Knowledge based Recursive Non-linear Partial Least Squares (RNPLS)

A. Merino[1], D. Garcia-Alvarez[b], G.I. Sainz-Palmero[c], L.F. Acebes[c], M.J. Fuente[c]

[a] *Department of Electromechanic Engineering, University of Burgos, Burgos, Spain*
[b] *Empresarios Agrupados Internacional, 47151 - Parque Tecnolgico Boecillo, Valladolid (Spain)*
[c] *Department of Systems Engineering and Automatic Control, University of Valladolid, Valladolid, Spain*

Abstract

Soft sensors driven by data are very common in modern industry to predict critical variables which are difficult to measure using other variables that are relatively easier to obtain. The use of soft sensors implies some challenges, such as the predictor variables colinearity, the time-varying and possible non-linear nature of the industrial process. To deal with the first challenge, the partial least square (PLS) regression has been employed for many applications to deal with the linear variable relationships, with noisy and highly correlated data. However, the PLS model needs to deal with the other two issues: the non-linear and the time-varying side behaviour of the processes. In this work, a new knowledge based methodology for a recursive non-linear PLS algorithm (RNPLS) is systematized to deal with these issues. Here, the non-linear PLS algorithm is made by carrying out the PLS regression over the augmented matrix of input, which includes knowledge based non-linear transformations of some of the variables. This transformation depends on the type of the system, and takes into account the available knowledge about the process, which is provided by expert knowledge or em-

[1] Corresponding author:
Alejandro Merino
Department of Electromechanic Engineering, Technical School, Avda. Cantabria, S/N, 09001, University of Burgos, Burgos, Spain,
e-mail: alejandromg@ubu.es
Tlf:

ulated using software tools. Then, the recursive exponential weighted PLS is used to modify and adapt the model according to the process changes. This RNPLS algorithm has been tested using two case studies according to the available knowledge, a real industrial evaporation station of the sugar industry where the expert knowledge about the process permits the formulation of the relationships and a simulated wastewater treatment plant where the needed knowledge about the process is obtained by a software tool. The results show that the methodology involving knowledge regarding the process is able to adjust the process changes, providing highly accurate predictions.

Keywords:
Soft Sensor, Partial Least Squares, Non-linear Mapping, Recursive Estimation, RNPLS

1. Introduction

Nowadays, industrial plants normally have a large number of sensors. Data obtained through these sensors are used to monitor and control the process. However, it is usual in industrial process plants for some variables not to be directly measured for several reasons: sometimes there are no sensors available on the market to measure a concrete physical or chemical variable, or if it exists, the cost is high. At other times, the expenses related to the sensor's maintenance are very high, etc. For these reasons, some process variables are obtained periodically by performing a laboratory analysis, which entails a considerable delay and low sampling time, with inappropriate data being obtained for control and monitoring purposes. In these cases, a soft sensor technique is a very effective method for estimating unmeasured variables, based on other process variables that are easier to measure.

In recent years, the industry has paied attention on soft sensors [1]. It is possible to find two categories of soft sensors: model-driven and data-driven methods. The former are principally based on a first principles model describing the physical and chemical relationships of the process [2] and can be obtained from process knowledge. However, given the complexity of industrial plants, developing and maintaining these models require a lot of time and effort. Nowadays, with the increased number of instruments in industrial plants, data-driven based soft sensors have gained in popularity. The most widely used techniques to develop soft sensors are the multivariate regression models, such as principal component regression (PCR) and partial least squares (PLS) [1]. However, these methods are linear and the industrial processes are normally non-linear, so different non-linear data-driven methods are used to cover this situation, such as support vector machines (SVM) [3] and neural networks (ANN) [2, 4].

The PLS algorithm has been shown to be a powerful multivariate statistical tool for modelling output variables when data are noisy and highly correlated as occurs in industrial plants. It has been demonstrated given two datasets, PLS is able to capture the maximal covariance between them [5], and the power of the PLS regression is that the relations between the set of observed variables are modelled by latent variables, which are the projections of the original process variables onto an orthogonal subspace of low dimension. These latent variables can be computed using the well-known algorithm called NIPALS (non linear and iterative partial least square) [6] or the kernel algorithm [7].

However, real plants can present significant non-linear characteristics, so the PLS regression is not adequate due to its linear nature. To solve this challenge, non-linear PLS (NPLS) algorithms have been developed. A first review of literature about NPLS can be found in [8]. There are two basic principles to develop NPLS algorithms.

The first is to apply non-linear transformations to the observed variables and to extend the input matrix to include those transformations, and then to apply the PLS algorithm over this augmented matrix [9]. The most used non-linear transformations are a polynomial one [10], spline transformations over the predictor variables, but in this case the coefficients of the splines are more unknown parameters that have to be calculated [11] and the outputs of an Radial Basis Function (RBF) network [12]. Information or knowledge about the process is not explicitly used to treat the process non-linearities. The problem with this approach is that the expansion with the transformations of all the input variables results in a very high increase in the number of variables, which leads to the appearance of the problem called ”curse of dimensionality”, which takes into account the high increase of the parameters to be estimated regarding the available data, while the number of observations remains unchanged.

The second principle is focused on modelling the inner relationship between the latent variables, in a non-linear way, without modifying the original values, but with a higher computational cost and optimization complexity. This relationship between the scores (or latent variables) can be modelled by a quadratic function [9, 13], artificial neural networks [14, 15], Radial Basis

Function Networks (RBFN) [16], Takagy-Sugeno-Kang fuzzy models [17, 18] or evolutionary computational (EC) methods [19].

A different approach to build an NPLS algorithm is when this is mapping the observed data into a high-dimensional feature space, where it is possible to build linear models based on the theory of support vector machines and kernel functions: so a kernel PLS (KPLS) method was proposed by [20] and used in [21] to predict some variables in a wastewater plant. However, although the KPLS can fit non-linear relationships, the model obtained is a black-box model with limited capabilities to explain the results in terms of the original variables.

Another approach was found in [22] where the logarithm transformation of inputs and outputs were carried out, in order to linearise both the dynamic response and the output profile. So, in this case the predictor matrix of the PLS algorithm are the logarithm transformations of the inputs to the system (temperatures) and the response matrix are the logarithm transformation of the outputs (compositions). Thus, with both transformations, the system and as consequence the model are linear and, in this case, it is possible to use the linear PLS regression. However, in a general industrial plant it is not easy and/or possible to transform the inputs and outputs so that the relationship between them is linear. A more general procedure it is necessary to deal with non-linear transformations between variables to be used in the PLS algorithm. Also, the objective, in this case, is not to linearise the model and to use a linear PLS algorithm, the aim is to introduce the non-linear characteristics of the process into the PLS algorithm to build a non-linear PLS (NPLS) method.

However, all these techniques are based on the assumption that the process is operating in a steady state, in which case the soft sensor can present accuracy and robustness problems when there exist time-varying changes such as those related to environmental conditions, the process raw materials, etc. To be able to cope with these effects, some kind of adaptation must be implemented in the soft sensor [23]. When data are collected continuously, it is desirable to recursively update the regression model as new data become available. Also, as the process changes with time, it is important to weight the novel data more deeply while discounting past data using a lower weight. [24] presents a recursive PLS algorithm where all input and output data are characterised by their respective loading matrices and new data are added to those matrices for updating the model. This algorithm keeps the size of the input data matrices for calculating the PLS regression model constant, i.e., they are not augmented with new data, but in the adaptation procedure, not all latent variables are retained in the loading matrix, so this can imply a loss of information and poor model performance. In [25], an RPLS algorithm was developed to overcome these problems and a moving time window was used to remove the oldest data. A fast recursive exponentially weighted PLS algorithm is proposed by [26], where the adaptation to new data is in the covariance matrices instead of the input and output data matrices. Some applications of the different RPLS algorithms can be found in [27, 28, 29, 30], but all these recursive algorithms are based on linear PLS models.

Another option to take into account the non-stationary process variations, i.e., the time-varying nature of the processes, is to use the fast moving window PLS method (FMWPLS) [31], to rule out the older data as new data become available, following the ideas of [32] whom propose a fast moving algorithm for monitoring time varying processes, adapting the PCA model. In this approach the computational load does not depend on the window size, which seems to be more adequate for on-line updating models. Also, this technique has been extended to monitor non-linear time-varying process with the Moving Window Kernel PCA (MWKPCA) [33]. Recently, much attention has been focused on the latent variable methods based on linear state space model for monitoring dynamic systems. Two major variants of the Linear State Spaces Models (LSSM) have been extensively adopted, particularly the linear Gaussian state space model LGSSM [34] and the canonical variate analysis CVA [35]. In these cases the state variables are the latent variables whose time-varying behaviour are used to monitor and detect anomalies in the system. These two latter methods are linear, to consider also the non-linear nature of the systems, a non-linear extension of the Gaussian estate space model (NGSSM) is proposed in [36].

Another aspect to take into account in the development of reliable industrial data-driven soft sensors is the structure of the available data (missing data, acquisition frequency, presence of outliers, variables resolution, etc.). In the scientific literature there are works that address the existence of outliers [37, 38, 39], missing data [40] and different sampling rates (multirate data) [41, 42, 43]. Facing this last aspect in different ways. A simple solution is to sample the system with the lowest frequency observed variable and build the model with this low sampling rate data. Another solution is to just do a numerical interpolation or to use lifting techniques to derive a model in order to include the missing low frequency observations. Finally, a better solution is based on finite impulse response (FIR) models used to weight past observations before their inclusion in the model [44]. Even it is possible to introduce a regularization approach to smooth out the coefficients over time to obtain a DPLS-TS (Dynamical PLS with temporal smoothness) [43] method or a SPLS method (sparse PLS) depending on the regularization approach used. Another aspect to consider is when the data present a mutiresolution structure, the process information has different grades of granularity (different resolutions). This happens when some variables contain instantaneous information and other variables represent averages with different time spans, shifts or production batches. This problem can be solved using mutiresolution soft sensors and multiresolution time series models [45, 46, 47].

In terms of knowledge acquisition (KA) in complex systems, it can be obtained in different ways, as it was explained in a review carried out by [48]. This KA is addressed in multiple fields of activity, as the engineering field [49]. The knowledge acquisition can be accomplished using human experts to obtain the expertise knowledge about a specific field, but also this task can be performed automatically based on data, applying machine agents. These machine agents employ different computational intelligence technologies, as statistical analysis, machine learning techniques as neural networks, neuro-fuzzy systems, support vector machine and so on, that allow them to perform an autonomous knowledge discovery process, generating knowledge models, hidden in the data collected from the plants. In this paper, the knowledge used to implement the software sensor is obtained using both sources: the human experts for the first case study and the machine agents for the second case study.

In this work, a new knowledge based methodology for Recursive Non-linear PLS is introduced, integrating knowledge about the process into PLS regression to overcome both the problems of non-linearity and the time-varying changes of the industrial processes. Firstly, an NPLS model is made using the PLS regression over an extended matrix of input. In this way, the input matrix is augmented with knowledge in way of expected generic non-linear relationships between the output variable and some of the original input variables, these relationships can be provided by experts, or sourced another way. These knowledge and expertise, can be expressed by means of fuzzy rules, polynomial, exponential, logarithmic, and so on, representations to deal with the non-linearity of the process. This knowledge can also consider delayed variables, which permits to face with the dynamic nature of the industrial processes. Finally, a recursive version of the NPLS, based on the recursive exponential weighted with the kernel algorithm [26], is used to modify the model and enable it to adapt to the process changes.

