Adaptive LAMDA applied to identify and regulate a process with variable dead time

Luis Morales
Dpto. de Automatización y Control Industrial
Escuela Politécnica Nacional
Quito, Ecuador
luis.moralesec@epn.edu.ec

David Pozo
Facultad de Ingeniería y Ciencias Aplicadas, Ingeniería en Electrónica y Automatización
Universidad de las Américas,
Quito, Ecuador
david.pozo@udla.edu.ec

Jose Aguilar
CEMISID, Facultad de Ingeniería
Universidad de Los Andes
Mérida, Venezuela
GIDITIC, Universidad EAFIT,
Medellin, Colombia
aguilar@ula.ve

Andrés Rosales
Dpto. de Automatización y Control Industrial
Escuela Politécnica Nacional
Quito, Ecuador
andres.rosales@epn.edu.ec

***Abstract*— In this paper, an adaptive intelligent controller based on the fuzzy algorithm called LAMDA (Learning Algorithm for Multivariable Data Analysis) is presented in order to identify and regulate a process with variable dead time. The original algorithm has been used for supervised and unsupervised learning, whose main field of application is the identification of functional states of the systems. In this work a modification of LAMDA has been implemented which is capable of online learning using hybrid techniques. The proposal consists of two stages: training stage to learn about the unknown plant in order to establish initial parameters to the controller, and a second phase, called application, in which the control strategy is updated using online learning. The proposed method is tested in the control objective of regulation of a process with variable dead time, to analyze the viability of its utilization in these types of systems in which their dynamics are variable and unknown.**

Keywords*— *LAMDA, intelligent controller, online learning, adaptive fuzzy models

I. INTRODUCTION

In control systems, the modeling of the process to be controlled is essential in order to obtain a good performance. One of the simplest ways to model processes is the First Order Plus Dead Time (FOPDT) approach. This modeling method is commonly used in process like: Heating, ventilating and air conditioning (HVAC) systems, level and pressure systems, chemical processes, among others [1]–[3]. The use of controllers focused on achieving control objectives properly on these systems, particularly in steady and transitory states, as well as time response, is a wide field of study. Most works have focused on control methods based on conventional PID (proportional–integral–derivative), and its modifications, such as: Fuzzy PID Controller [4], PI controllers using criteria of evolutionary computing with Multi-gene Genetic Programming [5], optimal Robust PID tuning [6]. New proposals out of traditional PID controllers have been developed to improve the systems behavior, such as: Smith predictor [7], Dynamic sliding mode control (DSMC) [8], PD+I Fuzzy Controllers optimized by Particle Swarm Optimization (PSO) [9], generalized Predictive Control Tuning [10], among others.

In chemical processes, the modeling could be a challenge due to the lack of knowledge about its implicit parameters which are not considered in a FOPDT model. Although this approach is a good way to represent the behavior of a process, it has some inaccuracies and errors due to the model approximation.

In general, the controllers to be implemented must be robust enough, in order to maintain good performance when there are uncertainties not considered in the model, or disturbances or changes in the dynamics (e.g., the dead time). In this context, we propose the identification (modeling) and control of processes using a fuzzy computational technique based on LAMDA (Learning Algorithm for Multivariable Data Analysis). LAMDA is a machine learning method based on fuzzy logic that was developed to be used specially for classification and clustering [11], specifically, for the identification of functional states of systems. LAMDA algorithm has been used in several works, especially in fields like data analytics of diesel engines, [12], classification applications on the oil industry [13], software tools [14] among others, and recently, an application as a controller has been proposed in [15].

Particularly, this paper presents a modification to the LAMDA algorithm, in which layers are added to the original model to use it as an adaptive controller. The control action is computed by the algorithm, while performing an online identification of the plant to auto-adjust its parameters. The learning is done through a hybrid technique. The proposed method can be applied to plants with variable characteristics, since it adapts to changes in their dynamics. Being a model with fixed number of layers, it has a great advantage over similar methods, such as neural networks in which this parameter must be defined based on the expertise of the designer. Furthermore, it is easy to implement due to the use of simple mathematical operations, without high computational complexity and with good error reduction in both, the modeling and control stages. Regarding conventional methods, this approach avoids performing the parameter calibration stage, e.g., scaling gains; and with respect to fuzzy controllers, the rule definition stage (based on knowledge) is omitted, which can be complex if the behavior of the controlled system is unknown.

978-1-7281-6932-3/20/$31.00 ©2020 IEEE

To demonstrate the adaptability of the proposed controller, a variable dynamic plant has been selected. Specifically, a system corresponding to a mixing tank with variable dead time has been chosen to verify if the proposal can adapt to these changes without losing performance. This parameter is compared with the response obtained with Fuzzy-PI and LAMDA-PI controllers, which are ruled-based methods.

This work is organized as follows: in section II, a detailed explanation about the adaptive LAMDA algorithm is presented. In section III, the adaptive LAMDA approach as a controller is shown. The simulations and results of the mixing tank model case study are presented in section IV. Finally, section V presents the conclusions of the paper.

II. Learning Algorithm for Multivariable Data Analysis

A. Structure of the modified LAMDA algorithm

LAMDA is a fuzzy algorithm used for classification and clustering. Compared to other algorithms, it performs a similitude analysis among the descriptors of one object, $X = [x_1; \ldots; x_j; \ldots; x_n]$, and the existing classes or clusters $C = \{C_1; \ldots; C_k; \ldots; C_m\}$, in order to establish its membership degree [14].

The procedure of the algorithm starts with the normalization of the descriptors of the object X, using (1).

$$\bar{x}_j = \frac{x_j - x_{jmin}}{x_{jmax} - x_{jmin}} \tag{1}$$

x_{jmax} is the maximum value of the descriptor x_j, x_{jmin} is the minimum value of the descriptor x_j; and $\bar{x}_j$ is the normalized j-descriptor.

