Online Mixture of Univariate Linear Regression Models for Adaptive Soft Sensors

Francisco Souza, *Student Member, IEEE,* and Rui Araújo, *Member, IEEE*

***Abstract*—This paper proposes a mixture of univariate linear regression models (MULRM) to be applied in time-varying scenarios, and its application to soft sensor problems. Offline and online solutions of MULRM will be obtained using the Expectation-Maximization Algorithm. A forgetting factor will be introduced in the online solution to discount the information of already learned data, so that it can be applied in time varying settings. The solution of the proposed method allows its online and recursive application in any regression problem, without the necessity of storing any past value of data. The recursive solution of the MULRM will then be applied in two time-varying real-world prediction problems. The proposed method is compared with four state of art algorithms. In all the experiments, the proposed method always exhibits the best prediction performance.**

***Index Terms*—adaptive soft sensors, prediction, regression, expectation-maximization, mixture of models, univariate models**

I. Introduction

MULTIVARIATE statistical techniques are being widely employed in industry, where their most common applications are in the soft sensors context. Basically, the soft sensor is a tool which integrates the available information coming from the hardware sensors and laboratory analysis into multivariate techniques to perform a specific task, such as the prediction of critical process variables, process monitoring and other tasks which are related to process control [1]–[6].

Most soft sensor applications concern the prediction of critical variables that cannot be determined directly with hardware sensors, and can be determined only by laboratory analysis, thus lacking the information of what happens in real time on such variables. In this context, a soft sensor comes as a tool which enables the acquisition of critical variables in real-time, by simulating a hardware sensor. This is done by learning a regression model using the sensors of the process as input of the model (easy-to-measure variables), and the critical variable as the target (hard-to-measure variable) [7]–[9].

Manuscript received on January 09, 2013. Accepted for publication September 16, 2013.

Francisco Souza, and Rui Araújo; Institute for Systems and Robotics (ISR-UC), and Department of Electrical and Computer Engineering (DEEC-UC), University of Coimbra, Pólo II, P-3030-290 Coimbra, Portugal. (e-mails: fasouza@isr.uc.pt, rui@isr.uc.pt).

Francisco Souza has been supported by Fundação para a Ciência e a Tecnologia (FCT) under grant SFRH/BD/63454/2009. The authors acknowledge the support of FCT project PEst-C/EEI/UI0048/2013, and Project SCIAD/2011/21531 co-financed by QREN, in the framework of the "Mais Centro - Regional Operational Program of the Centro", and by the European Union through the European Regional Development Fund (ERDF).

Soft sensor development has four main steps [1], [10]: 1) data acquisition and selection of historical data; 2) data pre-processing; 3) model selection, training and validation; 4) soft sensor maintenance. In the first stage, data is selected for training and evaluation of the model. Then data is submitted to pre-processing. The goals of this second stage are the handling of missing data and outliers and, if necessary the selection of most relevant variables. The model selection, training and validation phase requires the correct selection and learning of the model, so that it can correctly reproduce the hard-to-measure variable. The last step is soft sensor maintenance, where the goal is to maintain a good soft sensor response under the presence of process variations or some data change.

The mostly used models in soft sensors applications are based on multivariate statistical methods or artificial intelligence techniques [10], most of them use the multiple linear regression model (MLRM) of the form $f_L(\mathbf{x}, \boldsymbol{\theta}) = \theta_0 + \sum_j \theta_j x_j$, with Least Squares (LS) or Partial Least Squares (PLS) estimation methods [11], [12], Principal Component Analysis (PCA) [13] in combination with a prediction model, Artificial Neural Networks (ANN) [1], [14], or Fuzzy or Neuro-Fuzzy Systems (NFS) [15]–[17].

In time-varying processes, a good soft sensor maintenance strategy is necessary to maintain a good response over time. Generally, this is done by updating the soft sensor model online/recursively, in batch or sample wise mode, using the incoming samples of the process (in this context the soft sensors are called "adaptive soft sensors" [3]). Usually, a forgetting factor $0 < \lambda \leq 1$ is employed in the soft sensor model learning approach, such that the model could capture the information of the recent data [3], [18].

The most established and popular method for adaptive soft sensors is the PLS algorithm, with many applications in literature [3], [4], [19]–[27]. The PLS solution is the preferred and mostly applied solution of MLRM when comparing to LS, since it can handle data-collinearity, which is a common characteristic in industrial applications. When it comes to adaptive soft sensors, they are implemented with recursive online forms of the above methods, such as in recursive LS (RLS) [11], [12], and the recursive PLS (RPLS) [28].

Looking to reduce the dimensionality of the learning process, this work proposes a mixture of univariate linear regression models (MULRM), where the individual models are combined such that the prediction error is minimized.

Specifically, the MULRM has the following form:

$$f_M(\mathbf{x}, \boldsymbol{\Upsilon}, \boldsymbol{\Theta}) = \sum_{j=1}^{D} \upsilon_j f_j(x_j, \boldsymbol{\theta}^{(j)}), \text{ with: } \sum_{j=1}^{D} \upsilon_j = 1, \quad (1)$$

where $f_j(x_j, \boldsymbol{\theta}^{(j)}) = \theta_0^{(j)} + \theta_1^{(j)} x_j$ is a univariate linear regression model of variable x_j, $\boldsymbol{\theta}^{(j)} = [\theta_0^{(j)}, \theta_1^{(j)}]^T$, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(j)} | \, j = 1, \ldots, D\}$ denotes the set of all weight parameters, and $\boldsymbol{\Upsilon} = \{\upsilon_j | \, j = 1, \ldots, D\}$ denotes the set of mixing coefficients. The individual models are then combined, and the Expectation-Maximization (EM) algorithm [29], [30] is employed to jointly estimate the model parameters $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Theta}$. The recursive solution for the MULRM parameters, $\boldsymbol{\Theta}, \boldsymbol{\Upsilon}$ will be derived in the next sections. A forgetting factor will be introduced in the online solution to discount the information coming from the already learned data, so that it can be applied in time varying scenarios. The solution of the proposed method allows its online and recursive application in any regression problem, without the necessity to store any past value of data.

In the experimental part, the recursive solution of the MULRM is then applied in two time-varying real-world prediction problems: a polymerization data set [21], and a problem for fluorine estimation in a wastewater treatment system. Moreover, the proposed MULRM method is compared in these problems with four state of art algorithms: the RLS, the RPLS, the online sequential extreme learning machine (OS-ELM) [14], a fast learning algorithm for single layer feedforward ANN, with offline and online solutions and the recently proposed incremental local learning soft sensing (ILLSA) algorithm for adaptive soft sensors [21]. Regarding the comparison with the remaining methods, the PLS and the PCR in combination with a linear model have similar prediction accuracy in most of the cases [31]. The NFS are widely applied for prediction [10], [16], but their parameters are usually learned offline. Online tuning of NFS can be done by Evolving Fuzzy Systems (EFS) [32]. However, the implementation of such methods is very complex and time demanding. By this reason, this work will not compare the proposed approach with the NFS.

All the methods (RLS, RPLS, ILLSA, OS-ELM and MULRM) were tested for different percentages of availability of target data. The experimental results suggest that the recursive MULRM outperforms the RLS, RPLS, OS-ELM and ILSSA, when predicting in time-varying scenarios.

The paper is organized as follows. Section II presents background and notation. The proposed method is presented in Section III. Section IV presents experimental results. Finally, Section VI gives concluding remarks.

