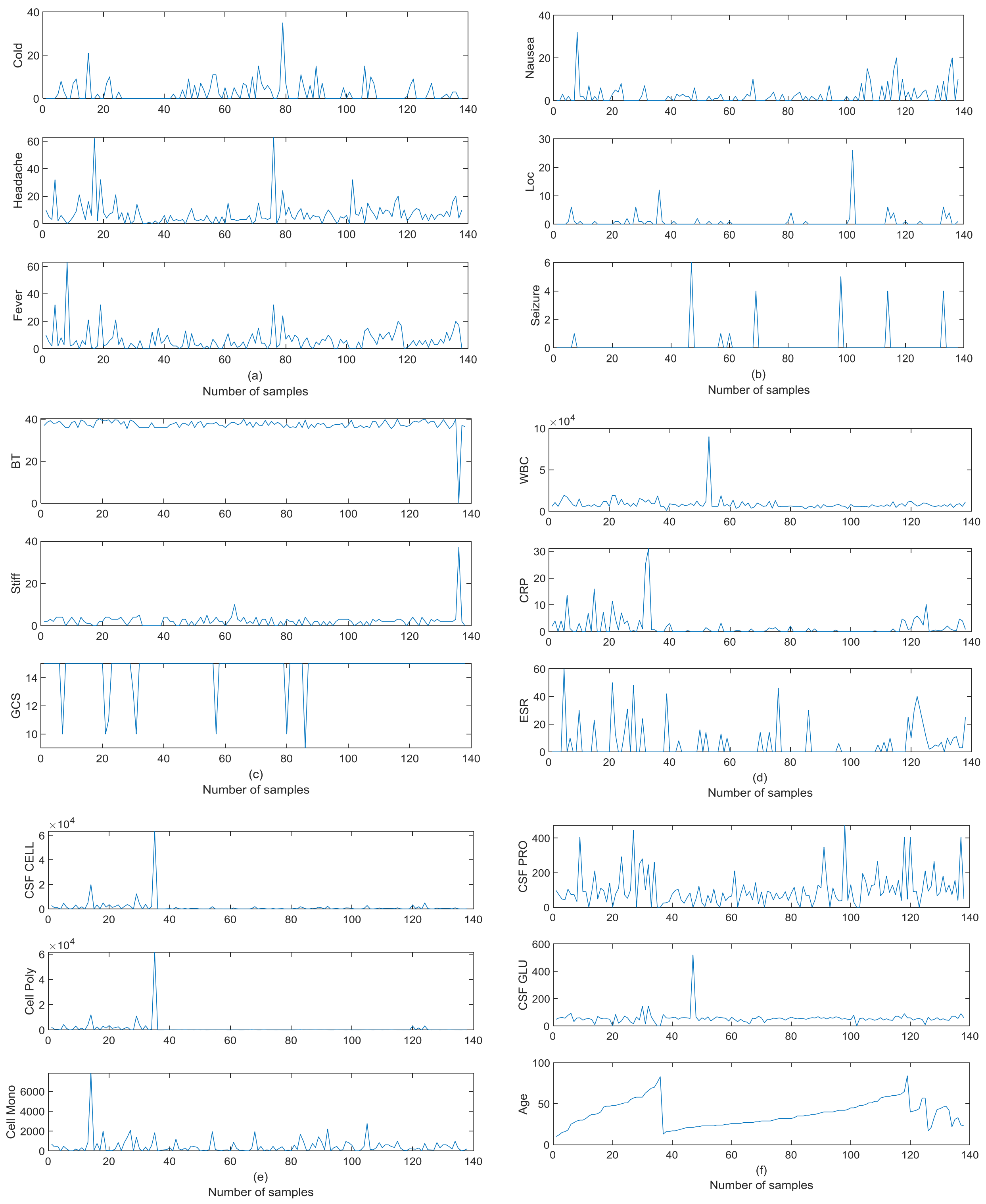


Figure 2. The 18 attributes: (a) cold, headache and fever, (b) nausea, loc and seizure, (c) BT, stiff and GCS, (d) WBC, CRP and ESR, (e) CSF cell, cell poly and cell mono, and (f) CSF pro, CSF pro, and age

3.1.3. Implementation of the Meningitis Classification Problem

Case 1: Bacterial Related Disease for the Cause of Meningitis

The implementation of these two algorithms (ANFIS with BPM and HANFA-ART with ACA) presented in this paper for a known cause of illness is carried out. The input data set is characterized by bacteria and the objective here is to evaluate the ability of the algorithms to classify the data set into this category and compared the obtained classification with a standard adaptive neuro-fuzzy inference system (ANFIS) trained with backpropagation with momentum (BPM) algorithm which is collectively called here as ANFIS with BPM [2,20,44].

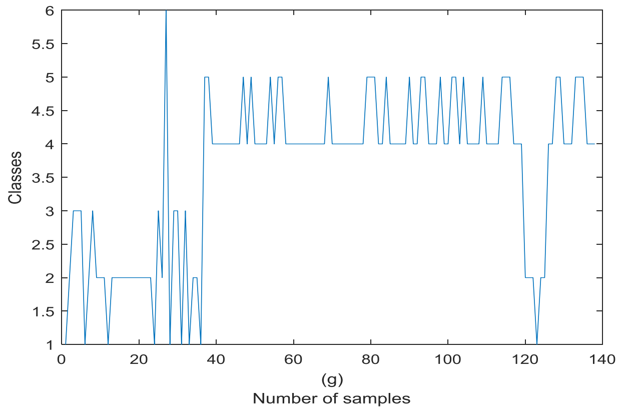
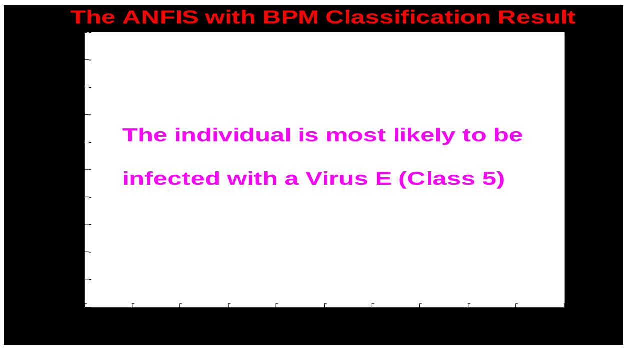
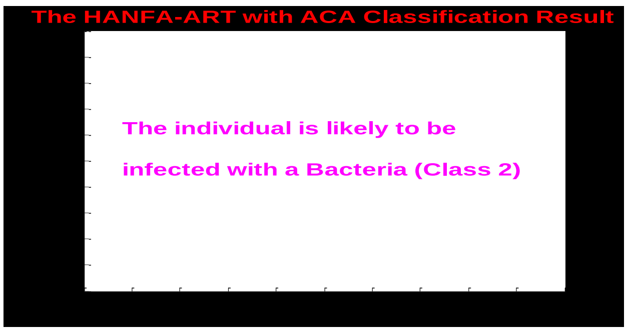


Figure 3. The six respective classes for the 18 attributes over the 140 experimental data set


The ANFIS with BPM Classification Result

The individual is most likely to be infected with a Virus E (Class 5)

(a)


The HANFA-ART with ACA Classification Result

The individual is likely to be infected with a Bacteria (Class 2)

(b)

Figure 4. Results for the classification of a bacterial disease based on (a) ANFIS with BPM and (b) HANFA-ART with ACA

Upon the implementation of the two algorithms, the result of Table 4 and the comments of Figure 4 (a) and (b) were obtained. The results of Figure 4 show the classification of a bacterial disease based on (a) ANFIS with BPM and (b) HANFA-ART with ACA. As it can be seen from Table 4 based on the results obtained using HANFA-ART with ACA, the major symptom is a bacterial related disease (Class 2) with 21.4286% followed by 20.0584% of a bacteria E related disease (Class 3) and a NAN% of virus related disease. Note although the 0% for virus may mean nothing but it is a value which indicate the absence of virus which when compared to NAN (Not A Number) which is assumed to have no effect nor contribute to the individual's illness.

Table 4. Class classification and class performance accuracies for an unknown cause of a disease suspected to be a bacterial related disease

Cases	Classes	Related Disease	ANFIS with BPM	HANFA-ART with ACA
Case 1 (Bacterial disease)	Class 1	Abscess	NAN %	NAN %
	Class 2	Bacteria	NAN%	21.4286%
	Class 3	Bacteria E	0%	20.0584 %
	Class 4	Virus	3.1394 %	NAN%
	Class 5	Virus E	8.3243%	0%
	Class 6	Tuberculosis	0%	NAN%

From the performance of the two classification algorithms shown in Figure 4, it can be seen that HANFA-ART with ACA gives the correct prediction and classification of 21.4286% and 20.0584% for the bacteria related disease when compared to the 3.1394% and 8.3243% obtained by ANFIS with BPM algorithm which suspects a viral infection E which is completely different from the true bacterial infection case.

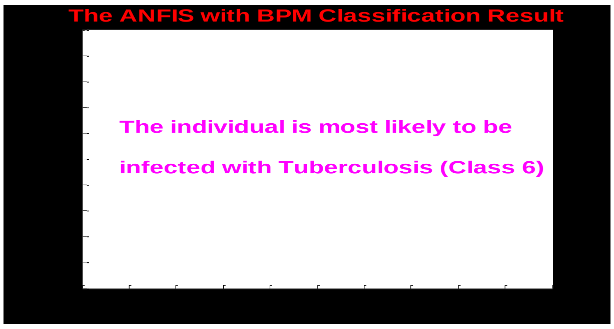
Case 2: Viral Related Disease for the Cause of Meningitis

The implementation of the two algorithms (ANFIS with BPM and HANFA-ART with ACA) presented in this work for a known viral cause of illness is carried out. The input data set is characterized by a virus and the objective here is to evaluate the ability of the algorithms to classify the data set into this category. Upon the implementation of the two algorithms, the result of Table 5 and the comments of Figure 5(a) and (b) were obtained. The results of Figure 5 show the classification results of a viral related disease based on (a) ANFIS with BPM and (b) HANFA-ART with ACA. As it can be seen from Table 5 based on the results obtained using HANFA-ART with ACA, the major symptom is a viral related disease (Class 4) with 15.6351% followed by another virus E (Class 5) with 9.4325. However, the ANFIS with BPM shows a tuberculosis (Class 6) and bacteria (Class 2) related diseases with 8.9462% and 5.7845% respectively for a known viral related disease and 0% for Class 4 which is the actual cause of the meningitis.

