applied sciences

MDPI

Article

A Regularized Mixture of Linear Experts for Quality Prediction in Multimode and Multiphase Industrial Processes

Francisco Souza [1,2,*], Jérôme Mendes [1] and Rui Araújo [1]

[1] Department of Electrical and Computer Engineering, Institute of Systems and Robotics, University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal; jermendes@isr.uc.pt (J.M.); rui@isr.uc.pt (R.A.)
[2] Department of Analytical Chemistry & Chemometrics, Radboud University, 6525 AJ Nijmegen, The Netherlands
* Correspondence: f.souza@science.ru.nl

Abstract: This paper proposes the use of a regularized mixture of linear experts (MoLE) for predictive modeling in multimode-multiphase industrial processes. For this purpose, different regularized MoLE were evaluated, namely, through the elastic net (EN), Lasso, and ridge regression (RR) penalties. Their performances were compared when trained with different numbers of samples, and in comparison to other nonlinear predictive models. The models were evaluated on real multiphase polymerization process data. The Lasso penalty provided the best performance among all regularizers for MoLE, even when trained with a small number of samples.

Keywords: multimode process; multiphase process; mixture of experts; polymerization

Citation: Souza, F.; Mendes, J.; Araújo, R. A Regularized Mixture of Linear Experts for Quality Prediction in Multimode and Multiphase Industrial Processes. *Appl. Sci.* **2021**, *11*, 2040. https://doi.org/10.3390/app11052040

Academic Editor: Maria Gabriella Xibilia

Received: 17 January 2021
Accepted: 20 February 2021
Published: 25 February 2021
Corrected: 7 December 2021

1. Introduction

Soft sensors are inferential models used for online prediction of quality variables [1–3]. Often, the online measurement of such variables is difficult due to high costs or lack of a physical sensor. In such cases, the quality variables are measured by means of laboratory analysis. An operator collects a sample during production and sends it away for laboratory analysis. Meanwhile, the operator cannot act on the process in response to the most recent measurement, whose response is not yet known, leading to possible production loss, or at least quality loss. After the analysis result returns back, the operator can then make a decision, but it might be late. The goal of soft sensor technology is to solve this issue. It builds a data-driven model that relates the operational process variables to the quality variable. By doing so, it allows the online inference of the quality variables, allowing operators to get earlier information about the quality of the process and take corrective actions on time. The common steps to deploy a soft sensor are: data cleaning/synchronization [4], feature selection [5], model learning/validation [2], and model maintenance [6]. During the learning phase, it is beneficial to take, as much as possible, the process properties into consideration. For example, in case of multimode or multiphase processes, such information can be used in the modeling efforts. Multimode process operational characteristics exist due to external factors, such as changes in feedstock, production, operation conditions, or the external environment. Multiphase process characteristics are commonly found in batch processes. A series of phases comprise a batch cycle of production, with its own characteristics [7]. Several authors appeal to using these multiple operating properties into the modeling phase. From now on, we will also refer to each mode/phase of multimode-multiphase processes as the operating mode.

Two main steps follow the modeling of multimode-multiphase processes for quality prediction. The first step is the characterization of the operating modes, which can be done manually [8] or be inferred from data [9]. The second step is the learning of the models for each mode. In [10], the authors derived and proposed the use of a mixture

of probabilistic principal component regression (MPPCR) for multimode data. In [11], the use of a two phase partial least squares (PLS) modeling approach was proposed. In the first phase, a separated intra-phase-PLS model is learned for each phase, and in the second phase a series of inter-phase-PLS are learned to model the relationships among different phases. In [12], in a case study for Mooney viscosity prediction in a rubber-mixing process, the authors employed a fuzzy C-means clustering algorithm to cluster the samples in different subclasses, and then taught single Gaussian process regression (GPR) models for each subclass. In [7], the main idea was to learn individual partial least squares (PLS) models for each phase and each mode. The distinction between modes and phases was made manually by the experts. Following this, in [13] the authors successfully implemented a Gaussian mixture regression (GMR) model to handle the multimode data in a penicillin process. In [14], the authors incorporated the PLS model into the GMM framework for quality prediction and fault detection in an industrial hot strip mill process. In [15], the authors expanded the Gaussian mixture models (GMM) framework to its semi-supervised form for dealing with incomplete data and multimode processes. Other approaches model multimode processes using just in time learning (JITL) [16,17]. In [18], the authors proposed a robust GMR based on the Student's-t distribution for dealing with noise and outliers in data.