II. Background

The notation used here is defined as follows, $\mathbf{x}(i) = [x_1(i), \ldots, x_D(i)]^T$ and $y(i)$ are the vector of input variables and the output target at instant i, $\mathbf{X}$, with elements $X_{ij} = x_j(i)$, and $\mathbf{y}$, with elements $y_{i,1} = y(i)$ are the input matrix and output vector containing all the k examples. Moreover, $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_D$, and $\mathcal{Y}$, denote the space of input variables values and the space of output values, respectively, where $\mathcal{X} \subset \mathbb{R}^D$ and $\mathcal{Y} \subset \mathbb{R}$. A subscript k will be used to denote the value of the corresponding variable after k samples.

In regression tasks, the objective is to make use of an input vector $\mathbf{x}$ to describe/approximate a target variable y, where a set of examples $\Phi = \{(\mathbf{x}(i), y(i)); i = 1, \ldots, k\}$ is used to train a model to do this. It is assumed that y can be approximated by a deterministic function $\hat{y} = f(\mathbf{x}, \boldsymbol{\theta})$, governed by a parameter vector $\boldsymbol{\theta}$, so that:

$$y = f(\mathbf{x}, \boldsymbol{\theta}) + \epsilon, \quad (2)$$

where ϵ, the approximation error, is a zero mean Gaussian random variable with variance ω. Under the assumption of Gaussian noise, the conditional probability of y given the input $\mathbf{x}$ can be represented by the normal distribution $p(y|\mathbf{x}, \boldsymbol{\theta}, \omega) = \mathcal{N}(y|f(\mathbf{x}, \boldsymbol{\theta}), \omega)$.

III. Mixture of Univariate Linear Regression Models

In this section, the formulas for the offline and online learning of the MULRM will be derived. In the offline solution, the parameters in (1) are found using a set of observations Φ. In the online solution the parameters in (1) are learned online and updated recursively when a new sample data becomes available. In the online learning, an exponential sample weighting strategy on the samples will be used to update the model parameters in (1), so that the model can represent the most recent data, allowing its application in time-varying scenarios.

Regarding the learning of parameters using exponential weighting of samples [28], it is important to note that the amount of data used to construct the multivariate model, represented here as $p(y|\mathbf{x})$, (i.e. the effective number of observations being used, defined as d), is related to the forgetting factor λ as $d = 1/(1-\lambda)$. Then, assuming that d plays a similar role as the number of training samples in the static case, it can be concluded that in time-varying scenarios small values of d (or small values of λ) lead to the same problems faced by learning a static model with small number of training samples, such as overfitting or poor generalization on the test set [33], mainly if $d < D$. Thus, from this assumption, then during learning in time-varying scenarios, the value of λ not only interferes in the speed of the model adaptation, but it also has influence in the model learning (i.e. in the recursive parameters learning) and its problems.

Then, the motivation behind the use of MULRM in time-varying scenarios, is that this approach has the benefit of separately estimating the pdf of y conditioned to each individual input variable, $p(y|x_j)$, rather than estimating the conditional pdf $p(y|\mathbf{x})$ with all variables. Normally, such estimate of $p(y|x_j)$ is more accurate than the estimate of $p(y|\mathbf{x})$ [34], mainly when d is small or $d < D$. The amount of data needed to obtain an accurate estimate increases with the dimensionality of the problem. Then, in some cases $p(y|x_j)$ can be estimated more reliably than $p(y|\mathbf{x})$. Moreover, this dimensionality reduction, while estimating $p(y|x_j)$, makes the learning problem much easier. However, for representing y, $p(y|x_j)$ is less representative than $p(y|\mathbf{x})$. To overcome this

effect, MULRM merges all the individual $p(y|x_j)$ models to estimate $p(y|\mathbf{x})$.

Moreover, due to the characteristics discussed above, the MULRM can be considered a good option for prediction in two possible scenarios. The first is when the input space is larger than the number of samples; i.e. in situations where $k < D$ (some regularization techniques such as elastic net [35] can also be employed in this case). The second situation (which is the main target of this work), is in soft sensor applications in time-varying scenarios. In these later scenarios, the forgetting factor λ employed to weight the samples is related to the effective number of samples by $d = 1/(1-\lambda)$, and the value chosen for λ can be small, leading to $d < D$, i.e. MULRM is a good option when the effective number of samples is less than the number of input variables.

A. MULRM - Offline Solution

The MULRM is based on the additive assumption [36]. More specifically, it is assumed that $f(\mathbf{x}, \boldsymbol{\theta})$ in (2) is approximated by (1). In the Bayesian framework, the MULRM approximates the true pdf $p(y|\mathbf{x})$ with the following superposition of individual pdfs:

$$p(y|\mathbf{x}, \boldsymbol{\Theta}, \boldsymbol{\Upsilon}, \boldsymbol{\Omega}) = \sum_{j=1}^{D} \upsilon_j p(y|f_j(x_j, \boldsymbol{\theta}^{(j)}), \boldsymbol{\Omega}), \tag{3}$$

where $p(y|f_j(x_j, \boldsymbol{\theta}^{(j)}), \boldsymbol{\Omega})$ is the pdf describing y given x_j, with mean $f_j(x_j, \boldsymbol{\theta}^{(j)})$ and additional pdf parameters $\boldsymbol{\Omega}$. $f_j(x_j, \boldsymbol{\theta}^{(j)}) : \mathcal{X}_j \to \mathcal{Y}$ is a linear model from input variable x_j, $\boldsymbol{\theta}^{(j)}$ is the vector of parameters of model j, υ_j is the mixing coefficient for model j ($j = 1, \ldots, D$), and $\sum_{j=1}^{D} \upsilon_j = 1$. From (3), prediction equation (1) is obtained as the following conditional mean of y given $\mathbf{x}$ [30, ch. 1], yielding a mixture of linear univariate models:

$$\begin{aligned} E[y|\mathbf{x}] &= f_M(\mathbf{x}, \boldsymbol{\Theta}, \boldsymbol{\Upsilon}) \\ &= \int y p(y|\mathbf{x}, \boldsymbol{\Theta}, \boldsymbol{\Upsilon}, \boldsymbol{\Omega}) \mathrm{d}y \\ &= \int y \sum_{j=1}^{D} \upsilon_j p(y|f_j(x_j, \boldsymbol{\theta}^{(j)}), \boldsymbol{\Omega}) \mathrm{d}y \\ &= \sum_{j=1}^{D} \upsilon_j f_j(x_j, \boldsymbol{\theta}^{(j)}). \end{aligned} \tag{4}$$

$f_M(\mathbf{x}, \boldsymbol{\Theta}, \boldsymbol{\Upsilon})$ is the function which minimizes the expected squared loss $E = \int\int \left(f_M(\mathbf{x}, \boldsymbol{\Theta}, \boldsymbol{\Upsilon}) - y\right)^2 p(\mathbf{x}, y) d\mathbf{x} dy$ [30].