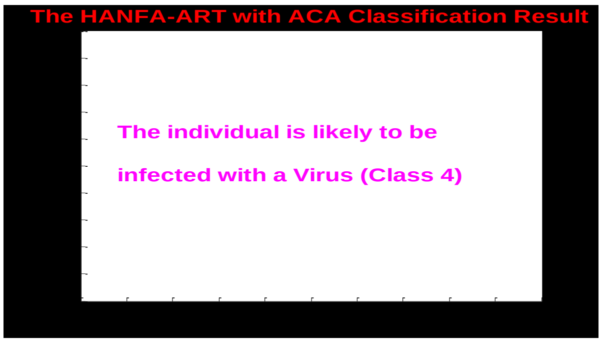
The performance of the two classification algorithms are shown in Figure 5. As it can be seen in Figure 5, the proposed HANFA-ART with ACA again gives the correct prediction and classification of 15.6351% and 9.4325 each for the viral related disease when compared to the results obtained from the ANFIS-BPM algorithm with 8.9462% and 5.7845% suspicion of tuberculosis and bacterial related problems respectively, which may or may not be present since it has a higher probability when compared to NAN.

Table 5. Class classification and class performance accuracies for an unknown cause of a disease suspected to be a viral related disease

Cases	Classes	Related Disease	ANFIS with BPM	HANFA-ART with ACA
Case 2 (viral disease)	Class 1	Abscess	NAN %	NAN %
	Class 2	Bacteria	5.7845%	0%
	Class 3	Bacteria E	NAN %	NAN %
	Class 4	Virus	0%	15.6351%
	Class 5	Virus E	NAN%	9.4325%
	Class 6	Tuberculosis	8.9462%	0%


The ANFIS with BPM Classification Result
The individual is most likely to be
infected with Tuberculosis (Class 6)

(a)


The HANFA-ART with ACA Classification Result
The individual is likely to be
infected with a Virus (Class 4)

(b)

Figure 5. Results for the classification of a viral disease based on (a) ANFIS with BPM and (b) HANFA-ART with ACA

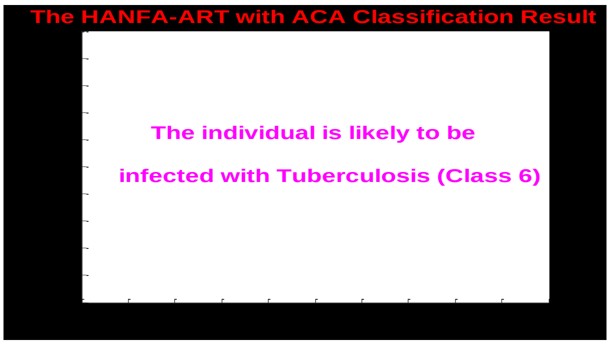
Case 3: Tuberculosis Related Infection for the Cause of Meningitis

The implementation of the two algorithms (ANFIS with BPM and HANFA-ART with BPM) for an unknown cause of illness is carried out. The input data set is characterized by a virus and the objective here is to evaluate the ability of the algorithms to classify the data set into this category. Upon the implementation of the three algorithms, the result of Table 6 and the comments of Figure 6(a) and (b) were obtained. The results of Figure 6 show the classification of a tuberculosis related disease based on (a) ANFIS with BPM and (b) HANFA-ART with ACA. As it can be seen from Table 6 based on the results obtained using HANFA-ART with ACA, the major symptom is a tuberculosis related infection (Class 6) with 24.3129% and 33.3333% as reported by ANFIS with BPM and HANFA-ART with ACA respectively. In addition to the high expectation probability of tuberculosis, the 2.6537% and 20% viral threat indicated by ANFIS with BPM and HANFA-ART with ACA respectively shows a high probability of the occurrence and presence of a viral related infection (Class 4). It should also be observed that HANFA-ART with ACA giving a 9.2645% also indicates an inevitable presence of a bacterial (Class 2) related infection.

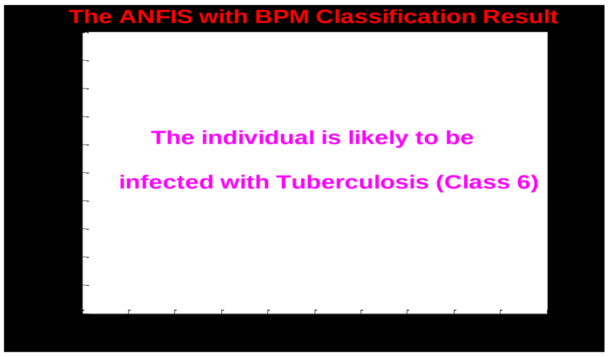
The performance of the two classification algorithms for the tuberculosis related infections are shown in Figure 6. As it can be seen in Figure 6, the ANFIS with BPM algorithm and the proposed HANFA-ART with ACA all give competitive correct and acceptable prediction and classification results. Finally, these three attributes (symptoms) could results in meningitis and must be addressed once observed.

Table 6. Class classification and class performance accuracies for an unknown cause of a disease suspected to be a tuberculosis

Cases	Classes	Related Disease	ANFIS with BPM	HANFA-ART with ACA
Case 3 (Tuberculosis)	Class 1	Abscess	NAN %	NAN %
	Class 2	Bacteria	0%	9.2645%
	Class 3	Bacteria E	NAN%	NAN %
	Class 4	Virus	2.6537%	20%
	Class 5	Virus E	NAN%	NAN%
	Class 6	Tuberculosis	24.3129%	33.3333%


The HANFA-ART with ACA Classification Result
The individual is likely to be
infected with Tuberculosis (Class 6)

(a)


The ANFIS with BPM Classification Result
The individual is likely to be
infected with Tuberculosis (Class 6)

(b)

Figure 6. Results for the classification of a tuberculosis related disease based on (a) ANFIS with BPM and (b) HANFA-ART with ACA

3.1.4. Summary of the Results for the Meningitis Classification Problem

The main task of this sub-section has been on how to correctly classify a disease (infection) or diseases (infections) in terms of 18 different attributes (symptoms) which could lead to meningitis. The 18 different attributes were classified into six (6) different classes of diseases (infections). The total data representing the 18 attributes is 2,520 (i.e. 140 x 18) which in addition to the 140 classes (i.e. 140 x 1) gives a grand total of 2660 data used in this study.

To this end, an HANFA-ART with ACA algorithm has been proposed in this paper. In order to evaluate the performance of the proposed algorithm, a second algorithm called ANFIS with BPM algorithm has also been presented. The performance of the proposed algorithm for a classification problem have shown acceptable results and are in agreement with the expected results except for the ANFIS with BPM which seldom gives good result. This is not surprising since networks trained with back-propagation related algorithm usually give poor performance since back-propagation algorithm on its own has several problems such as poor robustness, poor convergence property, long training cycles, large network parameters, etc.

The classification performance results obtained by the HANFA-ART with ACA demonstrate the classification efficiency and capability of the proposed algorithms. In order to place a benchmark, 500 iterations (epochs) was used throughout all the simulations and the number of input-to-hidden neurons were limited to 15 neurons. The simulation results show high prediction and classification accuracies when compared to the characteristics of the input data. The two algorithms proposed, implemented and demonstrated in this work can be adapted in hospitals and medical centers for immediate diagnosis based on the reports from patients.

3.2. Setpoint Determination for AS-WWTP Influent Pump Control Using a Takagi-Sugeno Fuzzy-Type Model

3.2.1. Description of the AS-WWTP System

The activated sludge model is a generic name for a group of mathematical methods to model activated sludge waste water treatment plant (AS-WWTP). The research in this area is coordinated by a task group of the International Water Association (IWA) [31,45]. The Activated sludge models are used in scientific research to study biological processes in hypothetical systems. They can also be applied on full scale wastewater treatment plants for optimization, when carefully calibrated with reference data for sludge production and nutrients in the effluent [31,45–48]. Modelling the activated sludge process has an important role in implementing efficient control actions for better process performance. However, the reliability of the proposed models depends on an increasing number of kinetic and stoichiometric parameters, which to a large extent depend on the characteristics of the actual wastewater and must therefore be experimentally determined.

There is little doubt that control strategies can be evaluated by model simulation. However, the protocol used in the evaluation is critical. To make unbiased comparisons, the influent pump control strategy must be evaluated under the simulation benchmark condition. Also, the effect of the control strategy must be compared to a fully defined and suitable reference output. The *'simulation benchmark'* defines such a protocol and provides a suitable reference outputs for operational performance assessment [31,45–48].

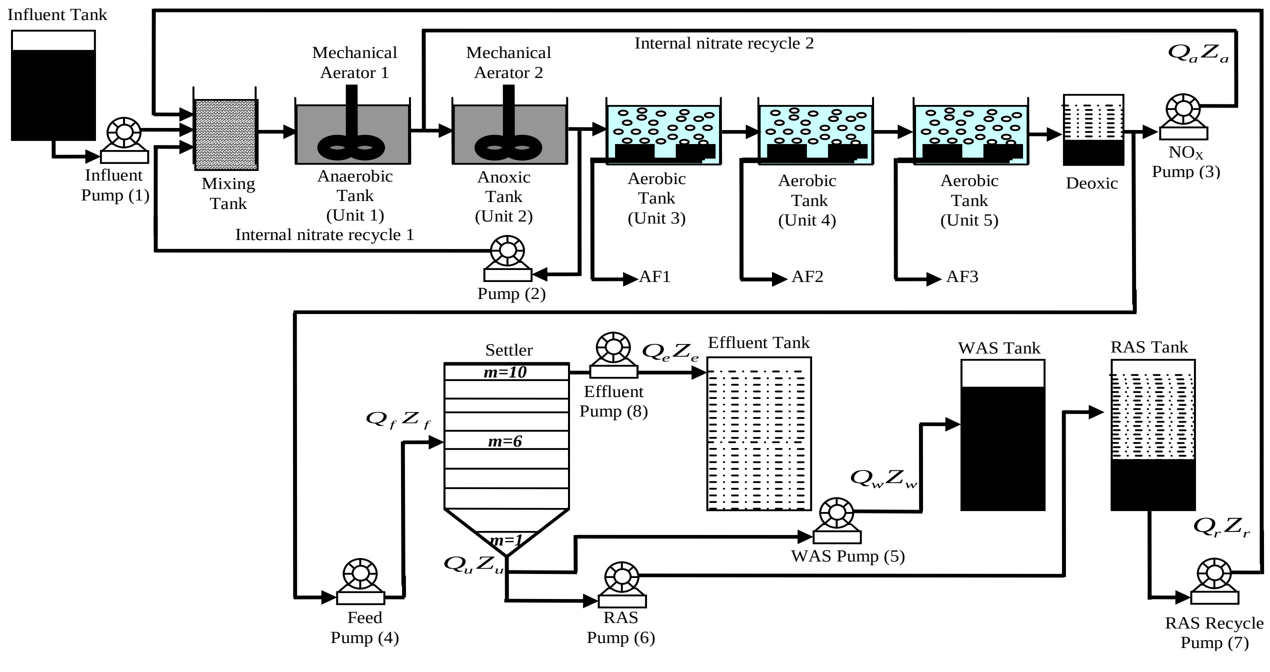


Figure 7. The schematic of the AS-WWTP process [31,46–48]

The *simulation benchmark* plant design comprises five reactors in series with a 10-layer secondary clarifier or settler as shown in the schematic representation of the layout of Figure 7 while the Simulink implementation of the benchmark model for experimental data generation with constant influent is shown in Figure 8. The detailed description and explanation of the AS-WWTP simulation benchmark and the Simulink model can be found in [31,45–48].

