Robust Evolving Cloud-based Controller in Normalized Data Space for Heat-Exchanger Plant

Goran Andonovski, Sašo Blažič
Faculty of Electrical Engineering
University of Ljubljana, Slovenia
goran.andonovski@fe.uni-lj.si,
saso.blazic@fe.uni-lj.si

Plamen Angelov
School of Coputing and Communications
Lancaster University, United Kingdom
p.angelov@lancaster.ac.uk

Igor Škrjanc
Faculty of Electrical Engineering
University of Ljubljana, Slovenia
igor.skrjanc@fe.uni-lj.si

***Abstract*—This paper presents an improved version and a modification of Robust Evolving Cloud-based Controller (RECCo). The first modification is normalization of data space in RECCo. As a consequence, some of the evolving and adaptation parameters become independent of the range of the process output signal. Thus the controller tuning is simplified which makes the approach more appealing for the use in practical applications. The data space normalization is general and is used with Euclidean norm, but other distance metrics could also be used. Beside the normalization new adaptation scheme of the controller gain is proposed which improves the control performance in the case of a negative initial error in starting phase of the evolving process. At the end, different simulation scenarios are tested and analyzed for further practical implementation of the Cloud-based controller into real environments. For that reason a detail simulation study of a plate heat exchanger is performed and different scenarios were analyzed.**

I. Introduction

Variety of PID controllers are already proposed and traditionally widely used in industrial control systems. Because of their simple structure and reliable operation they are commonly used for linear processes, therefore they work well in certain conditions. In the case of nonlinear processes classical PID controllers are commonly insufficient. Moreover, new approaches and techniques attempt to improve the control in case of nonlinear systems [1] [2]. Fuzzy controllers represent an alternative approach for solving the nonlinearity process control problem. New type of fuzzy rule-based (FRB) system ANYA was introduced in [3]. The main difference between this method and classical FRB systems such as Mamdani [4], [5] and Takagi-Sugeno (TS) [6] is the way of how the antecedent part is defined. While in TS fuzzy systems the antecedent part is fuzzy and the consequent part is functional, in the Mamdani FRB systems both parts are fuzzy. Defuzzification in Mamdani and TS systems could be realized using center-of-gravity or "winers takes all" methods and their variations. ANYA is a simplified FRB system and does not require an explicit definition of fuzzy sets (and their corresponding membership functions) for each input variable. Simplified antecedent part in this case is non-parametric and is formed using data clouds (sets of data samples in the data space) that have no specific shape, parameters or boundaries. In addition, the antecedents of ANYA FRB take into account the density of all previous data samples and it is calculated recursively.

On-line construction and evolving of the fuzzy rule-based controller without prior knowledge of the process was presented in the previous work [7] [8]. The mentioned approaches allow the controller structure (the membership function, fuzzy rules, fuzzy set, etc.) to be created based on the data collected on-line. New data samples are used for updating existing rules (clouds) or for creating new ones if a significant change of the operating point is detected. The proposed method starts with empty structure of the controller and just a few parameters which are needed for adaptation and evolving phases. The first phase is adaptation of control parameters in the consequent part of fuzzy rules. During the evolving phase controller structure is modified if needed.

In this paper we introduce an improved approach of robust evolving cloud-based controller (RECCo) for dealing with nonlinear process control. One of the novelties is normalization of the data space. This makes the performance of the system less sensitive on some design parameters (parameters required in the consequent of the fuzzy rules and parameters of the evolving algorithm) making the system more robust. Another improvement proposed here is the way how the PID parameters are adapted in the starting phase. In order to show the effective performance and advantages of the proposed method, it is applied to a simulated problem of a heat-exchanger pilot plant.

The rest of this article is organized as follows. Section II describes the robust evolving cloud-based controller, its structure and adaptation method in subsections II-A and II-B, respectively. In section III new approach for cloud space normalization is proposed. Finally in section IV different simulation scenarios are analyzed and presented as a proof of concept for the proposed methodology. The main conclusions are summarized in section V.

II. Robust Evolving Cloud-Based Controller (RECCo)

A special feature of the RECCo is that no a priori information about the controlled process is needed. Theoretically, the controller could be initialized from the first data sample received. But of course, any existing information can be used

to suitably initialize the controller parameters. After this, for every incoming sample the controller gains are adapted and if the certain conditions are satisfied, a new cloud is created.