Taking into account all this, the major contributions of this work can be summarized as: (1) Systematize a methodology for integrating knowledge about the process in the NPLS regression, here knowledge collecting involves a few well established raw non-linear relationships regarding variables, sourced by experts, or by an alternative way if these are not available; (2) extend the previous methodology including recursiveness in the non-linear PLS algorithm considering the non-linearity, the dynamics, the collinearity and the time-varying characteristics of the variables in industrial process; and (3) ensuring consistent results in real plants.

The remainder of this paper is organized as follows. In section 2, the PLS method and the RPLS algorithm proposed by [26] are briefly presented. In section 3, the RNPLS based on knowledge algorithm is detailed, and in the next section the effectiveness of this method is demonstrated by applying the RNPLS algorithm to two complex plants, a real industrial evaporation station of the sugar industry and a simulation benchmark of a Wastewater Treatment Plant. Finally, conclusions are given in the last section.

2. Recursive PLS algorithm

PLS regression [5, 50] is widely used by many reasons; it can calculate regression models from high collinearity data, allowing to analyze data with more variables than individuals, resulting models are able to provide stable predictions, minimizing the rate of adjustment, being able to manage data with missing information and to identify *outliers* in the data [38].

PLS determines a projection structure that models the relationship between a response matrix $\mathbf{Y} \in \Re^{n \times p}$ and the prediction matrix $\mathbf{X} \in \Re^{n \times m}$.

$$\mathbf{X} = \sum_{a=1}^{A} \mathbf{t}_a \mathbf{p}_a^T + \mathbf{E} = \mathbf{T}\mathbf{P}^T + \mathbf{E} \tag{1}$$

$$\mathbf{Y} = \sum_{a=1}^{A} \mathbf{t}_a \mathbf{q}_a^T + \mathbf{F} = \mathbf{T}\mathbf{Q}^T + \mathbf{F} \tag{2}$$

where $\mathbf{T}$ is known as the *score* matrix, $\mathbf{P}$ and $\mathbf{Q}$ are the *loadings* and $\mathbf{E}$ and $\mathbf{F}$ are the residual matrices of $\mathbf{X}$ and $\mathbf{Y}$ respectively, for a model with $\mathbf{a}$ latent variables determined using cross-validation. Each score is extracted through deflating $\mathbf{X}$ and $\mathbf{Y}$ matrices by the non-linear iterative partial least squares (NIPALS) algorithm [51].

$$\mathbf{X}_a = \mathbf{X}_{a-1} - \mathbf{t}_a \mathbf{p}_a^T, \quad \mathbf{Y}_a = \mathbf{Y}_{a-1} - \mathbf{t}_a \mathbf{q}_a^T \tag{3}$$

Once the PLS model is calculated using the process historical data until all the data structure variance can be explained, it is possible to carry out predictions of new estimates using the following equations:

$$\mathbf{T} = \mathbf{X}\mathbf{R}, \quad \mathbf{R} = \mathbf{W}\left(\mathbf{P}^T\mathbf{W}\right)^{-1} \tag{4}$$

$$\hat{\mathbf{Y}} = \mathbf{X}\mathbf{B}_{PLS}, \qquad \mathbf{B}_{PLS} = \mathbf{R}\mathbf{Q}^T \tag{5}$$

where $\mathbf{B}_{PLS}$ is the matrix of regression coefficients and $\mathbf{W}$ is the PLS weights matrix.

Regression coefficients provided by PLS can be used to predict the value of the observed variables on-line. However, the historical data used to build the PLS model should contain all possible future states and process conditions and this is not usually possible. This includes not only the states achieved under normal operation conditions but also those related to environmental conditions, raw materials changes, etc [52]. Additionally, most of the industrial processes exhibit time-varying behaviour and thus need some approach to face the on-line adaptation. So, when new sample data x_t, y_t became available, from the physical sensors or from the laboratory analysis, the parameters of the PLS model, eq. (5), need to be learned on-line and updated recursively (soft-sensors are usually called adaptive soft-sensors in this context). In this on-line learning, some strategy needs to be used to update the model parameters including recent data and forgetting the older ones, allowing its use in applications where the time-varying behaviour is relevant. This problem is usually solved by using an exponential weighting approach, usually a forgetting factor.

The adaptive soft sensor used in this paper is the recursive version of the PLS method, i.e., the RPLS method [26], which uses the current model, in this case in form of the covariance matrix, and new data samples to update the model. The adaptation requires weighting down the preceding model using a variable forgetting factor. This method is called exponentially weighted recursive PLS algorithm and combine the improved PLS kernel algorithm developed by [53], that is proven to be faster than the NIPALS algorithm, with the updating of the covariance matrices $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}^T\mathbf{Y}$. Those matrices are updated as new data are available and former data are exponentially discarded:

$$\left(\mathbf{X}^T\mathbf{X}\right)_t = \lambda_t \left(\mathbf{X}^T\mathbf{X}\right)_{t-1} + \mathbf{x}_t^T\mathbf{x}_t \tag{6}$$

$$\left(\mathbf{X}^T\mathbf{Y}\right)_t = \lambda_t \left(\mathbf{X}^T\mathbf{Y}\right)_{t-1} + \mathbf{x}_t^T\mathbf{x}_t \tag{7}$$

where $\mathbf{x}_t$ and $\mathbf{y}_t$ are the new $(1 \times m)$ and $(1 \times p)$ predictor and response vector observed at time t respectively and $(\mathbf{X}^T\mathbf{X})_t$ and $(\mathbf{X}^T\mathbf{Y})_t$ are the updated covariances at time t while the old data are discounted exponentially with a forgetting factor λ_t with $(0 < \lambda_t \leq 1)$ as new data are added. As it is known, if $\lambda_t = 1$ no discount of past data is done.

The discounting of old data when new data is available is necessary to account for the time varying nature of the processes. It is possible to do it with a constant forgetting factor, or a variable one. The problem with constant forgetting factor is that it is necessary the persistent excitation in the process, that is not always possible, in these cases the covariance matrix become ill-conditioned and the accuracy of the resulting model will be poor. To avoid that, forgetting factor for the variables, that discount old data only when there is information in the new data is a better solution. [54] proposed the use of equation (8) to estimate the value of λ_t. This equation is calculated in each sampling interval.

$$\lambda_t = 1 - \frac{[1 - \mathbf{x}_t(\mathbf{X}^T\mathbf{X})_t^{-1}\mathbf{x}_t^T]e_t^2}{\Sigma_0} \tag{8}$$

being λ_t narrowed using equation 9.

$$\lambda_t = max\{\lambda_{min}, min\{\lambda_t, \lambda_{max}\}\} \tag{9}$$

where e_t^2 is the error term and can be calculated using the PLS regression estimates, $\Sigma_0 = \sigma_0^2 N_0$, being σ_0^2 the expected measurement noise variance in the output variable and N_0 the nominal asymptotic memory length, that can be used to adjust the speed and smoothness of the adaptation.

In ordinary PLS algorithm, $\mathbf{x}_t$ and $\mathbf{y}_t$ variables need to be mean centred and scaled before using them to calculate the model. In time varying process, mean and variance may be changing with time, so both need to be updated at each sampling interval in order to mean centre and scale the new data that come from the plant. This can be done, in two forms, the first one, proposed by [26], instead of mean-centring the data, augments the dimension of the vector of observed variables $\mathbf{x}$ with an unity to account for the constant term, i.e., the new data is:

$$\mathbf{x}_t = [\mathbf{x}_{1t}\ \mathbf{x}_{2t}\ ...\ 1] \tag{10}$$

With this method the information regarding mean and variance can be extracted from covariance matrices. Another way to update the mean and variance is to use the exponential moving average. In this paper, the same equations that would be obtained from the extended matrices are used but calculated directly using equations (11) and (12) and executed every time a new sample is available.

$$\overline{\mathbf{x}}_t = \frac{\lambda_t \left(\sum \mathbf{x}\right)_{t-1} + \mathbf{x}_t}{N_t} \tag{11}$$

$$var\left(\mathbf{x}\right)_t = \frac{\left(\sum \mathbf{x}^2\right)_t - N_t \overline{\mathbf{x}}_t^2}{N_t - 1} \tag{12}$$

being $\overline{\mathbf{x}}$ the $(1 \times m)$ vector of means for the observed variables, $var\left(\mathbf{x}\right)_t$ the $(1 \times m)$ vector of variances for the observed variables and N_t is the effective window length. Similar equations can be used to calculate mean and variance for predicted variables $\mathbf{y}$.

Using the calculated values for the mean and variance, new data are centred and scaled before being introduced into the covariance matrices. If a variable forgetting factor would be used, the number of observations taken into account to calculate the moving average, i.e., the effective memory length would be:

$$N_t = \lambda_t N_{t-1} + 1 \tag{13}$$

3. Knowledge based RNPLS

Nowadays, especially in the continuous production processes, information and knowledge about the process is available through different sources and can be formalized in different ways. Technicians in charge of processes have gathered through years expert knowledge which can be expressed by fuzzy rules or qualitative mathematical formulations. Usually, in the design stage of industrial plants, simple models and simulators are used to size the equipment and to design the process and its control systems, being also alternatives sources of this knowledge, even in other cases a detailed dynamic model are available [55]. However, some times, the availability of experts is a challenge. In these cases, it is proposed to use some software tools such as Alamo (Automatic Learning of Algebraic MOdels) [56], to obtain some formulation of this knowledge to be included in the PLS regression.

Two cases studies are introduced in this work for testing the proposal: in the first case study, a real evaporation plant is involved, the knowledge was available in way of well-known relationships managed by technicians about process variables [57]. This information was used to find out base and raw mathematical expressions that relate the observed measurements with the output variable, extending the matrix of observed variables with these new calculated variables.

However, sometimes the expertise in the plant is not available, or it is a heavy challenge: this is the second case study introduced. In this case, it is proposed to use some software, such as the mentioned Alamo tool [56], to glimpse the relationship between observed and output variables.

Once the knowledge is obtained, this can be incorporate to the extended input matrix in order to compute the recursive non-linear PLS as it is described in the following subsection.