In this paper, it is assumed that each individual pdf $p(y|f_j(x_j, \boldsymbol{\theta}^{(j)}), \boldsymbol{\Omega})$ in (3) is described by a Gaussian distribution. Then, (3) can be rewritten as:

$$p(y|\mathbf{x}, \boldsymbol{\vartheta}) = \sum_{j=1}^{D} \upsilon_j \mathcal{N}(y|f_j(x_j, \boldsymbol{\theta}^{(j)}), \omega^{(j)}), \tag{5}$$

where $\omega^{(j)}$ is the variance of model j, and $\boldsymbol{\vartheta}$ denotes the set of all adaptive parameters in (5). Specifically, $\boldsymbol{\vartheta}$ includes the parameters of $\boldsymbol{\Theta}$, $\boldsymbol{\Upsilon}$, and $\boldsymbol{\Omega} = \{\omega^{(j)} | \, j = 1, \ldots, D\}$. From (5), the log likelihood of a given set of observations Φ is given by:

$$\begin{aligned} &\ln p(\mathbf{y}|\mathbf{X}, \boldsymbol{\vartheta}) \\ &\quad = \ln \left(\prod_{i=1}^{k} p(y(i)|\mathbf{x}(i), \boldsymbol{\vartheta}) \right) \\ &\quad = \sum_{i=1}^{k} \ln \left(\sum_{j=1}^{D} \upsilon_j \, \mathcal{N}\left(y(i) \middle| f_j\big(x_j(i), \boldsymbol{\theta}^{(j)}\big), \omega^{(j)}\right) \right). \end{aligned} \tag{6}$$

In order to maximize the likelihood function (6) the Expectation-Maximization (EM) algorithm will be employed. The EM algorithm is a general method for finding the maximum-likelihood estimate of the parameters of an underlying distribution from a given data set when the data has latents, unknown variables [29], [30]. In this work, a set of latent variables $\mathcal{Z} = \{z_j(i) | \, j = 1, \ldots, D, \, i = 1, \ldots, k\}$ is defined, where $z_j(i) \in \{0, 1\}$, and for each sample i, all variables $z_j(i)$ are zero, except for a single value of $z_j(i) = 1$, for some j. The hidden variable $z_j(i)$ indicates which model j is responsible for generating the data sample i. In this context, the log likelihood of the complete joint conditional distribution with the observed and latent variables is given by [30, ch. 9]:

$$\begin{aligned} &\ln p(\mathbf{y}, \mathcal{Z}|\mathbf{X}, \boldsymbol{\vartheta}) \\ &\quad = \sum_{i=1}^{k} \sum_{j=1}^{D} z_j(i) \ln \left(\upsilon_j \, \mathcal{N}\left(y(i) \middle| f_j\big(x_j(i), \boldsymbol{\theta}^{(j)}\big), \omega^{(j)}\right) \right). \end{aligned} \tag{7}$$

The EM algorithm starts with an initial value for the model parameters $\boldsymbol{\vartheta}$, called here as $\boldsymbol{\vartheta}^{\text{old}}$. Then, $\boldsymbol{\vartheta}^{\text{old}}$ is used to compute the responsibility of model f_j and sample i, $\gamma_j(i) = E[z_j(i)]$, which accounts for the probability of model j in generating the data sample i. Using Bayes's Theorem [30]:

$$\begin{aligned} \gamma_j(i) &= E[z_j(i)] \\ &= p\left(z_j(i) = 1 | f_j\big(x_j(i), \boldsymbol{\theta}^{(j)}\big), \boldsymbol{\vartheta}^{\text{old}}\right), \\ &= \frac{\upsilon_j \, \mathcal{N}\left(y(i) \middle| f_j\big(x_j(i), \boldsymbol{\theta}^{(j)}\big), \omega^{(j)}\right)}{\sum_{d=1}^{D} \upsilon_d \, \mathcal{N}\left(y(i) \middle| f_d\big(x_d(i), \boldsymbol{\theta}^{(d)}\big), \omega^{(d)}\right)}. \end{aligned} \tag{8}$$

Then, the responsibilities (8) are used to determine the expectation (E) of the complete data log likelihood (7) with respect to $\mathcal{Z}$, which is equal to:

$$\begin{aligned} Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}}) &= E_{\mathcal{Z}}\big(\ln p(\mathbf{y}, \mathcal{Z}|\mathbf{X}, \boldsymbol{\vartheta})\big) \\ &= \sum_{i=1}^{k} \sum_{j=1}^{D} \gamma_j(i) \Big[\ln(\upsilon_j) \\ &\qquad + \ln\left(\mathcal{N}\left(y(i)|f_j(x_j(i), \boldsymbol{\theta}^{(j)}), \omega^{(j)}\right)\right)\Big]. \end{aligned} \tag{9}$$

In the maximization step, function $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$ is maximized with respect to $\boldsymbol{\vartheta}$, using the responsibilities (8) computed in the expectation step. The parameters in $\boldsymbol{\vartheta}$ are $\boldsymbol{\theta}^{(j)}$, υ_j, and $\omega^{(j)}$ for all models f_j. Then, maximizing $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$ with respect to $\boldsymbol{\vartheta}$, i.e. solving equations $(\partial Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}}))/(\partial \boldsymbol{\theta}^{(j)}) = 0$,

$(\partial Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}}))/(\partial \upsilon_j) = 0$, and $(\partial Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}}))/(\partial \omega^{(j)}) = 0$, gives the following maximizing solution parameters:

$$\boldsymbol{\theta}^{(j)} = ((\mathbf{A}^{(j)})^T \boldsymbol{\Gamma}^{(j)} (\mathbf{A}^{(j)}))^{-1} (\mathbf{A}^{(j)})^T \boldsymbol{\Gamma}^{(j)} \mathbf{y}, \tag{10}$$

$$\upsilon_j = \frac{\sum_{i=1}^{k} \gamma_j(i)}{k}, \tag{11}$$

$$\omega^{(j)} = \frac{\sum_{i=1}^{k} \gamma_j(i) \left(y(i) - f_j(x_j(i), \boldsymbol{\theta}^{(j)})\right)^2}{\sum_{i=1}^{k} \gamma_j(i)}, \tag{12}$$

where $\mathbf{A}^{(j)}$ is the design matrix of model j, and $\boldsymbol{\Gamma}^{(j)} = \text{diag}(\gamma_j(1), \gamma_j(2), \ldots, \gamma_j(k))$ is a diagonal matrix.

The MULRM offline learning by the EM algorithm can then be summarized as follows:

1) Initialize $\boldsymbol{\vartheta}$ equal to some initial $\boldsymbol{\vartheta}^{\text{old}}$;
2) Repeat 3) and 4) until the EM algorithm converges;
3) **E** step:
   a) Compute the responsibilities (8) using $\boldsymbol{\vartheta}^{\text{old}}$;
   b) Compute the expectation $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$ using (9);
4) **M** step:
   a) Compute the values of $\boldsymbol{\vartheta}$ which maximize $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$ using (10)-(12):
      i) $\boldsymbol{\vartheta}^* = \arg\max_{\boldsymbol{\vartheta}} Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$;
      ii) $\boldsymbol{\vartheta}^{\text{old}} := \boldsymbol{\vartheta}^*$;
5) Return $\boldsymbol{\vartheta}^*$.

The convergence of the EM algorithm can be verified by analyzing the convergence of the expectation $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$. It is also possible to set a pre-specified maximum number of iterations. Equations (11)-(12) are the solutions of the maximization step in the EM algorithm, for the combination of the univariate models in the form of (1). However, this solution is limited to the offline case. The next section will provide an online algorithm for the learning of (1), with a sample weighting adaptation approach.

B. MULRM - Recursive/Online Solution

In the online learning of $\boldsymbol{\vartheta}$, each available sample corresponds to an iteration in the EM algorithm. The parameters computed by the offline equations (11)-(12) of the EM algorithm maximization step should be learned recursively, at each new sample. In the solution derived here, a forgetting factor λ will be employed to weight more recent data, making the MULRM able to be applied in time varying scenarios. $0 < \lambda \leq 1$, so that the smaller the λ parameter, the larger is the weight of the recent data, as in the traditional RLS.