A. The structure of the cloud-based controller

In this subsection we present the robust cloud-based controller structure with non-parametric antecedents. As we already mentioned, this method applies the concept of fuzzy data clouds and relative data density to define antecedents. Each data element is associated on-line to one of the existing clouds (if current data is close enough according to a chosen similarity measure) or a new fuzzy rule (cloud) is created. The concept does not employ membership functions in the classical sense. Degree of fulfilment is based on the distances between samples and the corresponding cloud relative density. At this point we have to mention that already two different similarity measures were used: Euclidean [3] [9] and Mahalanobis [10] distances. In this paper a simpler Euclidean distance is used according to the fact that both methods produced satisfying results.

The structure of ANYA was initially introduced in [3]. The authors proposed simplified FRB system of the following form:

$$\mathcal{R}^i : \text{IF} \quad (\mathbf{x} \sim X^i) \quad \text{THEN} \quad (u^i) \tag{1}$$

where the operator $\sim$ denotes the fuzzy membership expressed linguistically as '*is associated with*'. The number of the rules is defined by the number of the data clouds c ($i = 1, 2, \ldots, c$), and usually changes with time. X^i denotes the i-th cloud in the data space where $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ is the controller's input and u^i denotes its output in the i-th fuzzy domain. The contribution of a particular controller output in a certain fuzzy domain to the actual control output is given by the normalized relative density:

$$\lambda_k^i = \frac{\gamma_k^i}{\sum\limits_{j=1}^{c} \gamma_k^j} \quad i = 1, \ldots, c \tag{2}$$

where γ_k^i is the local density of the i-th cloud for the current data $\mathbf{x}_k$.

When a new data sample arrives, we compute c separate densities γ_k^j that define how "close" a new sample is relative to the existing clouds. According to the calculated densities, the current data sample is finally associated with the cloud with maximal density and all the parameters of this particular cloud are updated.

In [9] and [8] the local density of the i-th cloud is defined by Cauchy kernel as follows:

$$\gamma_k^i = \frac{1}{1 + \frac{\sum_{j=1}^{k-1}(d_{kj}^i)^2}{k-1}} \tag{3}$$

where k is the current time stamp and $\sum_{j=1}^{k-1}(d_{kj}^i)^2$ is the sum of the square of Euclidean distances ($d_{kj}^i = \|x_k - x_j\|^2$) between the new data $\mathbf{x}_k$ and all the data points of the i-th cloud. Furthermore, (3) can be recursively written as follows:

$$\gamma_k^i = \frac{1}{1 + \|x_k - \mu_k^i\|^2 + \sigma_k^i - \|\mu_k^i\|^2} \tag{4}$$

where μ_k^i is the mean value of the cloud's data points and σ_k^i is the variance. Both of them can be recursively calculated using following equations for mean value and variance, respectively:

$$\mu_k^i = \frac{k-1}{k}\mu_{k-1}^i + \frac{1}{k}x_k \tag{5}$$

$$\sigma_k^i = \frac{k-1}{k}\sigma_{k-1}^i + \frac{1}{k}\|x_k\|^2 \tag{6}$$

Initial condition for the mean value is $\mu_1^i = x_1$ and for the variance is $\sigma_1^i = \|x_1\|^2$.

Once we classified the current data sample to one of the clouds and updated its properties we can do the defuzzification of the FRB system. As we said above, the ANYA FRB system can work with both Mamdani and TS consequents. Therefore, if we consider the weighted average for the defuzzification, the output of the controller becomes:

$$u_k = \sum_{i=1}^{c} \lambda_k^i u^i = \frac{\sum\limits_{i=1}^{c} \gamma_k^i u^i}{\sum\limits_{i=1}^{c} \gamma_k^i} \tag{7}$$

where u^i denotes the i-th rule consequent and normalized relative density (2) is used.

It is unreal to expect that any controller can immediately bring the output process value to the reference. Depending on the system dynamics we defined first order linear reference-model with corresponding time constant τ. We have to note here that larger time constant usually produce better results especially in terms of robustness at the cost of a lower speed. In our case our reference model is defined as:

$$y_{k+1}^r = a_r y_k^r + (1 - a_r) r_k \quad 0 < a_r < 1 \tag{8}$$

where r_k is the reference signal set by the operator. Parameter a_r is the pole of the first order discrete filter and defines the transient dynamics. For systems with fast sampling, a_r can be approximated by $(1 - \frac{T_s}{\tau})$ where T_s defines the sampling period. In all our experiments we used $\tau = 40$ and $a_r = 0.95$ which is an acceptable time constant for the dynamics of the studied heat-exchanger plant.