3.2.2. Formulation of the AS-WWTP Influent Pump Control Problem

Here an attempt is made to use the HANFA-ART with ACA for the prediction and adaptive control of the influent pump (pump 1) of an AS-WWTP based on the setpoint generated by the HANFA-ART with ACA based the rules shown in Table 7. The constraints for the influent pump operation is between 0 and 1.20 x 10^5 revolution per day (rev/day) while the influent flow into the first biological reactor at a typical rate of 2.0 x 10^4 mg$\cdot l^{-1}$. [31,45–48] The various units for the eight parameters in Table 7 are given in Table 1. The minimum and maximum values are split into five levels Negative Big (NB), Negative Small (NB), Medium (M), Positive Small (PS), and Positive Big (NB) which indicates the appropriate setpoint for the influent pump control. Note that each of the 8 input parameters has 5 levels which give 8 x 5 x 5 = 200 possibilities of fuzzy rules that are used to manipulate the influent pump at each sampling time. A brief manipulation of the influent pump based on these parameters is highlighted in the following:

1). When all the values are met, then pump one is widely open i.e. Positive Big (PB);

2). When all other conditions are met but one of the parameter has a lesser value than the required value given by COST 624, then pump1 is open wide i.e. PB provided that FMR is exact;

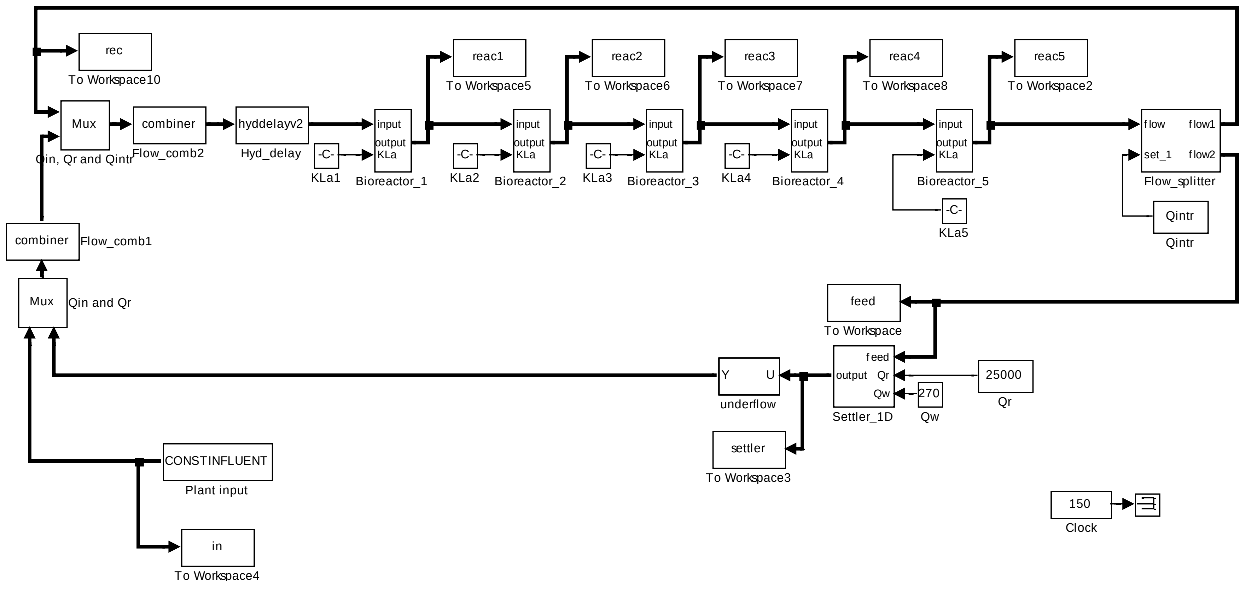
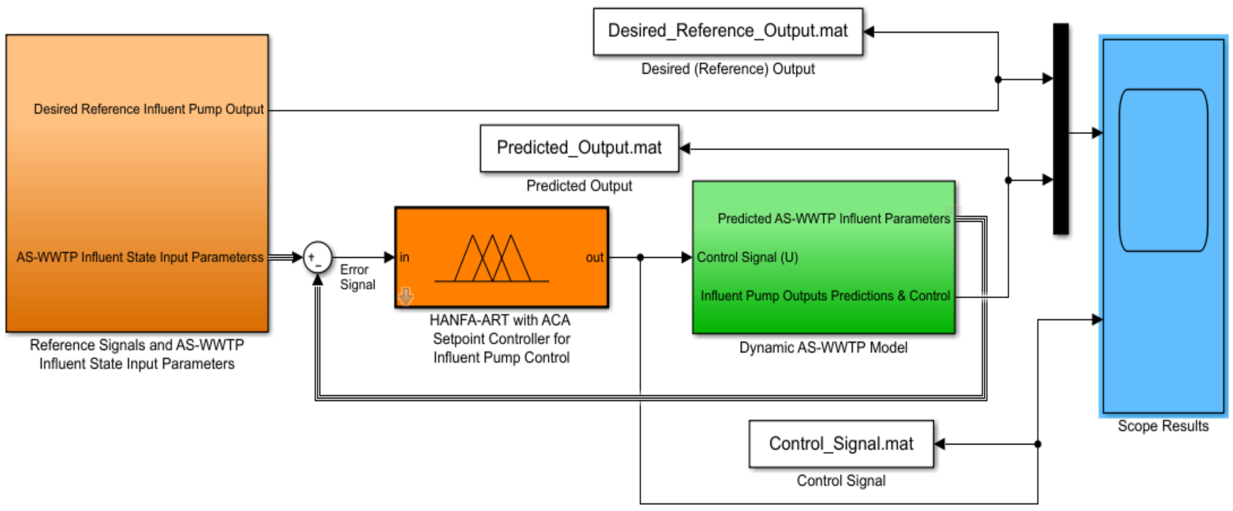


Figure 8. Open-loop steady-state benchmark simulation model for the AS-WWTP with constant influent [31,46–48]



Desired_Reference_Output.mat
Desired (Reference) Output
Desired Reference Influent Pump Output
Predicted_Output.mat
Predicted Output
AS-WWTP Influent State Input Parameterss
Reference Signals and AS-WWTP Influent State Input Parameters
Error Signal
in
out
HANFA-ART with ACA Setpoint Controller for Influent Pump Control
Predicted AS-WWTP Influent Parameters
Control Signal (U)
Influent Pump Outputs Predictions & Control
Dynamic AS-WWTP Model
Scope Results
Control_Signal.mat
Control Signal

Figure 9. The Simulink implementation of the HANFA-ART with ACA for setpoint determination for the control of AS-WWTP influent pump

Table 7. Minimum and maximum values of based on different combination of the influent input parameters for influent pump setpoint determination

S/N	Influent Input Parameters			Incoming Influent Parameter Levels				
				NB	**NS**	**M**	**PS**	**PB**
	Parameters	**Min**	**Max**	**0**	**25**	**50**	**75**	**100**
1.	BOD	1.40	2.60	1.41	1.66	2.00	2.29	2.60
2.	COD	33.74	62.66	33.90	40.50	48.20	55.10	62.7
3.	FMR	0.28	0.52	0.28	0.33	0.40	0.46	0.52
4.	SNH	22.05	40.95	22.20	26.70	31.50	36.30	41.00
5.	TSS	147.91	274.69	149.00	171.00	198.00	222.00	248.00
6.	IQ	$2.94\text{x}10^4$	$5.46\text{x}10^4$	$2.96\text{x}10^4$	$3.49\text{x}10^4$	$4.20\text{x}10^4$	$4.84\text{x}10^4$	$5.46\text{x}10^4$
7.	TKN	7.0	13.0	9.01	9.3	9.65	9.98	10.30
8.	Ntotal	$1.31\text{x}10^4$	$2.43\text{x}10^4$	$1.32\text{x}10^4$	$1.56\text{x}10^4$	$1.87\text{x}10^4$	$2.12\text{x}10^4$	$2.43\text{x}10^4$
9.	MLVSS	791	1,469	796.00	940.00	$1.13\text{x}10^4$	$1.30\text{x}10^4$	$1.47\text{x}10^4$

3). When at most three of the parameters has a lesser value while others meet the demand of the COST 624 for pump1, pump1 is partially open, i.e. Positive Small (PS) provided that FMR is exact. Another condition is that when all conditions are met but any one of the parameters having value above the demand, the pump is again allowed to be partially opened i.e. Positive Small (PS) provided that FMR, TSS and MLVSS are exact;

4). When more than three of the parameters do not meet the COST 624 demand, pump1 is partially closed i.e. Negative Small (NS); and

5). When more than one of any of the parameters have values higher than necessary as given by the COST 624, pump1 should be almost closed i.e. Negative Big (NB).

3.2.3. Implementation of the HANFA-ART with ACA for Setpoint Determination for AS-WWTP Influent Pump Control

The Simulink implementation of the HANFA-ART with ACA for the setpoint determination for the prediction and adaptive control of AS-WWTP influent pump is shown in Figure 9. As can be seen in Figure 9, the AS-WWTP constant influent state parameters are fed through a summer with the predicted AS-WWTP influent parameters in a negative feedback mode at the first sampling instant. The influent parameters from the summer forms the inputs to the HANFA-ART with ACA Takagi-Sugeno type fuzzy model which employs the rules defined by Table 7 to determine the appropriate setpoint for the influent pump output prediction, tracking and control against a desired reference influent pump output as shown in Figure 9.