This paper proposes a mixture of experts (MoE) methodology with the following two characteristics: (1) the characterization of different modes, and (2) the learning of models in a single unified manner. MoE consists of a set of experts and gates, applied for modeling heterogeneous or multi-mode data. The experts are assigned for each mode, while the gate defines the boundaries for the experts. Figure 1 illustrates the MoE.

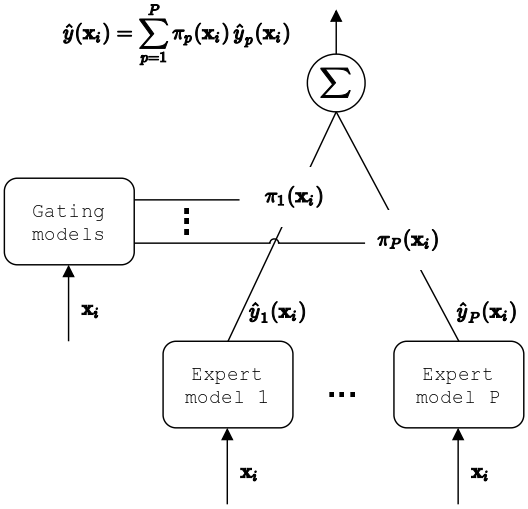

Figure 1. Mixture of experts (MoE) architecture with P experts.

MoE was introduced in [19,20], where the experts were neural network models and the gates were modeled by a softmax function. Since then, several extensions and variants of the MoE model have been proposed (see the review paper [21]). A variant of MoE is the mixture of linear experts (MoLE), wherein the experts are multivariate linear models. MoLE has the property of universal approximation, works for nonlinear modeling, is interpretable, and can automatically infer the different modes. All these characteristics make MoLE suitable for modeling multimode industrial data. However, the estimation of MoLE is unfeasible in the presence of collinearity or a small number of samples. Moreover, MoLE cannot handle irrelevant variables or perform feature selection. However, all these, are usual requirements to deal with industrial data. To solve the collinearity issue, in [9], the authors have proposed the use partial least squares (PLS) for modeling the gate and experts, defining the Mixture of Partial Least Squares Experts (Mix-PLS). It has been successfully applied to two industrial multimode-multiphase processes. In a

short paper, Ref. [22] modeled the results of MoLE with elastic net penalty for modeling a polymerization multiphase process.

Beyond the industrial context, authors have appealed to regularization to MoLE models [23–26] to perform of feature selection and allow the learning with small number of samples. The regularized MoLE methods reported in the literature use the l_1 penalty, also known as least absolute shrinkage and selection operator (Lasso) [27]; the l_2 penalty, also known as ridge regression (RR) or Tikonohov regularization [28]; a compromise between l_1 and l_2 norm [29], also known as elastic net (EN); and the smooth clipped absolute deviation (Scad) [30]. In [23], the authors have used a RR penalty for the gates and a Lasso and Scad penalties to experts in MoLE. They reported successful results, in both, performance and in finding parsimonious models. They compared the results with a RR regularized linear model. Similarly, Ref. [24] applied Lasso penalty on gates and experts for classification problems. Their regularized MoLE performed better than ordinary MoLE, and state of art classifiers. In [26], the authors employed Lasso penalty for experts, and EN penalty for the gates, and reported better results than ordinary MoLE. In [31] the authors have employed a proximal-Newton expectation maximization (EM) to avoid instability while learning the gates. In [32], the authors discuss the theoretical aspects of the use of Lasso in MoLE. All results report regularization as a viable approach to improve the performance of MoLE when dealing with irrelevant variables and small number of samples.

The goal of this paper is to check the performance of regularized MoLE models for quality prediction in multimode-multiphase processes. For this purpose, a regularized version of MoLE based on EN penalty has been derived, which has a flexible regularization form. Thereafter, this paper derives three regularized MoLE models, defined as MoLE-Lasso, MoLE-RR, and MoLE-EN. Besides, a set of experiments was run and analyzed, with all regularized MoLEs, and the Mix-PLS [9]. The experiments were run on real multiphase polymerization data, for predicting two quality variables, with a total of 31 batches. The performances of MoLE models with respect to the number of batches for training were checked. The results show that MoLE-Lasso gives the most stable results, even in learning with only a few batches. On the other hand, the Mix-PLS has a tendency to perform better when the number of batches increases. Finally, all regularized MoLE's were also compared to different state-of-the-art models. The results show that the regularized MoLE is a valid choice for modeling multimode processes data.

The paper is divided up as follows. Section 2 presents the proposed regularized MoLE. Section 3 presents the experimental results. Finally, Section 4 gives concluding remarks.