In this approach, the PID-based rule consequents of the following form are proposed:

$$u_k^i = P_k^i \varepsilon_k + I_k^i \Sigma_k^\varepsilon + D_k^i \Delta_k^\varepsilon + R_k^i \tag{9}$$

where $\varepsilon_k = y_k^r - y_{k-1}$ is the tracking error (the difference between the reference model and the process output). P_k^i, I_k^i, D_k^i are controller gains and R_k^i is compensation of the operating point for each cloud $i = 1, \ldots, c$. Discrete-time integral (Σ_k^ε) and discrete derivative (Δ_k^ε) of the tracking error ε_k are recursively computed as:

$$\Sigma_k^\varepsilon = \sum_{\kappa=0}^{k-1} \varepsilon_\kappa = \Sigma_{k-1}^\varepsilon + \varepsilon_{k-1} \tag{10}$$

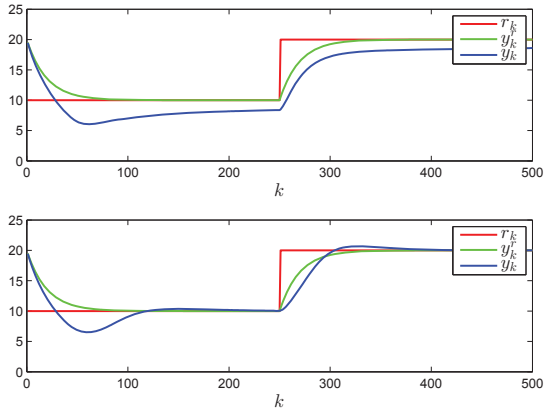

Fig. 1. The reference, model reference and the output signal for heat-exchanger pilot plant in the starting phase without calculating absolute value (upper plot) and with absolute value (lower plot) in adaptation of the controller gains ($y_o = 20\,^\circ C$).

$$\Delta_k^{\varepsilon} = \varepsilon_k - \varepsilon_{k-1} \tag{11}$$

B. Adaptation of the controller

An improved adaptation of the controller gains is proposed in this section, where the absolute value of error is used during starting phase of the simulation. New adaptation method proposed in (12) is used only in the starting phase (first 500 samples is enough) of the RECCo evolving system. Afterwards, the adaptation continues as originally proposed in [8] and [9], i.e., all the absolute values are omitted from (12). These absolute values prevent undesirable transient response of the system in the initial phase of the adaptation process. We can see the difference between calculating the adaptation with and without the absolute values in Fig. 1. It is obvious that in case of negative error in the initial phase (the current value is higher than the reference signal) the adaptation (12) produced far better results than the adaptation without absolute values. This is a real life scenario where the process has non-zero initial value. Even in the case of zero initial process value, the proposed adaptation of PID parameters preserves good results.

In the starting phase the controller gains are adapted as follows:

$$\begin{aligned}
\Delta P_k^i &= \alpha_P\, G_{sign} \lambda_k^i \left| \frac{e_k \varepsilon_k}{1 + r_k^2} \right| \\
\Delta I_k^i &= \alpha_I\, G_{sign} \lambda_k^i \left| \frac{e_k \Delta_k^{\varepsilon}}{1 + r_k^2} \right| \\
\Delta D_k^i &= \alpha_D\, G_{sign} \lambda_k^i \left| \frac{e_k \Delta_k^{\varepsilon}}{1 + r_k^2} \right| \\
\Delta R_k^i &= \alpha_R\, G_{sign} \lambda_k^i \frac{\varepsilon_k}{1 + r_k^2} \\
i &= 1, \ldots, c
\end{aligned} \tag{12}$$

where $e_k = r_k - y_{k-1}$ is the difference between the reference and the process output and λ_k^i is a normalized relative density. The constants $\alpha_P, \alpha_I, \alpha_D, \alpha_R$ are the adaptation gains. Furthermore, the controller gains vector is defined as $\theta_k^i = \left[P_k^i, I_k^i, D_k^i, R_k^i\right]^T$ and the adaptation of parameters is defined as follows:

$$\theta_k^i = \theta_{k-1}^i + \Delta\theta_k^i \tag{13}$$

The adaptation of the controller gains (13) is done only for the current active cloud while the others are kept constant.