1) Marginal Adequacy Degree (MAD)

With the normalized descriptors, the algorithm computes the $MADs$, which have information about the similarity of a descriptor with respect to the same descriptor in all the classes/clusters. $MADs$ are computed with probability density functions, such as: fuzzy binomial (2) or Gaussian (3) [13].

$$MAD_{k,j}(\bar{x}_j, \rho_{k,j}) = {\rho_{k,j}}^{\bar{x}_j}\left(1 - \rho_{k,j}\right)^{\left(1-\bar{x}_j\right)} \tag{2}$$

$$MAD_{k,j}(\bar{x}_j, \rho_{k,j}) = e^{-\frac{1}{2}\left(\frac{\bar{x}_j - \rho_{k,j}}{\sigma_{k,j}}\right)^2} \tag{3}$$

where $\rho_{k,j}$ is the average value of the descriptor j belonging to the class k, and it is computed by (4).

$$\rho_{k,j} = \frac{1}{n_{k,j}} \sum_{t=1}^{n_{k,j}} \bar{x}_j(t) \tag{4}$$

$\sigma_{k,j}$ is the standard deviation (dispersion) of the descriptor j belonging to the class k, and it is computed by (5).

$${\sigma_{k,j}}^2 = \frac{1}{n_{k,j} - 1} \sum_{t=1}^{n_{k,j}} (\bar{x}_j(t) - \rho_{k,j})^2 \tag{5}$$

LAMDA has a non-identified class (NIC) for the cases where an object cannot assigned to a known class. For the NIC, $\rho_{NIC,j} = 0.5$. With this value of the probability function (2), $MAD_{NIC} = 0.5$ and $\sigma_{NIC,j} = 0.25$ for any value of the descriptor $\bar{x}_j$.

2) Global Adequacy Degree (GAD)

The combination of the $MADs$ in each class results in the $GADs$. The $GADs$ are computed using fuzzy logic operators (aggregation functions). These functions are linear interpolations between the T-norm and S-norm presented in (6) and (7), respectively.

$$T(a, b) = ab \tag{6}$$

$$S(a, b) = a + b - ab \tag{7}$$

The GAD for each class is calculated as:

$$\begin{aligned} GAD_{k,\bar{X}}(MAD_{k,1}, \ldots MAD_{k,n}) &= \alpha T(MAD_{k,1}, \ldots, MAD_{k,n}) \\ &+ (1 - \alpha) S(MAD_{k,1}, \ldots, MAD_{k,n}) \end{aligned} \tag{8}$$

The exigency parameter $\alpha \in [0,1]$ is used to perform a strict or permissible classification. If α increases then a strict classification is performed, thus, more objects will be unrecognized (sent to the NIC), making the algorithm more selective. If α decreases, then the algorithm is permissible, this is, the individual is assigned to the most similar class although there is no great similarity. In cases where two or more descriptors are available, then (6) and (7) are computed recurrently. The object $\bar{X}$ is assigned to the class/cluster with the highest similarity degree (maximum GAD).

$$index = \max\left(GAD_{1,\bar{X}}, GAD_{k,\bar{X}}, \ldots, GAD_{m,\bar{X}}, GAD_{NIC,\bar{X}}\right) \tag{9}$$

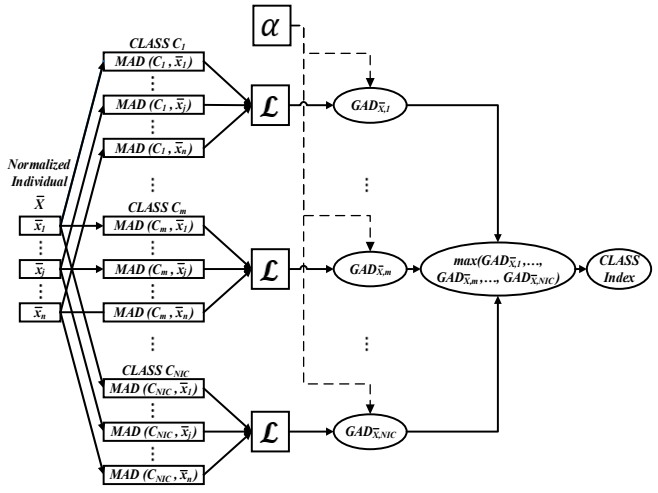

Fig. 1. Structure of the LAMDA algorithm

The original structure of the algorithm presented in Fig. 1 is simple and works properly in classification or clustering tasks. However, an online learning requires changes in its structure. Therefore, we propose to add layers to the model and the learning methodology known as hybrid is implemented. Unlike [15], where LAMDA is used as a

controller based on the expertise of the designer to define the classes and rules, in this work, it is proposed that the definition of the classes be automatic, depending on the behavior of the input-output data. Finally, an inference method, based on Takagi-Sugeno, is applied in order to calculate the output of the controller. Fig. 2 shows the new LAMDA structure. In layer 1 and layer 2 the calculations of the $MADs$ and the $GADs$ are performed respectively. The added layer 3 is used to normalize the GADs, resulting in $NGAD_{\bar{X},k}$ as:

$$NGAD_{\bar{X},k}\left(GAD_{\bar{X},1},\dots,GAD_{\bar{X},m}\right)=\frac{GAD_{\bar{X},k}}{\sum_{k=1}^{m}GAD_{\bar{X},k}} \quad (10)$$

Layer 4 corresponds to the product of the $NGAD_{\bar{X},k}$ with a first order Takagi-Sugeno function $H_k(\cdot)$ for each class k, using the descriptors of the analyzed individual (11). This function has $n+1$ parameters, depending on the number of descriptors of $\bar{X}$. These parameters are called consequent parameters.

