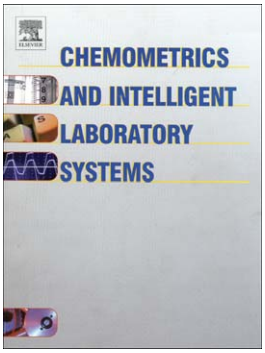


Mixture of Partial Least Squares Experts and Application in Prediction Settings with Multiple Operating Modes

Francisco A.A. Souza, Rui Araújo

PII: S0169-7439(13)00216-5
DOI: doi: 10.1016/j.chemolab.2013.11.006
Reference: CHEMOM 2732

To appear in: *Chemometrics and Intelligent Laboratory Systems*

Received date: 6 May 2013
Revised date: 1 August 2013
Accepted date: 13 November 2013

Please cite this article as: Francisco A.A. Souza, Rui Araújo, Mixture of Partial Least Squares Experts and Application in Prediction Settings with Multiple Operating Modes, *Chemometrics and Intelligent Laboratory Systems* (2013), doi: 10.1016/j.chemolab.2013.11.006

Mixture of Partial Least Squares Experts and Application in Prediction Settings with Multiple Operating Modes

Francisco A. A. Souza[a,b], Rui Araújo[a,b]

[a] *DEEC-Department of Electrical and Computer Engineering; University of Coimbra, Pólo II; Coimbra, Portugal*

[b] *ISR-Institute of Systems and Robotics; University of Coimbra, Pólo II; Coimbra, Portugal*

Abstract

This paper addresses the problem of online quality prediction in processes with multiple operating modes. The paper proposes a new method called mixture of partial least squares regression (Mix-PLS), where the solution of the mixture of experts regression is performed using the partial least squares (PLS) algorithm. The PLS is used to tune the model experts and the gate parameters. The solution of Mix-PLS is achieved using the expectation-maximization (EM) algorithm, and at each iteration of EM algorithm the number of latent variables of the PLS for the gate and experts are determined using the Bayesian information criterion. The proposed method, shows to be less prone to overfitting with respect to the number of mixture models, when compared to the standard [R2Q1]mixture of linear regression experts (MLRE). The Mix-PLS was successfully applied on three real prediction problems. The results were compared with five other regression algorithms. In all the experiments, the proposed method always exhibits the best prediction performance.

Keywords: soft sensors, mixture of experts, partial least squares, multiple modes, mix-pls

1. Introduction

Today, soft sensors have many applications in industry (e.g. fault detection, process monitoring, prediction of critical variables, and control) [1, 2, 3]. The major number of soft sensors applications consists on the prediction of critical or hard-to-measure[1] variables, where easy-to-measure variables (i.e. physical sensors) are used in a model to predict the hard-to-measure variable. Such model can be learned using the underlying knowledge about the process (white-box modeling), or using the available historical data to learn a data-driven model (data-driven modeling, or black-box modeling) or using both the underlying knowledge and the available data (gray-box modeling). The most popular data-driven models used in soft sensors applications are the multiple linear regression, with least squares (LS) or partial least squares (PLS) estimation methods, neural networks based models (NN), and

[1]The term hard-to-measure variable, employed here, refers to a variable which can not be measured by physical sensors, due the unavailability of sensor. Usually, this kind of variable is measured by laboratory analysis.

support vector regression (SVR) models. The PLS solution is the most popular and mostly applied solution when comparing to the other methods [4, 5, 6, 7, 8, 9]. Its popularity is motivated by its robustness under data collinearity, under measurement errors and under high dimensionality of input space, which are common characteristics in most industrial soft sensors applications. NN and SVR models are usually applied in situations where the input-output relationship is non-linear.

In almost all soft sensor applications, a single model is tuned using all available training samples, without distinguishing the operating modes of the process during the training phase. However, the existence of multiple operating modes in a process is an inherent characteristic of most industrial applications. Sometimes multiple operating modes result from external disturbances, as for example a change in feedstock or product grade or even changes such as the diurnal load variation of a power plant or the summer-winter operation of a refinery [10, 11]. In these situations, it would be beneficial for the prediction accuracy and reasonably, to consistently train a model for each operating mode of the process [12], or train a model for each set of correlated operating modes [13]; And during online operation, when a new sample is made available, the model which is the most adequate for this new sample is identified and then used to make the prediction. The identification of which model will be used is a key issue in the development [13, 14, 15], which can be done using expert knowledge [13] or using automatic tools, as finite mixture of Gaussian models (FMGM) [12].

In this context, in [13] the authors work on modeling the operating modes in a polymerization batch process case study. The correlated operating modes have been grouped, and then a separate PLS model is tuned for each set of correlated operating modes. During online operation, the incoming sample is assigned to the corresponding mode and its model is used for the prediction. However, in [13] the expert knowledge of operators has been used to determine the operating modes and in some cases this information can be not available.

Another approach, based on the FMGM, was proposed in [12]. In this work, the FMGM is used to automatically identify the different operating modes of the process. Then multiple localized Gaussian process regression models in the nonlinear kernel space were built to characterize the different dynamic relationships between process and quality variables within the identified operating modes. During online operation, the incoming sample is assigned automatically to the corresponding submodel, using the FMGM. The major drawback of [12] is that the determination of the operation modes and model tuning are done separately, i.e. the set of operating modes are determined independently of the model used. However, as verified in the case of study of [13], a model can be set for more than one operating mode, with the advantage of reducing the number of necessary models and increase the available number of samples for tuning each model. Another drawback of [12] is that the number of samples used for tuning each model is constrained by the number of samples of each operating mode, which can lead to poor modeling on the corresponding operating mode, depending on the chosen model and the available samples.

In this work, for the first time, it is proposed the use of a mixture of partial least squares (PLS) experts (Mix-PLS) for dealing with online prediction of critical variables in processes with multiple operating modes. The Mix-PLS will be derived from the framework of mixture of experts (ME) [16]. The ME models input-output observations by assuming that they have

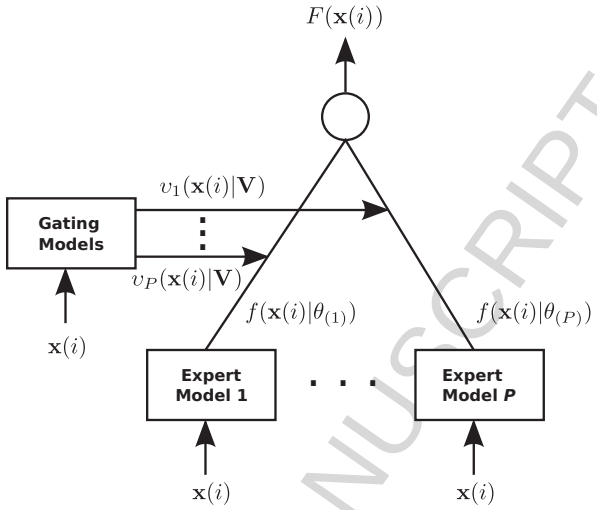


Figure 1: Mixture of linear regression models with P experts, where $\mathbf{x}(i)$ is an input sample, $\upsilon_p(\mathbf{x}(i), \mathbf{V})$ is the output of gating function for model p and $f(\mathbf{x}(i), \boldsymbol{\theta}_p)$ is the output of the linear model of expert p.

been produced by a set of different random sources (the random sources can be thought as operating modes). Each random source in the ME framework is modeled by an expert, and during the online operation the decision about which experts should be used is modeled by a gating function. Figure 1 illustrates this approach.

The learning of parameters in ME can be done using the maximum likelihood method and the expectation and maximization (EM) algorithm [17]. By modeling the experts by a [R2Q1,R2Q9]Gaussian linear regression and the [R2Q9]gating functions as a softmax function, the ME is then reduced to a mixture of linear regression experts (MLRE) [16, 18]. However, the standart MLRE cannot handle input collinearity, and the its solution is more prone to overfitting with respect to the number of experts used [19].

In this work the parameters of each expert and for each gating function are determined using the PLS algorithm. The solution of the parameters using the PLS algorithm overcomes the problem of collinearity of input data and also makes the Mix-PLS less prone to overfitting with respect to the number of mixture models. For the best of the authors's knowledge, there is no reference in the literature for solving the MLRE using PLS. See [19] for a [R2Q9]recent complete survey about mixture of experts.

In the experimental part, the Mix-PLS is then applied in three real prediction problems. Moreover, the proposed Mix-PLS method is compared in these problems with the state of the art algorithms for regression that are used in state of the art soft sensors methods: a single PLS model, a single layer neural network ([R1Q1]SLNN) trained using the gradient descent training algorithm, a least squares support vector regression (LS-SVR) with Gaussian kernel [20] [R1Q8]and with the multiplicative linear regression (MLR). The experimental results indicate that the recursive Mix-PLS outperforms the other methods. Moreover, the Mix-PLS has the advantage of being more interpretable than the non linear models with

respect to the parameters.

The paper is organized as follows. Section 3 reviews the PLS algorithm and its parameters selection. The proposed Mix-PLS method is presented in Section 4. Section 5 presents experimental results. Section 6 presents a discussion. Finally, Section 7 gives concluding remarks.

2. Notation

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.

3. Partial Least Squares

PLS regression is a method for finding the parameters $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_D]^T$ of a linear model of the form $f(\mathbf{x}, \boldsymbol{\theta}) = \theta_0 + \sum_{j=1}^{D} \theta_j x_j$ from a given a set of input-output samples $\Phi = \{(\mathbf{x}(i), y(i)); i = 1, \ldots, k\}$. This model is composed by a linear combination of the inputs for regression. The objective of the design of the linear combination is to maximize the covariance between the input and output spaces. The PLS estimation method is attractive because it works well on high dimensional data, noisy data, and data with collinearity, which are common characteristics in most industrial applications.