When dealing with adaptive control algorithms one needs to have in mind the potential instability of the system can occur [11]. There exist many known approaches that make adaptive laws more robust [12], [13]. In [8] and [9] several supervisory mechanisms were included in the adaptive law to prevent parameter drift and instability. In this article we will employ the same ones (dead zone d_{dead} in the adaptive law, parameter projection $\left[\underline{\theta}, \overline{\theta}\right]$, leakage σ_L in the adaptive law and interruption of adaptation $[u_{min}, u_{max}]$). The general idea behind the dead zone in the adaptive law is that the adaptation is simply switched off if the absolute value of the error is small [9]. In this article the output of the system is normalized and therefore we propose a fixed setting for the dead zone parameter (d_{dead} is related to the process range, in our case 1% of Δr). Other supervisory mechanisms of the adaptive law are the same as were proposed in [8] and [9].

III. Cloud space normalization

In the previous papers [9] [10] [8], the input data were chosen as $\mathbf{x} = \left[\ \varepsilon_k,\ y_k^r\ \right]^T$. Consequently, the calculated densities depend on the magnitude of the input data. This heavily affects the evolving part of the algorithm, e.g., the density where a new cloud is born (denoted by γ_{max}) has to be evaluated for different process ranges separately.

In this section a normalization of the cloud space is proposed:

$$x = \left[\ \frac{\varepsilon_k}{\Delta\varepsilon},\ \frac{y_k^r - r_{min}}{\Delta r}\ \right]^T \tag{14}$$

where $\Delta r = r_{max} - r_{min}$ and Δr depends on the operating system area and $\Delta\varepsilon = \frac{\Delta r}{2}$. In this case we are mostly interested in the region where we expect the majority of the data samples. Operator needs to choose, according to the process requirements, only two parameters r_{min} and r_{max}. The normalization transforms the cloud-space into the space of the constant size, and this allows us to fix some parameters, e.g., γ_{max}. In all our simulation studies and examples this value was constant and it is independent of the operating process range. Some other parameters such as the PID (initial) gains are also much easier to tune with normalized process values.

In Table I three different scenarios are presented and in each case the same value of parameter $\gamma_{max} = 0.93$ is used. Generated clouds are presented in Figs. 2, 3 and 9 for each of the scenarios (rows in Table I), respectively.

TABLE I
SIMULATION SCENARIOS WITH DIFFERENT PROCESS RANGE

	Reference range	y_o	γ_{max}
1.	$r_{min} = 5\,^\circ C$ $r_{max} = 35\,^\circ C$	$12\,^\circ C$	0.93
2.	$r_{min} = 20\,^\circ C$ $r_{max} = 45\,^\circ C$	$22\,^\circ C$	0.93
3.	$r_{min} = 10\,^\circ C$ $r_{max} = 50\,^\circ C$	$15\,^\circ C$	0.93

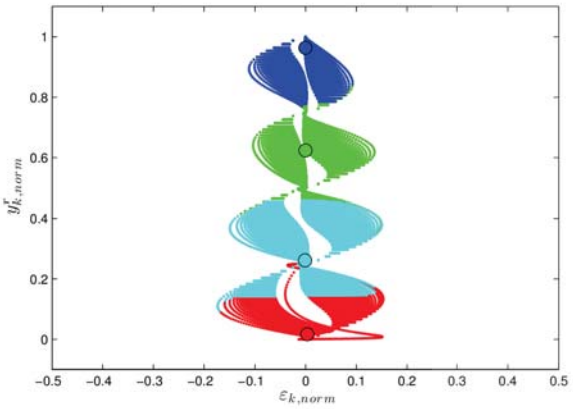

Fig. 2. The clouds in the input space $\mathbf{x} = \left[\frac{\varepsilon_k}{\Delta\varepsilon}, \frac{y^r_k - r_{min}}{\Delta r}\right]^T$ and ($r_{min} = 5\,^\circ C$, $r_{max} = 35\,^\circ C$, $y_o = 12\,^\circ C$).

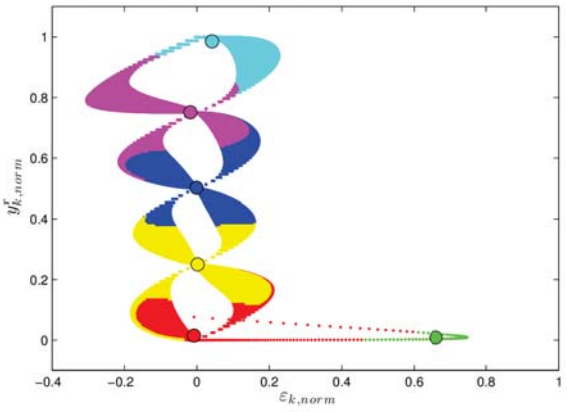

Fig. 3. The clouds in the input space $\mathbf{x} = \left[\frac{\varepsilon_k}{\Delta\varepsilon}, \frac{y^r_k - r_{min}}{\Delta r}\right]^T$ and ($r_{min} = 20\,^\circ C$, $r_{max} = 45\,^\circ C$, $y_o = 22\,^\circ C$).