$$H_k\left(\bar{X},h_{k1},\dots,h_{kj},\dots,h_{kn},h_k\right)$$
$$=\bar{x}_1h_{k1}+\cdots+\bar{x}_jh_{kj}+\cdots+\bar{x}_nh_{kn}+h_k \quad (11)$$

and the output of layer 4 is:

$$f_k\left(NGAD_{\bar{X},k},H_k\right)=NGAD_{\bar{X},k}H_k \quad (12)$$

Finally, Layer 5 has only the node O_L, which is computed as the sum of all the inputs as:

$$O_L(f_1,\dots,f_k,\dots f_m)=\sum_{k=1}^{m}f_k \quad (13)$$

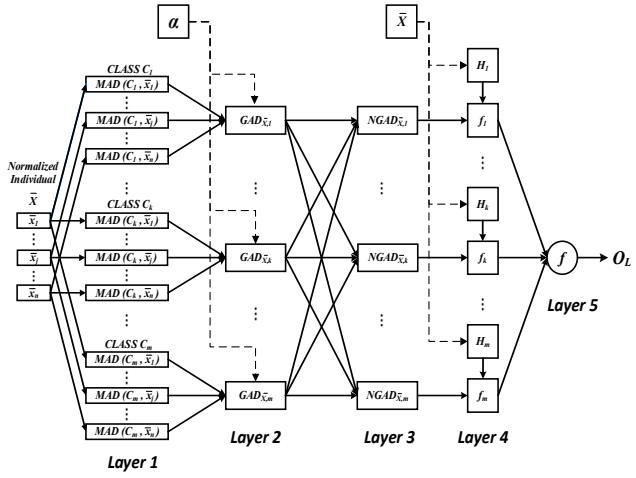

Fig. 2. Adaptive scheme for LAMDA

B. Hybrid learning

The proposed LAMDA architecture is composed of layers and nodes that have specific functions. The nodes of layer 1 and 4 have parameters that must be adapted during the learning process using the input-output data.

For this process, a hybrid learning is used, as is proposed in [16], [17]. It consists of a step forward and a step backward that improves the learning time, preventing that solutions be trapped in local minima [18]. The proposed hybrid learning scheme consisting of a step forward and a step backward is shown in Fig. 3.

1) Forward pass

The forward pass starts from layer 1, where antecedent parameters are fixed with initial values, then the computation of nodes until layer 4 is done. In this layer, the parameters H_k are calculated by the least squares method because the output of this layer is a linear combination of the consequent parameters. Developing (13), for the d-th individual $\bar{X}^d=\left[\bar{x}_1^d,\dots,\bar{x}_j^d,\dots,\bar{x}_n^d\right]$ gives as result the output o_L^d.

$$o_L^d=NGAD_{\bar{X}^d,1}H_1+\cdots+NGAD_{\bar{X}^d,k}H_k+\cdots$$
$$+NGAD_{\bar{X}^d,m}H_m \quad (14)$$

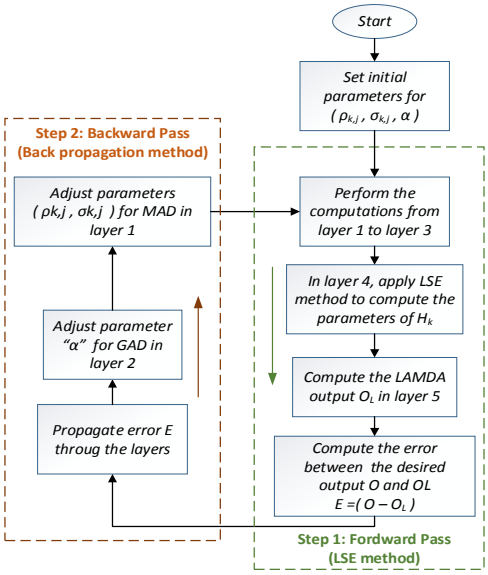

Fig. 3. Hybrid Learning Scheme applied to LAMDA

Solving (14) for all the desired values in the output $O=[o^1\dots o^d\dots o^D]^T$, it results in a standard linear regression problem that can be solved as follows:

$$O=Ah \quad (15)$$

If the matrix A is invertible, then (16) can be used; otherwise, the pseudoinverse is computed by (17).

$$h=A^{-1}O \quad (16)$$

$$h=(A^TA)^{-1}A^TO \quad (17)$$

The calculation of the inverse matrix is computationally expensive, thus, h is obtained with a sequential method applied to time-varying systems. This method uses the d-th row vector of the matrix A, defined by a_d^T, and the d-th element of O, defined by o^d. Then, h is iteratively computed using the covariance matrix P_{d+1}:

$$P_{d+1}=\frac{1}{\lambda}\left[P_d-\frac{P_da_{d+1}a_{d+1}^TP_d}{\lambda+a_{d+1}^TP_da_{d+1}}\right] \quad (18)$$

Where $\lambda\in[0,1]$ is the forgetting factor, and, h_{d+1} is computed as:

$$h_{d+1} = h_d + P_{d+1}a_{d+1}\left(o^{d+1} - a_{d+1}^T h_d\right);\ \forall d = \{1, \dots, D-1\} \tag{19}$$

2) Backward pass

Computed the consequent parameters h_{d+1} the data output set $O = [o^1 \dots o^d \dots o^D]^T$ is used, to spread the error backwards from layer 5 to layer 1. Being o^d the d-th data of the desired output O, and o_L^d the output computed by LAMDA for the $\bar{X}^d$ individual, the error in layer 5 is:

$$E_d = \frac{1}{2}[o^d - o_L^d]^2 \tag{20}$$

The aim is to propagate backward the error E_d, to obtain the derivative of the error E_d with respect to the antecedent parameters $\theta = \{\rho_{k,j}, \sigma_{k,j}, \alpha\}$ used in (3) and (8), to compute $MADs$ and $GADs$, respectively. With this adjustment, we avoid manually calibrating the parameters in θ, especially the parameter α, which modifies the exigency in clustering and classification tasks, improving the performance of the algorithm.

$$\frac{\partial E_d}{\partial \theta} \tag{21}$$

For all training data, we have:

$$\frac{\partial E}{\partial \theta} = \sum_{d=1}^{D} \frac{\partial E_d}{\partial \theta} \tag{22}$$

and the adjustment of θ at the time $(t\ +\ 1)$ is:

$$\theta(t+1) = \theta(t) - \eta \frac{\partial E}{\partial \theta} + \beta\big(\theta(t) - \theta(t-1)\big) \tag{23}$$

where $\eta \in [0,1]$ is the learning rate, and $\beta \in [0,1]$ is the momentum term.