From (11) it is possible to note that υ_j is given by the average over the samples of the responsibilities $\gamma_j(i)$ of model f_j. Then, υ_j can be learned using the adaptive recursive mean. Specifically, the mean of $\gamma_j(i)$ among k values, indicated by $\upsilon_j(k)$, can be updated when a new sample $\gamma_j(k+1)$ is available using the following adaptive mean formula, where λ is used to discount the information coming from the already learned data:

$$\upsilon_j(k+1) = \lambda \upsilon_j(k) + (1-\lambda)\gamma_j(k+1). \tag{13}$$

Using the same idea, equation (12) can be seen as a ratio between two means over the samples k:

$$\omega^{(j)} = \frac{\frac{\sum_{i=1}^{k} \gamma_j(i)\left(y(i) - f_j(x_j(i), \boldsymbol{\theta}^{(j)})\right)^2}{k}}{\frac{\sum_{i=1}^{k} \gamma_j(i)}{k}}, \tag{14}$$

where the denominator is equal to υ_j (11), and the numerator is equal to the weighted error $E_j = (\sum_{i=1}^{k} E_{\gamma_j}(i))/k$ of model f_j in predicting y, where $E_{\gamma_j}(i) = \gamma_j(i)\left(y(i) - f_j(x_j(i), \boldsymbol{\theta}^{(j)})\right)^2$. Similarly to (13), the recursive formula for E_j is given by:

$$E_j(k+1) = \lambda E_j(k) + (1-\lambda) E_{\gamma_j}(k+1). \tag{15}$$

Then, the value of $\omega^{(j)}$ when a new sample is available is:

$$\omega^{(j)}(k+1) = \frac{E_j(k+1)}{\upsilon_j(k+1)}. \tag{16}$$

When a new sample $\mathbf{a}_j(k+1) = [1, x_j(k+1)]^T$ is available, the closed form solution for $\boldsymbol{\theta}^{(j)}$ (10) is represented in expanded form in equation (17), where $\gamma_j(k+1)$ and $\boldsymbol{\Gamma}_k^{(j)}$ represent the values of the responsibilities of the current sample $\mathbf{a}_j(k+1)$ and of the previous samples, respectively. However, in time-varying systems, the update formula for the weights $\boldsymbol{\theta}^{(j)}$ of each model f_j, should also take into consideration the forgetting factor λ. Thus, a matrix of weights $\mathbf{W} = \text{diag}(\lambda^{k-1}, \lambda^{k-2}, \ldots, 1)$ is designed to affect the samples, so that the model could take into consideration recent data with more weight. Then, the full update equation for $\boldsymbol{\theta}^{(j)}$, taking into consideration the forgetting factor is given by (18). The recursive formulas for computing (10) are as follows:

$$\mathbf{P}_{k+1}^{(j)} = \lambda^{-1}\left(\mathbf{P}_k^{(j)} - \frac{\gamma_j(k+1)\mathbf{P}_k^{(j)}\mathbf{a}_j(k+1)\mathbf{a}_j^T(k+1)\mathbf{P}_k^{(j)}}{\lambda + \gamma_j(k+1)\mathbf{a}_j^T(k+1)\mathbf{P}_k^{(j)}\mathbf{a}_j(k+1)}\right), \tag{19}$$

$$g^{(j)}(k+1) = \mathbf{P}_{k+1}^{(j)}\mathbf{a}_j(k+1), \tag{20}$$

$$e^{(j)}(k+1) = \gamma_j(k+1)\left(y(k+1) - \mathbf{a}_j^T(k+1)\boldsymbol{\theta}^{(j)}(k)\right), \tag{21}$$

$$\boldsymbol{\theta}_{k+1}^{(j)} = \boldsymbol{\theta}_k^{(j)} + g^{(j)}(k+1)e^{(j)}(k+1). \tag{22}$$

The MULRM online learning by the EM algorithm is then summarized as follows:

1) Initialize $\mathbf{P}_k^{(j)}$, and $\boldsymbol{\vartheta}$ equal to some initial $\boldsymbol{\vartheta}^{\text{old}}$;
2) For each newly available sample $(\mathbf{x}(k+1), y(k+1))$:
3) **E** step:
   a) Compute the responsibilities (8) of sample $(k+1)$ using $\boldsymbol{\vartheta}^{\text{old}}$;
   b) Compute the expectation of sample $(k+1)$, $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})(k+1)$ using (9);
4) **M** step:
   a) Update the values of $\boldsymbol{\vartheta}$ which maximize $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{\text{old}})$:
      i) For each model, update υ_j using (13);
      ii) For each model, update $\omega^{(j)}$ using (16);
      iii) For each model, update $\boldsymbol{\theta}^{(j)}$ using (19)-(22);
5) Return the updated parameters.

$$\boldsymbol{\theta}_{k+1}^{(j)} = \left(\begin{bmatrix} \mathbf{A}_k^{(j)} \\ \mathbf{a}_j^T(k+1) \end{bmatrix}^T \begin{bmatrix} \boldsymbol{\Gamma}_k^{(j)} & 0 \\ 0 & \gamma_j(k+1) \end{bmatrix} \begin{bmatrix} \mathbf{A}_k^{(j)} \\ \mathbf{a}_j^T(k+1) \end{bmatrix} \right)^{-1} \begin{bmatrix} \mathbf{A}_k^{(j)} \\ \mathbf{a}_j^T(k+1) \end{bmatrix}^T \begin{bmatrix} \boldsymbol{\Gamma}_k^{(j)} & 0 \\ 0 & \gamma_j(k+1) \end{bmatrix} \begin{bmatrix} \mathbf{y}_k \\ y(k+1) \end{bmatrix} \tag{17}$$

$$\boldsymbol{\theta}_{k+1}^{(j)} = \left(\begin{bmatrix} \mathbf{A}_k^{(j)} \\ \mathbf{a}_j^T(k+1) \end{bmatrix}^T \begin{bmatrix} \lambda \mathbf{W} \boldsymbol{\Gamma}_k^{(j)} & 0 \\ 0 & \gamma_j(k+1) \end{bmatrix} \begin{bmatrix} \mathbf{A}_k^{(j)} \\ \mathbf{a}_j^T(k+1) \end{bmatrix} \right)^{-1} \begin{bmatrix} \mathbf{A}_k^{(j)} \\ \mathbf{a}_j^T(k+1) \end{bmatrix}^T \begin{bmatrix} \lambda \mathbf{W} \boldsymbol{\Gamma}_k^{(j)} & 0 \\ 0 & \gamma_j(k+1) \end{bmatrix} \begin{bmatrix} \mathbf{y}_k \\ y(k+1) \end{bmatrix} \tag{18}$$

The parameter $\mathbf{P}_k^{(j)}$ should be initialized as: $\mathbf{P}_{(k)}^{(j)} = \varphi \mathbf{I}_2$, where $\mathbf{I}_2$ is the identity matrix with size 2×2, and φ should be set as a large value.

The experimental results suggest that the online learning of the MULRM approximates to offline solution for a static case, when $\lambda = 1$, for a large number of samples. Since the MULRM will be applied in real time, then after some operating time a large number of samples will be under consideration.

C. Remarks on MULRM

1) Collinearity: The objective of the proposed method is on prediction rather than explaining the underlying relationships among the variables, i.e. the MULRM is to be applied on prediction problems and not explanation ones. When working with explanation, the problem of collinearity becomes a problem of understanding the relationships among the variables [37], and when working with prediction, the problem becomes to predict an output given a random input. According with Hocking [37], the collinearity among input variables is not necessarily harmful when working with prediction problems. Additive models are an example of prediction models which ignore the collinearity among the input variables, while providing good results in some applications [36].