In order to verify the efficiency of the HANFA-ART with ACA, a standard proportional-integral-derivative (PID) controller is also implemented for the AS-WWTP influent pump control. A major problem with PID controllers is the “wind up” of the integrator resulting in the saturation of the integral term for control signal of large magnitude which has been handled based on the techniques described in [49,50]. The complete mathematical description of the PID controller that computes the control signal is [49,50]:

$$\left.\begin{aligned} U(k) = K_P E(k) + K_I \frac{T}{2}\sum_{k=1}^{N}\left[E(k-1)+E(k)\right] \\ + K_D \frac{\left[E(k)-E(k-1)\right]}{K} \end{aligned}\right\} \tag{39}$$

where K_P, K_I and K_D are the proportional, integral and derivative gains respectively, K is the sampling time and $E(k) = R(k) - \hat{Y}(k)$ is the error between the desired reference $R(k)$ and predicted output $\hat{Y}(k)$, and N is the number of samples. The minimum and maximum constraints imposed on the PID controller to penalize changes on the AS-WWTP pump control inputs $U(k)$ and outputs $Y(k)$ are given as:

$$\left.\begin{aligned} U_{\min} \le U(k) \le U_{\max} \\ Y_{\min} \le Y(k) \le Y_{\max} \end{aligned}\right\} \tag{40}$$

Note that the minimum and maximum constraints used by the PID controller are generated by the HANFA-Art with ACA algorithm. The simulation results of the setpoint determination, output predictions and adaptive control of the influent pump is discussed in the next sub-section.

3.2.4. Simulation and Discussion of Results Setpoint Determination for the AS-WWTP Influent Tank Control Problem

The simulation results of the output predictions, tracking and adaptive control of the AS-WWTP influent pump based on accurate setpoint determination is shown in Figure 10 where (a) shows the comparison of the HANFA-ART with ACA and the PID controller for output predictions, tracking and adaptive control of the influent pump against the desired reference output of the influent pump while Figure 10(b) shows the output prediction errors between the predicted output by the PID controller and the HANFA-ART with ACA against the desired reference outputs; and Figure 10(c) compares the control signal generated by the PID controller and HANFA-ART with ACA for the adaptive control of the influent pump in order to tack the desired output. The control signal is the energy required to manipulate the influent pump to achieve excellent performance to tack with the desired reference output based on the setpoint.

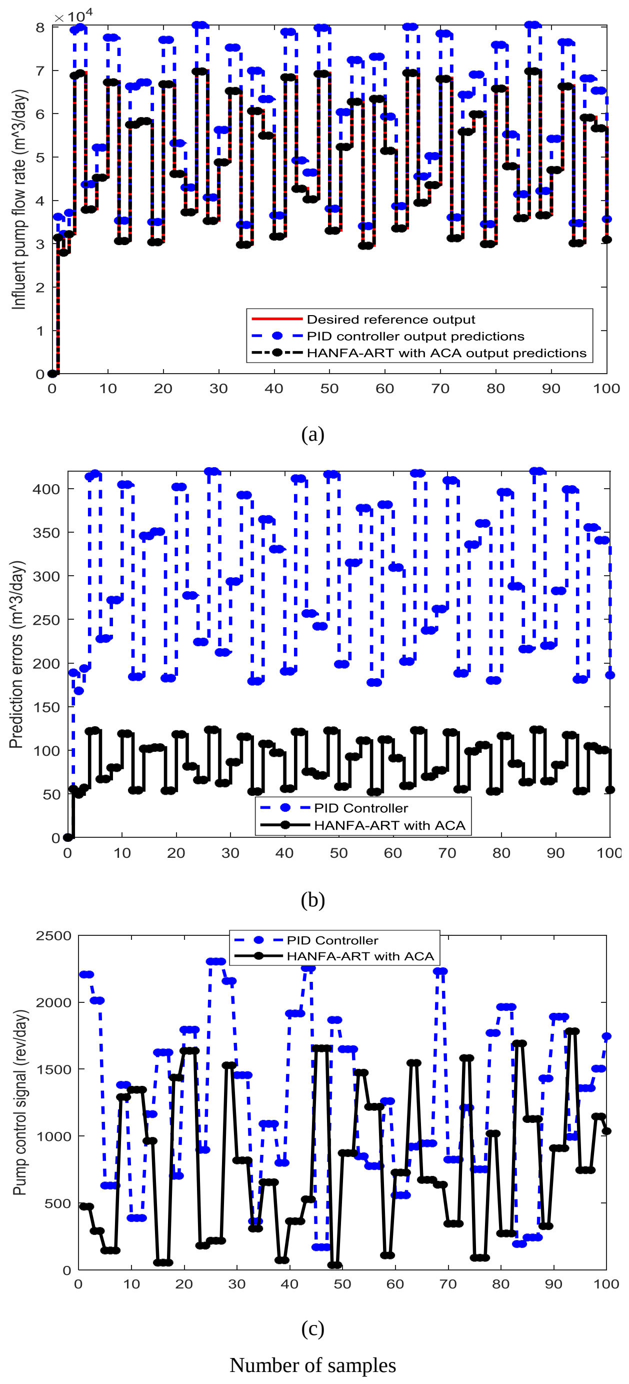


Figure 10. Output predictions, tracking and adaptive control of the AS-WWTP influent pump based on accurate setpoint determination: (a) comparison of the adaptive control and tracking of the desired reference output by the PID controller and HANFA-ART with ACA, (b) PID controller and the HANFA with ACA output prediction errors, and (c) control signal generated by PID controller and the HANFA-ART with ACA for the adaptive control of the influent pump

It is evident in the simulation results of Figure 10(a)-(c) that the HANFA-ART with ACA algorithm exhibit excellent setpoint determination which enhances the good output prediction, tracking and adaptive control of the influent point when compared the results obtained by the PID controller. The superior performance of the HANFA-ART with ACA over the PID controller is further justified by the maximum output prediction errors of 123.51 rev/day when compared to the maximum output prediction errors of 419.92 obtained by the PID controller. The minimum and maximum control effort by HANFA-ART with ACA varies between 36.36 and 1.64 x 10^3 rev/day respectively compared to the excessive minimum and maximum control effort of between 151.85 and 2.33 x 10^3 rev/day respectively control energy required by the PID controller.

4. Conclusions

The architecture and learning procedures for the HANFA-ART with ACA presented above is an adaptive fuzzy inference system implemented in the framework of adaptive neural networks and adaptive resonant theory as well as adaptive clustering algorithm. By using the proposed hybrid learning procedures, the proposed HANFA-ART with ACA has been able to construct an input-output mapping based on both human knowledge (in the form of fuzzy if-then rules) and adaptive input-output models.

The efficiency of the proposed HANFA-ART with ACA has been validated with two problems, namely: *(i)* classification of meningitis related diseases using HANFA-ART with ACA algorithm using the Mamdani-type fuzzy-type model, and *(ii)* setpoint determination for the output prediction, tracking and adaptive control of an AS-WWTP influent pump using Takagi-Sugeno fuzzy-type model.

The HANFA-ART with ACA algorithm clearly outperforms the standard ANFIS with BPM for the meningitis problem based on the excellent and accurate classification of the related disease that could be responsible for the meningitis.

Unlike conventional model reference adaptive control (MRAC) where plant parameters are adjusted to obtain the plant model for the controller design amid tight constraints or in model predictive control (MPC) where an explicit model of the plant is required for the controller design amid tight constraints [9,51]. However, the technique proposed and implemented in this work using the HANFA-ART with ACA is completely different. The idea here is to represent the plant behaviour as in Table 7 and the HANFA-ART with ACA is combined with the Takagi-Sugeno fuzzy-type in closed-loop for excellent output predictions, tracking of desired reference trajectory and adaptive control performance of the AS-WWTP influent pump as can be seen evidently in Figure 10(a)-(c) justified with small output prediction errors and minimum control effort when compared the well-celebrated PID controller.

It is worthwhile to note that clustering of numerical data forms the basis of many system modeling and identification, prediction and classification algorithm. The purpose of clustering such as the subtractive clustering used in this work is to identify natural groupings of data set to produce a concise representation of system's behaviour or trend within the data set.

Thus data clustering presents computationally intensive task. It is recommended that parallel computing be used for the implementation of the algorithms proposed in this work for real-time applications.

The optimization algorithms used in this work are all classical optimization algorithms which are gradient descent algorithms. The implication is that they are based on sequential gradient-based searching technique and a global optimum may not be easily obtained together with huge computational overhead for large multivariable data set or systems. Hence, the use of evolutionary optimization techniques such as any variation of genetic algorithms could alleviate this problem.

Other broad areas of applications of the proposed classification, prediction, tracking and adaptive control algorithms presented in this work are inexhaustible and should be investigated.

Appendix A: Gradient Updates for Membership Functions

A.1. Gradient Update for Gaussian Membership Functions

The parameterization of the membership functions is given by

$$\mu_j^i\left(x_i^t\right) = \exp\left(-\left(\frac{x_i^t - \bar{x}_j^i}{\sigma_j^i}\right)^2\right) \tag{A-1}$$

The updating formula for the parameters $\bar{x}_j^i$ *and* σ_j^i will be given by

$$\left.\begin{array}{l} \bar{x}_j^i(k+1) = \bar{x}_j^i(k) + \\ \alpha \displaystyle\sum_{t=1}^{N} \frac{\left(y^t - f\left(x^t\right)\right)}{B} \sum_{l\varepsilon u} 2\left(y^{-1} - f\left(x^t\right)\right)\mu_l\left(x^t\right)\frac{x_i^t - \bar{x}_j^i}{\sigma_j^{i\,2}} \end{array}\right\} \tag{A-2}$$

$$\left.\begin{array}{l} \sigma_j^i(k+1) = \sigma_j^i(k) + \\ \alpha \displaystyle\sum_{t=1}^{N} \frac{\left(y^t - f\left(x^t\right)\right)}{B} \sum_{l\varepsilon u} 2\left(y^{-1} - f\left(x^t\right)\right)\mu_l\left(x^t\right)\frac{\left(x_i^t - \bar{x}_j^i\right)^2}{\sigma_j^{i\,3}} \end{array}\right\} \tag{A-3}$$

where U is the set of rules with the antecedent term $\mu_j^i(\bullet)$.