The learning with the gradient descent method through the backpropagation of the error E_d from layer 5 to 1 is:

- Layer 5: $\epsilon^{(5)} = \frac{\partial E_d}{\partial o_L^d} = -(o^d - o_L^d)$ (24)
- Layer 4: The derivative of o_L^d with respect to ∂f_k is:

$$\frac{\partial o_L^d}{\partial f_k} = 1\ ;\ \forall k = 1, \dots, m \tag{25}$$

$$\epsilon_k^{(4)} = \epsilon^{(5)} \tag{26}$$

- Layer 3: In this layer the derivative to compute is:

$$\frac{\partial f_k}{\partial NGAD_{\bar{X},k}} = H_k\ ;\ \forall k = 1, \dots, m \tag{27}$$

$$\epsilon_k^{(3)} = \epsilon^{(5)} H_k\ ;\ \forall k = 1, \dots, m \tag{28}$$

- Layer 2: The partial derivatives of layer 3 are calculated with respect to the outputs of layer 2. Because each node k of layer 3 depends on all the outputs of layer 2, as is shown in (6), the term k_2 is used to refer to the nodes of layer 2.

$$\frac{\partial NGAD_{\bar{X},k}}{\partial GAD_{\bar{X},k_2}} = \begin{cases} \dfrac{\left(\sum_{k_2=1}^{m} GAD_{\bar{X},k_2}\right) - GAD_{\bar{X},k_2}}{\left(\sum_{k_2=1}^{m} GAD_{\bar{X},k_2}\right)^2} & if: k_2 = k \\ -\dfrac{GAD_{\bar{X},k}}{\left[\sum_{k=1}^{m} GAD_{\bar{X},k_2}\right]^2} & if: k_2 \neq k \end{cases} \tag{29}$$

$$\epsilon_k^{(2)} = \epsilon^{(5)} H_k \sum_{k=1}^{m} \frac{\partial NGAD_{\bar{X},k}}{\partial GAD_{\bar{X},k_2}}\ ;\ \forall k = 1, \dots, m \tag{30}$$

- Layer 1: The partial derivatives of layer 2 are computed with respect to the outputs of layer 1. Because the $GADs$ are calculated recursively, the term j_1 is used to refer to each of the nodes of the layer 1.

$$\frac{\partial GAD_{\bar{X},k}}{\partial MAD_{k,j}} = \alpha T\big(MAD_{k,1}, \dots, MAD_{k,j_1}, \dots, MAD_{k,n}\big) + (1-\alpha) S\big(MAD_{k,1}, \dots, MAD_{k,j_1}, \dots, MAD_{k,n}\big)\ ;\ \forall j_1 \neq j \tag{31}$$

and the propagated error in layer 1 is:

$$\epsilon_k^{(1)} = \epsilon_k^{(2)} \frac{\partial GAD_{\bar{X},k}}{\partial MAD_{k,j}} \tag{32}$$

The parameters $\rho_{k,j}, \sigma_{k,j}$ are updated in each class k and each descriptor j with (33) and (34), respectively, and α is updated with (35).

$$\frac{\partial MAD_{k,j}}{\partial \rho_{k,j}} = \frac{\left(\bar{x}_j - \rho_{k,j}\right)}{\left(\sigma_{k,j}\right)^2} \partial MAD_{k,j} \Longrightarrow \frac{\partial E_d}{\partial \rho_{k,j}} = \epsilon_k^{(1)} \frac{\partial MAD_{k,j}}{\partial \rho_{k,j}} \tag{33}$$

$$\frac{\partial MAD_{k,j}}{\partial \sigma_{k,j}} = \frac{\left(\bar{x}_j - \rho_{k,j}\right)^2}{\left(\sigma_{k,j}\right)^3} \partial MAD_{k,j} \Longrightarrow \frac{\partial E_d}{\partial \sigma_{k,j}} = \epsilon_k^{(1)} \frac{\partial MAD_{k,j}}{\partial \sigma_{k,j}} \tag{34}$$

$$\frac{\partial GAD_{\bar{X},k}}{\partial \alpha} = \big[T\big(MAD_{k,1}, \dots, MAD_{k,j}, \dots, MAD_{k,n}\big) - S\big(MAD_{k,1}, \dots, MAD_{k,j}, \dots, MAD_{k,n}\big)\big] \Longrightarrow \frac{\partial E_d}{\partial \alpha} = \sum_{k=1}^{m} \epsilon_k^{(2)} \frac{\partial GAD_{\bar{X},k}}{\partial \alpha} \tag{35}$$

Finally, the update of the terms is computed as:

$$\rho_{k,j}(t+1) = \rho_{k,j}(t) + \eta\left(-\frac{\partial E_d}{\partial \rho_{k,j}}\right) + \beta\left(\rho_{k,j}(t) - \rho_{k,j}(t-1)\right) \tag{36}$$

$$\sigma_{k,j}(t+1) = \sigma_{k,j}(t) + \eta\left(-\frac{\partial E_d}{\partial \sigma_{k,j}}\right) + \beta\left(\sigma_{k,j}(t) - \sigma_{k,j}(t-1)\right) \tag{37}$$

$$\alpha(t+1) = \alpha(t) + \eta\left(-\frac{\partial E_d}{\partial \alpha}\right) + \beta(\alpha(t) - \alpha(t-1)) \tag{38}$$

For the purpose of this work (online learning), this procedure is carried out every sample time.