2. Regularized MoLE

In this section, the regularized MoLE is derived. First, an introduction of MoLE is given. Afterwards, the derivation of regularized MoLE and its learning procedure are presented.

2.1. Notation

In this paper, finite random variables are represented by capital letters and their values by the corresponding lowercase letters, e.g., random variable A, and corresponding value a. Matrices and vectors are represented by boldface capital letters, e.g., $\mathbf{A} = [a_{ij}]_{n\times d}$ and boldface lowercase letters, e.g., $\mathbf{a} = [a_1, \ldots, a_d]^T$, respectively. The input and output/target variables are defined as $X = \{X_1, \ldots, X_d\}$ and Y, respectively. The variables $X_1, \ldots, X_d$ can take n different values as $\{x_{ij} \in X_j : j = 1, \ldots, d \text{ and } i = 1, \ldots, n\}$, and similarly for Y as $\{y_i \in Y : i = 1, \ldots, n\}$.

2.2. Definition

The MoLE follows the divide and conquer principle. In the learning phase, the input space is divided into soft regions and a linear "expert" is learned for each region, while the gates assign soft boundaries to the experts' domains. The output for a new data sample is

given by the weighted combination of the experts' outputs, where the new data is assigned to a specific or overlap of regions. The MoLE output is given by

$$\hat{y}(\mathbf{x}_i) = \sum_{p=1}^{P} g_p(\mathbf{x}_i)\, \hat{y}_p(\mathbf{x}_i), \tag{1}$$

where $\mathbf{x}_i$ is the vector of input variables, P is the total number of experts, $\hat{y}_p(\mathbf{x}_i)$ is the output of expert p, and $g_p(\mathbf{x}_i)$ is the gate output for expert p. The gates assign mixture proportions to the experts, and have the following constraints $\sum_{p=1}^{P} g_p(\mathbf{x}_i) = 1$ and $0 \leq g_p(\mathbf{x}_i) \leq 1$. From now on, $\hat{y}(\mathbf{x}_i)$, $\hat{y}_p(\mathbf{x}_i)$, $g_p(\mathbf{x}_i)$, are denoted by their shortened versions $\hat{y}_i$, $\hat{y}_{pi}$, and g_{pi}, respectively. Expert p has the linear form $\hat{y}_p(\mathbf{x}_i) = \mathbf{x}_i^T \boldsymbol{\theta}_p$, where $\boldsymbol{\theta}_p$ is the vector of regression coefficients of linear expert p; the bias has been omitted to simplify the derivation. For the gate, the following softmax function will be employed:

$$g_{pi} = \begin{cases} \frac{1}{1+\sum_{l=2}^{P} \exp\left(\mathbf{x}_i^T \mathbf{v}_l\right)}, & p = 1, \\ \frac{\exp\left(\mathbf{x}_i^T \mathbf{v}_p\right)}{1+\sum_{l=2}^{P} \exp\left(\mathbf{x}_i^T \mathbf{v}_l\right)}, & p = 2, \ldots, P, \end{cases} \tag{2}$$

where $\mathbf{v}_p$ is the gate parameter of expert p. This softmax function follows the required gate constraints. The MoLE formulation fits perfectly for multimode-multiphase processes, where each different mode can be modeled by an expert p, while the gates define the boundaries of the different modes. MoLE can infer the number of modes, or can use the expert information to define the number of modes. It has a very flexible format, and established learning algorithms. However, MoLE models cannot deal with collinearity or irrelevant variables. The next section will discuss how to apply regularization to MoLE models.

2.3. Formulation

The MoLE approximates the true probability distribution function (PDF), defined as $p(y_i|\mathbf{x}_i)$, by a superposition of PDF's:

$$p(y_i|\mathbf{x}_i, \boldsymbol{\Omega}) = \sum_{p=1}^{P} g_p(\mathbf{x}_i, \mathbf{G})\, p(y_i|\mathbf{x}_i, \boldsymbol{\Theta}_p), \tag{3}$$

where $p(y_i|\mathbf{x}_i, \boldsymbol{\Theta}_p) = \mathcal{N}(y_i|\mathbf{x}_i^T \boldsymbol{\theta}_p, \sigma_p^2)$ is the conditional PDF of expert p, governed by the parameters $\boldsymbol{\Theta}_p = \{\boldsymbol{\theta}_p, \sigma_p^2\}$, where σ_p^2 is the error variance of expert p, assumed to be zero mean. The collection of gates parameters is represented by $\mathbf{G} = \{\mathbf{v}_1, \ldots, \mathbf{v}_p\}$. The collection of all parameters is defined as $\boldsymbol{\Omega} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p, \sigma_1^2, \ldots, \sigma_p^2, \mathbf{v}_1, \ldots, \mathbf{v}_p\}$.