A.2. Gradient update for Trapezoidal Membership Functions

Assuming the parameterization given in the expression

$$\mu_j^i\left(x_i, a_j^i, b_j^i, c_j^i d_j^i\right) = \min\left[\begin{array}{l} \max\left(\dfrac{x_i - a_j^i}{b_j^i - a_j^i}, 0\right), \\ \max\left(1 - \dfrac{x_i - c_j^i}{d_j^i - c_j^i}, 0\right), 1 \end{array}\right] \tag{A-4}$$

The updating formulas will be

$$\left.\begin{array}{l} a_j^i(k+1) = a_j^i(k) + \\ \alpha \displaystyle\sum_{t=1}^{N} \frac{\left(y^t - f\left(x^t\right)\right)}{B} \sum_{l\varepsilon u}\left(y^{-1} - f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_j^i\left(x_i^t\right)}\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial a_j^i} \end{array}\right\} \tag{A-5}$$

$$\left.\begin{array}{l} b_j^i(k+1) = b_j^i(k) + \\ \alpha \displaystyle\sum_{t=1}^{N} \frac{\left(y^t - f\left(x^t\right)\right)}{B} \sum_{l\varepsilon u}\left(y^{-1} - f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_j^i\left(x_i^t\right)}\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial b_j^i} \end{array}\right\} \tag{A-6}$$

$$\left.\begin{array}{l} c_j^i(k+1) = c_j^i(k) + \\ \alpha \displaystyle\sum_{t=1}^{N} \frac{\left(y^t - f\left(x^t\right)\right)}{B} \sum_{l\varepsilon u}\left(y^{-1} - f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_j^i\left(x_i^t\right)}\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial c_j^i} \end{array}\right\} \tag{A-7}$$

$$\left.\begin{array}{l} d_j^i(k+1) = d_j^i(k) + \\ \alpha \displaystyle\sum_{t=1}^{N} \frac{\left(y^t - f\left(x^t\right)\right)}{B} \sum_{l\varepsilon u}\left(y^{-1} - f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_j^i\left(x_i^t\right)}\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial d_j^i} \end{array}\right\} \tag{A-8}$$

where the set $\Re$ is the set of rules that includes the function $\mu_j^i(x)$ in the antecedents and

$$\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial a_j^i} = \left\{\begin{array}{ll} 0 \quad , & if\ x_i^t < a_j^i \\ \dfrac{x_i^t - b_j^i}{\left(b_j^i - a_j^i\right)^2}, & if\ a_j^i < x_i^t < b_j^i \\ 0 \quad , & if\ x_i^t < b_j^i \end{array}\right\} \tag{A-9}$$

$$\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial b_j^i} = \left\{\begin{array}{ll} 0 \quad , & if\ x_i^t < a_j^i \\ \dfrac{a_i^t - x_i^t}{\left(b_j^i - a_j^i\right)^2}, & if\ a_j^i < x_i^t < b_j^i \\ 0 \quad , & if\ x_i^t < b_j^i \end{array}\right\} \tag{A-10}$$

$$\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial c_j^i} = \left\{\begin{array}{ll} 0 \quad , & if\ x_i^t < c_j^i \\ \dfrac{a_j^i - x_i^t}{\left(b_j^i - a_j^i\right)^2}, & if\ c_j^i < x_i^t < d_j^i \\ 0 \quad , & if\ x_i^t < d_j^i \end{array}\right\} \tag{A-11}$$

$$\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial d_j^i}=\left\{\begin{array}{cc} 0\quad , & if\ x_i^t<c_j^i \\ \dfrac{x_i^t-c_j^i}{\left(d_j^i-c_j^i\right)^2}, & if\ c_j^i<x_i^t<d_j^i \\ 0\quad , & if\ x_i^t<d_j^i \end{array}\right\} \quad \text{(A-12)}$$

It is important to remark that the updating should preserve the condition $a_j^i \le b_j^i \le c_j^i \le d_j^i$. This adaptation rule can be applied to triangular membership functions by just making $b_j^i = c_j^i$.

Appendix B: Gradient Update for Triangular Membership Functions with 0.5 Overlap

The membership functions are parameterized by using only their modal values. This parameterization not only preserves the overlap but also reduces the number of parameters to be turned. Triangular membership functions are parameterized by the position of their three vertices; but the condition of overlap 0.5 makes the lower right vertex of one membership functions to be at the same position as the modal value of the next membership function. So, instead of turning three parameters (the vertices), only one parameter is turned for each membership function.

The parameterization for a triangular membership function using the modal values as parameters is

$$\left.\begin{array}{l}\mu_j^i\left(x_i,m_{j-1}^i,m_j^i,m_{j+1}^i\right)= \\ \qquad \max\left[0,\min\left(\dfrac{x_i-m_{j-1}^i}{m_j^i-m_{j-1}^i},1-\dfrac{x_i-m_j^i}{m_{j+1}^i-m_j^i}\right)\right]\end{array}\right\} \quad \text{(B-1)}$$

The updating formula will be

$$\left.\begin{array}{l} m_j^i(k+1)=m_j^i(k)+ \\ \alpha\displaystyle\sum_{t=1}^{N}\frac{\left(y^t-f\left(x^t\right)\right)}{B}\left[\begin{array}{l}\displaystyle\sum_{l\varepsilon u}\left(y^{-1}-f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_{j-1}^i\left(x_i^t\right)}\frac{\partial\mu_{j-1}^i\left(x_i^t\right)}{\partial m_j^i} \\ +\displaystyle\sum_{l\in v}\left(y^{-1}-f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_j^i\left(x_i^t\right)}\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial m_j^i} \\ +\displaystyle\sum_{l\in w}\left(y^{-1}-f\left(x^t\right)\right)\frac{\mu_l\left(x^t\right)}{\mu_{j+1}^i\left(x_i^t\right)}\frac{\partial\mu_{j+1}^i\left(x_i^t\right)}{\partial m_j^i}\end{array}\right]\end{array}\right\} \quad \text{(B-2)}$$

where the set *U*, *V* and *W* are the set of rules that includes the functions $\mu_{j-1}^i(.), \mu_j^i(.)$ and $\mu_{j+1}^i(.)$, respectively, and with

$$\frac{\partial\mu_{j-1}^i\left(x_i^t\right)}{\partial m_j^i}=\left\{\begin{array}{cc} 0\quad , & if\ x_i^t<m_{j-1}^i \\ \dfrac{x_i^t-m_j^i}{\left(m_j^i-m_{j-1}^i\right)^2}, & if\ m_{j-1}^i<x_i^t<m_j^i \\ 0\quad , & if\ x_i^t<m_j^i \end{array}\right\} \quad \text{(B-3)}$$

$$\frac{\partial\mu_j^i\left(x_i^t\right)}{\partial m_j^i}=\left\{\begin{array}{cc} 0\quad , & f\ x_i^t<m_{j-1}^i \\ \dfrac{m_{j-1}^i-x_i^t}{\left(m_j^i-m_{j-1}^i\right)^2}, & if\ m_{j-1}^i<x_i^t<m_j^i \\ \dfrac{m_{j+1}-x_i^t}{\left(m_{j+1}^i-m_j^i\right)^2}, & if\ m_j^i<x_i^t<m_{j+1}^i \\ 0\quad , & f\ x_i^t>m_{j+1}^i \end{array}\right\} \quad \text{(B-4)}$$

$$\frac{\partial\mu_{j+1}^i\left(x_i^t\right)}{\partial m_j^i}=\left[\begin{array}{cc} 0\quad , & if\ x_i^t<m_j^i \\ \dfrac{x_i^t-m_{j+1}^i}{\left(m_{j+1}^i-m_j^i\right)^2}\quad , & if\ m_j^i<x_i^t<m_{j+1}^i \\ 0\quad , & if\ x_i^t>m_{j+1}^i \end{array}\right] \quad \text{(B-5)}$$

Here the adaptation must be constrained such that the condition $m_j^i \le m_{j+1}^i$ is preserved.

Appendix C: Gradient Expressions for Identification of Fuzzy Models with Membership Functions

C.1. Gradient Expressions for Identification of Fuzzy Models

The gradients derived in this appendix are useful in the optimization of dynamic models. The cost function to be minimized is the quadratic cost function, which is defined as

$$V_N(\theta)=\frac{1}{2N}\sum_{t=1}^{N}\left|y(t)-\hat{y}(t|\theta)\right|^2 \quad \text{(C-1)}$$

where $y(t)$ is the output of the "real" system at time *t*,

$$\hat{y}(t|\theta)=f\left(\varphi(t),\theta\right) \quad \text{(C-2)}$$

$\hat{y}(t|\theta)$ is the output of the constructed model parameterized by the vector θ. The vector θ describes the membership functions and the position of the singletons in the consequences and

$$\left.\begin{array}{l}\varphi(t)=\left[y(t-1),\ldots,y(t-m),\hat{y}(t-1),\ldots,\hat{y}(t-n),\ldots,\right. \\ \qquad \left.u(t),\ldots,u(t-k),\in(t-1),\ldots,\in(t-l)\right]\end{array}\right\} \quad \text{(C-3)}$$

is the set of regressors of the model. The derivations shown in this appendix can be applied to the structures NFIR, NARX, NOE, NARMAX and NBJ. This appendix initially shows the derivation of the gradient for the consequences of the rules; in the second part it shows the derivation for the parameters of different types of membership functions.