The proposed method can be applied for prediction in the presence or absence of collinearity in input data. Also, the MULRM cannot identify the joint effect among the variables (i.e. interaction terms of input variables are not considered in the MULRM), since in the MULRM the effects of the input variables on the output variables are individually assessed. However, this model formulation facilitates estimation since each component of the model can be addressed separately using (10)-(12). A special case of the proposed MULRM occurs when $\gamma_j(i) = 1/k, j = 1, \ldots, D$, $i = 1, \ldots, k$, which reduces to the LS solution in the case where the input variables are mutually independent. Moreover, it is not possible to assure the physical meanings for the slopes and intercepts in univariate linear regression models, then we assume that the slopes and intercepts does not have physical meaning at all.

2) Candidate models: The choice of linear form of $f_j(\cdot)$ during the development of MULRM lies on the good results provided by the predictor in our scenarios of interest. However, there is no restriction in the form of $f_j(\cdot)$, it can take any form, e.g:

1) Linear: $f_j(x_j) = \theta_0^{(j)} + \theta_1^{(j)} x_j$;
2) Polynomial: $f_j(x_j) = \sum_{l=0}^{n} \theta_l^{(j)} x_j^l$;
3) Other non-linear forms: e.g. $f_j(x_j) = \theta_0^{(j)} + \theta_1^{(j)} \ln(x_j)$ or $f_j(x_j) = \theta_0^{(j)} + \theta_1^{(j)} \sqrt{x}_j$.

3) Dealing with outliers: If a variable x_j is affected by an outlier at sample i, and such sample is used to update the parameters, the responsibility $\gamma_j(i)$ in (8) will take a small value. This happens because if the value of $x_j(i)$ is an outlier, the value of $\mathcal{N}\left(y(i) \middle| f_j(x_j(i), \boldsymbol{\theta}^{(j)}), \omega^{(j)}\right)$ in (8) will be small. This can affect the performance of MULRM whether variable x_j is relevant to predict the target or not. If x_j is relevant to predict the target and $\lambda < 1$ with $\gamma_j(i)$ being small, then, with the update for sample i, the parameters of model j, $\boldsymbol{\theta}^{(j)}$ (22), $\omega^{(j)}$ (14), and υ_j (13) will loose information learned from the previous samples and will not gain the information from the current sample, since it is an outlier. This can decrease the performance of the overall method. However, if variable x_j is not relevant, then the effect of the outlier on the overall performance will not be significant. Moreover, in any case, outlier detection is an essential step while building soft sensors, and this step is encouraged when applying the MULRM model in real time applications. See [3] for a review about methods to deal with outlier detection in soft sensors applications.

4) Accuracy, bias and precision from the measurement point of view: All data-driven soft sensors, as well as the proposed method, are data dependent; i.e. they perform the task of being a sensor, based on the learning of a model using the gathered data of the process, represented by $\Phi = \{(\mathbf{x}(i), y(i)); i = 1, \ldots, k\}$. The precision, accuracy and bias, of the soft sensor, from the measurement point of view, are directly related to the representativeness of Φ with respect to the future samples. After the data driven soft sensor is learned using Φ, and deployed for real operation, it is possible to affirm that the soft sensor is always precise, since the same input will always generate the same output. However, this does not mean that the soft sensor is accurate or not biased. In practice, due to the time-varying characteristics of most industrial processes, the soft sensor tends to deteriorate its accuracy over the time and also have the presence of bias. This happens because the data set Φ used for the learning is no longer representative when the time passes. However, this situation motivates the update of the soft sensor model with the most recent data, which is performed online on the proposed MULRM method, increasing its accuracy and reducing its bias. From the measurement point of view the sensor calibration of the soft sensor is done by updating the model using the most recent samples of the process.

5) Selecting λ: The value of λ can be fixed to a prespecified value or its value can vary iteratively in real time by using the following gradient descent method proposed in

[38]:

$$\lambda_{k+1} = \lambda_k + c_\lambda \text{sign}\left(\frac{\partial RSS}{\partial \lambda_k}\right), \tag{23}$$

where c_λ is a small constant. In the above equation the value of λ is moved in the direction in which the move minimizes the residual sum of squares (RSS). If λ is to be fixed, then it can be selected by a K-fold cross-validation procedure applied on the training set.

IV. Experimental Results

The online version of the proposed method was applied in two time-varying real-world data sets. The proposed recursive MULRM method will be compared with the RLS, RPLS solutions of the multivariate linear model, and with the OS-ELM [14] and ILLSA [21]. In all experiments, the values of both the training samples, and the testing samples, were normalized to be mean centered, and with unit variance, using the information of the sample mean and sample variance of the training samples. The original variables were used as input of all models: RLS, RPLS, MULRM, ILLSA and OS-ELM. Moreover, outliers detection was not considered. It has been assumed that data in both experiments were free of outliers.

In both data sets the normalized root mean square error (NRMSE) was used as a performance measure to compare the results of the methods:

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{k}\sum_{i=1}^{k}\left(y(i)-\hat{y}(i)\right)^2}}{\max(y)-\min(y)}, \tag{24}$$

where $y(i), \hat{y}(i)$ are the observed and predicted target, the quantities $\max(y)$, and $\min(y)$ are the maximum and minimum values of the observed target. This value is often expressed in percentage. The closer to 0 is the NRMSE the better is the quality of prediction. In a practical prediction perspective, a NRMSE value of less than 10% is acceptable.

A. Polymerization Data Set

The polymerization data set is a benchmark for adaptive soft sensors introduced in [21], [22], where the state of the art ensemble adaptive soft sensor method called ILLSA is also proposed. This data set describes a polymerization reactor. The objective is to predict the activity of the catalyst in the multitube. There are 15 input variables available, and some of them suffer from outliers, missing values, noise and automatic value interpolation by the data acquisition system. The data set covers 1 year of acquisition with 5800 available samples. In this data set, the degree of collinearity among the input variables, measured using the largest variance inflate factor (VIF) criterion, is equal to 91.19, which indicates a high degree of multicollinearity [37], [39].

This paper follows the same pre-processing procedure done in [21]: downsampling of the first 5800 samples by a factor of 10, the removal of variables $3, 4, 15$, and removing all samples which have missing values. This pre-processing resulted in a data set with 647 data points, where 194 are used for offline training and the remaining 457 are used to simulate the online data. To verify the performance of the soft sensor with respect to the availability of the target data, it was tested in the cases where 0%, 10%, 25%, 50%, and 100% of the available target values were made available. In these situations, the two extreme cases (i.e., 0% and 100%) represent a non-adaptive/static scenario, and a scenario where all the target values are used for the adaptation purposes, respectively. For evaluation purposes, if a new valid input-output pair $(\mathbf{x}(n), y(n))$ is available for update, then the output $y(n)$ will be first predicted using the input $\mathbf{x}(n)$, and then the model parameters will be updated. Moreover, the number of latent variables used in the RPLS algorithm and the number of neurons in the OS-ELM were respectively chosen to be 6 and 5, by applying a 10-fold cross validation scheme in the training set and selecting the number of latent variables and neurons which produced the averaged smallest error in all folds. The ILLSA algorithm was applied as in the original paper [21].

TABLE I
NRMSE values on the Polymerization and WWTP data sets for different forgetting factors, λ, and different percentages of target data.