III. Proposed Approach

The controller tested in this work is based on the strategy of inverse learning. It consists of two stages: learning and application. In the learning stage, the plant inverse model φ is obtained using the input-output data applied to the system, considering the existence of a NARX (nonlinear autoregressive model with exogenous input) model as regression vector:

$$\varphi = [u(k), \dots, u(k-p), x(k+1), x(k) \dots, x(k-q)] \quad (39)$$

For this stage (see Fig. 4), it is necessary to apply a time-variant input $u(k)$ and take the output data $x(k+1)$ and the previous states of the plant produced by that signal. LAMDA minimizes the error $e_u(k)$ following the procedure detailed in the previous section.

In the application stage, the algorithm is used for generating the control actions necessary to bring the system to a desired state. In order to guarantee an adequate response from the controller, which eliminates the error in a steady state, the adaptive online learning criterion is introduced. For this, the initial model obtained in the learning stage is used as a starting point, and the learning and control actions are updated at each sample time, as shown in Fig. 5.

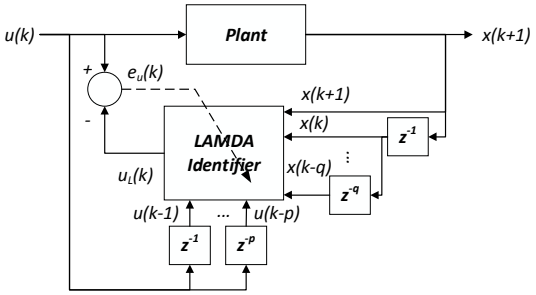

Fig. 4. Block diagram of the training phase of the inverse control method

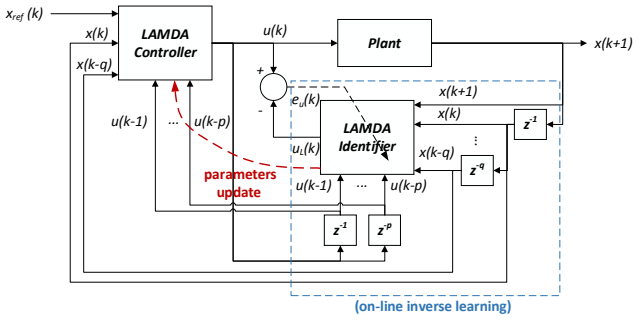

Fig. 5. Block diagram of the application phase of the inverse control method

IV. Simulations and Results

A. Mixing tank with variable dead-time

The process corresponds to the mixing of two fluids in a tank with a constant liquid volume (Fig. 6). The system has a hot stream $W_1(t)$ that mixes with a cold stream $W_2(t)$ controlled with a valve. The resulting mixture requires to remain at a desired temperature. The temperature transmitter is installed at a distance of $125\ [ft]$ from the tank outlet and has been calibrated to operate in a range of 100 to $200\ [°F]$ The distance between the tank outlet and the location of the temperature transmitter generates a dead time in the measurement.

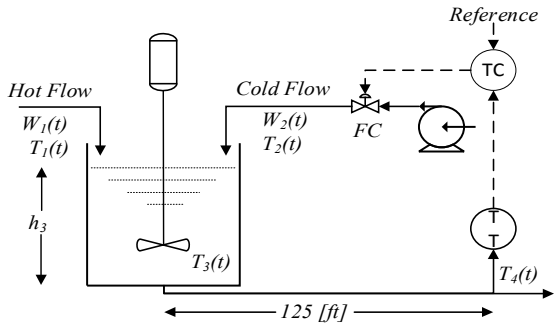

Fig. 6. Studied Process (Mixing Tank)

In the system analysis, the next aspects are considered:

- The tank contents are well mixed
- The pipe and the tank are well isolated
- The main disturbance of the system is the hot stream

The following expressions correspond to the mathematical process model used for the simulation:

- Energy balance in the mixing tank

$$W_1(t)Cp_1T_1(t) + W_2(t)Cp_2T_2(t) - \big(W_1(t) + W_2(t)\big)Cp_3T_3(t) = VCv_3\frac{dT_3(t)}{dt} \quad (40)$$

- Pipe delay (mixing tank and temperature sensor)

$$T_4(t) = T_3(t)\big(t - t_0(t)\big) \quad (41)$$

- Time delay

$$t_0(t) = \frac{LA\rho}{W_1(t) + W_2(t)} \quad (42)$$

- Temperature transmitter

$$\frac{dTO(t)}{dt} = \frac{1}{\tau_T}\left[\frac{T_4(t) - 100}{100} - TO(t)\right] \quad (43)$$

- Control valve position

$$\frac{dV_p(t)}{dt} = \frac{1}{\tau_{V_p}}\big[m(t) - V_p(t)\big] \quad (44)$$

- Control valve:

$$W_2(t) = \frac{500}{60}C_{VL}V_p(t)\sqrt{G_f\Delta P_v} \quad (45)$$

where

$W_1(t)$: mass flow of hot stream, lb/min.
$W_2(t)$: mass flow of cold stream, lb/min.
C_p: liquid heat capacity at constant pressure, $Btu/lb - °F$.
C_v: liquid heat capacity at constant volume, $Btu/lb - °F$.

h_3: tank content level, ft.
$T_1(t)$: hot flow temperature, $°F$
$T_2(t)$: cold flow temperature, $°F$
$T_3(t)$: liquid temperature in the mixing tank, $°F$
$T_4(t)$: temperature $T_3(t)$ considering the delay t_0, $°F$.
t_0: dead time, min.
ρ: density of the mixing tank contents, lb/ft^3.
C_{VL}: valve flow coefficient, $gpm/psi^{1/2}$
$TO(t)$: transmitter output signal normalized from 0 a 1, $p.u.$
$V_p(t)$: valve position, from 0 (closed valve) to 1 (open valve).
$m(t)$: fraction of controller output, from 0 to 1, $p.u.$
G_f: specific gravity, dimensionless.
ΔP_v: pressure drop across the valve, psi.
τ_T: time constant of the temperature sensor, min.
τ_{V_p}: time constant of the control valve, min.
A: pipe cross section, ft^2.
L: pipe length, ft.