The estimation of $\boldsymbol{\Omega}$ by maximum likelihood is unfeasible. Instead, the expectation maximization (EM) is commonly employed [33]. The EM uses a two-steps iterative procedure to perform the likelihood maximization of MoLE. In the first, called the expectation step (E-Step), the expectation of the log-likelihood (ELL) is evaluated, while in the second step, called maximization step (M-Step), new parameters are determined by maximum likelihood estimation from the ELL. Therefore, the maximization of likelihood in EM is achieved trough successive improvements of ELL; see [33] for further details on the application of the EM algorithm.

Since the EM is an iterative procedure, the superscript t is used to indicate the t-th iteration of the EM algorithm, e.g., $\mathbf{\Omega}^t$ is the vector of the estimated parameters of MoLE at iteration t. The ELL at the t-th E-Step, $Q^t(\mathbf{\Omega})$, is given by

$$Q^t(\mathbf{\Omega}) = Q_e^t(\mathbf{\Theta}) + Q_g^t(\mathbf{G}), \tag{4}$$

$$Q_e^t(\mathbf{\Theta}) = \sum_{i=1}^{n}\sum_{p=1}^{P} \gamma_p^t(\mathbf{x}_i) \log \mathcal{N}(y_i|\mathbf{x}_i^T\boldsymbol{\theta}_p, \sigma_p^2),$$

$$Q_g^t(\mathbf{G}) = \sum_{i=1}^{n}\sum_{p=1}^{P} \gamma_p^t(\mathbf{x}_i) \log g_p(\mathbf{x}_i, \mathbf{G}),$$

$$\gamma_p^t(\mathbf{x}_i) = \frac{g_p^t(\mathbf{x}_i)\mathcal{N}\left(y_i|\mathbf{x}_i^T\boldsymbol{\theta}_p^t, \sigma_p^2\right)}{\sum_{l=1}^{P} g_l^t(\mathbf{x}_i)\mathcal{N}\left(y_i|\mathbf{x}_i^T\boldsymbol{\theta}_l^t, \sigma_l^2\right)}, \tag{5}$$

where $Q_e^t(\cdot)$ and $Q_g^t(\cdot)$ account for the expert and gate contributions to the ELL, respectively, and $\gamma_p^t(\mathbf{x}_i)$ is the responsibility, which accounts for the probability of sample $\mathbf{x}_i$ belonging to the region covered by expert p; from now on $\gamma_p^t(\mathbf{x}_i)$ will also be referred as γ_{pi}^t. The convergence of the ELL, can be measured by computing the ELL difference $|Q^t(\mathbf{\Omega}^t) - Q^{t-1}(\mathbf{\Omega}^{t-1})|$ at each iteration. The complete derivation of ELL, and its relation to the log-likelihood of (3) can be found in ([9], Section 4).

In the M-Step, the objective is to estimate the new parameters $\mathbf{\Omega}^{t+1} = \{\mathbf{\Theta}^{t+1}, \mathbf{G}^{t+1}\}$, that maximize Equation (4). This is stated as the following optimization problem:

$$\mathbf{\Theta}^{t+1} = \arg\max_{\mathbf{\Theta}^*} Q_e^t(\mathbf{\Theta}^*), \tag{6}$$

$$\mathbf{G}^{t+1} = \arg\max_{\mathbf{G}^*} Q_g^t(\mathbf{G}^*). \tag{7}$$

The EM guarantees that the ELL, and consequently the log-likelihood of Equation (3), increases monotonically. The EM algorithm runs until the convergence of the ELL, where the convergence detection condition is defined as $|Q^t(\mathbf{\Omega}^t) - Q^{t-1}(\mathbf{\Omega}^{t-1})| \leq \xi_Q$, where ξ_Q is a convergence threshold.

The algorithmic solution is shown in Algorithm 1.