		Polymerization				WWTP			
Up.	λ, d	**RLS**	**RPLS**	**OS-ELM**	**Prop.**	**RLS**	**RPLS**	**OS-ELM**	**Prop.**
0%	$0.50, 2$	70.41	79.89	24.99	**19.00**	24.55	23.63	16.43	**13.43**
	$0.80, 5$	70.41	79.89	24.99	**19.00**	24.55	23.63	16.43	**13.43**
	$0.95, 20$	70.41	79.89	24.99	**19.00**	24.55	23.63	16.43	**13.43**
	$0.98, 50$	70.41	79.89	24.99	**19.00**	24.55	23.63	16.43	**13.43**
	$0.99{,}100$	70.41	79.89	24.99	**19.00**	24.55	23.63	16.43	**13.43**
	$1, \infty$	70.41	79.89	24.99	**19.00**	24.55	23.63	16.43	**13.43**
		ILLSA: 28.74				**ILLSA**: 14.59			
10%	$0.50, 2$	42.84	**11.65**	47.17	**11.65**	46.83	10.18	17.31	**9.40**
	$0.80, 5$	26.17	13.85	14.02	**7.78**	19.83	11.39	10.51	**9.65**
	$0.95, 20$	19.05	18.83	13.67	**10.62**	19.24	16.38	**10.53**	11.60
	$0.98, 50$	22.63	23.60	13.14	**12.31**	20.06	18.06	**11.01**	12.48
	$0.99{,}100$	24.19	26.80	14.44	**13.75**	20.43	19.58	**10.74**	12.86
	$1, \infty$	25.91	24.59	**15.13**	16.28	20.83	20.40	**11.20**	13.30
		ILLSA: 9.47				**ILLSA**: 11.30			
25%	$0.50, 2$	20.82	6.31	46.03	**4.54**	30.63	8.77	16.10	**9.93**
	$0.80, 5$	20.83	9.91	16.14	**5.83**	18.49	9.47	10.54	**8.22**
	$0.95, 20$	15.72	12.02	13.07	**8.31**	12.57	11.43	9.90	**9.51**
	$0.98, 50$	18.31	17.16	12.87	**10.52**	15.39	15.26	**10.28**	10.92
	$0.99{,}100$	20.19	19.23	12.26	**12.10**	16.37	15.20	**10.79**	11.77
	$1, \infty$	22.91	22.98	**13.57**	16.54	17.51	17.39	**11.25**	12.90
		ILLSA: 8.94				**ILLSA**: 10.01			
50%	$0.50, 2$	30.73	4.68	100.4	**2.90**	∞	**7.91**	19.28	30.73
	$0.80, 5$	10.17	5.92	16.16	**3.59**	13.89	8.33	9.80	**7.66**
	$0.95, 20$	15.55	11.00	10.36	**6.71**	9.12	9.07	8.74	**8.59**
	$0.98, 50$	14.80	12.10	11.31	**8.80**	10.74	10.53	**9.45**	9.62
	$0.99{,}100$	16.38	17.07	12.24	**10.24**	12.34	11.83	**10.09**	10.66
	$1, \infty$	20.66	20.55	**12.60**	16.28	14.41	17.52	**11.04**	12.53
		ILLSA: 6.81				**ILLSA**: 9.18			
100%	$0.50, 2$	20.92	3.58	6.54	**2.14**	∞	**7.79**	38.40	∞
	$0.80, 5$	7.85	4.14	10.29	**2.57**	∞	8.00	9.85	**7.05**
	$0.95, 20$	9.27	7.23	8.83	**5.12**	8.62	8.06	8.37	**7.77**
	$0.98, 50$	12.02	11.48	8.75	**7.20**	9.59	9.50	8.85	**8.68**
	$0.99{,}100$	12.26	12.17	10.15	**8.54**	9.59	10.06	9.69	**9.49**
	$1, \infty$	18.31	18.64	**12.49**	15.38	12.33	14.71	**11.03**	12.28
		ILLSA: 5.51				**ILLSA**: 9.60			

The results are summarized in Table I. The proposed MULRM method, the RLS, RPLS and the OS-ELM methods were applied for different values of λ (the corresponding value of $d = 1/(1-\lambda)$ is also indicated), and for different scenarios of availability of target data. By analyzing the performance of the models with different percentages of updates, it is possible to note the possibility of reducing, by using a soft sensor, the number of measurements of the hard to measure output variable that need to be obtained by real-sensors or laboratory

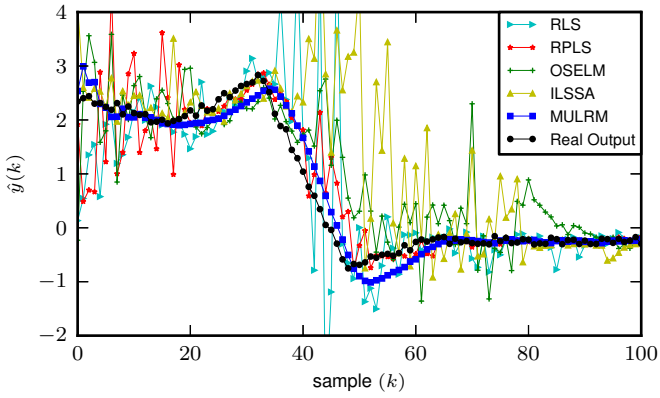

Fig. 1. Activity of catalyst prediction output of all models for the first 100 samples. The frequency of updates is 100% for all models. The value of λ for each model was chosen based on the best results of the respective model in Table I

analysis.

As can be noticed λ was set to be greater than or equal to 0.5. This was motivated by the physical meaning/interpretation of λ, since it is related with the effective number of samples. For example, assuming that the considered values of λ are less than 0.5, then d will lie between $1 \leq d < 2$ in the real interval, which does not have physical meaning or logic interpretation. Moreover, in our experiments, values of λ less than 0.5 did not provide any improvement to the experimental results.

As can be noticed, for the polymerization, the proposed MULRM method has the best results in almost all scenarios. In the case where the availability of target data is 0%, the non-adaptive scenario, the performance of the proposed method reached a NRMSE value of 19%, the smallest if compared with the other methods, but still larger than 10%, which, as mentioned before, is not an acceptable value in a practical application. Better results are reached with the increase of availability of target data. In the case where 100% of target data is used to update, the proposed MULRM method reached its smallest NRMSE value of 2.14% (with $\lambda = 0.50$), which is much better than the state of art methods. In the polymerization data set, the MULRM method seems to provide the best results when the values of λ, d, are small (which is the main motivation of the proposed approach), on the other hand, when $d = \infty$ (i.e. without forgetting the already learned data), the OS-ELM provides the best results, but they are not satisfactory. The prediction results of all models are exhibited in Figure 1.

B. Wastewater Treatment Plant

In this experiment the objective is to estimate the flour concentration in the effluent of a real-world urban wastewater treatment plant (WWTP). The data set of plant variables that is available for learning consists of 11 input variables, and one target output variable to be estimated. The variables correspond to physical values, such as pH, turbidity, color of the water and others. Table II presents further details about the variables. In this data set, the degree of collinearity among the

TABLE II
VARIABLES OF THE WASTEWATER TREATMENT PLANT DATA SET.