Table I shows the parameters at the desired operating point.

TABLE I. NUMERICAL VALUES FOR THE SYSTEM PARAMETERS AT THE OPERATING POINT

Variable	Value	Variable	Value
W_1	250 lb/min	C_{VL}	12 $gpm/psi^{1/2}$
W_2	191.17 lb/min	TO	0.5 $p.u.$
Cp_1	0.8 $Btu/lb-°F$	V_p	0.478
Cp_2	1.0 $Btu/lb-°F$	ΔP_v	16 psi
Cv_3	0.9 $Btu/lb-°F$	m	0.478 $p.u.$
Cp_3	0.9 $Btu/lb-°F$	G_f	1
T_1	250 $°F$	τ_T	0.5 min
T_2	50 $°F$	τ_{V_p}	0.4 min
T_3	150 $°F$	A	0.2006 ft^2
ρ	62.4 lb/ft^3	L	125 ft

This system has been used to test different types of controllers due to variable dead time modeled in (41). In general, the model can be approximated to a FOPDT [1]:

$$\frac{x(s)}{u(s)} = \frac{Ke^{-tos}}{\tau s + 1} \tag{46}$$

Also in [1], the dead time can be modeled using first-order Taylor series approximation, producing (47).

$$e^{-tos} \cong \frac{1}{t_0 s + 1} \tag{47}$$

Substituting (47) into (46), we have:

$$\frac{x(s)}{u(s)} \cong \frac{K}{(\tau s + 1)(t_0 s + 1)} \tag{48}$$

Resulting in a second order system in a discrete time:

$$\frac{x(z)}{u(z)} \cong \frac{az + b}{cz^2 + dz + f} \tag{49}$$

Developing (49), according to k, we obtain:

$$u(k) = \frac{cx(k+1) + dx(k) + fx(k-1) - bu(k-1)}{a} \tag{50}$$

Then, the LAMDA model will be trained with four inputs corresponding to $x(k+1),\ x(k),\ x(k-1)\ and\ u(k-1)$.

Euler's method has been used for the system discretization to perform the simulation of the process, and the input and output variables in the mixing tank are:

$$u(k) = m(k)\ ;\ x(k) = TO(k) \tag{51}$$

Since the first order approximations proposed in (46) and (47) can present uncertainties and errors in modeling because the process has nonlinear effects, we use LAMDA to perform the modeling and controlling of this process. The proposed scheme, after the training stage, is presented in Fig. 7.

We propose two scenarios to evaluate the performance of the adaptive model (LAMDA-Learning). The first scenario consists of a change of the system reference; the second scenario focuses on a regulation problem in which the control objective is to keep the temperature of the mixture at 150 $[°F]$ when the hot stream W_1 is changed (disturbance). The algorithm has been calibrated with 4 inputs, each with two classes, $\eta = 0.001, \beta = 0.001, \lambda = 1, T_s = 0.4\ min$.

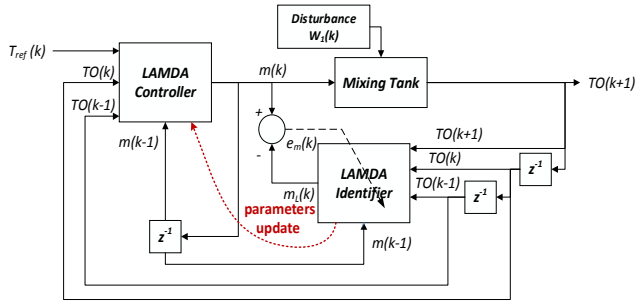

Fig. 7. Adaptive control structure for the mixing tank

1) Scenario 1: Process output with reference change

Fig. 8 shows the behavior of the system in scenario 1, changing the temperature reference from 150 $[°F]$ to 130 $[°F]$, without considering disturbances. The control action is presented in Fig. 9, which shows the comparative response of the proposed controller based on online learning (LAMDA-learning) against a FUZZY-PI controller (rules based) and LAMDA-PI (class based), which have been previously presented in [15].

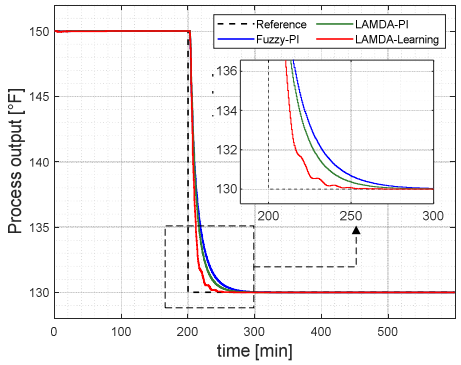

Fig. 8. Process output response to the reference change

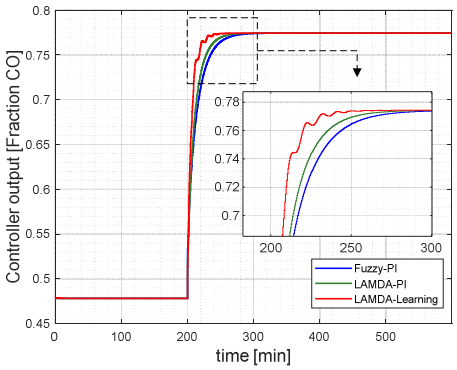

Fig. 9. Controller output response to the reference change

As noted, LAMDA-Learning presents a faster response compared to the other methods, in order to reach the desired reference. In the zoom of the image, the transient response of the system is shown, both for the process output and for the control action, where its behavior is observed in detail, which has minimum oscillations that do not affect the system performance, eliminating the error in steady state in less time.