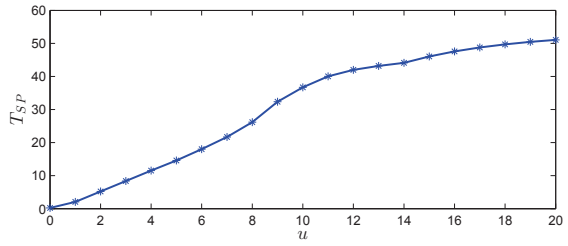

Fig. 4. Static characteristic of the heat-exchanger plant over whole process range.

IV. SIMULATION STUDY OF A HEAT-EXCHANGER PILOT PLANT

As we already mention above, in our experiment a model of heat exchanger (HE) process is used; it is explained in more detail in [14]. The heat-exchanger process consists of two separate water circuits (heating and cooling water). First circuit has a constant inlet temperature $T_{ec}(k)$ and motor driven valve to control the primary circuit flow $F_c(k)$ and represents the control process variable. The outlet water is returned to the reservoir which is electrically heated. The second circuit has inlet temperature $T_{ep}(k)$ and the water flow of cold water $F_p(k)$ on one side and controlled variable outlet temperature $T_{sp}(k)$ on the other side. This process is described by the following differential equation [14] :

$$\tau_2(T_{sp})\dot{T}_{sp} + T_{sp} = \delta T_{ep} + (1-\delta)T_{ec} \tag{15}$$

where the generalized formula of δ can be written as

$$\delta = \frac{1 + k_c(\frac{1}{F_c})^m}{1 + k_c((\frac{1}{F_c})^m + (\frac{1}{F_p})^m)} \tag{16}$$

where k_c and m are unknown constants and τ_2 is an unknown function of operating point. In [14] a fuzzy model of the first-order dynamics is given where small delay of the process is neglected. This model of heat-exchanger has nonlinear static characteristic (Fig. 4) and the reference signal is chosen to cover the whole range and to show the ability of learning.

Advantage of the RECCo is that we do not need a priori knowledge of the process; furthermore, the controller structure and membership functions are adapted and evolved during the simulation process which started without any fuzzy rules. All the simulations in this paper are started using the same initial conditions and the same values for all the parameters. These are divided into three groups (process, adaptive and evolving parameters). First group contains very intuitive parameters which are set according to the user requirements. The process range in this simulations was chosen as $r_{min} = 10\,^\circ C$ and $r_{max} = 50\,^\circ C$. We mentioned above that in the case of a different process range, no additional tuning of adaptive and evolving parameters is required. The time constant and the sampling time of the reference-model were chosen as $\tau = 40$ and $T_s = 2$, respectively. The process input or the actuator's range was given as $u_{min} = 0$ and $u_{max} = 20$. In the second group, the parameters of the adaptive laws are found. The dead

zone d_{dead} was chosen as 1% of process range Δr and the leakage parameter is set to $\sigma_L = 10^{-6}$. All adaptive gains $\alpha_P, \alpha_I, \alpha_D$ and α_R are set to 0.1. The evolving parameters from the third group define the rules when, why and "how many" clouds will be crated. Simulations were started with zero fuzzy rules (clouds) and maximal number of clouds was $c_{max} = 100$, which was not the case in any of the simulations. A new cloud was added when the maximal value of the local densities $\gamma_k^i,\ i = 1, \ldots, c$ is lower than the predefined value $\gamma_{max} = 0.93$. The parameter that defines the minimum time between two new clouds is defined as $n_{add} = 20$.

The simulation study presented in this section is divided into three groups of experiments:

1) The experiments where the effect of cloud space normalization is analyzed.
2) The effects of input disturbances and noise on the process output are examined.
3) A detailed study of the evolving learning method for the heat-exchanger plant is done.