3.1. Recursive Nonlinear PLS algorithm

Here, knowledge about the process serves for dealing with the challenges regarding non-linear characteristics and time-varying changes into a recursive non-linear PLS method. First, a non-linear PLS methodology is built combining PLS with the knowledge about the process to determine its non-linear characteristics, i.e., the physical-chemical knowledge about the process is used to glimpse raw non-linear relationships between some of the measured variables and the predicted variables. After that, when new information is taken from the plant, the NPLS is updated based on the new data and the former model, to obtain a Recursive Non-linear PLS (RNPLS) algorithm. So, this soft sensor is developed using two steps: building the non-linear PLS with mentioned qualitative knowledge of the process and updating the NPLS algorithm recursively when new information concerning the plant is available. The scheme of methodology proposed is shown in Fig. 1 and in the coming stages and steps:

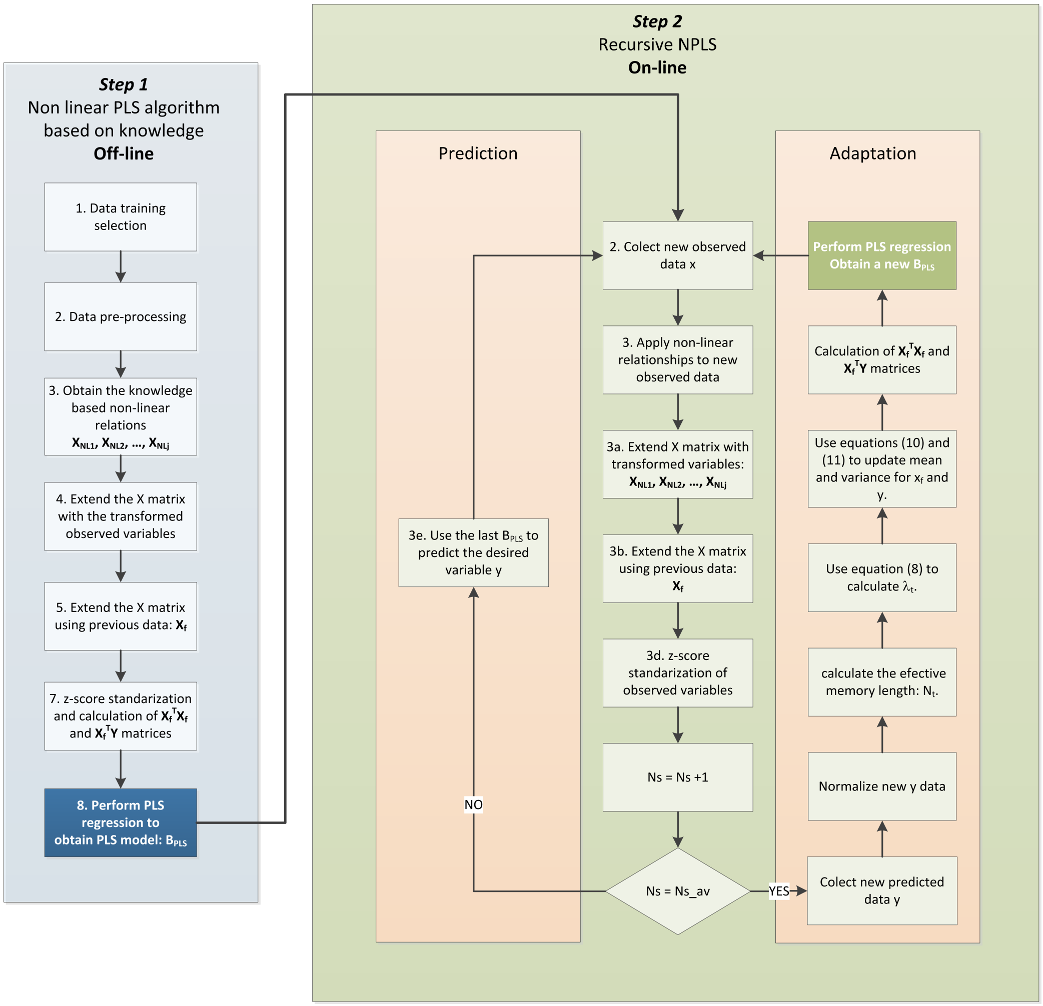
Step 1
Non linear PLS algorithm based on knowledge
Off-line
1. Data training selection
2. Data pre-processing
3. Obtain the knowledge based non-linear relations $X_{NL1}, X_{NL2}, \ldots, X_{NLj}$
4. Extend the X matrix with the transformed observed variables
5. Extend the X matrix using previous data: X_f
7. z-score standarization and calculation of $X_f^T X_f$ and $X_f^T Y$ matrices
8. Perform PLS regression to obtain PLS model: B_{PLS}
Step 2
Recursive NPLS
On-line
Prediction
Adaptation
2. Colect new observed data x
3. Apply non-linear relationships to new observed data
3a. Extend X matrix with transformed variables: $X_{NL1}, X_{NL2}, \ldots, X_{NLj}$
3b. Extend the X matrix using previous data: X_f
3d. z-score standarization of observed variables
Ns = Ns +1
Ns = Ns_av
NO
YES
3e. Use the last B_{PLS} to predict the desired variable y
Perform PLS regression Obtain a new B_{PLS}
Calculation of $X_f^T X_f$ and $X_f^T Y$ matrices
Use equations (10) and (11) to update mean and variance for x_f and y.
Use equation (8) to calculate λ_t.
calculate the efective memory length: N_t.
Normalize new y data
Colect new predicted data y

Figure 1: Flowchart of the proposed RNPLS based on knowledge soft sensor

Step 1. *Non-linear PLS algorithm based on knowledge.* The NPLS method is implemented using knowledge of the process in terms of qualitative relationships between the predictor variables and the predicted variable. This stage is made up of sub-steps as follows:

1. Data training acquisition showing the current operation conditions.
2. Data pre-processing, to choose the process variables to build the input or predictor matrix $\mathbf{X} \in \Re^{n \times m}$, where n is the number of observations and m the number of variables, and the output or predicted matrix $\mathbf{Y} \in$

$\Re^{n\times p}$, where p is the number of predicted or unknown variables that the sensor software has to calculate. Also in this step, it is necessary to eliminate outliers, take into account the missing variables, and so on.

3. Using the knowledge of the process to formulate relationships in a simple mathematical way between the output variable and some of the input variables of the matrix $\mathbf{X}$. This step depends on the nature of the process and the type of knowledge to be incorporated. If this knowledge of the process, i.e., if the physical-chemical properties of industrial process are not well known or are very complex to establish or this expertise is not available, this issue can be worked out using software capable of generating algebraic models from data, such as the Alamo method [56]. So, in this case, the qualitative knowledge is obtained from the data collected in the first sub-step. The output of this sub-step is a number of non-linear transformations of some of the input variables: $\mathbf{X}_{NL1}, \mathbf{X}_{NL2}, ..., \mathbf{X}_{NLj}$, i.e., j new variables are obtained.
4. Extend the input matrix $\mathbf{X}$ with these transformed variables to create the augmented matrix $\mathbf{X}_a$, i.e., $\mathbf{X}_a = [\mathbf{X}, \mathbf{X}_{NL1}, \mathbf{X}_{NL2}, ..., \mathbf{X}_{NLj}]$, with $\mathbf{X}_a \in \Re^{n\times(m+j)}$.
5. To capture the dynamic characteristics of the industrial process, a dynamic PLS (DPLS) method can be considered, taking into account, for each observation, its previous L observations and stacking the data matrix in the following manner: $\mathbf{X}_f = [\mathbf{X}_a(t), \mathbf{X}_a(t-1), ..., \mathbf{X}_a(t-L)]$, where $\mathbf{X}_f \in \Re^{(n-L)\times((m+j)(L+1))}$ is the final input matrix, $\mathbf{X}_a(t)$ is the matrix data $\mathbf{X}_a$ at the t time instant and $\mathbf{X}_a(t-L)$ at the time instant $t-L$, that is, with a L time samples delay.
6. This sub-step is optional. As the final input matrix $\mathbf{X}_f$ has a very large dimension, i.e., the number of variables (or columns) of this matrix can be very high $[(m+j)\cdot(L+1)]$ and in order to avoid the "curse of dimensionality" problem, a dynamic feature selection over $\mathbf{X}_f$ can be carried out [58], [59].
7. The final input matrix $\mathbf{X}_f$ and the output matrix $\mathbf{Y}$ are standarized using z-score. Calculate the covariance matrices: $\mathbf{X}_f^T\mathbf{X}_f$ and $\mathbf{X}_f^T\mathbf{Y}$.
8. Perform the PLS regression on those matrices using the PLS Kernel modified algorithm explained in section 2, i.e., the inputs to the algorithm are $[\mathbf{X}_f^T\mathbf{X}_f, \mathbf{X}_f^T\mathbf{Y}]$ and the outputs are $[\mathbf{W}, \mathbf{P}, \mathbf{Q}, \mathbf{R}, \mathbf{B}_{PLS}]$, defined in eqs. 4 and 5. So, $\mathbf{B}_{PLS}$ can be used to predict the output variables.

In this proposal, this step is calculated off-line, i.e., this first step is necessary to train the NPLS model. The pseudo-code of this NPLS method is shown in algorithm 1.

Algorithm 1 *Knowledge based NPLS pseudocode algorithm*

```
Initialize variables, and parametrize the problem: Number of training data: n, number of latent variables, etc.
NPLS Training
Read training data for the initial NPLS.
Select the input matrix $\mathbf{X}$ and the output matrix $\mathbf{Y}$
Apply non-linear transformations to the training data to obtain new observed variables $\mathbf{X}_{NL1}, \mathbf{X}_{NL2}, ..., \mathbf{X}_{NLj}$.
Define the augmented matrix with the observed variables and the transformed variables using non-linear relationships $\mathbf{X}_a = [\mathbf{X}, \mathbf{X}_{NL1}, \mathbf{X}_{NL2}, ..., \mathbf{X}_{NLj}]$
Use past values to define a dynamic problem $\mathbf{X}_f = [\mathbf{X}_a, \mathbf{X}_a(t-1), \mathbf{X}_a(t-2), ..., \mathbf{X}_a(t-L)]$.
[Optional] In the final matrix $\mathbf{X}_f$ defined before, performs the dynamic feature selection.
Normalize the input and output matrices: $\mathbf{X}_f$ and $\mathbf{Y}$ to z-score
Calculate the normalized covariance matrices $\mathbf{X}_f^T\mathbf{X}_f$ and $\mathbf{X}_f^T\mathbf{Y}$
Call PLS modified Kernel Algorithm with these covariance matrices as inputs [53]
Use $\mathbf{B}_{PLS}$ to predict the desired variables.
```

Step 2. *Recursive Non-linear PLS algorithm.* To face to the time-varying behaviour in industrial processes and to improve the on-line performance of the designed soft sensor, a recursive approach based on the ideas of [26] is also used. This algorithm is calculated on-line and predicts the value of the unknown variables using the knowledge based NPLS method training in Step 1, when a new data input vector $\mathbf{x} \in \Re^{1\times m}$ consisting of m variables becomes available, each sampling time. The adaptation of the NPLS algorithm is carried out when the target vector $\mathbf{y} \in \Re^{1\times p}$ is also available, which does not occur in each sampling time, because in realistic industrial scenarios, the target vector is sampled with a very low sampling rate, called in this paper N_s. This procedure is explained in algorithm 2 and is made up of various sub-steps:

1. Start with the NPLS algorithm based on knowledge trained in Step 1.

2. Collect new data from the plant $\mathbf{x} \in \Re^{1\times m}$. If the actual sampling time is different from $k \cdot N_s$, with $k = 1, 2, ...$ go to the prediction procedure. If the actual sampling time coincides with the sampling rate of the target vector, $k \cdot N_s$, collect the new value of these variables from the plant: $\mathbf{y} \in \Re^{1\times p}$ and go to the updating procedure.
3. Prediction procedure:
   (a) Use the non-linear transformations obtained in the training process to calculate the j new variables so as to extend the input matrix using the new data collected from the plant: $\mathbf{x}_{NL1}, \mathbf{x}_{NL2}, ..., \mathbf{x}_{NLj}$. The augmented matrix is now: $\mathbf{x}_a = [\mathbf{x}, \mathbf{x}_{NL1}, \mathbf{x}_{NL2}, ..., \mathbf{x}_{NLj}]$, with $\mathbf{x}_a \in \Re^{1\times(m+j)}$.
   (b) Use the L previous observations to implement the DPLS method and to calculate: $\mathbf{x}_f = [\mathbf{x}_a(t), \mathbf{x}_a(t-1), ..., \mathbf{x}_a(t-L)]$, the final input matrix, where now $\mathbf{x}_f \in \Re^{1\times((m+j)(L+1))}$
   (c) Apply the dynamic feature selection over $\mathbf{x}_f$ if this sub-step was done in the training procedure of Step 1.
   (d) Normalize the new data matrix: $\mathbf{x}_f$ to z-score, using the last mean and variance calculated.
   (e) Call the NPLS trained in Step 1, i.e., use the $\mathbf{B}_{PLS}$ obtained in Step 1 to predict the desired variables: $\hat{\mathbf{y}}$
4. Updating procedure:
   (a) Use the same three first steps of the prediction procedure explained above to get the final input matrix: $\mathbf{x}_f$.
   (b) The recursive method discounts the old data when new data is available using a forgetting factor, λ_t. So now, this variable forgetting factor is calculated as explained in section 2, and it is used to compute the mean and the variance of the new data, $\mathbf{x}$ and $\mathbf{y}$, for the next iteration, because as the system is time-varying, it is normal that the mean and variance can change. In addition, it is used in the updating of the covariance matrices $\mathbf{X}_f^T\mathbf{X}$ and $\mathbf{X}_f^T\mathbf{Y}$, taking into account the old model and the new available data (eqs. 6 and 7 respectively).
   (c) Perform the PLS regression on these new covariance matrices using the PLS Kernel modified algorithm to obtain an updated model.
   (d) Then, use this new updated $\mathbf{B}_{PLS}$ to predict the output variable: $\hat{\mathbf{y}}$.

Algorithm 2 *RNPLS pseudocode algorithm*

```
 1: Start with the initial NPLS method trained in algorithm 1
 2: Define period between new observations of the target vector $N_s$
 3: Initialize variables, and parametrize the problem: number of data for recursive estimation($N_r, \infty$ in on-line real application), number of latent variables, etc.
 4: *Recursive NPLS*
 5: for i=1 to $N_r$ do
 6:     Add new $\mathbf{x}$ data
 7:     if (i % $N_s$==0) then
 8:         Add new $y$ data
 9:     end if
10:     Calculate the new variables applying non-linear transformations to the observed variables $\mathbf{x}_{NL1}, \mathbf{x}_{NL2}, ..., \mathbf{x}_{NLj}$.
11:     Build the augmented matrix with the observed variables and the transformed variables using non-linear relationships $\mathbf{x}_a = [\mathbf{x}, \mathbf{x}_{NL1}, \mathbf{x}_{NL2}, ..., \mathbf{x}_{NLj}]$.
12:     Use $L$ past values to define extra observed variables $\mathbf{x}_f = [\mathbf{x}_a, \mathbf{x}_a(t-1), \mathbf{x}_a(t-2), ..., \mathbf{x}_a(t-L)]$.
13:     [Optional] Apply the dynamic feature selection over $\mathbf{x}_f$ if this sub-step was done in the training procedure of Step 1.
14:     Normalize new $\mathbf{x}_f$ data using the last calculated value.
15:     if (i % $N_s$==0) then
16:         Normalize new $\mathbf{y}$ data using the new information available.
17:         Use equation (13) to calculate the effective memory length: $N_t$.
18:         Use equation (8) to calculate $\lambda_\mathbf{t}$.
19:         Use equations (11) and (12) to update mean and variance for $\mathbf{x}_f$ and $\mathbf{y}$.
20:         Update normalized covariance matrices $\mathbf{X}_f^T\mathbf{X}$ and $\mathbf{X}_f^T\mathbf{Y}$ using equations (6) and (7) respectively.
21:         Call Modified Kernel Algorithm using all the previous calculated variables.
22:     end if
23:     Use $\mathbf{B}_{PLS}$ to predict the value for the desired variables.
24: end for
```

4. Experimental results

Here, the proposal is applied in two case studies, and the corresponding results are shown. The first one is a real-world case study addressing the prediction of the sugar concentration in an evaporation station of a real sugar industry, in where the expertise about this process is available by qualitative knowledge, which is involved trough generic mathematical relationships. The second case is the estimation of the chemical oxygen demand (COD) variable in a benchmark regarding a wastewater treatment plant. Here, the expert knowledge is not available but through the ALAMO software this expertise was emulated in order to glimpse qualitative knowledge to be included in the computation. In order to check the performance of this proposal (RNPLS), its results have been compared with other approaches such as DPLS and knowledge based NPLS.

4.1. Case Study 1: Evaporation station of a real sugar plant

The first case study is the design of a °Brix soft sensor in an evaporation section of a sugar factory. Sugar plants produce sugar crystals mainly from sugar beets or sugar cane, involving various stages. The process generally begins with the sucrose extraction, obtaining a juice, which needs to be refined and concentrated in order to crystallise the sugar in batch vacuum pans [60]. Evaporation is very important in sugar factories, it is the most energy consuming section of the plant and also provides steam to the rest of the plant equipment [61]. Evaporation is formed by several evaporators arranged in series, called a multi-effect evaporation, in which the steam produced in one effect is used in the next one, that operates at a lower pressure (Figure 2).

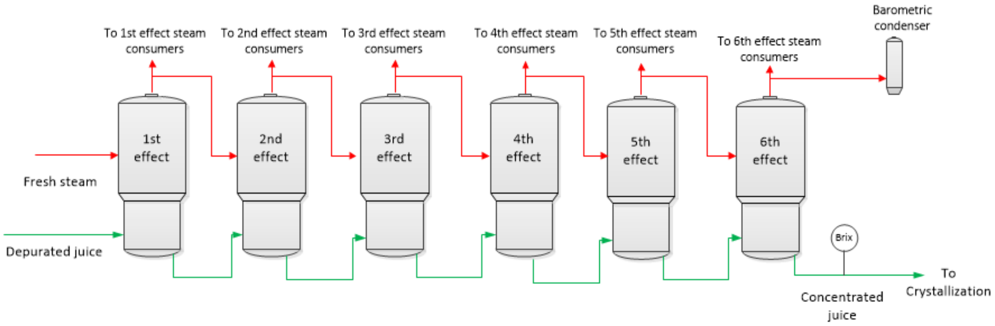
To 1st effect steam consumers
To 2nd effect steam consumers
To 3rd effect steam consumers
To 4th effect steam consumers
To 5th effect steam consumers
To 6th effect steam consumers
Barometric condenser
1st effect
2nd effect
3rd effect
4th effect
5th effect
6th effect
Fresh steam
Depurated juice
Brix
To Crystallization
Concentrated juice

Figure 2: Evaporation section scheme

The main variable to control in the evaporation process is the °Brix, which is defined as the amount of soluble solid content present in the juice, mainly sucrose, expressed as a percentage. Together with the purity, gives an idea of the concentration of sugar in the juice, and it should be kept within a specific range of values to avoid problems. If the °Brix are too high, caramelization can occur and a crust of sugar can be formed in the equipment. If the sugar concentration is too low, the vacuum pans, where crystallization takes place, would need to evaporate the remaining water, increasing the batch time.

Several instruments can be used to measure °Brix on-line, such as refractometers, densimeters, microwaves and infra-red waves. These sensors are expensive and require frequent maintenance, which limits their use to critical points in the process. This fact makes this variable an excellent candidate to be estimated using a soft sensor, as they can also be used as a replacement for the real sensor during maintenance or recalibration.

For the experiments, real plant data, sampled every 10 seconds, were used. Fifty variables related with the evaporation are considered, shown in table 1. Various data sets were gathered up from the plant representing ordinary operation conditions; specifically, a training data set of 7000 sampled data and a validation data set with 15000 sampled data were collected from the plant to design the soft sensor.

Table 1: Variables description

Id.	Description	Units	Id.	Description	Units
P1	Steam pressure in evaporator 1	bar	P2	Steam pressure in evaporator 2	bar
P3	Steam pressure in evaporator 3	bar	P4	Steam pressure in evaporator 4	bar abs
P5	Steam pressure in evaporator 5	bar abs	P6	Steam pressure in evaporator 6	mbar abs
T1	Temperature of the juice at evaporator 1 inlet	°C	T2	Temperature of the juice at R10 heat exchanger inlet	°C
T3	Temperature of the juice at R13 heat exchanger inlet	°C	T4	Temperature of the juice at R14 heat exchanger inlet	°C
T5	Temperature of the juice at R3 heat exchanger inlet	°C	T6	Temperature of the juice at R3B heat exchanger inlet	°C
T7	Temperature of the juice at R8 heat exchanger inlet	°C	T8	Temperature of the juice at R9 heat exchanger inlet	°C
T9	Temperature of the juice at thin juice tank outlet	°C	T10	Temperature of the juice at R10 heat exchanger outlet	°C
T11	Temperature of the juice at R11 heat exchanger outlet	°C	T12	Temperature of the juice at R12 heat exchanger outlet	°C
T13	Temperature of the juice at R3 heat exchanger outlet	°C	T14	Temperature of the juice at R3A heat exchanger outlet	°C
T15	Temperature of the juice at R3B heat exchanger outlet	°C	T16	Temperature of the juice at R4 heat exchanger outlet	°C
T17	Temperature of the juice at R5 heat exchanger outlet	°C	T18	Temperature of the juice at R6 heat exchanger outlet	°C
T19	Temperature of the juice at R7 heat exchanger outlet	°C	T20	Temperature of the juice at R8 heat exchanger outlet	°C
T21	Temperature of the juice at R9 heat exchanger outlet	°C	T22	Steam temperature in evaporator 1	°C
T23	Steam temperature in evaporator 2	°C	T24	Steam temperature in evaporator 3	°C
T25	Steam temperature in evaporator 4	°C	T26	Steam temperature in evaporator 5	°C
T27	Steam temperature evaporator 6	°C	T28	Temperature of the steam from boilers to evaporation	°C
W1	Mass flow of juice to heat exchanger R10	t/h	W2	Mass flow of juice to heat exchanger R3	t/h
W3	Mass flow of juice to heat exchanger R3B	t/h	W4	Mass flow of juice to heat exchanger R4	t/h
W5	Mass flow of juice to heat exchanger R8	t/h	W6	Mass flow of juice to heat exchanger R9	t/h
W7	Mass flow of juice from thin juice tank	t/h	W8	Mass flow of juice evaporator 6 outlet	t/h
W9	Mass flow of steam from boilers to evaporation	t/h	T29	Temperature of the juice in evaporator 1	°C
T30	Temperature of the juice in evaporator 2	°C	T31	Temperature of the juice in evaporator 3	°C
T32	Temperature of the juice in evaporator 4	°C	T33	Temperature of the juice in evaporator 5	°C
T34	Temperature of the juice in evaporator 6	°C	P7	Pressure of the steam from boilers to evaporation	bar

4.1.1. Experimental methodology

The aforementioned training data are used for knowledge based NPLS method, following the different steps introduced in Algorithm 1. The first step consists of using the available expertise to implement glimpsed new variables, usually as non-linear relationships of the original variables, i.e., to calculate $\mathbf{X}_{NL1}, \mathbf{X}_{NL2}, ..., \mathbf{X}_{NLj}$.