Algorithm 1 Expectation maximization (EM) for mixture of linear experts (MoLE) learning.

procedure MoLE(ξ_Q)
  Initialize $\mathbf{\Omega}^0$, $t \leftarrow 0$, Done $\leftarrow$ FALSE
  while not Done **do**
    E-Step: compute responsibilities, using Equation (5).
    if $t > 0$ **then**
      if $|Q^t(\mathbf{\Omega}^t) - Q^{t-1}(\mathbf{\Omega}^{t-1})| \leq \xi_Q$ **then** ▷ Check convergence.
        Done$\leftarrow$TRUE
      end if
    end if
    M-Step: $\mathbf{\Omega}^{t+1} \leftarrow \arg\max_{\mathbf{\Omega}^*} Q^t(\mathbf{\Omega}^*)$ ▷ Find new parameters.
    $t \leftarrow t + 1$
  end while
  return $\hat{\mathbf{\Omega}} \leftarrow \mathbf{\Omega}^t$
end procedure

As input, the MoLE receives the convergence threshold ξ_Q, and outputs the learned parameters $\hat{\mathbf{\Omega}}$.

2.4. Regularization on Experts

The next step is to solve the maximization (6) for the experts. The experts' contribution to the ELL, $Q_e^t(\boldsymbol{\Theta})$, can be further decomposed to account for the contribution of each expert separately, as follows

$$Q_e^t(\boldsymbol{\Theta}) = \sum_{p=1}^{P} Q_{ep}^t(\boldsymbol{\Theta}), \tag{8}$$

$$Q_{ep}^t(\boldsymbol{\Theta}) = \sum_{i=1}^{n} \gamma_{pi}^t \log \mathcal{N}\left(y_i | \mathbf{x}_i^T \boldsymbol{\theta}_p, \sigma_p^2\right), \tag{9}$$

where $Q_{ep}^t(\cdot)$ is the individual contribution of expert p to the ELL. The new vector of expert coefficients $\boldsymbol{\theta}_p^{t+1}$ is the one that maximizes $Q_{ep}^t(\boldsymbol{\Theta})$, which is the solution of the following weighted least squares problem

$$\boldsymbol{\theta}_p^{t+1} = \arg\min_{\boldsymbol{\theta}_p^*} \frac{1}{2} \sum_{i=1}^{n} \gamma_{pi}^t (y_i - \mathbf{x}_i^T \boldsymbol{\theta}_p^*)^2 = (\mathbf{X}^T \boldsymbol{\Gamma}_p \mathbf{X})^{-1} \mathbf{X}^T \boldsymbol{\Gamma}_p \mathbf{y}, \tag{10}$$

where $\boldsymbol{\Gamma}_p = \text{diag}\left(\gamma_{p1}^t, \ldots, \gamma_{pn}^t\right)$ is the matrix of responsibilities of expert p. However, in presence of collinearity, the inverse $(\mathbf{X}^T \boldsymbol{\Gamma}_p \mathbf{X})^{-1}$ becomes ill conditioned. To overcome this situation, a EN regularization is added as follows to penalize the loss function

$$\boldsymbol{\theta}_p^{t+1} = \arg\min_{\boldsymbol{\theta}_p^*} \frac{1}{2} \sum_{i=1}^{n} \gamma_{pi}^t (y_i - \mathbf{x}_i^T \boldsymbol{\theta}_p^*)^2 + \lambda_p^e \left(\alpha \sum_i |\theta_{pi}^*| + (1-\alpha) \sum_i |\theta_{pi}^*|^2 \right), \tag{11}$$

where λ_p^e is the regularization parameter that controls the sparsity of the solution, and α is the EN penalty. The EN regularization allows the use of the Lasso penalty when $\alpha = 1$, the RR penalty when $\alpha = 0$, or the EN penalty when $\alpha = 0.5$ (a trade-off between Lasso and RR). The Lasso penalty is known to promote sparse solutions by shrinking the regression coefficient towards zero, being adequate when dealing with a large number of inputs. However, the Lasso penalty does not consider the group effect, i.e. for correlated features, it will tend to select one input while shrinking the coefficient of others to zero. The RR penalty alleviates the ill posed problem of $(\mathbf{X}^T \boldsymbol{\Gamma}_p \mathbf{X} + \lambda_p^e \mathbf{I})^{-1}$ by adding a regularization factor to it. In RR, all coefficients will have a contribution to prediction. On the other hand, the EN penalty integrates the benefits of both, it provides sparse solutions (Lasso penalty) while considering the effect group problem (RR penalty).

The error variance update becomes the solution of the following maximization problem

$$\sigma_p^{2\,t+1} = \arg\max_{\sigma_p^{2*}} \sum_{i=1}^{n} \gamma_{pi}^t \log \mathcal{N}\left(y_i | \mathbf{x}_i^T \boldsymbol{\theta}_p, \sigma_p^{2*}\right), \tag{12}$$

which is equal to

$$\sigma_p^{2\,t+1} = \frac{\sum_{i=1}^{n} \gamma_{pi}^t (y_i - \mathbf{x}_i^T \boldsymbol{\theta}_p^{t+1})^2}{\sum_{i=1}^{n} \gamma_{pi}^t}, \tag{13}$$

where the updated $\boldsymbol{\theta}_p^{t+1}$ is used to compute the updated variance term.