C.2. Gradient for the Singleton Consequences

The expression for the gradient is given by

$$\frac{\partial V_N(\theta)}{\partial \bar{y}^1} = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|\theta)(-\frac{\partial \hat{y}(t|\theta)}{\partial \bar{y}^1}) \qquad \text{(C-4)}$$

With $$\frac{\partial \hat{y}(t|\theta)}{\partial \bar{y}^1} = \hat{y}\frac{\partial w_l(\varphi(t))}{\partial \bar{y}^1} + w_l(\varphi(t)) \qquad \text{(C-5)}$$

Observe that if the model is NARX or NFIR the term $\partial w_l(\varphi(t)) / \partial \bar{y}^1 = 0$ and the expression for the gradient will be the same used for static function approximation. The term $\partial w_l(\varphi(t)) / \partial \bar{y}^1$ is dependent from previous gradient values and must be generated dynamically.

$$\frac{\partial w_l(\varphi(t))}{\partial \bar{y}^1} = \frac{\left\{\begin{matrix} \dfrac{\partial \mu_l(\varphi(t))}{\partial \bar{y}^1}\sum_{i=1}^{L}\mu_l(\varphi(t)) \\ -\mu_l(\varphi(t))\dfrac{\partial \sum_{i=1}^{L}\mu_i(\varphi(t))}{\partial \bar{y}^1} \end{matrix}\right\}}{\sum_{i=1}^{L}\mu_l(\varphi(t))} \qquad \text{(C-6)}$$

where

$$\left.\begin{aligned} \frac{\partial \mu_l(\varphi(t))}{\partial \bar{y}^1} &= \sum_{j\in\varepsilon}\frac{\mu_l(\varphi(t))}{\mu_l^i(\hat{y}(t-k(i)))}\frac{\partial \mu_l^i(\hat{y}(t-k(i)))}{\partial \bar{y}^1} \\ &+ \sum_{j\in\varepsilon}\frac{\mu_l(\varphi(t))}{\mu_l^j(\varepsilon(t-k(i)))}\frac{\partial \mu_l^j(\varepsilon(t-k(j)))}{\partial \bar{y}^1} \end{aligned}\right\} \qquad \text{(C-7)}$$

where y represents the set of inputs related with the regressors $\hat{y}(\bullet)$ with delay $k(i)$ and ε represents the set of inputs related with the regressors $\varepsilon(\bullet)$ with delay $m(j)$, and

$$\frac{\sum_{I=1}^{L}\mu_I(\varphi(t))}{\partial \bar{y}^1} = \sum_{l=1}^{l}\left\{\begin{matrix} \sum_{i\in U}\dfrac{\mu_l(\varphi(t))}{\mu_l^i(\hat{y}(t-k(i)))}\dfrac{\partial \mu_l^i(\hat{y}(t-k(i)))}{\partial \bar{y}^1} \\ +\sum_{j\in\varepsilon}\dfrac{\mu_l(\varphi(t))}{\mu_l^j(\varepsilon(t-m(i)))}\dfrac{\partial \mu_l^j(\varepsilon(t-m(j)))}{\partial \bar{y}^1} \end{matrix}\right\} \qquad \text{(C-8)}$$

Finally, according to the type of membership functions used in the terms $\partial \mu_l^j(\varepsilon(t-m(j))) / \partial \bar{y}^1$ and $\partial \mu_l^j(\bar{y}(t-k(i))) / \partial \bar{y}^1$ will have the following expressions:

C.2.1. Gradient for the Singleton Consequences with Gaussian Membership Functions

For Gaussian membership functions using the parameterization presented in (A-1):

$$\left.\begin{aligned} &\frac{\partial \mu_l^i(\hat{y}(t-k(i)))}{\partial \bar{y}^1} = \\ &-2\frac{(\hat{y}(t-k(i)) - \bar{x}_l^i)}{\sigma_l^{i2}}\mu_l^i(\hat{y}(t-k(i)))\frac{\partial \hat{y}(t-k(i))}{\partial \bar{y}^1} \end{aligned}\right\} \text{(C-9)}$$

$$\left.\begin{aligned} &\frac{\partial \mu_l^j(\in(t-m(j)))}{\partial \bar{y}^1} = \\ &2\frac{(\in(t-m(j)) - \bar{x}_l^j)}{\sigma_l^{i2}}\mu_l^j(\in(t-m(j)))\frac{\partial \hat{y}(t-m(j))}{\partial \bar{y}^1} \end{aligned}\right\} \text{(C-10)}$$

C.2.2. Gradient for the Singleton Consequences with Trapezoidal Membership Functions

For trapezoidal membership functions using the parameterization given in Equation (A-4)

$$\left.\begin{aligned} &\frac{\partial \mu_l^i(\hat{y}(t-k(i)))}{\partial \bar{y}^1} = \\ &\begin{cases} 0 \quad , & if\ \hat{y}(t-k(i)) < a_l^i \\ \dfrac{1}{b_l^i - a_l^i}\dfrac{\partial \hat{y}(t-k(i))}{\partial \bar{y}^1}, & if\ a_l^i < \hat{y}(t-k(i)) < b_l^i \\ 0 \quad , & if\ b_l^i < \hat{y}(t-k(i)) < c_l^i \\ \dfrac{-1}{d_l^i - c_l^i}\dfrac{\partial \hat{y}(t-k(i))}{\partial \bar{y}^1}, & if\ c_l^i < \hat{y}(t-k(i)) < d_l^i \\ 0 \quad , & if\ \hat{y}(t-k(i)) > d_l^i \end{cases} \end{aligned}\right\} \text{(C-11)}$$

$$\left.\begin{aligned} &\frac{\partial \mu_l^j(\in(t-(j)))}{\partial \bar{y}^1} = \\ &\begin{cases} 0 \quad , & if \in(t-m(j)) < a_l^j \\ \dfrac{-1}{b_l^j - a_l^j}\dfrac{\partial \hat{y}(t-m(j))}{\partial \bar{y}^1}, & if\ a_l^j < \in(t-m(j)) < b_l^j \\ 0 \quad , & if\ b_l^j < \in(t-m(j)) < c_l^j < c_l^i \\ \dfrac{1}{d_l^j - c_l^j}\dfrac{\partial \hat{y}(t-m(j))}{\partial \bar{y}^1}, & if\ c_l^j < \in(t-m(j)) < d_l^j \\ 0 \quad , & if \in(t-m(j)) > d_l^j \end{cases} \end{aligned}\right\} \text{(C-12)}$$

C.3. Gradient for the parameters of the Membership Functions

The expression for the gradient is given by

$$\frac{\partial V_N(\theta)}{\partial \bar{y}^1} = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|\theta)(-\frac{\partial \hat{y}(t|\theta)}{\partial \alpha}) \qquad \text{(C-13)}$$

where $\alpha \subset \theta$ represents all the adjustable parameter in the model excluding the consequences, and

$$\left.\begin{aligned}\frac{\partial \hat{y}(t|\theta)}{\partial \alpha_n} &= \frac{1}{\sum_{l=1}^{L}\mu_l(\varphi(t))}\sum_{l\in u}\left(\bar{y}^1-\hat{y}(t|\theta)\right)\\ &\times\frac{\mu_l(\varphi(t))}{\mu_j^i(\hat{y}(t-k(l)))}\frac{\partial \mu_j^i(\hat{y}(t-k(l)))}{\partial \alpha_n}\end{aligned}\right\} \quad \text{(C-14)}$$

where α_n is a parameter of the membership function $\mu_j^i(.)$ and U is the set of rules that include in the antecedents the membership function $\mu_j^i(.)$. According to the type of membership functions and the regressors the term $\partial\mu_j^i(\hat{y}(t-k(l)))/\partial\alpha_n$ will have expressions, which are presented in the following lines.

C.3.1. Gradient for the parameters of the Membership Functions with Gaussian Membership Functions

With Gaussian membership functions α_n could be $\bar{x}_j^i, \sigma_j^i$ and the gradients will be:

$$\left.\begin{aligned}\frac{\partial \mu_l^i(\hat{y}(t-k(i)))}{\partial \bar{x}_j^i} &= -2\frac{\left(\hat{y}(t-k(l))-\bar{x}_j^i\right)}{\sigma_j^{i2}}\\ &\times\mu_l^i(\hat{y}(t-k(l)))\left[\frac{\partial \hat{y}(t-k(l))}{\partial \bar{x}_j^i}-1\right]\end{aligned}\right\} \quad \text{(C-15)}$$

$$\left.\begin{aligned}\frac{\partial \mu_l^i(\varepsilon(t-k(i)))}{\partial \bar{x}_j^i} &= -2\frac{\left(\varepsilon(t-k(l))-\bar{x}_j^i\right)}{\sigma_j^{i2}}\\ &\times\mu_l^i(\varepsilon(t-m(l)))\left[\frac{\partial \hat{y}(t-m(l))}{\partial \bar{x}_j^i}-1\right]\end{aligned}\right\} \quad \text{(C-16)}$$

$$\left.\begin{aligned}&\frac{\partial \mu_l^i(\hat{y}(t-k(i)))}{\partial \sigma_j^i} =\\ &\quad -2\frac{\left(\hat{y}(t-k(l))-\bar{x}_j^i\right)}{\sigma_j^{i3}}\mu_l^i(\hat{y}(t-k(l)))\\ &\quad \times\left[\sigma_j^i\frac{\partial \hat{y}(t-k(l))}{\partial \bar{x}_j^i}-\left(\hat{y}(t-k(l))-\bar{x}_j^i\right)\right]\end{aligned}\right\} \text{(C-17)}$$

$$\left.\begin{aligned}&\frac{\partial \mu_l^i(\in(t-m(i)))}{\partial \sigma_j^i} =\\ &\quad -2\frac{\left(\in(t-m(l))-\bar{x}_j^i\right)}{\sigma_j^{i3}}\mu_l^i(\in(t-m(l)))\\ &\quad \times\left[-\sigma_j^i\frac{\partial \hat{y}(t-m(l))}{\partial \sigma_j^i}-\left(\in(t-k(l))-\bar{x}_j^i\right)\right]\end{aligned}\right\} \text{(C-18)}$$

C.3.2. Gradient for the parameters of the Membership Functions with Trapezoidal Membership Functions

For trapezoidal membership functions α_n could be $a_i^j, b_i^j, c_i^j, d_i^j$ and the gradients will be

$$\left.\begin{aligned}&\frac{\partial \mu_j^i(\hat{y}(t-k(l)))}{\partial a_i^j} =\\ &\begin{cases}0, & \hat{y}(t-k(l))<a_i^j\\ \dfrac{\left\{\dfrac{\partial \hat{y}(t-k(l))}{\partial a_i^j}\left(b_i^j-a_i^j\right)\right\}+\left(\hat{y}(t-k(l))-b_i^j\right)}{\left(b_i^j-a_i^j\right)^2}, & a_i^j<\hat{y}(t-k(l))<b_i^j\\ 0, & \hat{y}(t-k(l))>b_i^j\end{cases}\end{aligned}\right\} \text{(C-19)}$$

$$\left.\begin{aligned}&\frac{\partial \mu_j^i(\hat{y}(t-m(l)))}{\partial a_i^j} =\\ &\begin{cases}0, & \varepsilon(t-m(l))<a_i^j\\ \dfrac{\left\{-\dfrac{\partial \hat{y}(t-m(l))}{\partial a_i^j}\left(b_i^j-a_i^j\right)\right\}+\left(\varepsilon(t-m(l))-b_i^j\right)}{\left(b_i^j-a_i^j\right)^2}, & a_i^j<\varepsilon(t-m(l))<b_i^j\\ 0, & \varepsilon(t-m(l))>b_i^j\end{cases}\end{aligned}\right\} \text{(C-20)}$$