2) Scenario 2: Process output with disturbances

In this scenario, the response of the system considering the disturbances produced by varying hot stream W_1 is presented. As is shown in (43), the dead time changes depending on W_1. The changes of the hot stream are shown in Fig. 10a, and the variation of the dead time is presented in Fig. 10b, here it is observed that it changes in a range of 3.6 to 4.9 $[min]$. The variable dead time causes the dynamics of the system to change depending on this parameter, so it is appropriate to use an adaptive method such as LAMDA.

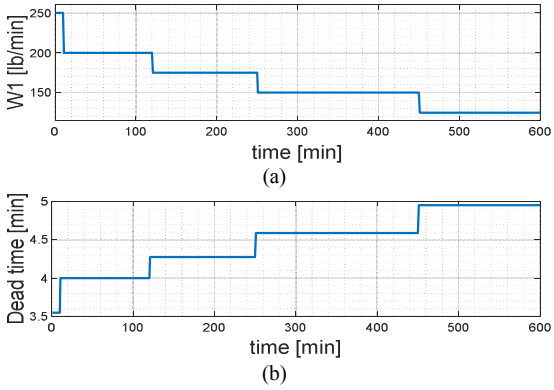

Fig. 10. Change of: a) W_1 and b) dead time

Similar to the previous case, the tests are carried out for the proposed controller (LAMDA-Learning), the FUZZY-PI and LAMDA-PI. Fig. 11 shows the response of temperature T_4 for the changes presented in Fig. 10a. The control action of the different methods is presented in Fig. 12.

The behavior of LAMDA-Learning is adequate, considering that the learning of the algorithm is automatic. It is observed that the response of the controller improves as the time increases, i.e., a slow response is seen in the disturbance given at $10min$; however, it is improving as more data is received. In the disturbance given to the $450\,min$, it is observed that although there is a presence of oscillations in the response, it is quickly attenuated, even faster than the other two proposals, without degrading and maintaining its performance and stability.

The numerical criteria for the evaluation of the controllers in this experiment is the IAE (integral absolute error), which is an index used to measure the performance. The IAE reflexes the cumulative error, i.e., how far the response is with respect to the applied reference. Therefore, the controller that reaches the minimum index is the best in terms of performance.

TABLE II. IAE OF LAMDA-PI, FUZZY PI AND LAMDA-LEARNING APPROACHES

Scenario	*Index*	*FUZZI PI*	*LAMDA PI*	*LAMDA Learning*
Reference change	***IAE***	2.979	2.661	1.960
	Δ	41.26%	30.34%	-
Disturbance	***IAE***	5.719	5.185	4.544
	Δ	22.90%	13.18%	-

Table 2 shows that the index with the lowest value is LAMDA-Learning, because it reaches the reference in less time and the number of oscillations decreases faster than the other proposals. In the reference change, it can be seen that the adaptive proposal is better in 30%, and in the case of disturbances in 13%, with respect to LAMDA-PI (approach based on classes), percentages that show the potential of the learning algorithm in control systems.

Regarding the computational complexity, the average time used in an iteration composed by the calculation of the control action and the learning task has been measured, considering the 4 inputs established in (50), resulting in 0.8 ms for 16 classes, 7 ms for 48 classes and 112 ms for 256 classes.

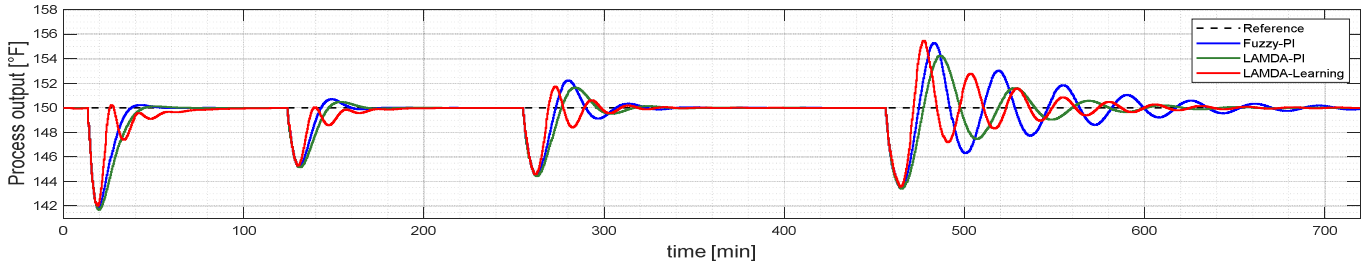

Fig. 11. Comparative response of the system (temperature)