Var.	Description	Var.	Description
x_1	Chlorine in the influent;	x_2	Chlorine in the effluent;
x_3	Turbidity in the raw water;	x_4	Turbidity in the influent;
x_5	Turbidity in the effluent;	x_6	Ph in the raw water;
x_7	Ph in the influent;	x_8	Ph in the effluent;
x_9	Color in the raw water;	x_{10}	Color in the influent;
x_{11}	Color in the effluent;	y	Flourine in the effluent.

input variables, measured using the largest VIF value, is equal to 6.19, which is considered an acceptable value for VIF [37], [39], and it indicates that the collinearity will not interfer in the LS solution.

The methodology used in the WWTP experiment was the same as the one used in the polymerization experiment (Sec. IV-A). The available data set was split into 30% for training, and the remaining 70% of data was used to simulate the online data, and it is delivered as a stream of samples. The scenarios of availability of target data were 0%, 10%, 25%, 50%, and 100%.

The historical data set comprises 3 years of acquisition, with 13512 data samples, with a sample rate of 2 [h], for the variables acquired by sensors (input variables). The target variable, the fluorine, is laboratory measured at every 24 [h]. The samples with missing fluorine data were removed, resulting in a data set with 1002 samples, where 294 were used for training, and the remaining 708 were used to simulate the online data.

The number of latent variables for the RPLS model, and the number of neurons in the OS-ELM model, as result of the 10-fold cross-validation on the training data set, were 4, and 7, respectively. The parameters used in the ILLSA algorithm, found by a 10-fold cross-validation (check [21] for more details), were: 11 receptive fields, $\sigma = 10^{-3}$, and $\sigma^{adapt} = 10^{-6}$.

Table I shows the results regarding the WWTP data set. In the non-adaptive scenario, 0%, the results of the proposed MULRM method are the best among all, with a NRMSE = 13.43%. Despite these results the values of NRMSE are still larger than 10%, which are not acceptable values. The results of all methods get better with the increase in the number of updates, and all methods reach their minimum NRMSE when 100% of target data is used to update the model, and the proposed method reaches a NRMSE of 7.05 with $\lambda = 0.80$.

For all percentages of availability, when $d \leq 20$, the best results are obtained by the MULRM method, in most of the cases. When $d > 20$, and for updates between 10% and 50%, OS-ELM is the method with the best results. It is important to also note that, motivated by the small values of VIF, the LS solution provides good results in some experiments, which in some cases is even better than the PLS regression.

In the WWTP experiment, the MULRM produced large values of NRMSE when $\lambda = 0.5$, $d = 2$, in the case where 50% and 100% of data are used to update the model. This happened because some variables in the WWTP data set are very noisy, and with frequencies of updates of 50% and 100%

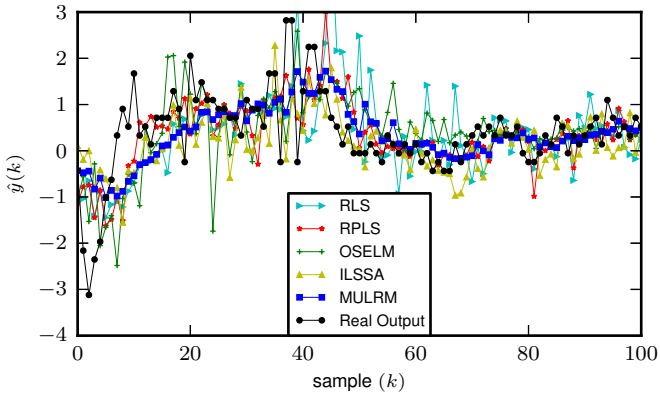

Fig. 2. Fluorine prediction output of all models for the first 100 samples. The frequency of updates is 100% for all models. The value of λ for each model was chosen based on the best results of the respective model in Table I.

more samples of these variables are used in the updates. Thus, in these scenarios it is not advisable to use $d = 2$, since it can lead to the learn the noise. Values of $\lambda > 0.8$, and $d > 5$, seem to be more appropriate. Another alternative to set λ is to use the systematic procedure discussed in Section III-C5.

The proposed method also reaches acceptable values of NRMSE for 25% and 50% of target data available for update. This suggests that the frequency of laboratory measurements can be reduced by a half or less, if this soft sensor is going to be applied, reducing the costs associated with the laboratory measurements. The predictions of all models are exhibited in Figure 2, validating and showing the effectiveness of the proposed method to perform prediction in time-varying environments.

V. Discussion

Both data sets are time-varying real world data sets, then to track the time-varying parameters of each model a forgetting factor was used in the recursive learning of the parameters. For both data sets, it is possible to see that the best results, in almost all models (the OS-ELM seems to be the most constant among the other models), are achieved when $d = 2$, $d = 5$ and $d = 20$ (small values of λ); i.e. the results mean that the best performances are achieved when the most recent samples are used to compose the learning of the models parameters. Moreover, the best prediction performance achieved by the proposed MULRM method, is in general better when using small values of (λ, d). However, although these results are representative, they are also conditioned by the problems under study, i.e. it is not possible to assure that they are general for all other conceivable problems. Nevertheless, MULRM can be a good option for soft sensor applications in time-varying scenarios. In some experiments, the proposed method still does not satisfy the requirement for NRMSE $< 10\%$. This happens mainly when the updating of the model does not occur (0%) or when the frequency of model updating is low (10% and 25%).

The advantage of MULRM on the presented experiments, in comparison with the other methods, is mainly on the prediction performance, since the execution times for all methods are similar (except ILLSA, which is much more time demanding). The major drawback of MULRM is that it cannot identify the joint effect among the variables, thus it cannot be used as an explanation method.

VI. Conclusions

This paper proposed the use of a mixture of univariate linear models for adaptive regression, in a new method called MULRM. The formulas for the offline and online learning were derived based on the EM algorithm. Furthermore, in this work the proposed method has been evaluated and compared with the current state of art methods on two real world data sets.

On the polymerization data set, the proposed MULRM method was compared with the RLS, RPLS, OS-ELM and ILLSA methods, with better results in the almost all experiments. On the WWTP data set, the performance of the proposed method was much better, in the cases where $d < D$, when compared with the other state of the art RLS, RPLS, OS-ELM and ILLSA methods. Moreover, the application of the proposed method on the WWTP plant, to predict the fluorine, will allow the reduce of costs associated with the laboratory analysis, since it has been verified that the rate of model updates can be reduced by a half or more, while still attaining good prediction results.

References

[1] L. Fortuna, S. Graziani, A. Rizzo, and M. G. Xibilia, *Soft Sensors for Monitoring and Control of Industrial Processes*, 1st ed., ser. Advances in Industrial Control. Springer, December 2006.

[2] ——, "Comparison of soft-sensor design methods for industrial plants using small data sets," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 8, pp. 2444–2451, 2009.

[3] P. Kadlec, R. Grbic, and B. Gabrys, "Review of adaptation mechanisms for data-driven soft sensors," *Computers & Chemical Engineering*, vol. 35, no. 1, pp. 1–24, 2011.

[4] D. Wang, J. Liu, and R. Srinivasan, "Data-driven soft sensor approach for quality prediction in a refining process," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 1, pp. 11–17, february 2010.

[5] D. Schutz, A. Wannagat, C. Legat, and B. Vogel-Heuser, "Development of plc-based software for increasing the dependability of production automation systems," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 2397–2406, November 2012.

[6] M. Grbovic, W. Li, N. A. Subrahmanya, A. K. Usadi, and S. Vucetic, "Cold start approach for data driven fault detection," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 2264–2273, November 2013.

[7] J. Zhao, Q. Liu, W. Pedrycz, and D. Li, "Effective noise estimation-based online prediction for byproduct gas system in steel industry," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 4, pp. 953–963, november 2012.