More specifically, PLS projects the information of the data into a low dimensional space defined by a small number of orthogonal latent vectors $\mathbf{t}_m$ and $\mathbf{u}_m$, with $\mathbf{T} = (\mathbf{t}_1, \ldots, \mathbf{t}_M) \in \mathbb{R}^{k \times M}$ (with $M \leq D$ as the number of latent variables) and $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_M) \in \mathbb{R}^{k \times M}$:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E} = \sum_{m=1}^{M} \mathbf{t}_m \mathbf{p}_m^T + \mathbf{E}, \tag{1}$$

$$\mathbf{y} = \mathbf{T}\mathbf{B}\mathbf{Q}^T + \mathbf{F} = \sum_{m=1}^{M} \mathbf{u}_m \mathbf{q}_m^T + \mathbf{F}, \tag{2}$$

where [R2Q5]$\mathbf{U} = \mathbf{T}\mathbf{B}$, $\mathbf{P} = (\mathbf{p}_1, \ldots, \mathbf{p}_M) \in \mathbb{R}^{D \times M}$ and $\mathbf{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_M) \in \mathbb{R}^{1 \times M}$ are the loading matrices, $\mathbf{E}$ and $\mathbf{F}$ are the input and output data residuals, $\mathbf{B} = \text{diag}(b_1, \ldots, b_M)$ is a diagonal matrix with the regression weights b_m. Then, the estimated output $\hat{y}$, given an input sample $\mathbf{x}$, is given by:

$$\hat{y} = \mathbf{x}^T \boldsymbol{\theta}, \tag{3}$$

where $\boldsymbol{\theta} = \mathbf{P}^{\dagger}\mathbf{B}\mathbf{Q}^T$, and $\mathbf{P}^{\dagger} = (\mathbf{P}\mathbf{P}^T)^{-1}\mathbf{P}$ is the pseudo-inverse of $\mathbf{P}$. The values of b_m $(m = 1, \ldots, M)$, $\mathbf{T}$, $\mathbf{P}$, $\mathbf{U}$, $\mathbf{Q}$ from the above problem can be computed by using the classical Nonlinear Iterative Partial Least Squares ([R1Q2]NIPLS or NIPALS) method [21].

3.1. Selecting the Number of Latent Variables

Let $\mathcal{M}$ be such that $M \in \mathcal{M}$, for any possible/eligible number of latent variables, M. The major concern regarding the PLS algorithm is to select the number of latent variables M. Usually it is determined by a K-fold cross-validation procedure applied on the training set [R1Q3][22, 23, 24]. In K-fold cross validation the training set is split randomly into K subsets or folds, then the PLS is trained using the samples from the $(K-1)$ folds and evaluated in the remaining fold using any performance metric, usually the residual sum of squares (RSS); e.g. lower values of RSS indicate better models. It is repeated for all folds K, and with different values for the number of latent factors. The selected number of latent factors M is the one that produced the lowest average cross-validation performance metric among these K realizations. However, the K-fold cross-validation procedure is very efficient as long as k (the number of samples) is not too large, since it needs to run the PLS algorithm $K|\mathcal{M}|$ times. A fast way of selecting the number of latent variables is using information criterion methods, like the Akaike Information Criterion (AIC) [25] or the Bayesian Information Criterion (BIC) [26], which measure the quality of a model in terms of its accuracy-complexity trade-off (ACT). Using information criterion methods, the PLS algorithm runs just $|\mathcal{M}|$ times [27].

However, the major concern when applying information criterion methods to evaluate the ACT in the PLS algorithm is to determine the number of its degrees of freedom (DOF) (number of free parameters) of the PLS. Usually the DOF is set to be equal to the number of latent variables, but this is a wrong assumption and does not lead to satisfactory in the selection of the number of latent variables [28, 29]. This problem of determining the DOF in a PLS model was addressed in [29], where it has been proposed an unbiased estimate of the DOF. The use of 10-fold cross validation (using the RSS measure), AIC and BIC criteria (both with the proposed DOF estimate) to select the number of latent variables has been compared. It has been concluded that BIC and 10-fold cross validation provide the best results, with similar performance for both, and with much lower computational cost associated with the BIC computations.

Thus, in this work, the BIC criterion will be used to select the number of latent vectors for the PLS algorithm, for each expert and each gate of the Mix-PLS (the proposed implementation will be detailed [R2Q9]in Section 4). Assume that variable y has an approximation uncertainty modeled by a Gaussian pdf $\mathcal{N}(y(i)|f(\mathbf{x}(i),\boldsymbol{\theta}),\sigma^2)$, where $f(\mathbf{x},\boldsymbol{\theta})$ is the mean, and σ^2 is the variance. For a linear model $f(\mathbf{x},\boldsymbol{\theta}) = \mathbf{x}^T\boldsymbol{\theta}$, where $\boldsymbol{\theta}$ is determined using the PLS method with $m \leq |\mathcal{M}|$ latent vectors, the BIC of the model for the data set $\{\mathbf{X},\mathbf{y}\}$ is equal to:

$$\mathrm{BIC}(m) = -2\ln\prod_{i=1}^{k}\mathcal{N}(y(i)|f(\mathbf{x}(i),\boldsymbol{\theta}),\sigma^2) + \frac{1}{2}d(m,\mathbf{X},\mathbf{y},\mathbf{T})\ln(k), \tag{4}$$

where the quantity $\ln\prod_{i=1}^{k}\mathcal{N}(y(i)|f(\mathbf{x}(i),\boldsymbol{\theta}),\sigma^2)$ is the log likelihood which accounts for the model accuracy, and the second term $d\,(m,\mathbf{X},\mathbf{y},\mathbf{T})$ is the number of DOF of the PLS regressor, which relates to model complexity (see [29] for implementation details of $d(\cdot)$).

4. Mixture of Partial Least Squares Regression Experts

In this section, the formulas for the learning of the Mix-PLS [R2Q9] is going to be derived. For the learning, the parameters of the Mix-PLS are tuned using a set of observations Φ. This section also discusses the determination of the number of experts to be used.

4.1. Mixture of Experts

The ME approximates the true pdf $p(y(i)|\mathbf{x}(i))$ with the following superposition of individual pdfs:

$$p(y(i)|\mathbf{x}(i), \boldsymbol{\vartheta}) = \sum_{p=1}^{P} \upsilon_p(\mathbf{x}(i), \mathbf{V})\, p(y(i)|f_p(\mathbf{x}(i), \boldsymbol{\theta}_p), \boldsymbol{\Omega}), \tag{5}$$

where P is the number of experts, $\boldsymbol{\vartheta} = \{\mathbf{V}, \boldsymbol{\mathcal{E}}\}$, $\mathbf{V}$ and $\boldsymbol{\mathcal{E}} = \{\boldsymbol{\Theta}, \boldsymbol{\Omega}\}$ are defined as the sets of parameters of the gates and model experts, respectively, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_p | \, p = 1, \ldots, P\}$, $\upsilon_p(\mathbf{x}(i), \mathbf{V})$ is the gating function of expert p, and $p(y(i)|f_p(\mathbf{x}(i), \boldsymbol{\theta}_p), \boldsymbol{\Omega})$ is the pdf of expert model p, with mean $f_p(\mathbf{x}(i), \boldsymbol{\theta}_p)$ and additional pdf parameters $\boldsymbol{\Omega}$. From (5), prediction equation of ME is obtained as the following conditional mean of y:

$$\begin{aligned} F(\mathbf{x}(i)) &= \int y\, p(y|\mathbf{x}(i), \boldsymbol{\vartheta}) \mathrm{d}y = \int y \sum_{p=1}^{P} \upsilon_p(\mathbf{x}(i), \mathbf{V})\, p(y|f_p(\mathbf{x}(i), \boldsymbol{\theta}_p), \boldsymbol{\Omega}) \mathrm{d}y \\ &= \sum_{p=1}^{P} \upsilon_p(\mathbf{x}(i), \mathbf{V})\, f_p(\mathbf{x}(i), \boldsymbol{\theta}_p). \end{aligned} \tag{6}$$

In the ME the log likelihood of (5), given a set of observations Φ is given by [16]:

$$\begin{aligned} \ln p(\mathbf{y}|\mathbf{X}, \boldsymbol{\vartheta}) &= \ln \left(\sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \mathbf{V})\, p(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\mathcal{E}}) \right) = \ln \left(\prod_{i=1}^{k} p(y(i)|\mathbf{x}(i), \boldsymbol{\vartheta}) \right), \\ &= \ln \left(\prod_{i=1}^{k} \sum_{\mathbf{z}(i)} p(\mathbf{z}(i)|\mathbf{x}(i), \mathbf{V})\, p(y(i)|\mathbf{x}(i), \mathbf{z}(i), \boldsymbol{\mathcal{E}}) \right), \end{aligned} \tag{7}$$

where $\mathbf{Z}$ denotes a set of hidden variables $\mathbf{Z} = \{z_p(i) | \, p = 1, \ldots, P, \, i = 1, \ldots, k\}$, and $\mathbf{z}(i) = [z_1(i), \ldots, z_P(i)]^T$ is the vector of hidden variables for a sample i, where $z_p(i) \in \{0, 1\}$, and for each sample i, all variables $z_p(i)$ are zero, except for a single value of $z_p(i) = 1$, for some p. The hidden variable $z_p(i)$ indicates which expert p was responsible for generating the data point i. The distributions $p(\mathbf{z}(i)|\mathbf{x}(i), \mathbf{V})$ and $p(y(i)|\mathbf{x}(i), \mathbf{z}(i), \boldsymbol{\mathcal{E}})$ are defined as

Algorithm 1 EM Algorithm

1. Initialize $\boldsymbol{\vartheta}$ to be equal to some initial $\boldsymbol{\vartheta}^{(\text{old})}$;
2. Repeat 3) to 5) until the EM algorithm converges*;
3. **E** step:
   a) Estimate the distribution $p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta}^{(\text{old})})$ using (12);
4. **M** step:
   a) Find the new parameters values $\boldsymbol{\vartheta}^{(\text{new})}$, which maximize the expectation of the complete-data log likelihood $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{(\text{old})})$.
      i. $\boldsymbol{\vartheta}^{(\text{new})} = \arg\max_{\boldsymbol{\vartheta}} Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{(\text{old})}) =$
      $= \arg\max_{\boldsymbol{\vartheta}} \left(\sum_{\mathbf{Z}} \ln p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta}) p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta}^{(\text{old})}) \right)$ (Equation (17));
5. Set $\boldsymbol{\vartheta}^{(\text{old})} \leftarrow \boldsymbol{\vartheta}^{(\text{new})}$;
6. Return $\boldsymbol{\vartheta}^{(\text{new})}$.