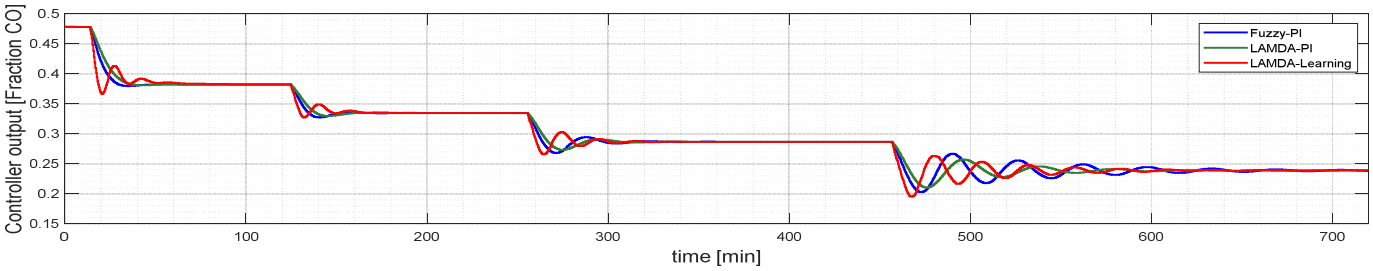

Fig. 12. Applied control actions to the system

V. Conclusion

In this work, the implementation of a controller based on the adaptive LAMDA algorithm has been tested on a system with variable dead time. LAMDA based on online learning for system identification and control has presented satisfactory results when it is compared with class-based LAMDA and a conventional FUZZY controller, which present good results if they have an adequate calibration depending on a good knowledge of the plant by the designer.

This paper has allowed to observe that LAMDA-Learning is capable of modeling and controlling systems with variable dynamics, due to its ability of online learning from the behavior of the system, which allows an auto-adjusting of its internal parameters, which is a great advantage over the other methods shown since the definition of rules and classes is not required.

As future work, it is proposed to test the algorithm in a real process, and to analyze its stability properties.

References

[1] O. Camacho and C. A. Smith, "Sliding mode control: an approach to regulate nonlinear chemical processes," *ISA Trans.*, vol. 39, no. 2, pp. 205–218, 2000.

[2] P. Bagheri and A. Khaki Sedigh, "Robust tuning of dynamic matrix controllers for first order plus dead time models," *Appl. Math. Model.*, vol. 39, no. 22, pp. 7017–7031, 2015.

[3] Nitin Sundriyal, Pratibha Yadav, and Mayank Chaturvedi, "A Novel Approach for Designing PID Controller for Set-Point Tracking for a HVAC Process," vol. 479, R. Singh and S. Choudhury, Eds. Singapore: Springer Singapore, 2017, pp. 161–166.

[4] Y. Zhou, B. Qi, S. Huang, and Z. Jia, "Fuzzy PID Controller for FOPDT system based on a hardware-in-the-loop simulation," *Chinese Control Conf. CCC*, vol. 2018-July, pp. 3382–3387, 2018.

[5] D. Pozo, L. Morales, D. Maldonado, and J. Aguilar, "A Novel Methodology to obtain Optimal PI Controller Gains using Multi-gene Genetic Programming for FOPTD Systems," in *2018 IEEE Third Ecuador Technical Chapters Meeting (ETCM)*, 2018, pp. 1–6.

[6] T. Sato, I. Hayashi, Y. Horibe, R. Vilanova, and Y. Konishi, "Optimal Robust PID Control for First- and Second-Order Plus Dead-Time Processes," *Appl. Sci.*, vol. 9, no. 9, p. 1934, May 2019.

[7] B. S. Bright and R. Swarnalatha, "Performance evaluation of two degree of freedom conventional controller adopting the smith principle for first order process with dead time," *Int. J. Electr. Comput. Eng.*, vol. 9, no. 4, p. 3002, Aug. 2019.

[8] J. Yang, J. Su, S. Li, and X. Yu, "High-Order Mismatched Disturbance Compensation for Motion Control Systems Via a Continuous Dynamic Sliding-Mode Approach," *IEEE Trans. Ind. Informatics*, vol. 10, no. 1, pp. 604–614, Feb. 2014.

[9] J. Campos, S. Jaramillo, L. Morales, O. Camacho, and D. Chavez, "PD + I Fuzzy Controller optimized by PSO applied to a variable dead time process," in *IEEE ETCM*, 2018, vol. i, pp. 1–6.

[10] K. Manjari and S. Majhi, "Generalized Predictive Control Tuning for Desired Frequency Domain Specification," in *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, 2019, pp. 2245–2250.

[11] J. F. Botía Valderrama and D. J. L. Botía Valderrama, "On LAMDA clustering method based on typicality degree and intuitionistic fuzzy sets," *Expert Syst. Appl.*, vol. 107, pp. 196–221, 2018.

[12] F. A. Ruiz, C. V. Isaza, A. F. Agudelo, and J. R. Agudelo, "A new criterion to validate and improve the classification process of LAMDA algorithm applied to diesel engines," *Eng. Appl. Artif. Intell.*, vol. 60, no. October 2016, pp. 117–127, 2017.

[13] L. Morales, H. Lozada, J. Aguilar, and E. Camargo, "Applicability of LAMDA as classification model in the oil production," *Artif. Intell. Rev.*, p. 32, 2019.

[14] L. Morales, C. A. Ouedraogo, J. Aguilar, C. Chassot, S. Medjiah, and K. Drira, "Experimental comparison of the diagnostic capabilities of classification and clustering algorithms for the QoS management in an autonomic IoT platform," *Serv. Oriented Comput. Appl.*, vol. 13, no. 3, pp. 199–219, 2019.

[15] L. Morales, J. Aguilar, A. Rosales, J. A. G. de Mesa, and D. Chavez, "An Intelligent Controller based on LAMDA," in *2019 IEEE 4th Colombian Conference on Automatic Control (CCAC)*, 2019, pp. 1–6.

[16] J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Trans. Syst. Man. Cybern.*, vol. 23, no. 3, pp. 665–685, 1993.

[17] M. A. Denaï, F. Palis, and A. Zeghbib, "Modeling and control of non-linear systems using soft computing techniques," *Appl. Soft Comput. J.*, vol. 7, no. 3, pp. 728–738, 2007.

[18] J. S. R. Jang and Chuen-Tsai Sun, "Neuro-fuzzy modeling and control," *Proc. IEEE*, vol. 83, no. 3, pp. 378–406, Mar. 1995.