The maximization problem in Equation (11) can be achieved using the coordinate gradient optimization descent algorithm, described in [34]. The coordinate gradient descent algorithm minimizes the loss function for each coordinate at a time. It converges to the

optimal value if the loss function is convex and differentiable, conditions that hold for the loss function (11). The updated coefficient of variable j and expert p equals to

$$\theta_{pj}^{t+1} = \frac{S\left(\sum_{i=1}^{n} \gamma_{pi}^{t} x_{ij}(y_i - \tilde{y}_{pi}^{j}), \lambda_p^e \alpha\right)}{\sum_{i=1}^{n} \gamma_{pi}^{t} x_{ij}^2 + \lambda_p^e (1-\alpha)}, \tag{14}$$

where $\tilde{y}_{pi}^{j} = \sum_{l \neq j}^{d} x_{il}\theta_{pl}$ is the fitted value of local expert p, without the contribution of variable j. $S(z, \eta)$ is the soft threshold operator, given by

$$S(z, \eta) = \text{sign}(z)(|z| - \eta)_+ = \begin{cases} z - \eta, & \text{if } z > 0 \text{ and } \eta < |z|, \\ z + \eta, & \text{if } z < 0 \text{ and } \eta < |z|, \\ 0, & \text{if } \eta \geq z. \end{cases} \tag{15}$$

Further details on the derivation of EN learning by coordinate gradient descent can be found in [35]. For the experiments, the glmnet package [34] has been used. It is a computationally efficient procedure that uses cyclical coordinate descent, computed along a regularization path for solving the EN problem.

Another issue is to select a proper value of λ_p^e, which controls the overfitting and the sparsity of the solution. For such purpose, here it is adopted the Bayesian information criterion (BIC), which measures the trade off between accuracy and complexity, and has the following format

$$\text{BIC}^e(\lambda_p^e, \alpha) = n_p^e \log\left(\sum_{i=1}^{n} \gamma_{pi}^{t}\left(y_i - \mathbf{x}_i^T \boldsymbol{\theta}_p^{t+1}\right)^2\right) + \log(n_p^e)\psi_p^e, \tag{16}$$

where $n_e^p = \sum_{i=1}^{n} \gamma_{pi}^{t}$ is the number of effective samples of expert p, and ψ_p^e is the number of the degree of freedom of expert p, and is the number of non zero elements in $\boldsymbol{\theta}$. The BIC has a tendency to penalize complex models, due to the $\log(n_p^e)$ multiplicative factor, giving preference to simpler models. Thus, the selected value of λ_p^e is the one that minimizes the value of $\text{BIC}(\lambda_p^e, \alpha)$.

2.5. Regularization on Gates

On the experts' update, the EN regularization was easily added to penalize the experts' parameters (Section 2.4). On the other hand, during the gates learning, the application of the regularization term is not explicit. The contribution of gates to the ELL, $Q_g^t(\mathbf{G})$, is given by

$$\begin{aligned} Q_g^t(\mathbf{G}) &= \sum_{i=1}^{n}\sum_{p=1}^{P} \gamma_{pi}^{t} \log g_{pi} \\ &= \sum_{i=1}^{n}\left[\sum_{p=2}^{P} \gamma_{pi}^{t} \mathbf{x}_i^T \mathbf{v}_p - \sum_{p=1}^{P} \gamma_{pi}^{t} \log\left(1 + \sum_{k=2}^{P} \exp\left(\mathbf{x}_i^T \mathbf{v}_k\right)\right)\right]. \end{aligned} \tag{17}$$

The solution for the new parameters $\mathbf{G}^{t+1}$ by direct maximization of Equation (17) is not straightforward. Instead, the iterative re-weighted least squares (IRLS) method will be employed. The IRLS algorithm works in the following way. First, define the following auxiliary variable $\hat{\mathbf{v}}_p^k$ as the auxiliary gate parameter in the k-th iteration of the IRLS algorithm. It is updated as follows:

$$\begin{aligned} \hat{\mathbf{v}}_p^{k+1} &= \hat{\mathbf{v}}_p^k + \left[\frac{\partial^2 Q_g^t(\mathbf{G})}{\partial \mathbf{v}_p^k (\mathbf{v}_p^k)^T}\right]^{-1}\left[\frac{\partial Q_g^t(\boldsymbol{G})}{\partial \mathbf{v}_p^k}\right], \\ &= (\mathbf{X}^T \mathbf{R}_p \mathbf{X})\mathbf{X}^T \mathbf{u}_p = (\mathbf{X}^T \mathbf{R}_p \mathbf{X})\mathbf{X}^T \mathbf{R}_p \hat{\mathbf{z}}_p^k, \end{aligned} \tag{18}$$