Here, this knowledge of the process involves non-linear relationships regarding the thermodynamic calculation of the vapour pressure of the juice, which depends on the dry substance content [62].

In fact, the soft sensor could be developed using only the physical model equations with reasonable results, but the use of the physical model has some drawbacks that are described in [2]. Physical equations are for pure sucrose solutions in ideal conditions and thermodynamic equilibrium, and this is not true in an industrial environment. Also, these equations are very sensitive to small variations in the two variables involved, steam pressure and juice temperature at the evaporator output, so the calculations are not very robust. This can be observed in Figure 3, where the estimation is close to the real value, but is very noisy. When using PLS and NPLS methods, more process variables are involved in the °Brix calculation, so the resulting model provides more robust estimates.

Nevertheless, although the direct use of these models does not deliver a very good estimation, the knowledge of the plant and the use of these models provide expertise and knowledge to establish some underlying behavioural relationships between variables, which are useful into the PLS problem. These relationships are the same in each evaporation effect, so $\mathbf{bx}$, $\mathbf{T_j}$, $\mathbf{T_v}$ and $\mathbf{T_x}$ can be related for the six effects in the following forms:

$$\mathbf{nlneq1} = \ln([\mathbf{T_j} \quad \mathbf{T_v}]) \tag{14}$$

$$\mathbf{nlneq2} = \mathbf{T_j}^2 \tag{15}$$

$$\mathbf{nlneq3} = \ln(1/\ln([\mathbf{T_j} \quad \mathbf{T_v}])) \tag{16}$$

$$\mathbf{nlneq4} = \sqrt{\ln([\mathbf{T_j} \quad \mathbf{T_v}])} \tag{17}$$

$$\mathbf{nlneq5} = \sqrt{1/\ln(\mathbf{T_x})} \tag{18}$$

$$\mathbf{nlneq6} = \sqrt{\ln\left(\mathbf{T_j}/\mathbf{T_x}^2 + \mathbf{T_x}\right)} \tag{19}$$

where $\mathbf{T_j}$ are the juice temperatures in °C at the output of all effects

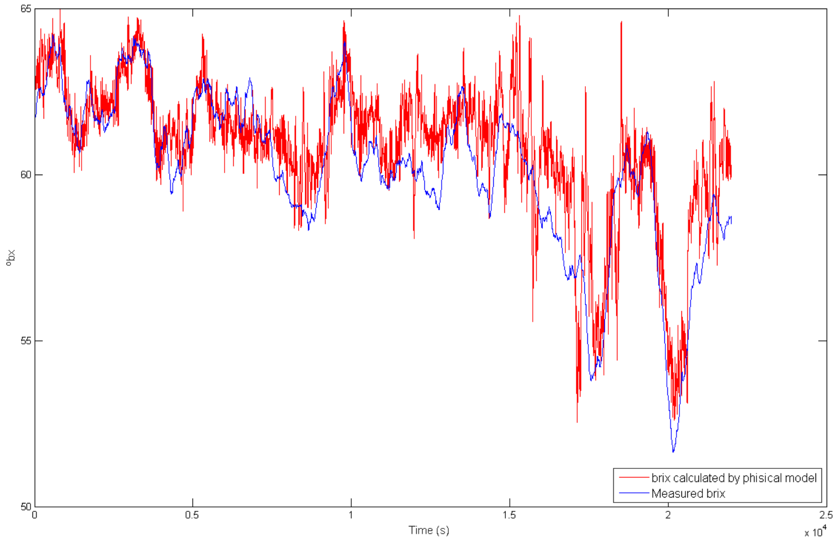

Figure 3: Results for °Brix calculated using the physical model

corresponding to variables T29 to T34 in table 1, $\mathbf{T_v}$ are the temperatures of the steam leaving the evaporator chambers for all the effects corresponding to variables T22 to T27 in table 1, $\mathbf{T_x}$ is the correction of the temperature of the solution due to the presence of sugar. $\mathbf{T_x}$ can be supposed as the difference between the juice temperatures $\mathbf{T_j}$ and the steam temperatures $\mathbf{T_v}$. In equations 14, 16, 17 and 18, $[\mathbf{T_j} \quad \mathbf{T_v}]$ represent the concatenation of $\mathbf{T_j}$ and $\mathbf{T_v}$ vectors.

Also, the steam pressure of the vapour generated in the evaporators $\mathbf{P}$ and the °Brix can be guessed as:

$$nlneq7 = \ln(\log_{10}(\mathbf{P})) \tag{20}$$

Finally, it is known that the °Brix obtained in the evaporation basically is inversely proportional to the amount of juice flowing through the evaporation $\mathbf{F}$, so another relationship can be included.

$$nlneq8 = 1/\mathbf{F} \tag{21}$$

The second step (see Algorithm 1) is to build the augmented matrix. So in this case, the variables resulting from applying these eight equations for each effect to the corresponding variables are added as extra columns to the matrix of input variables $\mathbf{X}$ to form the augmented matrix $\mathbf{X}_a$.

Also, in order to take into account the influence of past values in the calculation of the predicted variable, the dynamic problem is solved including past values of the variables as extra columns in the observed matrix to obtain the final input matrix $\mathbf{X}_f$ defined in Algorithm 1. To be precise for this model $L = 4$ lag variables are introduced, i.e., $\mathbf{X}_f = [\mathbf{X}_a(t), [\mathbf{X}_a(t-1), ..., [\mathbf{X}_a(t-4)]$. This value is calculated as a trade-off between improving the performance of the static NPLS and not increasing too much the dimension of the final input matrix: $\mathbf{X}_f$, i.e., its number of columns: $(m+j)*(L+1) = (50+46)*(4+1)$. With this final matrix, the Modified Kernel PLS algorithm is executed to obtain the NPLS soft sensor.

4.1.2. Results and discussion

Here, Step 2 of the proposal is applied, i.e., the on-line prediction of the °Brix, using the RNPLS method (see Algorithm 2). To determine the performance of the proposed RNPLS, the estimation made using the RNPLS algorithm is compared with dynamic PLS (DPLS) method and with the recursive dynamic PLS both with $L = 4$ past values of the observed variables, i.e., $\mathbf{X}_f = [\mathbf{X}(t), \mathbf{X}(t-1), ..., \mathbf{X}(t-4)]$ and with the knowledge based NPLS trained in the first step. The comparison is made in terms of the mean square error (MSE) between the real value of the variable and its predicted value with validation data for each of the methods considered. The results of the mean squared error (MSE) obtained for the different approaches with the validation data can be seen in table 2.

The worst results are obtained for the case of the physical model, with a very noisy prediction, that can be observed in Figure 3.

For the case of the non-recursive approaches, the results obtained using the knowledge based NPLS are compared with the DPLS method. In both cases, the number of latent variables considered are 12. The results can be seen in Figure 4 and it can be clearly observed as the NPLS approach improves significantly the estimation.

In both linear and non-linear predictions, i.e., in DPLS and NPLS soft sensors, it is clear that the main error is due to the fact that the mean value in the validation zone, i.e. with new data collected from the plant, is lower than the mean for the training zone, because of the time-varying behaviour

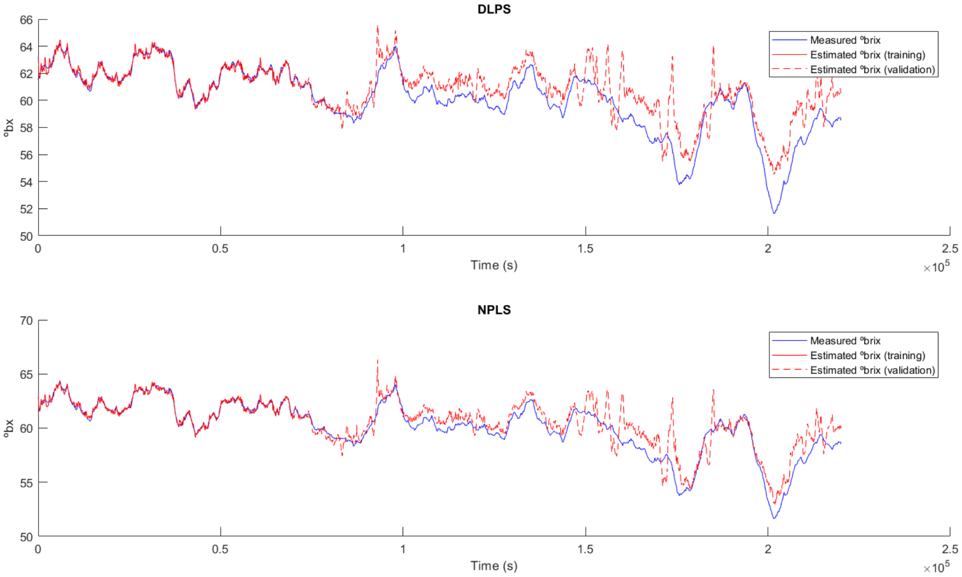

Figure 4: Comparison between °Brix predictions made with the dynamic PLS (upper plot) and the non-linear dynamic PLS (lower plot).

of the real industrial process. This produces higher values for the predicted variable than the real ones. This problem can be dealt with in the recursive version of the algorithm.

The recursive approach considers that the NPLS is able to adapt to the new operation conditions by including new information that can be added to the previously available information and use it to update the covariance matrices. In this case, the soft sensor is used on-line, making predictions concerning the process variable, while the information relative to the output can be added when new laboratory analysis results are available, which are obtained each N_s sampling periods. The goodness of the adaptation depends on the frequency of this analysis.

Now, with the NPLS trained in Step 1, new input data, $\mathbf{x}$, are collected from the plant in each sampling time, and it is supposed that new information for the target vector, $\mathbf{y}$, is available every 1440, 720 or 360 sampling periods, i.e., three study cases are under consideration, that corresponds with 1, 2 and 4 hours of sampling.

So, following Step 2 of the method (Algorithm 2), the recursion or adap-

Type of sensor	Linear MSE	Non-linear MSE
Dynamic PLS	2.5175	1.5871
Recursive PLS Ns = 1440	2.0153	1.3210
Recursive PLS Ns = 720	1.7501	1.2252
Recursive PLS Ns = 360	1.3113	0.9147
Physical model	2.7205	

Table 2: MSE for the tested approximations

tation of the NPLS model is carried out every N_s sampling times. When this recursive approach is used, the NPLS sensor software improves its performance. In Table 2, the results for all the performed experiments are shown, it is possible to observe thatthe proposed method, the RNPLS algorithm, obtains better results than the linear recursive PLS for all cases, and as expected, the more frequently new information is added, i.e., the lower N_s is, the better the prediction of the unknown variable. i.e., the best results are for the RNPLS with new information included in the model every $N_s = 360$ sampled periods (1 hour). In Figure 5, the results for the recursive RNPLS approximation are also shown.

To visualize the contribution of the different non-linear terms to the °Brix estimation, the absolute value of the higher 50 PLS regression coefficients $\mathbf{B}_{PLS}$ for the observed variables X have been plotted in Figure 6. The original non transformed variables are named following the notation defined in Table 4, adding $\_D1$, $\_D2$, $\_D3$ and $\_D4$ to represent delayed variables. The variables obtained via non-linear transformations, have been named using the correspondent non-linear equation, followed by the name variable described in Table 4 and finally the delay indication. A simple observation of the upper plot of Figure 6, shows that different non-linear terms contribute significantly to the predicted variable estimation. The coefficients shown correspond to the values obtained using the training data of the RNPLS algorithm with $N_s = 360$, but these are not constant and evolve during the recursive estimation. The evolution of the 5 most significant $\mathbf{B}_{PLS}$ coefficients is plotted in the lower graph of Figure 6. It can be seen as the coefficients are stable during the validation experiment but their values vary to adapt the model to the new information as it becomes available.