*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 pre-specified maximum number of iterations.

follows [30]:

$$p(\mathbf{z}(i)|\mathbf{x}(i), \boldsymbol{\vartheta}) = p(\mathbf{z}(i)|\mathbf{x}(i), \mathbf{V}),$$
$$= \prod_{p=1}^{P} \left[p(z_p(i)|\mathbf{x}(i), \mathbf{V}) \right]^{z_p(i)} = p\left(z_p(i) = 1|\mathbf{x}(i), \mathbf{V} \right), \tag{8}$$

$$p(y(i)|\mathbf{x}(i), \mathbf{z}(i), \boldsymbol{\vartheta}) = p(y(i)|\mathbf{x}(i), \mathbf{z}(i), \boldsymbol{\mathcal{E}}),$$
$$= \prod_{p=1}^{P} \left[p(y(i))|\mathbf{x}(i), z_p(i), \boldsymbol{\mathcal{E}}) \right]^{z_p(i)} = p\left(y(i)|z_p(i) = 1, \mathbf{x}(i), \boldsymbol{\mathcal{E}} \right). \tag{9}$$

Then, from (7)-(9):

$$\ln p(\mathbf{y}|\mathbf{X}, \boldsymbol{\vartheta}) = \sum_{i=1}^{k} \ln \left(\sum_{p=1}^{P} p\left(z_p(i) = 1|\mathbf{x}(i), \mathbf{V} \right) \; p\left(y(i)|z_p(i) = 1, \mathbf{x}(i), \boldsymbol{\mathcal{E}} \right) \right). \tag{10}$$

The maximization of (10) is not straightforward [30, 16]. In order to maximize (10) the Expectation-Maximization (EM) algorithm [R2Q9]is going to 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 hidden variables [17, 30]. The learning of the mixture of experts by the EM algorithm is summarized in Algorithm 1. During the Expectation step (E step) of the EM, the current parameter values $\boldsymbol{\vartheta}^{(\text{old})}$ are used to estimate the posterior distribution of hidden variables $p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta}^{(\text{old})})$. Then, in the Maximization step (M step), this posterior distribution is used to find the new parameters

values $\boldsymbol{\vartheta}^{(\text{new})}$, which maximize the expectation of the complete-data (output and hidden variables) log likelihood

$$Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{(\text{old})}) = \mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})] = \sum_{\mathbf{Z}} \ln p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})\, p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta}^{(\text{old})}). \tag{11}$$

To perform the E step, the Bayes theorem and equations (7)-(9) are used to calculate the posterior distribution of the hidden variables, $p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta})$, as follows:

$$\begin{aligned} p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta}) &= \frac{p(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\vartheta}) p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})}{p(\mathbf{y}|\mathbf{X}, \boldsymbol{\vartheta})}, \\ &= \prod_{i=1}^{k} \prod_{p=1}^{P} \left(\frac{p\left(y(i)|z_p(i), \mathbf{x}(i), \boldsymbol{\mathcal{E}}\right) p\left(z_p(i)|\mathbf{x}(i), \mathbf{V}\right)}{\sum_{p=1}^{P} \left[p\left(z_p(i)|\mathbf{x}(i), \mathbf{V}\right) p\left(y(i)|z_p(i), \mathbf{x}(i), \boldsymbol{\mathcal{E}}\right)\right]} \right)^{z_p(i)}. \end{aligned} \tag{12}$$

For the M step, the value of $p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})$, necessary to compute $Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{(\text{old})})$ (11) is obtained using (8)-(9) as follows:

$$\begin{aligned} p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta}) &= p(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\vartheta})\, p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta}), \\ &= \prod_{i=1}^{k} \prod_{p=1}^{P} \left[p\left(z_p(i)|\mathbf{x}(i), \mathbf{V}\right) p\left(y(i)|z_p(i), \mathbf{x}(i), \boldsymbol{\mathcal{E}}\right) \right]^{z_p(i)}. \end{aligned} \tag{13}$$

The expectation of the complete-data log likelihood (11) can be computed using (12) and (13). First, taking the logarithm of $p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})$:

$$\ln p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta}) = \sum_{i=1}^{k} \sum_{p=1}^{P} \left(z_p(i) \left[\ln p\left(z_p(i) = 1|\mathbf{x}(i), \mathbf{V}\right) + \ln p\left(y(i)|z_p(i) = 1, \mathbf{x}(i), \boldsymbol{\mathcal{E}}\right) \right] \right), \tag{14}$$

and then computing the expectation of $\ln p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})$ with respect to the posterior distribution of hidden variables $\mathbf{Z}$:

$$\begin{aligned} Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{(\text{old})}) &= \sum_{\mathbf{Z}} \ln p(\mathbf{y}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\vartheta})\, p(\mathbf{Z}|\mathbf{y}, \mathbf{X}, \boldsymbol{\vartheta}), \\ &= \sum_{i=1}^{k} \sum_{p=1}^{P} \gamma_p^{(\text{old})}(i) \ln p\left(z_p(i) = 1|\mathbf{x}(i), \mathbf{V}\right) + \sum_{i=1}^{k} \sum_{p=1}^{P} \gamma_p^{(\text{old})}(i) \ln p\left(y(i)|z_p(i) = 1, \mathbf{x}(i), \boldsymbol{\mathcal{E}}\right) = \\ &= Q_g(\mathbf{V}, \boldsymbol{\vartheta}^{(\text{old})}) + Q_e(\boldsymbol{\mathcal{E}}, \boldsymbol{\vartheta}^{(\text{old})}), \end{aligned} \tag{15}$$

where $\gamma_p^{(\text{old})}(i)$, defined as the responsibility of model p, is the expectation of $z_p(i)$ with respect to its distribution (12), and it accounts for the probability of model p generating the data sample i:

$$\gamma_p^{(\text{old})}(i) = \frac{p\left(z_p(i) = 1|\mathbf{x}(i), \mathbf{V}^{(\text{old})}\right) p\left(y(i)|z_p(i) = 1, \mathbf{x}(i), \boldsymbol{\mathcal{E}}^{(\text{old})}\right)}{\sum_{l=1}^{P} \left[p\left(z_l(i) = 1|\mathbf{x}(i), \mathbf{V}^{(\text{old})}\right) p\left(y(i)|z_l(i) = 1, \mathbf{x}(i), \boldsymbol{\mathcal{E}}^{(\text{old})}\right)\right]}. \tag{16}$$

In (15), Q_g and Q_e are the contributions of gate and expert parameters for the expectation of complete-data log likelihood. Then, the M step of the EM algorithm can be performed, by separately maximizing the gate and expert contributions, as follows:

$$\begin{aligned}\boldsymbol{\vartheta}^{(\text{new})} &= \arg\max_{\boldsymbol{\vartheta}} Q(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^{(\text{old})}),\\ &= \left\{ \arg\max_{\mathbf{V}} Q_g(\mathbf{V}, \boldsymbol{\vartheta}^{(\text{old})}), \arg\max_{\boldsymbol{\mathcal{E}}} Q_e(\boldsymbol{\mathcal{E}}, \boldsymbol{\vartheta}^{(\text{old})}) \right\}. \end{aligned} \tag{17}$$

Thus, the determination of the parameters for the gates $\mathbf{V}$ and the experts $\boldsymbol{\mathcal{E}}$ is independently performed by the maximizations in (17). In the Mix-PLS, such maximizations are done using the PLS algorithm, as derived in Subsections 4.2 and 4.3 below.

4.2. Modeling the Experts With the PLS Algorithm

In this paper, it is assumed that each pdf $p\left(y(i)|z_p(i)=1,\mathbf{x}(i),\boldsymbol{\mathcal{E}}\right)$ in $Q_e(\boldsymbol{\mathcal{E}},\boldsymbol{\vartheta}^{(\text{old})})$ (15) is described by a Gaussian distribution $\mathcal{N}(y(i)|f_p(\mathbf{x}(i),\boldsymbol{\theta}_p),\omega_p)$, where $f_p(\mathbf{x}(i),\boldsymbol{\theta}_p)$, and ω_p are the mean and variance of the model of expert p, respectively. The mean is modeled by a linear model $f_p(\mathbf{x}(i),\boldsymbol{\theta}_p)=\mathbf{x}(i)^T\boldsymbol{\theta}_p$. Specifically, the experts parameters $\boldsymbol{\mathcal{E}}=\{\boldsymbol{\Theta},\boldsymbol{\Omega}\}$, include the parameters of $\boldsymbol{\Theta}=\{\boldsymbol{\theta}_p|\, p=1,\ldots,P\}$, and $\boldsymbol{\Omega}=\{\omega_p|\, p=1,\ldots,P\}$. Thus, the contribution $Q_e(\boldsymbol{\mathcal{E}},\boldsymbol{\vartheta}^{(\text{old})})$ of all experts to the expectation of complete data log likelihood (15) can be rewritten as:

$$Q_e(\boldsymbol{\mathcal{E}},\boldsymbol{\vartheta}^{(\text{old})}) = \sum_{p=1}^{P} Q_{e,p}\left(\{\boldsymbol{\theta}_p,\omega_p\},\boldsymbol{\vartheta}^{(\text{old})}\right), \tag{18}$$

$$Q_{e,p}\left(\{\boldsymbol{\theta}_p,\omega_p\},\boldsymbol{\vartheta}^{(\text{old})}\right) = \sum_{i=1}^{k} \gamma_p^{(\text{old})}(i)\ln\mathcal{N}(y(i)|f_p(\mathbf{x}(i),\boldsymbol{\theta}_p),\omega_p), \tag{19}$$

where $Q_{e,p}\left(\{\boldsymbol{\theta}_p,\omega_p\},\boldsymbol{\vartheta}^{(\text{old})}\right)$ is the contribution of expert p, and from (16) the responsibility $\gamma_p^{(\text{old})}(i)$ is equal to:

$$\gamma_p^{(\text{old})}(i) = \frac{\upsilon_p^{(\text{old})}(i)\mathcal{N}(y(i)|f_p(\mathbf{x}(i),\boldsymbol{\theta}_p^{(\text{old})}),\omega_p^{(\text{old})})}{\sum_{l=1}^{P}\upsilon_l^{(\text{old})}(i)\mathcal{N}(y(i)|f_l(\mathbf{x}(i),\boldsymbol{\theta}_l^{(\text{old})}),\omega_l^{(\text{old})})}, \tag{20}$$

where $\upsilon_p^{(\text{old})}(i)=p\left(z_p(i)=1|\mathbf{x}(i),\mathbf{V}^{(\text{old})}\right)$ is the probability of model p generating sample i, which [R2Q9]is going to be determined in Section 4.3.