where

$$\begin{aligned}\mathbf{R}_p &= \text{diag}\big(g_{p1}(1-g_{p1}),\ldots,g_{pn}(1-g_{pn})\big),\\ \mathbf{u}_p &= [g_{p1}-\gamma_{p1}^t,\ldots,g_{pn}-\gamma_{pn}^t]^T,\\ \hat{\mathbf{z}}_p^k &= \mathbf{X}\hat{\mathbf{v}}_p^k - \mathbf{R}_p^{-1}\mathbf{u}_p.\end{aligned}$$

The IRLS algorithm runs until convergence. The employed convergence detection condition is $||\hat{\mathbf{v}}_p^{k+1} - \hat{\mathbf{v}}_p^k||_2 < \xi_I$. At the end of the algorithm, the new gate parameter is updated as $\mathbf{v}_p^{t+1} = \hat{\mathbf{v}}_p^K$, where K is the last iteration of the IRLS algorithm. In practice, few K iterations are necessary in the IRLS algorithm. The IRLS solution can become unstable due to the ill-conditioned inverse $\mathbf{R}_p^{-1}$. Many authors have proposed different alternatives to IRLS, to overcome this problem, such as the proximal-Newton EM in [31] that avoids the matrix inversion in the gates update. Here, a regularization term is added, which works well in practice. Specifically, the regularization term $\zeta\mathbf{I}$ is added such that the inverse becomes $(\mathbf{R}_p + \zeta\mathbf{I})^{-1}$. In experiments, $\zeta = 10^{-3}$ is used.

However, similarly to the experts, the solution for the gates at each iteration of IRLS becomes ill conditioned in the presence of collinearity. Thus, through the closed form solution of the inner loops of the IRLS algorithm in Equation (18), the results derived for the experts are mimicked to the gates. By modifying so, the value of $\hat{\mathbf{v}}_p^k$ to be found at the k-th IRLS algorithm iteration with the EN regularization added, becomes

$$\hat{\mathbf{v}}_p^{k+1} = \arg\min_{\hat{\mathbf{v}}_p^*} \frac{1}{2}\sum_{i=1}^{n} r_{pi}(\hat{z}_{pi}^k - \mathbf{x}_i^T\hat{\mathbf{v}}_p^*)^2 + \lambda_p^g\left(\alpha\sum_i|\hat{v}_{pi}^*| + (1-\alpha)\sum_i|\hat{v}_{pi}^*|^2\right). \tag{19}$$

Then, at each iteration of IRLS, an EN penalty is added to the loss function. In total, Kp EN maximization problems are computed. In a way similar to case of the experts, the solution of (19) can be achieved by using the coordinate gradient descent algorithm, described in [34]. In that case

$$v_{pj}^{k+1} = \frac{S\left(\sum_{i=1}^n r_{pi}x_{ij}(\hat{z}_{pi}^k - \tilde{z}_{pi}^j), \lambda_p^g\alpha\right)}{\sum_{i=1}^n r_{pi}^t x_{ij}^2 + \lambda_p^g(1-\alpha)}, \tag{20}$$

where $\tilde{z}_{pi}^j = \sum_{l\neq j}^d x_{il}v_{pl}$ is the fitted value of gate p, without the contribution of variable j.

The major issue here is to find the most appropriate λ_p^g for each gate p and at each iteration of the IRLS algorithm, where the value of λ_p^g controls the sparsity of the solution. For such purpose, it is adopted the BIC, which measures the trade off between accuracy and complexity, and for the gates, it has the following format

$$\text{BIC}^g(\lambda_p^g,\alpha) = n_p^g\log\left(\sum_{i=1}^n r_{pi}\left(z_{pi} - \mathbf{x}_i^T\boldsymbol{v}_p^k\right)^2\right) + \log(n_p^g)\psi_p^g, \tag{21}$$

where $n_p^g = \left(\sum_i r_{pi}\right)$ is the number of effective samples of gate p, and ψ_p^g is the degrees of freedom of expert p which is equal to the number of non zero elements in $\boldsymbol{v}$. Thus, the selected value of λ_p^g is the one that minimizes the value of $\text{BIC}^g(\lambda_p^g,\alpha)$. The IRLS algorithm with EN penalty is described in Algorithm 2.