A more detailed analysis of the contribution of the non-linear terms to the estimation is displayed in Table 3 where the sum of the $\mathbf{B}_{PLS}$ coefficients

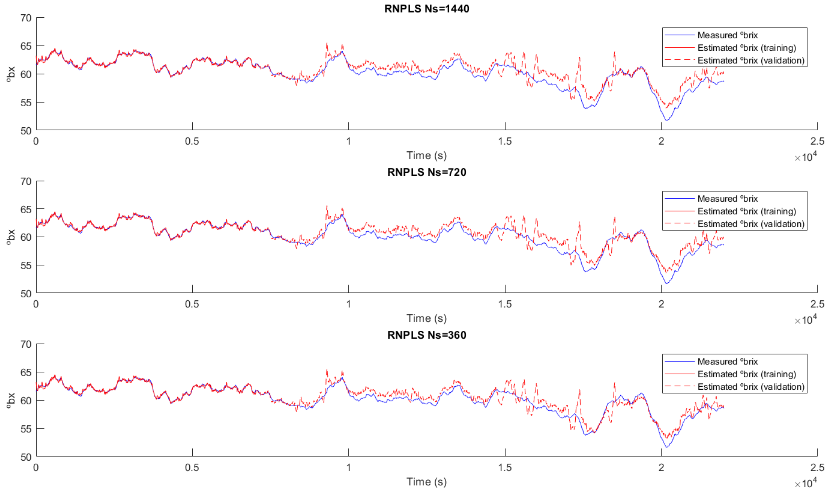

Figure 5: Results for the Soft Sensor using the RNPLS. Upper plot includes new information every 3600 seconds, intermediate plot adds new information every 7200 s and the lower figure adds new information every 14400 s.

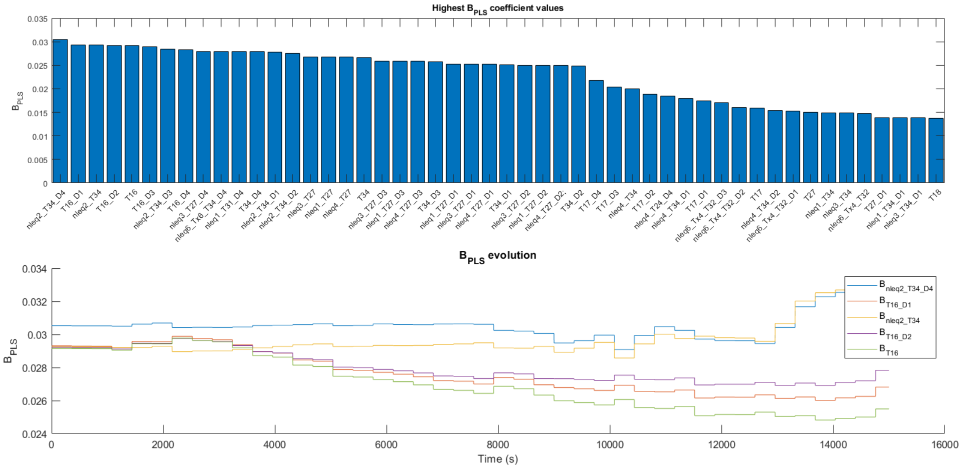

Figure 6: B_{PLS} values for the $Ns = 360$ recursive NPLS. Upper plot: 50 highest values of the B_{PLS} coefficients. Lower plot: evolution of the 5 highest $\mathbf{B}_{PLS}$ coefficients during the recursive estimation.

for each non-linear transformation and its percentage over the sum of the $\mathbf{B}_{PLS}$ coefficients is shown. It can be seen as the contribution of the terms obtained using non-linear transformations is bigger to 58%. The most significant contributions correspond to the transformations achieved applying equations 14, 16 and 17.

Observation	$\sum \mathbf{B}_{PLS}$	Percentage of the total
Linear terms	1.3264	41.63 %
Non-linear terms	1.8598	58.37%
nleq1 (14)	0.3452	10.83%
nleq2 (15)	0.1431	4.49%
nleq3 (16)	0.3455	10.84%
nleq4 (17)	0.3725	11.69%
nleq5 (18)	0.1463	4.59%
nleq6 (19)	0.2086	6.55%
nleq7 (20)	0.1383	4.34%
nleq8 (21)	0.1603	5.03%
Total	3.1862	100%

Table 3: Contribution of $\mathbf{B}_{PLS}$ coefficients

4.2. Case Study 2: Wastewater Treatment Plant

The second example in which the RNPLS method is tested is the Benchmark Simulation Model no. 2 (BSM2) developed by the Working Groups of COST Action 682 and 624 and the IWA Task Group [63]. This benchmark consists of a Wastewater Treatment Plant (WWTP) that purifies contaminated water coming from urban activities making the effluent adequate for pouring into a river or for use in other applications.

The layout of the BMS2 is shown in Figure 7. The plant can be divided into various stages: a primary clarifier, activated sludge reactors where biological reactions are carried out to remove nitrogen and the organic matter, a secondary clarifier, where clean water is obtained, and an anaerobic digester where pathogenic microbes are removed from the sludge to make them suitable to be sent to a dumping site. A complete review on data driven soft sensors for wastewater treatment plants can be found in [64] and the first applications of PLS in WWTP to predict process variables are in [65, 66, 67].

In the case described in this paper, the selected variable to be estimated by the soft sensor is the Chemical Oxygen Demand (COD) of the effluent, i.e.,

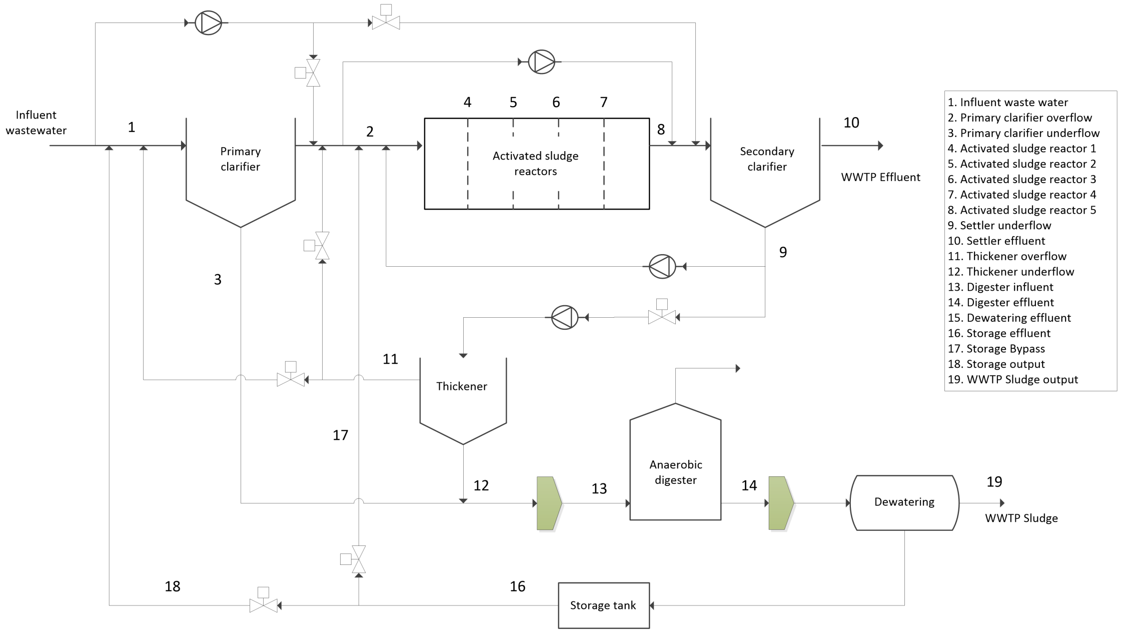

Figure 7: Process Diagram of the BMS2 plant

the output matrix **Y**. As secondary variables, used to estimate the COD, 114 process variables have been included to make up the original input matrix **X**. These variables correspond to the 6 variables shown in Table 4, for the 19 currents numbered 1 to 19 in Figure 7. The benchmark plant model is coded in Simulink (Matlab), and the simulation model has been executed to generate data for 146 days under normal operation conditions. Measurements were recorded every 15 minutes, so there are 14000 samples, the first 7000 samples for the training data set and the rest of the data for the testing data set, i.e., the other 7000 samples.

4.2.1. Experimental methodology

The first step of the proposal is to apply the available knowledge about the system to obtain extra generic underlaying relationships (Algorithm 1) between variables for improving the performance during the computation of the RNPLS. In this case, the physical-chemical properties of the process are very complex and not very well known, so the underlaying relationships between the measurements observed from the process variables and the estimated variable are not really known, so in order to save this knowledge gap we use a software capable of generating algebraic models from data,

Identifier	Variable	Units
ALK	Alkalinity	$g_{COD} \cdot m^3$
O2	Dissolved Oxygen	$g_{COD} \cdot m^3$
TSS	Total suspended Solids	$g_{SS} \cdot m^3$,
Q	Wastewater Flow	$m^3 \cdot d^{-1}$
T	Wastewater Temperature	$°C$
N2	Total Nitrogen	$mg \cdot L^{-1}$

Table 4: List variables used

in this case the Alamo software [56], to obtain underlaying knowledge about the behaviour between variables, complementing the expertise skill available. According to all this, the following non-linear relationships were established using both knowledge sources:

$$\begin{aligned}
\mathbf{nlneq1} &= 1/\mathbf{TSS} \quad (22)\\
\mathbf{nlneq2} &= \mathbf{TSS}^2 \quad (23)\\
\mathbf{nlneq3} &= 1/\mathbf{ALK} \quad (24)\\
\mathbf{nlneq4} &= \mathbf{ALK}^2 \quad (25)\\
\mathbf{nlneq5} &= 1/\mathbf{O2} \quad (26)\\
\mathbf{nlneq6} &= \mathbf{N2}^2 \quad (27)\\
\mathbf{nlneq7} &= 1/\mathbf{N2} \quad (28)\\
\mathbf{nlneq8} &= 1/\mathbf{T} \quad (29)\\
\mathbf{nlneq9} &= 1/\exp(\mathbf{O2}) \quad (30)
\end{aligned}$$

The relationship between process variables are supposed to be the same for all the process currents, so the previous equations are applied to all of them.

Next, the augmented matrix $\mathbf{X}_a$ is built, including the non-linear relationships in the original matrix variable $\mathbf{X}$. Also, due to the important transport delays of the plant, a dynamic PLS has been implemented, where past measurements are included as independent variables in the NPLS model, for this example $L = 2$ lag variables are introduced. This value, as before, is calculated as a trade-off between improving the performance of the static NPLS and not increasing by too much the number of columns of the final input

Type of sensor	Normal operation MSE		Operation disturbances MSE	
	Linear	Non-linear	Linear	Non-linear
Dynamic PLS	5.3411	3.9522	244.0255	72.1629
Recursive PLS Ns = 100	5.3410	4.1454	10.9437	10.3699
Recursive PLS Ns = 30	4.3807	3.6725	15.2080	4.8380
Recursive PLS Ns = 10	3.9300	3.5956	6.6147	3.9988

Table 5: MSE for the tested approximations

matrix: $\mathbf{X}_f$, $((m+j)*(L+1))$. In this case, the final input matrix is:

$$\mathbf{X}_f = \begin{bmatrix}\mathbf{X}_a(t) & \mathbf{X}_a(t-1) & \mathbf{X}_a(t-2)\end{bmatrix} \tag{31}$$

The covariance matrices are calculated with this matrix, as it was explained in section 3.1, and the Modified Kernel PLS algorithm is executed to obtain the knowledge based NPLS soft sensor.