[8] D. Wang, "Robust data-driven modeling approach for real-time final product quality prediction in batch process operation," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 2, pp. 371 –377, May 2011.

[9] C. Gao, L. Jian, and S. Luo, "Modeling of the thermal state change of blast furnace hearth with support vector machines," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 2, pp. 1134–1145, 2012.

[10] P. Kadlec, B. Gabrys, and S. Strandt, "Data-driven soft sensors in the process industry," *Computers & Chemical Engineering*, vol. 33, no. 4, pp. 795–814, 2009.

[11] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, 1st ed. Prentice Hall, September 1997.

[12] M.-D. Ma, J.-W. Ko, S.-J. Wang, M.-F. Wu, S.-S. Jang, S.-S. Shieh, and D. S.-H. Wong, "Development of adaptive soft sensor based on statistical identification of key variables," *Control Engineering Practice*, vol. 17, no. 9, pp. 1026–1034, 2009.

[13] I. T. Jolliffe, *Principal component analysis*. Springer, 2002.
[14] L. Nan-Ying, H. Guang-Bin, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.
[15] M. A. Shoorehdeli, M. Teshnehlab, and A. K. Sedigh, "Training anfis as an identifier with intelligent hybrid stable learning algorithm based on particle swarm optimization and extended kalman filter," *Fuzzy Sets and Systems*, vol. 160, pp. 922–948, 2009.
[16] J. Mendes, F. Souza, R. Araújo, and N. Gonçalves, "Genetic fuzzy system for data-driven soft sensors design," *Applied Soft Computing*, vol. 12, no. 10, pp. 3237–3245, 2012.
[17] V. Behbood, J. Lu, and G. Zhang, "Fuzzy refinement domain adaptation for long term prediction in banking ecosystem," *IEEE Transactions on Industrial Informatics*, 2013, to be published. [Online]. Available: http://dx.doi.org/10.1109/TII.2012.2232935
[18] A. Tsymbal, "The problem of concept drift: Definitions and related work," Department of Computer Science, Trinity College: Dublin, Ireland, Tech. Rep., 2004.
[19] O. Haavisto and H. Hyötyniemi, "Recursive multimodel partial least squares estimation of mineral flotation slurry contents using optical reflectance spectra," *Analytica Chimica Acta*, vol. 642, no. 1-2, pp. 102–109, 2009, papers presented at the 11th International Conference on Chemometrics in Analytical Chemistry - CAC 2008.
[20] K. Helland, H. E. Berntsen, O. S. Borgen, and H. Martens, "Recursive algorithm for partial least squares regression," *Chemometrics and Intelligent Laboratory Systems*, vol. 14, no. 1-3, pp. 129–137, 1992.
[21] P. Kadlec and B. Gabrys, "Local learning-based adaptive soft sensor for catalyst activation prediction," *AIChE Journal*, vol. 57, no. 5, pp. 1288–1301, 2011.
[22] ——, "Adaptive online prediction soft sensing without historical data," in *The 2010 International Joint Conference on Neural Networks (IJCNN'10)*, 2010, pp. 1 –8.
[23] T. Komulainen, M. Sourander, and S. L. J. Jounela, "An online application of dynamic pls to a dearomatization process," *Computers & Chemical Engineering*, vol. 28, no. 12, pp. 2611–2619, 2004.
[24] C. Li, H. Ye, G. Wang, and J. Zhang, "A recursive nonlinear pls algorithm for adaptive nonlinear process modeling," *Chemical Engineering & Technology*, vol. 28, pp. 141–152, 2005.
[25] S. Mu, Y. Zeng, R. Liu, P. Wu, H. Su, and J. Chu, "Online dual updating with recursive pls model and its application in predicting crystal size of purified terephthalic acid (pta) process," *Journal of Process Control*, vol. 16, no. 6, pp. 557–566, 2006.
[26] P. Facco, F. Bezzo, and M. Barolo, "Nearest-neighbor method for the automatic maintenance of multivariate statistical soft sensors in batch processing," *Industrial & Engineering Chemistry Research*, vol. 49, no. 5, pp. 2336–2347, 2010.
[27] R. Muradore and P. Fiorini, "A pls-based statistical approach for fault detection and isolation of robotic manipulators," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 8, pp. 3167–3175, 2012.
[28] B. S. Dayal and J. F. MacGregor, "Recursive exponentially weighted pls and its applications to adaptive control and prediction," *Journal of Process Control*, vol. 7, no. 3, pp. 169–179, 1997.
[29] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1–38, 1977.
[30] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. Springer, 2006.
[31] P. D. Wentzell and L. V. Montoto, "Comparison of principal components regression and partial least squares regression through generic simulations of complex mixtures," *Chemometrics and Intelligent Laboratory Systems*, vol. 65, no. 2, pp. 257–279, 2003.
[32] P. Angelov and A. Kordon, "Adaptive inferential sensors based on evolving fuzzy models," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, no. 2, pp. 529–539, 2010.
[33] S. J. Raudys and A. K. Jain, "Small sample size effects in statistical pattern recognition: recommendations for practitioners," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 3, pp. 252–264, mar 1991.
[34] E. Frank, L. Trigg, G. Holmes, and I. H. Witten, "Technical note: Naive bayes for regression," *Machine Learning*, vol. 41, no. 1, pp. 5–25, June 2000.
[35] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, no. 1, pp. 1–22, 2010.
[36] A. Buja, T. Hastie, and R. Tibshirani, "Linear smoothers and additive models," *The Annals of Statistics*, vol. 17, no. 2, pp. 453–510, 1989.
[37] R. R. Hocking, *Methods and Applications of Linear Models: Regression and the Analysis of Variance*. John Wiley & Sons, Inc. 2003, ch. Collinearity in Multiple Linear Regression, pp. 151–192.
[38] C. Anagnostopoulos, D. Tasoulis, D. J. Hand, and N. M. Adams, "Online optimization for variable selection in data streams," in *Proceedings of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence*. Amsterdam, The Netherlands: IOS Press, 2008, pp. 132–136.
[39] D. A. Belsley, E. Kuh, and R. E. Welsch, *Regression Diagnostics*. John Wiley & Sons, Inc. 2005, ch. Detecting and Assessing Collinearity, pp. 85–191.

Francisco Souza was born in Fortaleza, Ceará - Brazil, 1986. He received the B.Sc. degree in Electrical Engineering (Automation branch) from the University Federal of Ceará, Brazil, in 2008. He is currently pursuing his Ph.D. degree in Electrical and Computer Engineering at the University of Coimbra. Since 2009, he is a Researcher at the "Institute for Systems and Robotics - University of Coimbra" (ISRUC). His research interests includes machine learning and pattern recognition with application to industrial processes.

Rui Araújo received the B.Sc. degree ("Licenciatura") in Electrical Engineering, the M.Sc. degree in Systems and Automation, and the Ph.D degree in Electrical Engineering from the University of Coimbra, Portugal, in 1991, 1994, and 2000 respectively. He joined the Department of Electrical and Computer Engineering of the University of Coimbra where he is currently an Assistant Professor. He is a founding member of the Portuguese Institute for Systems and Robotics (ISR-Coimbra), where he is now a researcher. His research interests include computational intelligence, intelligent control, computational learning, fuzzy systems, neural networks, estimation, control, robotics, mobile robotics and intelligent vehicles, robot manipulators control, sensing, soft sensors, automation, industrial systems, embedded systems, real-time systems, and in general architectures and systems for controlling robot manipulators, mobile robots, intelligent vehicles, and industrial systems.