Then, $Q_e(\boldsymbol{\mathcal{E}},\boldsymbol{\vartheta}^{(\text{old})})$ is maximized with respect to $\boldsymbol{\mathcal{E}}$ by solving equations $\frac{\partial Q_e(\boldsymbol{\mathcal{E}},\boldsymbol{\vartheta}^{(\text{old})})}{\partial\boldsymbol{\theta}_p}=0$, and $\frac{\partial Q_e(\boldsymbol{\mathcal{E}},\boldsymbol{\vartheta}^{(\text{old})})}{\partial\omega_p}=0$, which gives the following solution:

$$\boldsymbol{\theta}_p^{(\text{new})} = \left(\mathbf{X}^T\boldsymbol{\Gamma}_p\mathbf{X}\right)^{-1}\mathbf{X}^T\boldsymbol{\Gamma}_p\mathbf{y}, \tag{21}$$

$$\omega_p^{(\text{new})} = \frac{\sum_{i=1}^{k}\gamma_p^{(\text{old})}(i)\left(y(i)-f_p(x_j(i),\boldsymbol{\theta}_p^{(\text{new})})\right)^2}{\sum_{i=1}^{k}\gamma_p^{(\text{old})}(i)} = \frac{\left|\left|\mathbf{y}_{(\boldsymbol{\Gamma},p)}-\mathbf{X}_{(\boldsymbol{\Gamma},p)}\boldsymbol{\theta}_p^{(\text{new})}\right|\right|^2}{\text{Tr}(\boldsymbol{\Gamma}_p)}, \tag{22}$$

where $\mathbf{\Gamma}_p = \text{diag}(\gamma_p^{(\text{old})}(1), \gamma_p^{(\text{old})}(2), \ldots, \gamma_p^{(\text{old})}(k))$ is a diagonal matrix, and $\mathbf{y}_{(\mathbf{\Gamma},p)}$ and $\mathbf{X}_{(\mathbf{\Gamma},p)}$ are defined in (23)-(24). As can be noticed, the maximization of Q_e (18) is equivalent to a weighted least squares problem, where the responsibility $\gamma_p^{(\text{old})}(i)$ is the importance of each sample.

In this work, the parameters of each model $\boldsymbol{\theta}_p^{(\text{new})}$ (21) which [R2Q9]is going to be solved using the PLS algorithm. In the PLS algorithm, from (1)-(2), the inputs $\mathbf{X}$ and output $\mathbf{y}$ are traditionally represented through their approximation with M latent and loading variables representation, i.e. $\mathbf{X} \approx \mathbf{T}\mathbf{P}^T$ and $\mathbf{y} \approx \mathbf{T}\mathbf{B}\mathbf{Q}^T$. However, solving (21) after replacing these approximations is not straightforward. A simpler approach is to multiply both $\mathbf{X}$ and $\mathbf{y}$ by $\sqrt{\mathbf{\Gamma}_p}$, so that the weighted representation of $\mathbf{X}$ and $\mathbf{y}$ becomes equal to:

$$\mathbf{X}_{(\mathbf{\Gamma},p)} = \sqrt{\mathbf{\Gamma}_p}\mathbf{X} \approx \mathbf{T}_{(\mathbf{\Gamma},p)}\mathbf{P}_{(\mathbf{\Gamma},p)}^T, \tag{23}$$

$$\mathbf{y}_{(\mathbf{\Gamma},p)} = \sqrt{\mathbf{\Gamma}_p}\mathbf{y} \approx \mathbf{T}_{(\mathbf{\Gamma},p)}\mathbf{B}_{(\mathbf{\Gamma},p)}\mathbf{Q}_{(\mathbf{\Gamma},p)}^T, \tag{24}$$

where $\mathbf{X}_{(\mathbf{\Gamma},p)}$ and $\mathbf{y}_{(\mathbf{\Gamma},p)}$ are the weighted inputs and output matrices of model p with weight matrix $\mathbf{\Gamma}_p$. $\mathbf{T}_{(\mathbf{\Gamma},p)}$ and $\mathbf{P}_{(\mathbf{\Gamma},p)}$ are the PLS latent and loading matrices of the weighted input $\mathbf{X}_{(\mathbf{\Gamma},p)}$, and $\mathbf{B}_{(\mathbf{\Gamma},p)}$ and $\mathbf{Q}_{(\mathbf{\Gamma},p)}^T$ are the PLS latent and loading matrices of the weighted output $\mathbf{y}_{(\mathbf{\Gamma},p)}$. It is assumed that the weighted input and output decomposition for expert p through the PLS algorithm is made with Me_p latent variables.

Then, by replacing (23) and (24) into (21), the parameters of model p can be written as:

$$\begin{aligned}
\boldsymbol{\theta}_p^{(\text{new})} &= \left(\mathbf{X}_{(\mathbf{\Gamma},p)}^T\mathbf{X}_{(\mathbf{\Gamma},p)}\right)^{-1}\mathbf{X}_{(\mathbf{\Gamma},p)}^T\mathbf{y}_{(\mathbf{\Gamma},p)}, \\
&= \left(\left(\mathbf{T}_{(\mathbf{\Gamma},p)}\mathbf{P}_{(\mathbf{\Gamma},p)}^T\right)^T\left(\mathbf{T}_{(\mathbf{\Gamma},p)}\mathbf{P}_{(\mathbf{\Gamma},p)}^T\right)\right)^{-1}\left(\mathbf{T}_{(\mathbf{\Gamma},p)}\mathbf{P}_{(\mathbf{\Gamma},p)}^T\right)^T\mathbf{T}_{(\mathbf{\Gamma},p)}\mathbf{B}_{(\mathbf{\Gamma},p)}\mathbf{Q}_{(\mathbf{\Gamma},p)}^T, \\
&= \left(\mathbf{P}_{(\mathbf{\Gamma},p)}\mathbf{P}_{(\mathbf{\Gamma},p)}^T\right)^{-1}\mathbf{P}_{(\mathbf{\Gamma},p)}\mathbf{B}_{(\mathbf{\Gamma},p)}\mathbf{Q}_{(\mathbf{\Gamma},p)}^T.
\end{aligned} \tag{25}$$

As at each new iteration of the EM algorithm, the values of responsibility $\gamma_p^{(\text{old})}(i)$ computed in the expectation step change. Consequently the values of weighted input matrix $\mathbf{X}_{(\mathbf{\Gamma},p)}$ and output vector $\mathbf{y}_{(\mathbf{\Gamma},p)}$ change. Then, the number of latent variables Me_p necessary to represent $\mathbf{X}_{(\mathbf{\Gamma},p)}$ and $\mathbf{y}_{(\mathbf{\Gamma},p)}$ should be recomputed for a proper representation.

As discussed before, the use of K-fold cross validation to determine Me_p would computationally overload the EM algorithm, since at each new iteration the cross validation would need to be run $K|\mathcal{M}|$ times. Then, at each new iteration, the number of latent variables which [R2Q9]is going to be be determined using the BIC measure (4), which needs to run just $|\mathcal{M}|$ times. Since each sample $y(i)$ has a weight $\gamma_p^{(\text{old})}(i)$, then the weighted log-likelihood (WLL, $\ln \mathcal{L}_w$) [31] which [R2Q9]is going to be be used instead of the log-likelihood in the first term of the r.h.s. of (4). Thus, to compute the BIC for expert p, it is necessary to determine the WLL of its approximation model. From the definition of weighted likelihood [31], the WLL of a PLS model with sample weights $\gamma_p^{(\text{old})}(i)$, is equal to:

$$\ln \mathcal{L}_w = \ln \prod_{i=1}^{k} \mathcal{N}(y(i)|f_p(\mathbf{x}(i), \boldsymbol{\theta}_p), \omega_p)^{\gamma_p^{(\text{old})}(i)} = \sum_{i=1}^{k} \gamma_p^{(\text{old})}(i) \ln \mathcal{N}(y(i)|f_p(\mathbf{x}(i), \boldsymbol{\theta}_p), \omega_p), \tag{26}$$

and it is equal to $Q_{e,p}\left(\{\boldsymbol{\theta}_p, \omega_p\}, \boldsymbol{\vartheta}^{(\text{old})}\right)$ in (19). Then, the BIC when using m latent variables for expert p is:

$$\begin{aligned}
\text{BIC}_{\text{E}}(p,m) &= -2Q_{e,p}\left(\{\boldsymbol{\theta}_p, \omega_p\}, \boldsymbol{\vartheta}^{(\text{old})}\right) + \frac{1}{2}d\left(m, \sqrt{\boldsymbol{\Gamma}_p}\mathbf{X}, \sqrt{\boldsymbol{\Gamma}_p}\mathbf{y}, \mathbf{T}_{(\boldsymbol{\Gamma},p)}\right)\ln(k), \\
&= -2\sum_{i=1}^{k}\gamma_p^{(\text{old})}(i)\ln\mathcal{N}\left(y(i)|f_p(\mathbf{x}(i),\boldsymbol{\theta}_p),\omega_p\right) + \frac{1}{2}d\left(m, \mathbf{X}_{(\boldsymbol{\Gamma},p)}, \mathbf{y}_{(\boldsymbol{\Gamma},p)}, \mathbf{T}_{(\boldsymbol{\Gamma},p)}\right)\ln(k), \\
&= \sum_{i=1}^{k}\gamma_p^{(\text{old})}(i)\left(\ln(2\pi\omega_p) + \frac{\left(\mathbf{x}(i)\boldsymbol{\theta}_p - y(i)\right)^2}{\omega_p}\right) + \frac{1}{2}d\left(m, \mathbf{X}_{(\boldsymbol{\Gamma},p)}, \mathbf{y}_{(\boldsymbol{\Gamma},p)}, \mathbf{T}_{(\boldsymbol{\Gamma},p)}\right)\ln(k), \\
&= \text{Tr}(\boldsymbol{\Gamma}_p)\ln(2\pi\omega_p) + \frac{||\mathbf{X}_{(\boldsymbol{\Gamma},p)}\boldsymbol{\theta}_p - \mathbf{y}_{(\boldsymbol{\Gamma},p)}||^2}{\omega_p} + \frac{1}{2}d\left(m, \mathbf{X}_{(\boldsymbol{\Gamma},p)}, \mathbf{y}_{(\boldsymbol{\Gamma},p)}, \mathbf{T}_{(\boldsymbol{\Gamma},p)}\right)\ln(k). \qquad (27)
\end{aligned}$$

Then, at each iteration of the EM algorithm, the number of latent variables used for the PLS model of expert p is determined by:

$$Me_p = \arg\min_{m\in\mathcal{M}} \text{BIC}_{\text{E}}(p,m). \qquad (28)$$

4.3. Modeling the Gates with the PLS Algorithm

Let the gate parameters be $\mathbf{V} = \{\mathbf{v}_p | \, p = 2,\ldots,P\}$, where $\mathbf{v}_p$ is the regression coefficient of gate p. In this work, the gate of each expert in (5) is modeled using the softmax function as follows:

$$\upsilon_p(i) = p\left(z_p(i)=1|\mathbf{x}(i),\mathbf{V}\right) = \begin{cases} \frac{1}{1+\sum_{l=2}^{P}\exp(\mathbf{x}^T(i)\mathbf{v}_l)}, & p=1, \\ \frac{\exp\left(\mathbf{x}^T(i)\mathbf{v}_p\right)}{1+\sum_{l=2}^{P}\exp(\mathbf{x}^T(i)\mathbf{v}_l)}, & p=2,\ldots,P, \end{cases} \qquad (29)$$

where $\upsilon_p(i)$ is used as a simplified notation for $\upsilon_p(\mathbf{x}(i),\mathbf{V})$.