4.2.2. Results and discussion

Here, Step 2 of the proposal is applied, i.e., the on-line prediction of the *COD* using the RNPLS method (see Algorithm 2). The estimation performance of the proposal is compared, as before, with the dynamic PLS method (DPLS), with only the original variables $\mathbf{X}$ and $L = 2$ past values, i.e., $\mathbf{X}_f = [\mathbf{X}(t), \mathbf{X}(t-1), \mathbf{X}(t-2)]$ and the knowledge based NPLS trained in the first step and with the recursive linear dynamic PLS model (RDPLS) with also two lag variables. The comparison is made taking into account the mean square error (MSE) between the real value of the variable and its predicted value with validation data for each one of the methods considered.

In Figure 8, DPLS and the proposed NPLS using non-linear relationships are compared. It can be seen how the NPLS based on knowledge achieves some improvement over the linear DPLS model. This can be observed even better in Table 5, where the MSE for the different models considered in this paper are compared with validation data. In the first column of Table 5, it is possible to observe how the NPLS can reduce the MSE index of the linear DPLS model under normal conditions.

Next, the proposed RNPLS method is applied on-line, making predictions on the output process variable each sampling time. However, the recursive updating action is only carried out when new information of the target vari-

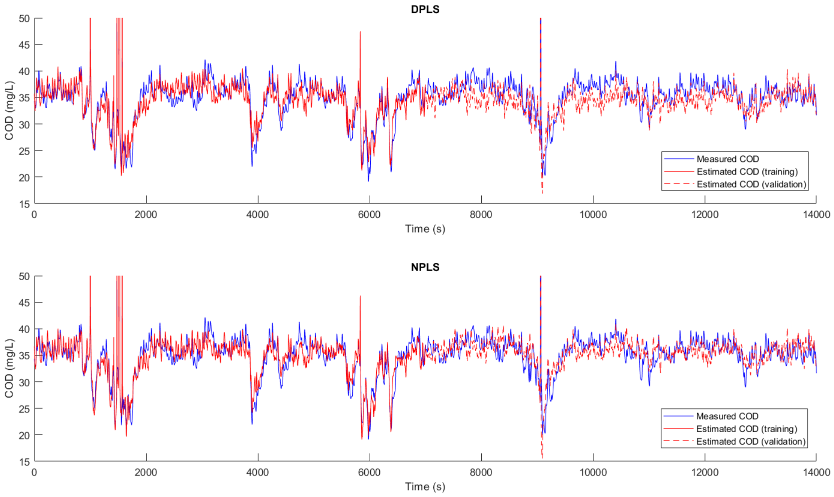

Figure 8: Comparison between COD predictions of the dynamic linear PLS (DPLS) (Upper figure) and the knowledge based dynamic non-linear PLS (NLPS) (lower plot).

able is obtained from the plant i.e. every N_s sampling period, because of the slow sampling rate of the output variable in real industrial scenarios.

As before, three scenarios are studied, the first is when $\mathbf{y}$ is measured or obtained by laboratory analysis each $N_s = 10$ samples, i.e. every 150 minutes, a time relatively close to the sampling time of the system, 15 minutes, as explained above in this section. The second case is when $N_s = 30$ sampled times, i.e. every 7.5 hours, an intermediate time; while the third case is when $N_s = 100$ sampled times, i.e. every 25 hours, a time very far from the sampling time of the system. In Figure 9, the results for the RNPLS are shown for the chosen update frequencies. The new information provides better approximations for the predicted variable, i.e., the RNPLS obtains better results than the recursive DPLS in all three cases and, for two cases, it is better than the NPLS, and as expected, the more frequently the information is added to the RNPLS, the better the results, as it is possible to see in the first column of Table 5, where the best result is obtained when the RNPLS soft sensor is updated every $N_s = 10$ sample times.

However, in this case, the improvement is not very impressive and it can be hardly observed in Figure 9, due to the fact that the process is maintained close to the operation point, i.e., within a limited range of operation. When

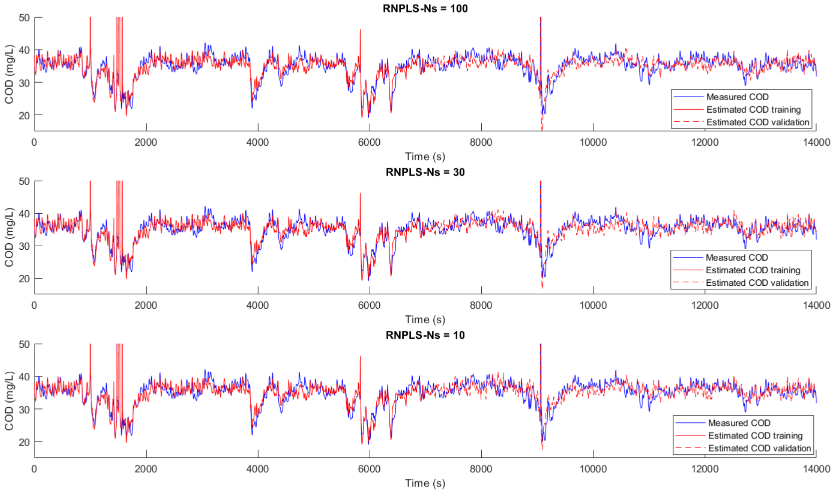

Figure 9: Results for the Soft Sensor using the RNPLS approach with various periods in which new information is added, i.e., when the updating procedure is carried out, upper figure $N_s = 100$, intermediate figure $N_s = 30$ and lower figure $N_s = 10$ sampled times

the process changes abruptly due to changes in the operation conditions, the improvement provided by the RNPLS is much more significant. In the next experiment, while the soft sensor is working on-line, an increase is provoked in one of the input variables, specifically in the flow of the thickener overflow. In this situation, both the DPLS and the NPLS methods are not able to provide good results, showing a high bias in the predicted variable, while the RNPLS is able to adapt to the perturbation in the three scenarios considered, as can be seen in Table 5. The second column of Table 5 shows the MSE index when the aforementioned disturbance is introduced in the process, and it is possible to see how, in this case, the improvement of the RNPLS over the other methods is very important. Finally, in Figure 10, the NPLS and RNPLS with an update period of $N_s = 30$ samples are compared, to show the best results obtained with the proposed RNPLS soft sensor.

5. CONCLUSIONS

This paper presents the design of software sensors based on knowledge for the estimation of unknown variables in real industrial processes. This work proposes a new methodology to integrate the available knowledge about the

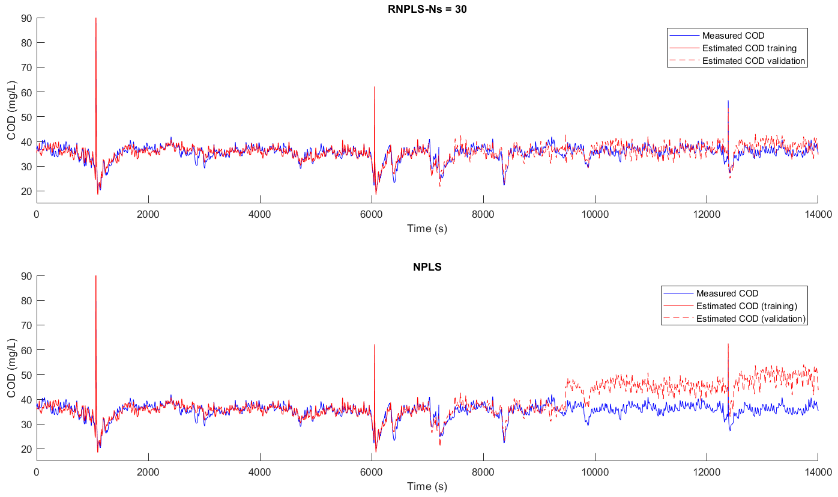

Figure 10: Comparison between NPLS and RNPLS with Ns = 30 for an experiment with a deviation in the flow of the thickener overflow.

process in a recursive non-linear PLS (RNPLS) method. This permit both problems of the non-linearity and the time-varying feature of the industrial processes to be overcome. First, an NPLS model is made by carrying out the usual PLS regression over an extended matrix of input, built with the original process variables and with non-linear transformations of some of those variables sourced from the expert knowledge available around the process, which permits to deal with the non-linearity nature of the industrial processes. Also, some delayed variables can be introduced in the augmented matrix to consider the dynamic nature of the real processes. Finally, a recursive version of the NPLS is used to modify the model and adapt it to the process changes when new information concerning the target vector is available.

This RNPLS algorithm was applied to two case studies: to estimate the sugar concentration in an evaporation station of a real sugar industry and to predict the Chemical Oxygen Demand variable in a benchmark of a wastewater treatment plant. It was also compared with the PLS and knowledge based NPLS algorithms. The results show that when the system is working in an operation point without changes in the system, the RNPLS gives a better result, but one which is not very far from the other methods for predicting the unknown variable, as happens in the example of the Wastewater treatment plant. However, if the system is time-varying, as is usual in real industrial plants, the proposed RNPLS algorithm will give better results, with quite significant improvements regarding the other methods, as tested in the two study cases. Thus, the proposed methodology including knowledge in the RNPLS algorithm allows an improvement in the prediction for non-linear and time-varying processes.

Acknowledgements

This work has been partially supported by the Spanish Government and the European Regional Development Fund (FEDER) through the Project no. (MINECO/FEDER) DPI2015-67341-C2-2-R.

References

[1] P. Kadlec, B. Gabrys, S. Strandt, Data-driven soft sensors in the process industry, Computers & Chemical Engineering 33 (2009) 795–814.

[2] D. Garcia-Alvarez, A. Merino, R. Marti, M. J. Fuente, Soft sensor design for dry substance content estimation in the sugar industry, Sugar Industry 137 (2012) 645–653.

[3] W. Yan, H. Shao, X. Wang, Soft sensing modeling based on support vector machine and Bayesian model selection, Computers & Chemical Engineering 28 (2004) 1489–1498.

[4] D. Aguado, A. Ferrer, A. Seco, J. Ferrer, Comparison of different predictive models for nutrient estimation in a sequencing batch reactor for wastewater treatment, Chemometrics and Intelligent Laboratory Systems 84 (2006) 75–81.

[5] A. Höskuldsson, PLS regression methods, Journal of Chemometrics 2 (1988) 211–228.

[6] H. Wold, Non linear estimation by iterative least squares procedures, Wiley, New York, 1966.

[7] F. Lindgren, P. Geladi, S. Wold, The kernel algorithm for PLS, Journal of Chemometrics 7 (1993) 45–59.

[8] P. A. Hassel, Nonlinear Partial Least Squares, Ph.D. thesis, University of Newcastle (2003).