In Table II the advantage of the normalization of the cloud space is presented. Our goal is to generate rational number of clouds for each case/scenario. Four different reference ranges were chosen and without normalization we need to estimate the value of parameter γ_{max} for each scenario separately (third column in Table II). With the normalization the value of the evolving parameter γ_{max} is constant (fourth column in Table II).

TABLE II
COMPARASION BETWEEN ORIGINAL AND NORMALIZED CLOUD SPACE

	Reference range	γ_{max}	$\gamma_{max,norm}$	Number of generated clouds
1.	$r_{min} = 10\,^\circ C$ $r_{max} = 50\,^\circ C$	0.019	0.93	5
2.	$r_{min} = 5\,^\circ C$ $r_{max} = 25\,^\circ C$	0.032	0.93	4
3.	$r_{min} = 4\,^\circ C$ $r_{max} = 20\,^\circ C$	0.046	0.93	4
4.	$r_{min} = 2\,^\circ C$ $r_{max} = 10\,^\circ C$	0.221	0.93	4

Next, we studied the effect of input disturbance added to the control signal and output noise with characteristics $\mathcal{N}(0, 0.1)$. The results are shown in Figs. 5 (in the starting phase of the adaptation) and 6 (after the transient of the adaptation). In both figures the disturbance was added to the control signal in the middle of the visible time window (the size of the disturbance: 2.5 units in magnitude and 50 time stamps in duration). We can see that evolving algorithm adapts quickly to the new circumstances and is not sensitive to input disturbance and output noise.

In the case of the third scenario (the third row in Table I) more aspects will be shown in the paper where some new improvements (normalization and adaptation) of robust evolving cloud-based controller were tested in this section. Efficient learning and evolving of the RECCo is presented during next several figures. A close look to the starting phase

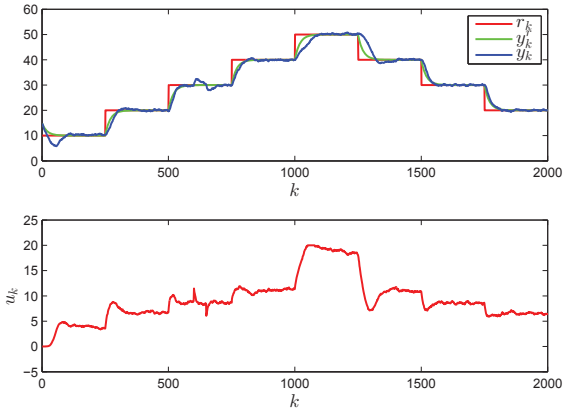

Fig. 5. The effect of the input disturbance and output process noise in the starting phase of the adaptation.

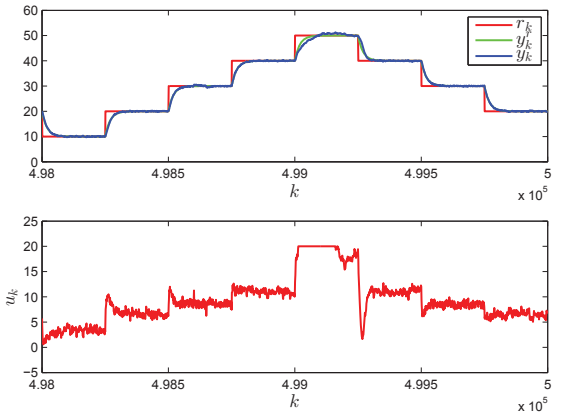

Fig. 6. The effect of the input disturbance and output process noise after the transient of the adaptation.

of simulation is shown in Fig. 7 where the reference, the model reference, the output, and the control signal are given. At this point we want to notice that in this simulation a new adaptation method is used (mentioned in subsection II-B). In Fig. 8 the phase after the transient of the adaption is shown. An important aspect of adapting and learning method proposed in this article is the convergence of the adaptive gains shown in Fig. 10 for all the generated clouds during simulations. We have to note here that a new cloud is created and initialized with the mean value of previous cloud's parameters. In Fig. 9 all the created clouds during the simulation are shown. In this case we created five clouds. The tracking error is shown in Fig. 11 where its decreasing with time can clearly be noticed.

V. CONCLUSION

In this paper a new approach of RECCo with normalized data space and improved adaptation of controller parameters is proposed. The normalization of the data space is used for determination of the evolving and the adaptation parameters.