$$\left.\begin{aligned}&\frac{\partial \mu_j^i(\hat{y}(t-k(l)))}{\partial b_i^j} =\\ &\begin{cases}0, & \hat{y}(t-k(l))<a_i^j\\ \dfrac{\left\{\dfrac{\partial \hat{y}(t-k(l))}{\partial b_i^j}\left(b_i^j-a_i^j\right)\right\}-\left(\hat{y}(t-k(l))-a_i^j\right)}{\left(b_i^j-a_i^j\right)^2}, & a_i^j<\hat{y}(t-k(l))<b_i^j\\ 0, & \hat{y}(t-k(l))>b_i^j\end{cases}\end{aligned}\right\} \text{(C-21)}$$

$$\left.\begin{aligned}&\frac{\partial \mu_j^i(\in(t-m(l)))}{\partial b_i^j} =\\ &\begin{cases}0, & \in(t-m(l))<a_i^j\\ \dfrac{\left\{-\dfrac{\partial \hat{y}(t-m(l))}{\partial b_i^j}\left(b_i^j-a_i^j\right)\right\}-\left(\in(t-m(l))-a_i^j\right)}{\left(b_i^j-a_i^j\right)^2}, & a_i^j<\in(t-m(l))<b_i^j\\ 0, & \in(t-m(l))>b_i^j\end{cases}\end{aligned}\right\} \text{(C-22)}$$

$$\frac{\partial \mu_j^i\left(\hat{y}\left(t-k(l)\right)\right)}{\partial c_i^j} = \begin{cases} 0 & \hat{y}\left(t-k(l)\right) < c_i^j \\ \dfrac{\left\{ -\dfrac{\partial \hat{y}\left(t-k(l)\right)}{\partial c_i^j}\left(d_i^j - c_i^j\right) + \left(d_i^j - \hat{y}\left(t-k(l)\right)\right) \right\}}{\left(d_i^j - c_i^j\right)^2}, & c_i^j < \hat{y}\left(t-k(l)\right) < d_i^j \\ 0, & \hat{y}\left(t-k(l)\right) > d_i^j \end{cases} \quad \text{(C-23)}$$

$$\frac{\partial \mu_j^i\left(\in\left(t-m(l)\right)\right)}{\partial c_i^j} = \begin{cases} 0 & \in\left(t-m(l)\right) < c_i^j \\ \dfrac{\left\{ \dfrac{\partial \hat{y}\left(t-m(l)\right)}{\partial c_i^j}\left(d_i^j - c_i^j\right) + \left(d_i^j - \in\left(t-m(l)\right)\right) \right\}}{\left(d_i^j - c_i^j\right)^2}, & c_i^j < \in\left(t-m(l)\right) < d_i^j \\ 0, & \in\left(t-m(l)\right) > d_i^j \end{cases} \quad \text{(C-24)}$$

$$\frac{\partial \mu_j^i\left(\hat{y}\left(t-k(l)\right)\right)}{\partial d_i^j} = \begin{cases} 0 & \hat{y}\left(t-k(l)\right) < c_i^j \\ \dfrac{\left\{ -\dfrac{\partial \hat{y}\left(t-k(l)\right)}{\partial d_i^j}\left(d_i^j - c_i^j\right) - \left(d_i^j - \hat{y}\left(t-k(l)\right)\right) \right\}}{\left(d_i^j - c_i^j\right)^2}, & c_i^j < \hat{y}\left(t-k(l)\right) < d_i^j \\ 0, & \hat{y}\left(t-k(l)\right) > d_i^j \end{cases} \quad \text{(C-25)}$$

$$\frac{\partial \mu_j^i\left(\in\left(t-m(l)\right)\right)}{\partial d_i^j} = \begin{cases} 0 & \in\left(t-m(l)\right) < c_i^j \\ \dfrac{\left\{ \dfrac{\partial \hat{y}\left(t-m(l)\right)}{\partial d_i^j}\left(d_i^j - c_i^j\right) - \left(d_i^j - \in\left(t-m(l)\right)\right) \right\}}{\left(d_i^j - c_i^j\right)^2}, & c_i^j < \in\left(t-m(l)\right) < d_i^j \\ 0, & \in\left(t-m(l)\right) > d_i^j \end{cases} \quad \text{(C-26)}$$

C.3.3. Gradient for the parameters of the Membership Functions with Triangular Membership Functions with 0.5 Overlap

For triangular membership functions with 0.5 overlap the α_n parameters could be m_{j-1}`, m_j, m_{j+1} and the derivatives will be given by

$$\frac{\partial \mu_j^i\left(\hat{y}\left(t-k(l)\right)\right)}{\partial m_j^i} = \begin{cases} 0 & \hat{y}\left(t-k(l)\right) < m_{j-1}^i \\ \dfrac{\left\{ \dfrac{\partial \hat{y}\left(t-k(l)\right)}{\partial m_j^i}\left(m_j^i - m_{j-1}^i\right) - \left(\hat{y}\left(t-k(l)\right) - m_{j-1}^i\right) \right\}}{\left(m_j^i - m_{j-1}^i\right)^2}, & m_{j-1}^i < \hat{y}\left(t-k(l)\right) < m_j^i \\ \dfrac{\left\{ \dfrac{\partial \hat{y}\left(t-k(l)\right)}{\partial m_j^i}\left(m_{j+1}^i - m_j^i\right) - \left(\hat{y}\left(t-k(l)\right) - m_{j+1}^i\right) \right\}}{\left(m_{j+1}^i - m_j^i\right)^2}, & m_j^i < \hat{y}\left(t-k(l)\right) < m_{j+1}^i \\ 0 & \hat{y}\left(t-k(l)\right) > m_{j+1}^i \end{cases} \quad \text{(C-27)}$$

$$\frac{\partial \mu_j^i\left(y \in\left(t-m(l)\right)\right)}{\partial m_j^i} = \begin{cases} 0 & \in\left(t-m(l)\right) < m_{j-1}^i \\ \dfrac{\dfrac{\partial \hat{y}\left(t-m(l)\right)}{\partial m_j^i}\left(m_j^i - m_{j-1}^i\right) - \left(\in\left(t-m(l)\right) - m_{j-1}^i\right)}{\left(m_j^i - m_{j-1}^i\right)^2}, & m_{j-1}^i < \in\left(t-m(l)\right) < m_j^i \\ \dfrac{\dfrac{\partial \hat{y}\left(t-m(l)\right)}{\partial m_j^i}\left(m_{j+1}^i - m_j^i\right) - \left(\in\left(t-m(l)\right) - m_{j+1}^i\right)}{\left(m_{j+1}^i - m_j^i\right)^2}, & m_j^i < \in\left(t-m(l)\right) < m_{j+1}^i \\ 0 & \in\left(t-m(l)\right) > m_{j+1}^i \end{cases} \quad \text{(C-28)}$$

$$\frac{\partial \mu_{j+1}^i\left(\hat{y}\left(t-k(l)\right)\right)}{\partial m_i^j} = \begin{cases} 0 & \hat{y}\left(t-k(l)\right) < m_j^i \\ \dfrac{\left\{ \dfrac{\partial \hat{y}\left(t-k(l)\right)}{\partial m_j^i}\left(m_{j+1}^i - m_j^i\right) + \left(\hat{y}\left(t-k(l)\right) - m_{j+1}^i\right) \right\}}{\left(m_{j+1}^i - m_j^i\right)^2}, & m_j^i < y\left(t-k(l)\right) < m_{j+1}^i \\ 0, & \hat{y}\left(t-k(l)\right) > m_{j+1}^i \end{cases} \quad \text{(C-29)}$$

$$\frac{\partial \mu_{j+1}^{i}\left(\in\left(t-m(l)\right)\right)}{\partial m_i^j}= \left\{\begin{array}{ll} 0 & , \quad \varepsilon\left(t-m(l)\right)<m_j^i \\ \dfrac{\left\{\begin{array}{l} -\dfrac{\partial \hat{y}\left(t-m(l)\right)}{\partial m_j^i}\left(m_{j+1}^i-m_j^i\right) \\ +\left(\in\left(t-m(l)\right)-m_{j+1}^i\right) \end{array}\right\}}{\left(m_{j+1}^i-m_j^i\right)^2} & , \quad m_j^i<\varepsilon\left(t-m(l)\right)<m_{j+1}^i \\ 0 & , \quad \varepsilon\left(t-m(l)\right)>m_{j+1}^i \end{array}\right. \qquad \text{(C-30)}$$

$$\frac{\partial \mu_{j-1}^{i}\left(\hat{y}\left(t-k(l)\right)\right)}{\partial m_i^j}= \left\{\begin{array}{ll} 0 & , \quad \hat{y}\left(t-k(l)\right)<m_{j-1}^i \\ \dfrac{\left\{\begin{array}{l} \dfrac{\partial \hat{y}\left(t-k(l)\right)}{\partial m_j^i}\left(m_j^i-m_{j-1}^i\right) \\ +\left(\hat{y}\left(t-k(l)\right)-m_{j-1}^i\right) \end{array}\right\}}{\left(m_j^i-m_{j-1}^i\right)^2} & , \quad m_{j-1}^i<y\left(t-k(l)\right)<m_j^i \\ 0 & , \quad m_{j-1}^i<\hat{y}\left(t-k(l)\right)<m_j^i \end{array}\right. \qquad \text{(C-31)}$$

$$\frac{\partial \mu_{j-1}^{i}\left(\in\left(t-m(l)\right)\right)}{\partial m_i^j}= \left\{\begin{array}{ll} 0 & , \quad \varepsilon\left(t-m(l)\right)<m_{j-1}^i \\ \dfrac{\left\{\begin{array}{l} -\dfrac{\partial \hat{y}\left(t-m(l)\right)}{\partial m_j^i}\left(m_j^i-m_{j-1}^i\right) \\ +\left(\in\left(t-m(l)\right)-m_{j-1}^i\right) \end{array}\right\}}{\left(m_j^i-m_{j-1}^i\right)^2} & , \quad m_{j-1}^i<\varepsilon\left(t-m(l)\right)<m_j^i \\ 0 & , \quad \varepsilon\left(t-m(l)\right)>m_j^i \end{array}\right. \qquad \text{(C-32)}$$

Observe again that when there is no feedback in the model, then $\partial\hat{y}\left(t-m(l)\right)/\partial\alpha_n=0$ and the gradient expressions are reduced to the ones used for static functions. The presence of the terms $\partial\hat{y}\left(t-m(l)\right)/\partial\alpha_n=0$ in the gradients demands for computation of the gradients the calculation of a numerical solution of a discrete dynamic system, which must be updated each time a new data point is presented to the model.

Conflict of Interest

The authors declare that they have no conflict of interest.