[9] S. Wold, N. Kettaneh-Wold, B. Skagerberg, Nonlinear PLS modeling, Chemometrics and Intelligent Laboratory Systems 7 (1989) 53–65.

[10] A. Berglund, S. Wold, INLR, Implicit Non-linear Latent Variable Regression, Journal of Chemometrics 11(2) (1997) 141–156.

[11] J. F. Durand, R. Sabatier, Additive splines for partial least squares regression, Journal of the American Statistical Association 92 (440) (1997) 1546–1554. doi:10.1080/01621459.1997.10473676.

[12] C. Li, H. Ye, G. Wang, J. Zhang, A recursive non linear PLS algorithm for adaptive nonlinear process modelling, Chemical Engineering Technology 28(2) (2005) 141–152.

[13] G. Baffi, E. Martin, A. Morris, Non-linear projection to latent structures revisited: the quadratic PLS algorithm, Computers & Chemical Engineering 23 (1999) 395–411.

[14] S. J. Qin, T. J. McAvoy, Nonlinear PLS modeling using neural networks, Computers & Chemical Engineering 16(4) (1992) 379–391.

[15] G. Baffi, E. Martin, A. Morris, Non-linear projection to latent structures revisited (the neural network PLS algorithm), Computers & Chemical Engineering 23 (1999) 1293–1307.

[16] D. Wilson, G. Irwin, G. Lightbody, Nonlinear PLS modelling using radial basis functions, in: Proceedings of the 1997 American Control Conference (Cat. No.97CH36041), IEEE, 1997, pp. 3275–3276 vol.5. doi:10.1109/ACC.1997.612069.

[17] Y. Bang, C. Yoo, I. Lee, Nonlinear PLS modeling with fuzzy inference system, Chemometrics and Intelligent Laboratory Systems 64 (2003) 137–155.

[18] C. K. Yoo, Y. H. Bang, I. B. Lee, P. A. Vanrolleghem, C. Rosen, Application of fuzzy partial least squares (FPLS) modeling nonlinear biological processes, Korean Journal of Chemical Engineering 21(6) (2004) 1087–1097.

[19] D. Searson, M. Willis, G. Montague, Co-evolution of non-linear PLS model components, Journal of Chemometrics 21 (2007) 592–603.

[20] X. Zhang, W. Yan, H. Shao, Nonlinear multivariate quality estimation and prediction based on kernel partial least squares, Industrial & Engineering Chemistry Research 47 (2008) 1120–1131.

[21] S. H. Woo, C. O. Jeon, Y. S. Yun, H. Choi, C. Lee, D. S. Lee, On-line estimation of key process variables based on kernel partial least squares in an industrial cokes wastewater treatment plant, Journal of Hazardous Materials 161(1) (2009) 538–544.

[22] T. Mejdell, S. Skogestad, Composition estimator in a pilot-plant distillation column using multiple temperatures, Industrial & Engineering Chemistry Research 30 (12) (1991) 2555–2564. `doi:10.1021/ie00060a008`.

[23] R. Bakirov, B. Gabrys, D. Fay, Multiple adaptive mechanisms for data-driven soft sensors, Computers & Chemical Engineering 96 (2017) 42–54.

[24] K. Helland, H. E. Bernsten, O. Borgen, H. Martens, Recursive algorithm for partial least squares regression , Chemometrics and Intelligent Laboratory Systems 14 (1991) 129–137.

[25] S. Qin, Recursive PLS algorithms for adaptive data modeling, Computers & Chemical Engineering 22(4-5) (1998) 503–514.

[26] B. S. Dayal, J. F. MacGregor, Recursive exponentially weighted PLS and its applications to adaptive control and prediction, Journal of Process Control 7(3) (1997) 169–179.

[27] X. Wang, U. Kruger, B. Lennox, Recursive partial least squares algorithms for monitoring complex industrial processes, Control Engineering Practice 11 (2003) 613–652.

[28] H. W. Lee, M. W. Lee, J. Park, Robust adaptive partial least squares modeling of a full-scale industrial wastewater treatment process, Industrial & Engineering Chemistry Research 46(3) (2007) 955–964.

[29] F. Ahmed, S. Nazir, Y. K. Yeo, A recursive PLS-based soft sensor for prediction of the melt index during grade change operations in HDPE plant, Korean Journal of Chemical Engineering 26(1) (2009) 14–20.

[30] O. Xu, Y. Fu, H. Su, L. Li, A selective moving window partial least squares method and its application in process modeling, Chinese Journal of Chemical Engineering 22(7) (2014) 799–804.

[31] J. Liu, D. S. Chen, J. F. Shen, Development of self-validating soft sensors using fast moving window partial least squares, Industrial & Engineering Chemistry Research 49 (2010) 11530–11546.

[32] X. Wang, U. Kruger, G. Irwin., Process monitoring approach using fast moving window PCA, Industrial & Engineering Chemistry Research 44 (2005) 5691–5702.

[33] I. Jaffel, O. Taouali, M. F. Harkat, H. Messaoud, Moving window KPCA with reduced complexity for nonlinear dynamic process monitoring, ISA Transactions 64 (2016) 184–192.

[34] L. Zhou, G. Li, Z. Song, S. J. Qin, Autoregressive dynamic latent variable models for process monitoring, IEEE Transactions on Control Systems Technology 25 (2017) 366–373.

[35] L. Shang, J. Liu, Y. Zhang, W. G, Efficient recursive canonical variate analysis approach for monitoring time-varying processes, Journal of Chemometrics 31 (2017) 1–10.

[36] H. Yu, F. Khan, Improved latent variable models for nonlinear and dynamic process monitoring, Chemical Engineering Science 168 (2017) 325–338.

[37] I. Stanimirova, M. Daszykowski, B. Walczak, Dealing with missing values and outliers in principal component analysis, Talanta 72(1) (2007) 172–178.

[38] A. Ferrer, D. Aguado, S. Vidal-Puig, J. M. Prats, M. Zarzo, PLS: A versatile tool for industrial process improvement and optimization, Applied Stochastic Models in Business and Industry 24(6) (2008) 551–567.

[39] G. Heo, P. Gader, H. Frigui, RKF-PCA: robust kernel fuzzy PCA., Neural networks 22(5-6) (2009) 642–650.

[40] F. Arteaga, A. Ferrer, Framework for regression-based missing data imputation methods in on-line MSPC, Journal of Chemometrics 19 (2005) 439–447.

[41] B. Lin, B. Recke, T. Schmidt, J. Knudsen, S. Jrgensen, Data-driven soft sensor design with multiple-rate sampled data: a comparative study., Industrial & Engineering Chemistry Research 48(11) (2009) 5379–5387.

[42] N. Lu, Y. Yang, F. Gao, F. Wang, Multirate dynamic inferential modeling for multivariable processes, Chemical Engineering Science 59(4) (2004) 855–864.

[43] C. Shang, X. Huang, J. Suykens, D. Huang, Enhancing dynamic soft sensors based on DPLS: A temporal smoothness regularization approach., Journal of Process Control 28 (2015) 17–26.

[44] L. Xie, H. Yang, B. Huang, FIR model identification of multirate processes with random delays using EM algorithm, AIChE Journal 59(11) (2013) 4124–4132.

[45] T. J. Rato, M. S. Reis, Multiresolution soft sensors: a new class of model structures for handling multiresolution data, Industrial & Engineering Chemistry Research 56 (2017) 3640–3654.

[46] A. Llus-Serra, S. Vila-Marta, T. Escobet-Canal, Formalism for a multiresolution time series data base model, Information Systems 56 (2016) 19–35.

[47] M. Reis, T. Rato, Multiresolution analystics for large scale industrial processes, IFAC Paper on-line 51-18 (2018) 464–469.

[48] G. Leu, H. Abbass, A multi-disciplinary review of knowledge acquisition methods: From human to autonomous eliciting agents, Knowledge-Based Systems 105 (2016) 1–22.

[49] S. Quintana-Amate, P. Bermell-Garcia, A. Tiwari, Transforming expertise into Knowledge-Based Engineering tools: A survey of knowledge sourcing in the context of engineering design, Knowledge-Based Systems 84 (2015) 89–97.

[50] I. Helland, Some theoretical aspects of partial least squares regression, Chemometrics and Intelligent Laboratory Systems 58 (2) (2001) 97–107.

[51] P. Geladi, B. Kowalski, Partial least-squares regression: a tutorial, Analytica Chimica Acta 185 (1) (1986) 1–17.

[52] P. Kadlec, R. Grbić, B. Gabrys, Review of adaptation mechanisms for data-driven soft sensors, Computers & Chemical Engineering 35 (1) (2011) 1–24.

[53] B. S. Dayal, J. F. MacGregor, Improved PLS algorithms, Journal of Chemometrics 11 (1997) 73–85.

[54] T. Fortescue, L. Kershenbaum, B. Ydstie, Implementation of self-tuning regulators with variable forgetting factors, Automatica 17(6) (1981) 831–835.

[55] D. S. Patle, Z. Ahmad, G. P. Rangaiah, Operator training simulators in the chemical industry: review, issues, and future directions, Reviews in Chemical Engineering 30 (2). `doi:10.1515/revce-2013-0027`.

[56] Z. T. Wilson, N. V. Sahinidis, The ALAMO approach to machine learning, Computers & Chemical Engineering 106 (2017) 785–795.

[57] A. Merino, R. Alves, L. Acebes, A training simulator for the evaporation section of a beet sugar production process, The 2005 European Simulation and Modelling.

[58] A. Sanchez-Fernandez, F. J. Baldan, G. Sainz-Palmero, J. Benitez, M. J. Fuente, Fault detection based on time series modeling and multivariate statistical process control, Chemometrics and Intelligent Laboratory Systems 182 (2018) 57–69.

[59] R. Rendall, I. Castillo, A. Schmidt, S. T. Chin, L. H. Chiang, M. Reis, Wide spectrum feature selection (WiSe) for regression model building, Computers & Chemical Engineering 121 (2019) 99–110.

[60] R. A. Mc Ginnis, Beet Sugar Technology, 3rd Edition, Literary Licensing, LCC.

[61] A. Merino, L. F. Acebes, R. Alves, C. de Prada, Real Time Optimization for steam management in an evaporation section, Control Engineering Practice 79 (2018) 91–104. `doi:10.1016/j.conengprac.2018.07.010`.

[62] Z. Bubnik, P. Kadlec, D. Urban, Sugar technologists manual : chemical and physical data for sugar manufacturers and users, Albert Bartens, 1995.

[63] IWA, IWA Task Group on Benchmarking of Control Strategies for WWTPs.
URL `http://www.benchmarkwwtp.org`

[64] H. Haimi, M. Mulas, F. Corona, R. Vahala, Data-derived soft-sensors for biological wastewater treatment plants: an overview, Environmental Modelling and Software 47 (2013) 88–107. `doi:10.1016/j.envsoft.2013.05.009`.

[65] P. Aarnio, P. Minkkinen, Application of partial least-squares modelling in the optimization of a waste-water treatment plant, Anal 191 (1986) 457–460.

[66] H. A. Blom, Indirect measurement of key water quality parameters in sewage treatment plants, Journal of Chemometrics 10 (5-6) (1996) 697–706.

[67] P. Teppola, S. Mujunen, P. Minkkinen, Partial least squares modeling of an activated sludge plant: A case study, Chemometrics and Intelligent Laboratory Systems 38 (1997) 197–208.