It can be seen that (29) keeps valid the constraint $\sum_{p=1}^{P} p\left(z_p(i)=1|\mathbf{x}(i),\mathbf{V}\right)=1$. Then, the gate contribution $Q_g(\mathbf{V},\boldsymbol{\vartheta}^{(\text{old})})$ to $Q(\boldsymbol{\vartheta},\boldsymbol{\vartheta}^{(\text{old})})$ (see (15), (17)) can be rewritten as:

$$\begin{aligned}
Q_g(\mathbf{V},\boldsymbol{\vartheta}^{(\text{old})}) &= \sum_{i=1}^{k}\sum_{p=1}^{P}\gamma_p^{(\text{old})}(i)\ln p\left(z_p(i)=1|\mathbf{x}(i),\mathbf{V}\right), \\
&= \sum_{i=1}^{k}\left[\sum_{p=2}^{P}\gamma_p^{(\text{old})}(i)\,\mathbf{x}^T(i)\mathbf{v}_p - \sum_{p=1}^{P}\gamma_p^{(\text{old})}(i)\ln\left(1+\sum_{l=2}^{P}\exp(\mathbf{x}^T(i)\mathbf{v}_l)\right)\right]. \qquad (30)
\end{aligned}$$

In order to find the parameters $\mathbf{V}$ to update the gating parameters in the M step, it is necessary to maximize equation (30). The maximization of $Q_g(\mathbf{V},\boldsymbol{\vartheta}^{(\text{old})})$ with respect to each gate parameter $\mathbf{v}_p$ which [R2Q9]is going to be be obtained by the iterative reweighted least squares (IRLS) method [18, 32] as follows:

$$\mathbf{v}_p^{(\text{new})} = \mathbf{v}_p^{(\text{old})} + \left[-\frac{\partial^2 Q_g(\mathbf{V},\boldsymbol{\vartheta}^{(\text{old})})}{\partial\mathbf{v}_p\mathbf{v}_p^T}\right]^{-1}\left[\frac{\partial Q_g(\mathbf{V},\boldsymbol{\vartheta}^{(\text{old})})}{\partial\mathbf{v}_p}\right]. \qquad (31)$$

From (30), the derivatives in (31) can be obtained:

$$\left[-\frac{\partial^2 Q_g(\mathbf{V}, \boldsymbol{\vartheta}^{(\text{old})})}{\partial \mathbf{v}_p \mathbf{v}_p^T}\right]^{-1} = (\mathbf{X}^T \mathbf{R}_p \mathbf{X})^{-1}, \tag{32}$$

$$\left[\frac{\partial Q_g(\mathbf{V}, \boldsymbol{\vartheta}^{(\text{old})})}{\partial \mathbf{v}_p}\right] = \mathbf{X}^T \mathbf{u}_p, \tag{33}$$

where $\mathbf{R}_p = \text{diag}(\upsilon_p(1)(1-\upsilon_p(1)), \upsilon_p(2)(1-\upsilon_p(2)), \ldots, \upsilon_p(k)(1-\upsilon_p(k)))$ is a diagonal matrix and $\mathbf{u}_p = [\gamma_p^{(\text{old})}(1)-\upsilon_p(1), \gamma_p^{(\text{old})}(2)-\upsilon_p(2), \ldots, \gamma_p^{(\text{old})}(k)-\upsilon_p(k)]^T$. After some manipulations, equation (31) can be transformed to:

$$\mathbf{v}_p^{(\text{new})} = \left(\mathbf{X}^T \mathbf{R}_p \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{R}_p \mathbf{z}_p, \tag{34}$$

where $\mathbf{z}_p = \mathbf{X}\mathbf{v}_p^{(\text{old})} - \mathbf{R}_p^{-1}\mathbf{u}_p$. Now the parameters $\mathbf{v}_p$ for $p > 1$ can be solved using the PLS algorithm, similarly to the method that was used to determine the expert parameters (Section 4.2). Using (1)-(2), the weighted input and output values are written in terms of their latent and loading variables as follows:

$$\mathbf{X}_{(\mathbf{R},p)} = \sqrt{\mathbf{R}_p}\,\mathbf{X} \approx \mathbf{T}_{(\mathbf{R},p)} \mathbf{P}_{(\mathbf{R},p)}^T, \tag{35}$$

$$\mathbf{z}_{(\mathbf{R},p)} = \sqrt{\mathbf{R}_p}\,\mathbf{z}_p \approx \mathbf{T}_{(\mathbf{R},p)} \mathbf{B}_{(\mathbf{R},p)} \mathbf{Q}_{(\mathbf{R},p)}^T, \tag{36}$$

where $\mathbf{X}_{(\mathbf{R},p)}$ and $\mathbf{y}_{(\mathbf{R},p)}$ are the weighted input matrix and weighted output vector of model p with weight matrix $\mathbf{R}_p$, and $\mathbf{T}_{(\mathbf{R},p)}$ and $\mathbf{P}_{(\mathbf{R},p)}$ are the latent and loading matrices of weighted input $\mathbf{X}_{(\mathbf{R},p)}$ and similarly, $\mathbf{B}_{(\mathbf{R},p)}$ and $\mathbf{Q}_{(\mathbf{R},p)}^T$ are the latent and loading matrices of weighted output $\mathbf{z}_{(\mathbf{R},p)} = [z_{(\mathbf{R},p)}(1), \ldots, z_{(\mathbf{R},p)}(k)]^T$. It is assumed that the weighted input and output decomposition through the PLS algorithm is made with Mg_p latent variables.

Then, from (34)-(36) the parameters vector of each gate p is updated using the PLS algorithm as follows:

$$\begin{aligned}
\mathbf{v}_p^{(\text{new})} &= \left(\mathbf{X}_{(\mathbf{R},p)}^T \mathbf{X}_{(\mathbf{R},p)}\right)^{-1} \mathbf{X}_{(\mathbf{R},p)}^T \mathbf{z}_{(\mathbf{R},p)}, \\
&= \left(\left(\mathbf{T}_{(\mathbf{R},p)} \mathbf{P}_{(\mathbf{R},p)}^T\right)^T \left(\mathbf{T}_{(\mathbf{R},p)} \mathbf{P}_{(\mathbf{R},p)}^T\right)\right)^{-1} \left(\mathbf{T}_{(\mathbf{R},p)} \mathbf{P}_{(\mathbf{R},p)}^T\right)^T \mathbf{T}_{(\mathbf{R},p)} \mathbf{B}_{(\mathbf{R},p)} \mathbf{Q}_{(\mathbf{R},p)}^T, \\
&= \left(\mathbf{P}_{(\mathbf{R},p)} \mathbf{P}_{(\mathbf{R},p)}^T\right)^{-1} \mathbf{P}_{(\mathbf{R},p)} \mathbf{B}_{(\mathbf{R},p)} \mathbf{Q}_{(\mathbf{R},p)}^T.
\end{aligned} \tag{37}$$

As in the case of the expert model parameters, the number of latent variables to represent $\mathbf{X}_{(\mathbf{R},p)}$ and $\mathbf{z}_{(\mathbf{R},p)}$ should be recomputed at each new iteration. The parameter vector solution (37) of gate p has a weighted least squares solution, similar to the solution (25) of parameter vector of expert p. Then, the BIC for a gate p can be computed by adapting the expression for the BIC of expert p (27) by changing the weighted input, $\mathbf{X}_{(\mathbf{\Gamma},p)}$, and output, $\mathbf{y}_{(\mathbf{\Gamma},p)}$, to $\mathbf{X}_{(\mathbf{R},p)}$ and $\mathbf{z}_{(\mathbf{R},p)}$, respectively, and redefining the variance ω_p to ϖ_p. Then, the BIC value for a gate p, represented by $\text{BIC}_\text{G}(p, m)$ is equal to:

$$\begin{aligned}
&\text{BIC}_\text{G}(p, m) = \\
&= \text{Tr}(\mathbf{R}_p) \ln(2\pi\varpi_p) + \frac{||\mathbf{X}_{(\mathbf{R},p)}\mathbf{v}_p - \mathbf{z}_{(\mathbf{R},p)}||^2}{\varpi_p} + \frac{1}{2} d(m, \mathbf{X}_{(\mathbf{R},p)}, \mathbf{z}_{(\mathbf{R},p)}, \mathbf{T}_{(\mathbf{R},p)}) \ln(k),
\end{aligned} \tag{38}$$

where ϖ_p is the variance of the Gaussian model that models the uncertainty of $z_{(\mathbf{R},p)}(i)$:

$$\varpi_p = \frac{\left|\left|\mathbf{z}_{(\mathbf{R},p)} - \mathbf{X}_{(\mathbf{R},p)}\mathbf{v}_p\right|\right|^2}{\mathrm{Tr}(\mathbf{R}_p)}. \tag{39}$$

Then, the number of latent variables Mg_p used for the PLS gate at each iteration is determined by:

$$Mg_p = \arg\min_{m \in \mathcal{M}} \mathrm{BIC}_{\mathrm{G}}(p, m). \tag{40}$$

[R2Q3]The parameter $\mathbf{v}_p$ for $p = 1, \ldots, P$, of the softmax function, (29), is known to suffer from instability in the maximum likelihood estimation of the parameters when the data samples are separable or quasi-separable. In these situations, the vector $\mathbf{v}_p$ tends to infinity in the maximization of log likelihood. (30). However, the PLS estimation (37) tends to alleviate this problem by combining the input variables into a new set of latent variables, reducing the effect of input variables which are responsible for the data separation. Nonetheless, during the Mix-PLS learning by the EM algorithm, it is possible to detect the instability of parameter estimation by using the Hessian matrix. (32). If the values of the terms in (32) are very large or it is not possible to compute the inverse, then it is possible to restart the learning of Mix-PLS or just reset the value of vector $\mathbf{v}_p$ to its initial value.