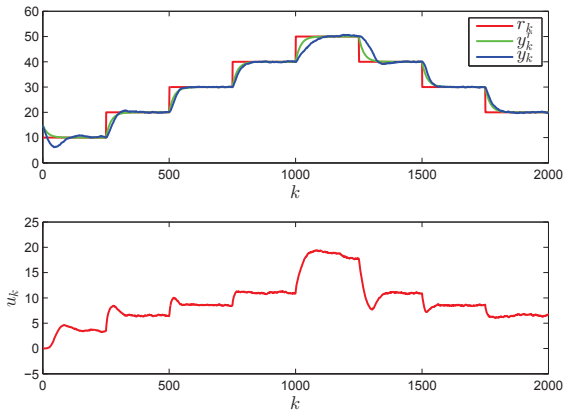

Fig. 7. The reference, model reference and output signal tracking and the control signal for plate heat exchanger in the starting phase ($y_o = 15\,^\circ C$).

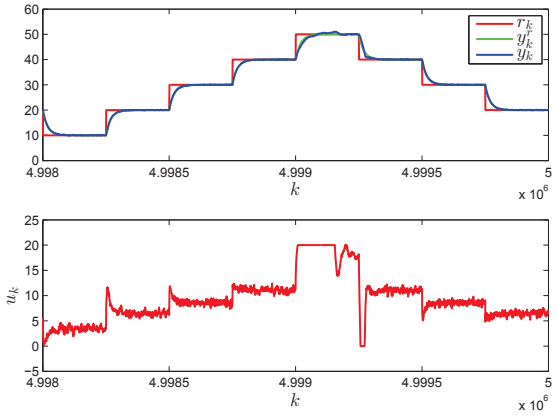

Fig. 8. The reference, model reference and output signal tracking and the control signal for plate heat exchanger in the finishing phase ($y_o = 15\,^\circ C$).

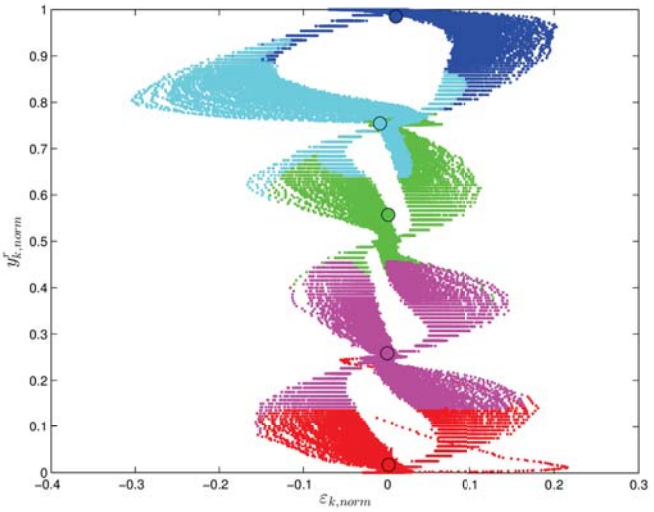

Fig. 9. The clouds in the input space $\mathbf{x} = \left[\frac{\varepsilon_k}{\Delta\varepsilon}, \frac{y_k^r - r_{min}}{\Delta r}\right]^T$ and ($r_{min} = 10\,^\circ C$, $r_{max} = 50\,^\circ C$, $y_o = 15\,^\circ C$).

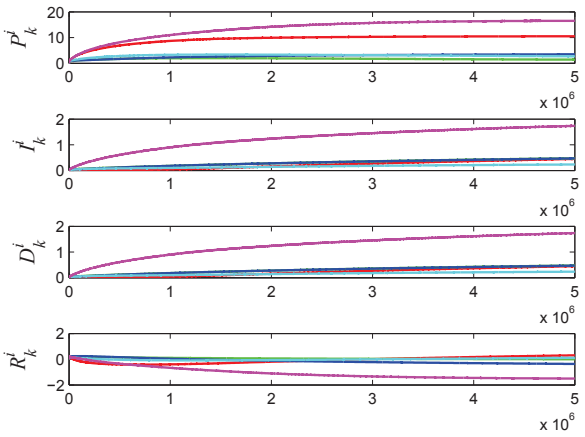

Fig. 10. The evolving parameters of PID controller for each cloud ($y_o = 15\,^\circ C$).

In case of the negative initial process error a new adaptation of the controller is proposed for the starting phase of the evolving process. Both modifications, normalization and adaptation, are thoroughly tested and analyzed during different simulation scenarios for heat-exchanger pilot plant. Irrespective of the process range, the same initial values of the parameters are used in all the simulations which is one of the most valuable benefits of the proposed modifications. The effect of output noise and input disturbances was also analyzed to test the robustness of the controller. The main advantages of the RECCo controller are the self-evolving procedure which starts with no a priori knowledge and effectively deals with nonlinear processes.