REFERENCES

[1] C. T. Sun and J. S. R. Jang, "A neuro-fuzzy classifier and its applications". *Proc. of IEEE Int. Conf. on Fuzzy Systems*, San Francisco 1: 94–98, 1993.

[2] B. Cetisli and A. Barkana, "Speeding up the scaled conjugate gradient algorithm and its application in neuro-fuzzy classifier training", *Soft Computing*, vol. 14, no. 4, pp. 365 – 378, 2010.

[3] B. Cetisli, "Development of an adaptive neuro-fuzzy classifier using linguistic hedges: Part 1", *Expert Systems with Applications*, vol. 37, no. 8, pp. 6093–6101, 2010.

[4] M. Al-Mahasneh, M. T. Aljarrah, T. Rababah and M. H. Aludatt, "Application of Hybrid Neural Fuzzy System (ANFIS) in Food Processing and Technology, *Food Engineering Reviews*, vol. 8, no. 3, pp. 1 – 18, 2016.

[5] M. M. Jafari, H. Ojaghlou, M. Zare and G. J. Schumann, "Application of a Novel HybridWavelet-ANFIS/Fuzzy C-Means Clustering Model to Predict Groundwater Fluctuations", *Atmosphere*, vol. 12, no. 9, pp. 1 – 15, 2020.

[6] S. Chopra, G. Dhiman, A. Sharma, M. Shabaz, P. Shukla and M. Arora, "Taxonomy of Adaptive Neuro-Fuzzy Inference System in Modern Engineering Sciences", Computational Intelligence and Neuroscience, vol. 2021, pp. 1 – 14, 2021.

[7] S. O. Sada and S. C. Ikpeseni, "Evaluation of ANN and ANFIS modeling ability in the prediction of AISI 1050 steel machining performance", *Heliyon*, vol. 7, pp. 1 – 9, 2021.

[8] S. A. Saadat, S. M. Ghamari, H. Mollaee and F. Khavari, "Adaptive neuro-fuzzy inference systems (ANFIS) controller design on single-phase full-bridge inverter with a cascade fractional-order PID voltage controller", *IET Power Electronics*, vol. 14, no. 11, 2021.

[9] M. Nørgaard, O. Ravn, N. K. Poulsen and L.K. Hansen, "*Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook*". London: Springer-Verlag, 2000.

[10] J. Mikleš and M. Fikar. *Process Modelling, Identification and Control.* Springer-Verlag, Heidelberg, 2007.

[11] T. Bohlin. *Practical Grey-Box Process Identification.* Springer-Verlag, London, 2006.

[12] C. J. Lin and C. T. Lin, "An ART-based fuzzy adaptive learning control network, *IEEE Transactions on Fuzzy System*, vol. 5, no. 4, Nov. 1997.

[13] A. Garrett, "Fuzzy ART and fuzzy ARTMAP neural networks", 22nd December, 2003.

[14] C. M. Bishop, "*Pattern Recognition and Machine Learning*", Springer Science, Singapore, 2006.

[15] B. Balasko, J. Abonyi and B. Feil, "*Fuzzy Clustering and Data: Analysis Toolbox For Use with MATLAB*", Technical Report, Department of Process Engineering University of Veszprem, Veszprem, Hungary.

[16] F. Hoppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis*", John Wiley & Sons, Inc., Chichester, 1999.

[17] S. N. Sivanandam, S. Sumathi and S. N. Deepa, "*Introduction to Fuzzy Logic using MATLAB*", pp. 304-309,

Springer-Verlag, New York, 2007.

[18] J. Espinosa, J. Vandewalle and V. Wertz, "*Fuzzy Logic, Identification and Predictive Control*", Springer-Verlag, London, 2005.

[19] K. M. Passino and S. Yurkovich, "*Fuzzy Control*", Addison-Wesley Longman Inc., California, 1998.

[20] L. H. Tsoukalas and R. E. Uhrig, "*Fuzzy and Neural Approaches in Engineering*", John Wiley & Sons, Inc., Canada, 1997.

[21] C. T. Lin and C. S. G. Lee, "*Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*", Prentice-Hall, Upper Saddle River, 1996.

[22] S. Haykin. *Neural Networks: A Comprehensive Foundation*, 2nd Edition, Prentice-Hall, Upper Saddle River, New Jersey, 1999.

[23] H.B. Demuth, M.H. Beale and M.T Hagan. *Neural Network Design*. University of Colorado, 2007.

[24] M. M. Gupta, L. Jin and N. Homma, "Static and Dynamic Neural Networks: From Fundamental to Advanced Theory". *Hoboken, New Jersey: John Wiley & Sons*, 2003.

[25] The Math Works Inc., *MATLAB® & Simulink® R2021a*, Natick, USA. www.mathworks.com.

[26] J. S. R. Jang, C. T. Sun and E. Mizutani, "Neuro-fuzzyand soft-computing: A computational approach to learning and machine intelligence", Prentice-Hall, Upper Saddle River, NJ, 1997.

[27] T. O. S. Hanafy, A. S. Al-Osaimy, M. M. Al-Harthi, and A. A. Aly, "Identification of uncertain nonlinear MIMO spacecraft systems using Coactive neuro fuzzy inference system (CANFIS)", *International Journal of Control, Automation and Systems*, vol. 3, no. 2, 2014.

[28] J. M. Keller and H. Tahani, "Back-propagation neural networks for fuzzy logic", *Information Science*, vol. 62, pp. 205–221, 1992.

[29] I. Guler and E. D. Ubeyli, "Adaptive neuro-fuzzy inference system for classification of EEG signals using wavelet coefficients," *Journal of Neuroscience Methods*, vol. 148, pp. 113-121, 1992.

[30] X. G. Wang, Z. Tang, H. Tamura, M. Ishii and W. D. Sun, "An improved backpropagation algorithm to avoid the local minima problem", *Neurocomputing*, vol. 56, pp. 455 – 460, 2004.

[31] V. A. Akpan, "*Development of new model adaptive predictive control algorithms and their implementation on real-time embedded systems*", Aristotle university of Thessaloniki, GR-54124, Thessaloniki, Greece, Ph.D. Dissertation, 517 pages, July, 2011. Available [Online]: http://invenio.lib.auth.gr/record/127274/files/GRI-2011-7292.pdf.

[32] V. A. Akpan and G. D. Hassapis, "Training dynamic feedforward neural networks for online nonlinear model identification and control applications". *International Reviews of Automatic Control: Theory & Applications*, vol. 4, no. 3, pp. 335 – 350, 2011.

[33] V. A. Akpan and G. D. Hassapis, "Nonlinear model identification and adaptive model predictive control using neural networks", *ISA Transactions*; vol. 5, no. 2, pp. 177–94, 2011.

[34] A. Rizzi, M. Biancavilla and F. M. F. Mascioli, "Incremental Min-Max Network. Part 1: Continuous and Dicrete Spaces", *In M. Marinaro and R. Tagliaferri (eds): Neural Nets WIRN VIETRI-98. Perspectives in Neural Computing*, Springer-Verlag, London, 1999.

[35] F. M. F. Mascioli, M. Biancavilla and A. Rizzi, "Incremental Min-Max Network. Part 2: Continuous and Dicrete Spaces", *In M. Marinaro and R. Tagliaferri (eds): Neural Nets WIRN VIETRI-98. Perspectives in Neural Computing*, Springer-Verlag, London, 1999.

[36] D. Simon, "Training radial basis function neural networks with the extended Kalman filter", *Neurocomputing*, vol. 48, pp. 455 – 475, 2002.

[37] L. X. Wang, "*Adaptive Fuzzy Systems and Control*", Prentice Hall, Englewood Cliffs, New Jersey, 1994.

[38] R. Chiong, "Intelligent Systems for Automated Learning and Adaptation: Emerging Trends and Applications". *Hershey PA, USA: Information Science Reference,* 2010, Chapter 4,

[39] J. E. Dennis and R. B. Schnabel, "Numerical Methods for Unconstrained Optimization and Nonlinear Equations". Englewood Cliffs, NJ: SIAM, Prentice-Hall, 1996.

[40] C. T. Kelley, "Iterative Methods for Optimization". Philadelphia: SIAM, 1999.

[41] Verneda Lights and Elizabeth Boskey, "What Do You Want to Know About Meningitis?" Retrieved 28th August, 2021. Available [Online]: https://www.healthline.com/health/meningitis#types.

[42] Centers for Disease Control and Prevention, Retrieved 28th August, 2021. Available [Online]: https://www.cdc.gov/meningitis/index.html.

[43] U.S Department of Health and Human Services, "what you need to know", U.S. Department of health and human services, centre for disease control and prevention, Retrieved 28th August, 2021. Available [Online]: www.immunize.org/viz.10/14/2011.

[44] V. V. Phansalkar and P. S. Sastry, "Analysis of Back-Propagation Algorithm with Momentum", *IEEE Transactions on Neural Networks*, vol. 5, no. 3, pp. 505 – 506, 1994.

[45] Working Groups of COST Actions 632 and 624. (Apr., 2008). IWA Task Group on Benchmarking of Control Strategies for WWTPs: http://www.ensic.inplnancy.fr/benchmarkWWTP/Bsm1/Benchmark1.htm.

[46] V. A. Akpan and R. A. O. Osakwe, "Online Prediction of Influent Characteristics for Wastewater Treatment Plants Management Using Adaptive Recursive NNARMAX Model", *American Journal of Intelligent Systems*, vol. 4, no. 3, pp. 107 – 130, 2014. Available [Online]: http://article.sapub.org/pdf/10.5923.j.ajis.20140403.03.pdf.

[47] V. A. Akpan and R. A. O. Osakwe, "Multivariable NNARMAX Model Identification of an AS-WWTP Using ARLS (Part 1: Dynamic Modeling of the Biological Reactors)", *American Journal of Intelligent Systems*, vol. 4, no. 2, pp. 43 – 72, 2014. Available [Online]: http://article.sapub.org/pdf/10.5923.j.ajis.20140402.03.pdf.

[48] V. A. Akpan and R. A. O. Osakwe, "Multivariable NNARMAX Model Identification of an AS-WWTP Using ARLS (Part 2: Dynamic Modeling of the Secondary Settler and Clarifier*)", American Journal of Intelligent Systems*, vol. 4, no. 3, pp. 77 – 106, 2014. Available [Online]: http://article.sapub.org/pdf/10.5923.j.ajis.20140403.02.pdf.

[49] A. Visioli, *Practical PID Control* (Springer-Verlag Ltd., 2006).

[50] P. Hippe, *Windup in Control*, (Springer-Verlag Ltd., 2006).

[51] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4 – 27, 1990.