4.4. Selecting the Number of Mixture Models

The standard mixture of linear regression models ([R2Q1]MLRE) is sensitive to the number of experts used to compose the mixture. As the number of expert models increases, the training data is better fitted. However, the mixtures with too many experts tend to overfit the training data and show poor generalization performance.

On the other side the Mix-PLS is less prone to overfitting, even with a large number of models. This happens because the parameters of each expert and each gate are solved in a low dimensional space spanned by the results of the PLS algorithm. Moreover, the number of latent variables selected to represent each expert and each gate through the PLS algorithm is determined using the BIC criterion which penalizes complex models, then avoiding overfitting.

4.4.1. Mix-PLS and Overfitting

A small example was studied to demonstrate the robustness of Mix-PLS to overfitting with respect to the number of experts. An artificial data set containing 500 samples was created to compare the performance of Mix-PLS with the [R2Q1]MLRE with respect to the number of mixture models. The output y of the artificial model is defined as follows:

$$y(k) = \begin{cases} 2x_1(k) + \mathcal{N}(0, 0.1), \text{if } x_1(k) \leq 0.5, \\ 2 - 2x_1(k) + \mathcal{N}(0, 0.1), \text{if } x_1(k) > 0.5, \end{cases} \tag{41}$$

where x_1 was randomly generated with a uniform distribution over $[0, 1]$ and $\mathcal{N}(0, 0.1)$ is a zero-mean Gaussian random variable with 0.1 variance. From the 500 generated samples,

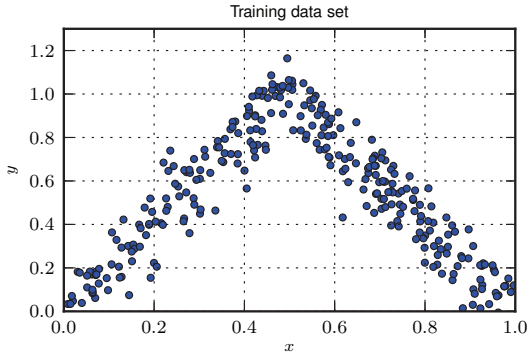
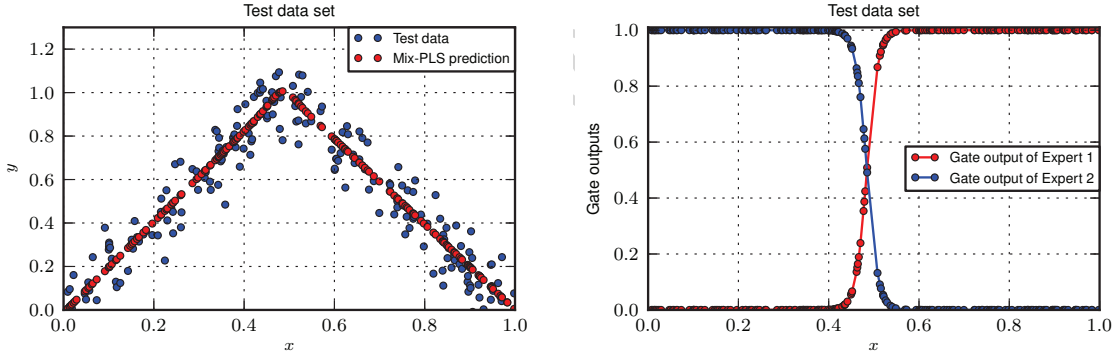


Figure 2: Output y defined in equation (41).



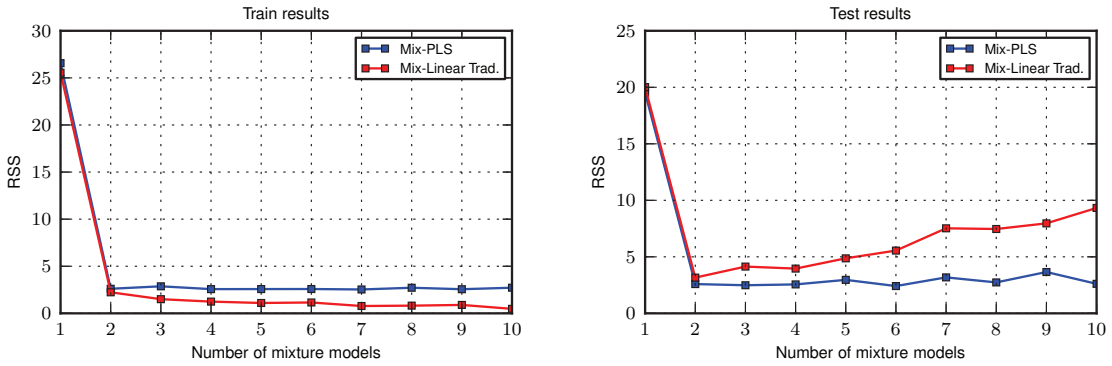
(a) **(b)**

Figure 3: (a) Prediction results and (b) gate outputs on the Mix-PLS on the test set of the artificial data set.

300 were used for training and the remaining 200 were used to testing. The output y of the training data set is represented in Figure 2. In this experiment the Mix-PLS and the $^{\text{R2Q1}}$MLRE were learned using variable x_1 jointly with more 20 irrelevant variables which were added to the data set. The irrelevant variables were generated from a multivariate Gaussian distribution with randomly selected mean and covariance matrix. The values of variables were normalized to be over $[0, 1]$.

The results of using Mix-PLS with two mixture models ($P = 2$) to learn the function (41) are shown in Figure 3. Figure 3a shows the fitting results on the test data set, where it is possible to conclude that the performance of Mix-PLS is good. Figure 3b shows the output of the gating functions, used to select which model is responsible to predict the output.

Figures 4a and 4b show the performance of Mix-PLS and the $^{\text{R2Q1}}$MLRE. As can be noticed, on the training data set, the traditional solution fits better as the number of expert models increases. On the other hand, the Mix-PLS results shows a constant performance on the training data set. On the test results, it is possible to see that the $^{\text{R2Q1}}$MLRE tends to overfit the training data, then providing poor generalization results. The performance of



(a) **(b)**

Figure 4: Performance comparison between the Mix-PLS and the [R2Q1]MLRE on the artificial data set for different numbers of mixture models: (a) training data set, and (b) test data set.

the Mix-PLS on the test data set is much better, and as mentioned before Mix-PLS is less prone to overfitting.

4.4.2. Number of Experts Selection

To select the number of mixture models this paper will use the criterion suggested by [33, 34], where for each expert p, a *worth index* is defined as:

$$I_p = \frac{1}{k}\sum_{i=1}^{k}\gamma_p(i). \tag{42}$$

In a mixture of P_e experts, without loss of generality assume that $I_1 \geq I_2 \geq \ldots \geq I_{P_e}$. Then, as defined in [33], the number of experts, P, is selected as the minimum number of experts with the largest worth indices for which the sum of their worth indices exceeds some threshold value τ, i.e.:

$$P = \min\left\{P^* : \sum_{p=1}^{P^*} I_p > \tau, \text{ and } P^* \leq P_e, \text{ and } I_1 \geq I_2 \geq \ldots \geq I_{P_e}\right\}. \tag{43}$$

The $(P_e - P)$ models with the lowest worth indices can be pruned from the mixture of experts. In [33] it is suggested the value of $\tau = 0.8$, which has shown to work well in practice.

5. Experimental Results

This section presents experimental results of the Mix-PLS applied in three real prediction problems. In two of the three data sets, two targets are to be predicted. The prediction will be performed separately for each of the outputs in these data sets. A summary of data sets is given in Table 1, As the objective of this work is to evaluate the proposed method, and

Table 1: Summary of data sets.

Data set name	#Inputs	#Train samples	#Test samples
SRU: (H_2S) [1]	20	2000	8074
SRU: (SO_2) [1]	20	2000	8074
Polymerization (Viscosity) [13]	24	521	133
Polymerization (Acidity) [13]	24	521	133
Spectra [35]	401	48	12

not discuss the process itself. Only a short description of each process/dataset is given as follows:

1. SRU: This data set covers the estimation of hydrogen sulfide (H_2S) and sulfur dioxide (SO_2) in the tail stream of a sulfur recovery unit [1, Chapter 5]. The original data set contains 10072 samples, and in this work the learning set includes the first 2000 samples for training and the remaining 8072 samples for test (as in the original work [1]). The data set contains five input variables: x_1 , x_2 , x_3, x_4, x_5. By considering lagged inputs, the inputs considered in the models, are: $x_1(k), x_1(k-5), x_1(k-7), x_1(k-9), \ldots, x_5(k), x_5(k-5), x_5(k-7), x_5(k-9)$, making a total of 20 input variables. According to the authors, the prefered models are the ones that are able to accurately predict peaks in the H_2S and SO_2 concentrations in the tail gas.
2. Polymerization: The objective in this data set is the estimation of the quality of a resin produced in an industrial batch polymerization process [13]. The resin quality is determined by the values of two chemical properties: the resin acidity number (N_A) and the resin viscosity (μ). The data set is composed of 24 input variables and [R1Q4]the authors [13] have predefined 521 samples for train and 133 for test.
3. [R1Q8]Spectra: The objective in this data set is the estimation of octane ratings based on the near infrared (NIR) spectral intensities of 60 samples of gasoline at 401 wavelengths [35]. This data set was split in 80% for training and the remaining 20% was used for test.

[R1Q5]In all experiments, the values of both the training samples, and the testing samples, were normalized to have zero mean and unit variance. In the experiments with exception for the Spectra data set, the Mix-PLS, [R2Q1]MLRE, [R1Q8]MLR and PLS models will be tuned by using as input of the model the original variables plus the squared values of these variables; the objective while using the squared values of input variables is to introduce some nonlinearity into the linear models (Mix-PLS, [R2Q1]MLRE and PLS). In the experiments, for all data sets presented in Table 1, the proposed Mix-PLS method will be compared with the [R2Q1]MLRE, a single PLS model, a [R1Q1]SLNN trained using the gradient descent training algorithm, and a LS-SVR with Gaussian kernel [20, Chapter 3]. From the results, it can be seen that Mix-PLS attains better results when compared with [R2Q1]MLRE, PLS and to the [R1Q1]SLNN and LS-SVR non-linear models. Moreover, the Mix-PLS has the advantage of

Table 2: Parameters selected for each model and for each data set.