REFERENCES

[1] W.-D. Chang, R.-C. Hwang, and J.-G. Hsieh, "A self-tuning pid control for a class of nonlinear systems based on the lyapunov approach," *Journal of Process Control*, vol. 12, pp. 233–242, 2002.

[2] Y. Kansha, L. Jia, and M.-S. Chiu, "Self-tuning {PID} controllers based on the lyapunov approach," *Chemical Engineering Science*, vol. 63, no. 10, pp. 2732 – 2740, 2008.

[3] P. Angelov and R. Yager, "Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density," in *Evolving and Adaptive Intelligent Systems (EAIS), 2011 IEEE Workshop on*, April 2011, pp. 62–69.

[4] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *Int. J. Hum.-Comput. Stud.*, vol. 51, no. 2, pp. 135–147, Aug. 1999. [Online]. Available: http://dx.doi.org/10.1006/ijhc.1973.0303

[5] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-3, no. 1, pp. 28–44, Jan 1973.

[6] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-15, no. 1, pp. 116–132, Jan 1985.

[7] P. Angelov, R. Buswell, J. A. Wright, and D. Loveday, "Evolving rules-based control," in *EUNITE Symposium*, December 2001, pp. 36–41.

[8] P. Angelov, I. Škrjanc, and S. Blažič, "Robust evolving cloud-based controller for a hydraulic plant," in *Evolving and Adaptive Intelligent Systems (EAIS), 2013 IEEE Conference on*, April 2013, pp. 1–8.

[9] I. Škrjanc, S. Blažič, and P. Angelov, "Robust evolving cloud-based pid

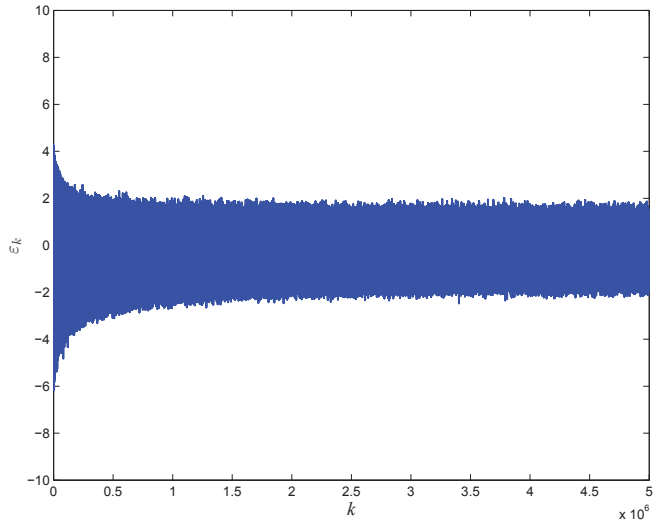

Fig. 11. The tracking error ε_k for heat-exchanger pilot plant ($y_o = 15\,^\circ C$).

control adjusted by gradient learning method," in *Evolving and Adaptive Intelligent Systems (EAIS), 2014 IEEE Conference on*, June 2014, pp. 1–8.

[10] S. Blažič, D. Dovžan, and I. Škrjanc, "Cloud-based identification of an evolving system with supervisory mechanisms," in *Intelligent Control (ISIC), 2014 IEEE International Symposium on*, Oct 2014, pp. 1906–1911.

[11] C. Rohrs, L. Valavani, M. Athans, and G. Stein, "Robustness of continuous-time adaptive control algorithms in the presence of unmodeled dynamics," *Automatic Control, IEEE Transactions on*, vol. 30, no. 9, pp. 881–889, Sep 1985.

[12] S. Blažič, I. Škrjanc, and D. Matko, "A new fuzzy adaptive law with leakage," in *Evolving and Adaptive Intelligent Systems (EAIS), 2012 IEEE Conference on*, May 2012, pp. 47–50.

[13] ——, "Globally stable direct fuzzy model reference adaptive control," *Fuzzy Sets and Systems*, vol. 139, no. 1, pp. 3 – 33, 2003.

[14] I. Škrjanc and D. Matko, "Predictive functional control based on fuzzy model for heat-exchanger pilot plant," *Fuzzy Systems, IEEE Transactions on*, vol. 8, no. 6, pp. 705–712, Dec 2000.