Data set name	Mix-PLS	R2Q1MLRE	PLS	R1Q1SLNN	LS-SVR
SRU: (H_2S) [1]	$P = 2$ $Me_p = \{14, 17\}$ $Mg_p = \{7\}$	$P = 2$	$M = 10$	$N = 9$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 5$
SRU: (SO_2) [1]	$P = 2$ $Me_p = \{14, 15\}$ $Mg_p = \{10\}$	$P = 2$	$M = 12$	$N = 3$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 5$
Poly.: (Viscosity) [13]	$P = 2$ $Me_p = \{18, 8\}$ $Mg_p = \{2\}$	$P = 2$	$M = 10$	$N = 3$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 10$
Poly.: (Acidity) [13]	$P = 2$ $Me_p = \{20, 15\}$ $Mg_p = \{2\}$	$P = 2$	$M = 17$	$N = 3$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 25$
Spectra [35]	$P = 4$ $Me_p = \{40, 25, 26, 27\}$ $Mg_p = \{1, 1, 36\}$	$P = -$	$M = 24$	$N = 6$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 25$

having more interpretability with respect to its parameters when compared with non linear models R1Q1SLNN and LS-SVR.

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

$$\text{NRMSE} = \frac{\sqrt{\sum_{i=1}^{k} \left(y(i) - \hat{y}(i)\right)^2}}{\max\left(y\right) - \min\left(y\right)}, \tag{44}$$

where $y(i)$, and $\hat{y}(i)$ are the observed and predicted targets, respectively, and $\max(y)$, and $\min(y)$ are the maximum and minimum values of the observed target. NRMSE is often expressed in percentage. The closer the NRMSE is to 0 the better is the quality of prediction.

5.1. Evaluation and Discussion

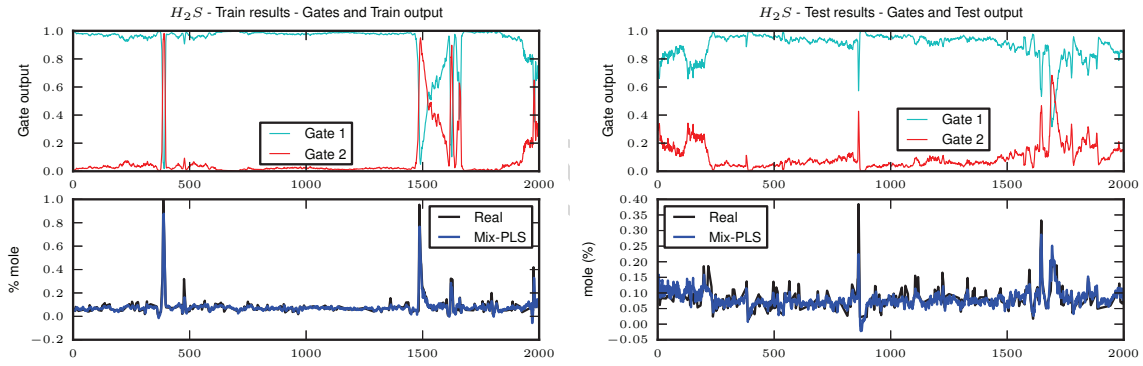
The number of hidden nodes N of the R1Q1SLNN and the regularization parameter $\gamma_{\text{LS-SVR}}$ and the Gaussian kernel parameter $\sigma_{\text{LS-SVR}}$ of the LS-SVR were determined using a 10-fold cross validation. For the PLS model the number of latent variables M, was determined using the BIC criterion as discussed in Section 3.1. For the R2Q1MLRE, and Mix-PLS the numbers of experts P were obtained from (43). Additionally, for the Mix-PLS the set that contains the number of latent variables for each expert $Me = \{Me_1, \ldots, Me_p\}$ was obtained from (28), and the corresponding set of numbers of latent variables for the gates $Mg = \{Mg_2, \ldots, Mg_p\}$ was obtained from (40). Table 2 shows the parameters obtained for each model and for each data set in the experiments.

5.1.1. SRU Data-Set

For the prediction of H_2S in the SRU data set, the NRMSE performances on the test set for all models, are indicated in Table 3. These results indicate that the Mix-PLS has the best performance among all the models. Further analysis on the Mix-PLS results, in Figure

Table 3: NRMSE results on the test set.

Data set name	Mix-PLS	[R2Q1]MLRE	PLS	[R1Q1]SLNN	LS-SVR	[R1Q8]MLR
SRU: (H_2S) [1] (C)	**4.59**	5.75	6.43	10.41	9.14	7.40
SRU: (SO_2) [1] (C)	**3.35**	5.36	3.57	3.95	5.66	5.54
Poly.: (Viscosity) [13] (B)	**8.07**	23.43	24.23	9.95	12.38	14.52
Poly.: (Acidity) [13] (B)	**3.62**	5.54	4.25	3.93	5.94	7.93
Spectra [35] (C)	**6.91**	–	9.14	8.61	28.52	7.26



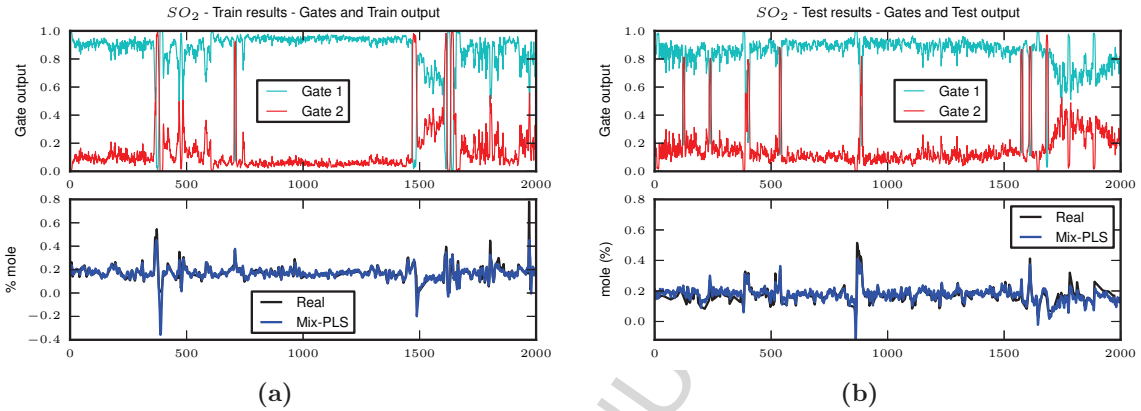
(a) **(b)**

Figure 5: Plots of H_2S prediction on the SRU data set. (a) Train results, gates and prediction. (b) Test results, gates and prediction. For better visualization, only 2000 samples are shown.

5, indicates that for the H_2S prediction, the Mix-PLS was able to identify two different operating modes, which are modeled by two experts. The first expert is the most used for predicting in the regular operation and the second expert is most used to predict peaks, as can be verified by the gates output in Figure 5. The prediction results on the test set, shown in 5b, indicate that, on unseen data, the Mix-PLS performs very well during the prediction, including in the prediction in peak periods.

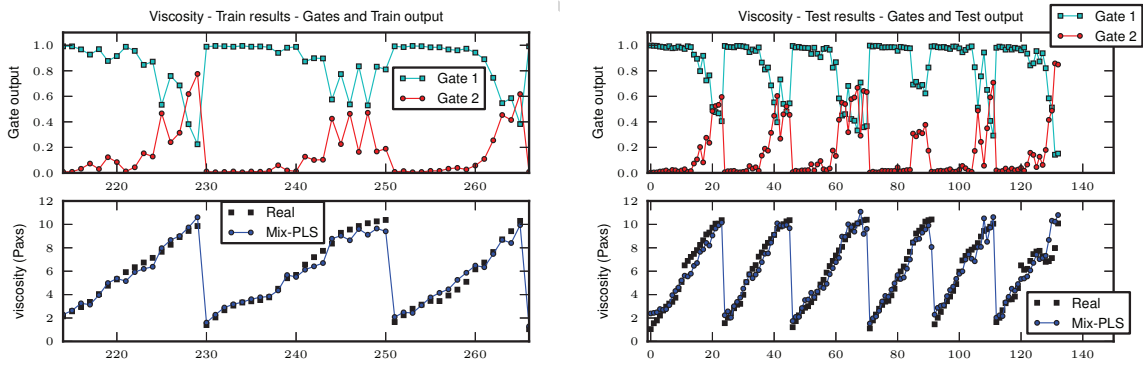
For the SO_2 prediction, the performance of all models using the NRMSE criterion are indicated in Table 3. It is shown that in this experiment, the Mix-PLS has the best performance among all the models, and the [R1Q1]SLNN model has results close to Mix-PLS. However, the Mix-PLS is more attractive than the [R1Q1]SLNN, because of the interpretability of its parameters. On this data set, the Mix-PLS was able also to identify two operating modes. The prediction results on the train and test set are shown in Figure 6.

From the H_2S and SO_2 results on the SRU data set, it was possible to conclude that the Mix-PLS was able to identify two different operating modes, in the two data sets. According with [1], on the SRU data set, the preferred models are the ones that are able to accurately predict peaks. From the SRU results it is possible to note that one expert is more responsible for predicting the regular operation mode, while the other expert is able to predict the peaks.



(a) **(b)**

Figure 6: Plots of SO_2 prediction on SRU data set. (a) Train results, gates and prediction. (b) Test results, gates and prediction. For better visualization, only 2000 samples are shown.



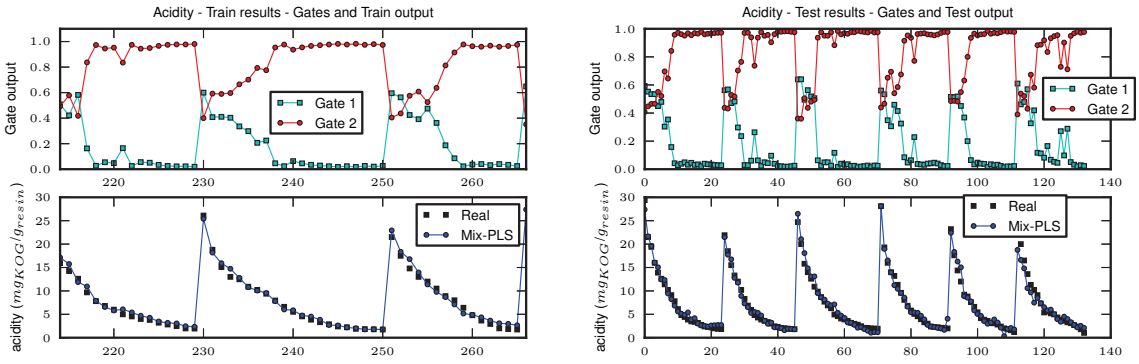
(a) **(b)**

Figure 7: Plots of viscosity prediction on Polymerization data set. (a) Train results, gates and prediction. (b) Test results, gates and prediction.

5.1.2. Polymerization Data-Set

This data set was studied in [13], and the objective is to estimate the viscosity and acidity of a resin produced in an industrial batch polymerization process. According with Table 3, for predicting the viscosity, the Mix-PLS reached the best results among all the models in terms of NRMSE. Inspecting the results from the gates activation on the train and test set which are presented in Figure 7, it is possible to note that the prediction of the first expert is predominant at the beginning of each batch, and the prediction of the two models are combined, usually at the end of each batch. The Mix-PLS suggests, that for viscosity prediction, just two models are necessary and that their prediction should be combined at the end of each batch.

For predicting the acidity, the Mix-PLS also reached the best results in terms of NRMSE, as indicated in Table 3. The Mix-PLS used 2 experts to predict the acidity. The plots of



(a) **(b)**

Figure 8: Plots of acidity prediction on Polymerization data set. (a) Train results, gates and prediction. (b) Test results, gates and prediction.

gates and prediction on the train and test sets are shown in Figure 8. Differently from the viscosity prediction, the models are combined at the beginning of each batch and then, one expert is predominant in the rest of the batch.

As can be seen the Mix-PLS was successfully applied on the Polymerization data set, delivering satisfactory prediction results. Moreover, Mix-PLS has shown better results when compared with the nonlinear models.

5.1.3. Spectra data set

R1Q8This Spectra data set was analyzed in [35], and the objective is the estimation of the octane ratings based on the near infrared (NIR) spectral intensities of 60 samples of gasoline at 401 wavelengths. This data set is characterized by having only a few samples and a large number of input variables. Moreover, it is known a priori that this data set does not have multiple operating modes, then the analysis is focused in the prediction performance. According to Table 3, the Mix-PLS reached the best results among all the models in terms of NRMSE and the MLRE method did not converge in this experiment. Moreover, Mix-PLS has shown much better results when compared with the nonlinear models in this data set.

6. Discussion

R2Q7The selection of the number of latent variables on each iteration of Mix-PLS algorithm, in our case by the BIC criterion, is not obligatory, but it is recommended. Other options are to run the Mix-PLS algorithm with a fixed number of latent variables or select it after the overall run of the algorithm. The use of a validation data set can also be a good option to select the number of latent variables.

R2Q8The expectation of the complete data log likelihood value (11) in EM algorithm with the PLS and the selection of the number of latent variables (i.e. the Mix-PLS) is monotonically increasing in most iterations. This is more evident in the first iterations of the algorithm, however, very infrequently, in some iterations the likelihood decreases its

value. However, the overall trend is to obtain an increasing likelihood. Such characteristic is expected in the proposed Mix-PLS approach since, to avoid overfitting in the training data, the selection of the latent variables by the BIC criterion, which is directly built into the model learning, and is performed at each iteration, then penalizes the likelihood value of the algorithm.

R2Q4It is already know that the first two data sets, Polymerization and SRU, have multiple operating modes, and the analysis of the results in both data sets has emphasized this case. From the results it is seen that Mix-PLS is more than a good non-linear regression method, also it picks/assigns different operating modes in/to different experts. However, although these results are representative, they are also conditioned to the problem under study, i.e. it is not possible to assure that the separate assignment of different modes to different experts is a general property that holds for all other conceivable problems. However, the application of the proposed approach is not limited to multiple operating modes and it can also be used as a non-linear regression method, as in the case of Spectra data set.

7. Conclusion

This paper proposed the use mixture of linear regression models for dealing with multiple operating modes in soft sensor applications. In the proposed Mix-PLS method, the solution of the mixture of linear regression models is done using the partial least squares regression model. The formulas for 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 three real-world data sets, encompassing the prediction of five variables.

In comparison with the traditional solution of the mixture of linear regression models, the Mix-PLS is much less prone to overfitting with respect to the number of mixture models to be used, while still attaining good prediction results, as demonstrated in an artificial data set experiment. In the real-world data sets experiments, all the results obtained with Mix-PLS were superior when compared with a R2Q1MLRE, a single PLS, a R1Q1SLNN, LS-SVR R1Q8and MLR models. Differently of the non linear models, the Mix-PLS gives more interpretability to the prediction.

Future directions of this work are to research on the implementation of the method in an online manner, further increasing the applicability.

8. Bibliography

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

[2] P. Kadlec, B. Gabrys, S. Strandt, Data-driven soft sensors in the process industry, Computers & Chemical Engineering 33 (4) (2009) 795–814.

[3] L. H. Chiang, E. L. Russell, R. D. Braatz, Fault diagnosis in chemical processes using fisher discriminant analysis, discriminant partial least squares, and principal component analysis, Chemometrics and Intelligent Laboratory Systems 50 (2) (2000) 243–252.

[4] 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.

[5] O. Haavisto, H. Hyötyniemi, Recursive multimodel partial least squares estimation of mineral flotation slurry contents using optical reflectance spectra, Analytica Chimica Acta 642 (1-2) (2009) 102–109, papers presented at the 11th International Conference on Chemometrics in Analytical Chemistry - CAC 2008.

[6] K. Helland, H. E. Berntsen, O. S. Borgen, H. Martens, Recursive algorithm for partial least squares regression, Chemometrics and Intelligent Laboratory Systems 14 (1-3) (1992) 129–137.

[7] C. Li, H. Ye, G. Wang, J. Zhang, A recursive nonlinear PLS algorithm for adaptive nonlinear process modeling, Chemical Engineering & Technology 28 (2005) 141–152.

[8] S. Mu, Y. Zeng, R. Liu, P. Wu, H. Su, 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 16 (6) (2006) 557–566.

[9] P. Facco, F. Bezzo, M. Barolo, Nearest-neighbor method for the automatic maintenance of multivariate statistical soft sensors in batch processing, Industrial & Engineering Chemistry Research 49 (5) (2010) 2336–2347.

[10] M. Matzopoulos, Dynamic process modeling: Combining models and experimental data to solve industrial problems, in: M. C. Georgiadis, J. R. Banga, E. N. Pistikopoulos (Eds.), Process Systems Engineering, Wiley-VCH Verlag GmbH & Co. KGaA, 2010, pp. 1–33.

[11] F. Wang, S. Tan, J. Peng, Y. Chang, Process monitoring based on mode identification for multi-mode process with transitions, Chemometrics and Intelligent Laboratory Systems 110 (1) (2012) 144–155.

[12] J. Yu, A nonlinear kernel gaussian mixture model based inferential monitoring approach for fault detection and diagnosis of chemical processes, Chemical Engineering Science 68 (1) (2012) 506–519.

[13] P. Facco, F. Doplicher, F. Bezzo, M. Barolo, Moving average PLS soft sensor for online product quality estimation in an industrial batch polymerization process, Journal of Process Control 19 (3) (2009) 520–529.

[14] J. Camacho, J. Picó, Online monitoring of batch processes using multi-phase principal component analysis, Journal of Process Control 16 (10) (2006) 1021–1035.

[15] N. Lu, F. Gao, Stage-based process analysis and quality prediction for batch processes, Industrial & Engineering Chemistry Research 44 (10) (2005) 3547–3555.

[16] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton, Adaptive mixtures of local experts, Neural Computation 3 (1) (1991) 79–87.

[17] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, Series B 39 (1) (1977) 1–38.

[18] M. I. Jordan, Hierarchical mixtures of experts and the EM algorithm, Neural Computation 6 (2) (1994) 181–214.

[19] S. E. Yuksel, J. N. Wilson, P. D. Gader, Twenty years of mixture of experts, IEEE Transactions on Neural Networks and Learning Systems 23 (8) (2012) 1177–1193.

[20] J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, 2002.

[21] H. Wold, Path models with latent variables: The NIPALS approach, in: H. M. B. et al. (Ed.), Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building, Academic Press, 1975, pp. 307–357.

[22] B.-H. Mevik, H. R. Cederkvist, Mean squared error of prediction (MSEP) estimates for principal component regression (PCR) and partial least squares regression (PLSR), Journal of Chemometrics 18 (9) (2004) 422–429.

[23] D. M. Hawkins, The problem of overfitting, Journal of Chemical Information and Computer Sciences 44 (2004) 1–12.

[24] D. Toher, G. Downey, T. B. Murphy, A comparison of model-based and regression classification techniques applied to near infrared spectroscopic data in food authentication studies, Chemometrics and Intelligent Laboratory Systems 89 (2) (2007) 102–115.

[25] H. Akaike, A new look at the statistical model identification, IEEE Transactions on Automatic Control 19 (6) (1974) 716–723.

[26] G. Schwarz, Estimating the dimension of a model, Annals of Statistics 6 (2) (1978) 461–464.
[27] B. Li, J. Morris, E. B. Martin, Model selection for partial least squares regression, Chemometrics and Intelligent Laboratory Systems 64 (1) (2002) 79–89.
[28] N. Kramer, M. L. Braun, Kernelizing PLS, degrees of freedom, and efficient model selection, in: Proc. 24th International Conference on Machine learning, ICML'07, ACM, New York, NY, USA, 2007, pp. 441–448.
[29] N. Kramer, M. Sugiyama, The degrees of freedom of partial least squares regression, Journal of the American Statistical Association 106 (494) (2011) 697–705.
[30] C. M. Bishop, Pattern Recognition and Machine Learning, 1st Edition, Springer, 2006.
[31] M. A. Newton, A. E. Raftery, Approximate bayesian inference with the weighted likelihood bootstrap, Journal of the Royal Statistical Society. Series B (Methodological) 56 (1) (1994) 3–48.
[32] I. T. Nabney, Efficient training of rbf networks for classification, in: Proc. Ninth International Conference on Artificial Neural Networks, 1999 (ICANN 99), Vol. 1, Edinburgh, Scotland, 1999, pp. 210–215.
[33] R. A. Jacobs, F. Peng, M. A. Tanner, A bayesian approach to model selection in hierarchical mixtures-of-experts architectures, Neural Networks 10 (2) (1997) 231–241.
[34] S.-K. Ng, G. J. McLachlan, A. H. Lee, An incremental EM-based learning approach for on-line prediction of hospital resource utilization, Artificial Intelligence in Medicine 36 (3) (2006) 257–267.
[35] J. H. Kalivas, Two data sets of near infrared spectra, Chemometrics and Intelligent Laboratory Systems 37 (2) (1997) 255–259.

Highlights

- This paper proposed a method called Mix-PLS for regression.
- The Mix-PLS was successfully applied on two real prediction problems.
- The results were compared with four other regression algorithms.
- The proposed method always exhibits the best prediction performance.