Algorithm 2 Iterative re-weighted least squares (IRLS) with elastic net (EN) regularization.

procedure IRLS-EN($\mathbf{v}_p^t$, ξ_I,$\mathbf{u}_p$,$\mathbf{R}_p$,λ_p^g,α)
  Initialize $\hat{\mathbf{v}}_p^0 \leftarrow \mathbf{v}_p^t$, $k \leftarrow 0$, Done $\leftarrow$ FALSE
  while not Done **do**
    $\hat{\mathbf{z}}_p^k \leftarrow \mathbf{X}\hat{\mathbf{v}}_p^k - \mathbf{R}_p^{-1}\mathbf{u}_p$
    $\hat{\mathbf{v}}_p^{k+1} = \arg\min_{\hat{\mathbf{v}}_p^*} \frac{1}{2}\sum_{i=1}^n r_{pi}(\hat{z}_{pi}^k - \mathbf{x}_i^T\hat{\mathbf{v}}_p^*)^2 + \lambda_p^g\left(\alpha\sum_i|\hat{v}_{pi}^*|^2 + (1-\alpha)\sum_i|\hat{v}_{pi}^*|\right)$
    if $||\hat{\mathbf{v}}_p^{k+1} - \hat{\mathbf{v}}_p^k||_2 \leq \xi_I$ **then**
      Done$\leftarrow$TRUE
    end if
    $k \leftarrow k+1$
  end while
  return $\mathbf{v}_p^{t+1} = \hat{\mathbf{v}}_p^k$ ▷ t: EM index; k: IRLS index.
end procedure

2.6. Model Selection and Stop Condition

In ordinary MoLE learning, the ELL is employed as the measure of convergence. Here, in addition to ELL, the BIC criterion will be employed to select the best MoLE architecture along the EM iterations. In that case, the parameters $\mathbf{\Omega}$ to be selected is the ones were the BIC is minimal, instead of ELL. For this purpose, at each EM iteration, the BIC criterion is computed:

$$\text{BIC}(\mathbf{\Omega}) = m\log\left(\frac{\sum_{i=1}^n(y_i-\hat{y}_i)^2}{m}\right) + \log(m)\sum_{p=1}^P\left(\psi_p^e + \psi_p^g\right). \tag{22}$$

Smaller values of BIC means better models. The BIC will increase as the complexity of the MoLE architecture increases. That selection allows the selection of less complex architectures, and overcome the problem of overfitting in the prediction phase. The ELL measures the convergence of the MoLE, while the BIC is considered as the criteria to select the best model. Here, the number of experts selected is not considered a concern, it is assumed that this information is known a priori, and comes from the process to be modeled. However, this criterion can also be employed for model selection. In [36], there is a short discussion on different modeling selections for MoE.

2.7. Different Regularized MoLE

The regularized MoLE learning, presented in previous sections will be used to derive three main regularized MoLE models for the experimental part. First, $\alpha = 1$ is considered to derive the MoLE-Lasso; then $\alpha = 0.5$ is used to derive the MoLE-EN; and $\alpha = 0$ is used to derive the MoLE-RR. The selection for regularized MoLE regularization parameters will follow the BIC procedure, as previously discussed. The source code will be made available at the author's github page (www.github.com/faasouza (accessed on 19 February 2021)).

3. Experimental Results

This section presents the experimental results of using a real industrial polymerization dataset from [8]. The goal in this case study was the estimation of two quality variables related to the resin production in a batch polymerization process. These variables are two chemical properties: the resin acidity number (N_A) and the resin viscosity (μ). The datasets comprise a total of 34 variables measured over the course of the process, such as temperatures, pressures, valve openings, and controller set-points (manually adjusted). The provided dataset contains data from 33 batches (16 months of operating effort). The data were divided into two subsets: 26 batches constitute the training set, and the remaining 6 represent the test set. The properties of dataset are detailed in Table 1.

Table 1. Properties of the polymerization dataset.

Property	Values
# batches train	25
# batches test	6
# samples per batches train	13, 17, 22, 16, 17, 19, 26, 18, 19, 17, 16, 20, 14, 13, 16, 18, 20, 21, 23, 17, 16, 18, 23, 20, 19, 18
average samples per batches train	18
# samples per batches test	23, 21, 24, 20, 19, 20
average samples per batches test	21

The average number of samples per batch for training data is 18 samples, while for the test is 20. The total samples for the test dataset are 127 samples.

3.1. Experimental Settings

The results are divided into two steps. In the first step, the behavior of different regularized MoLEs is evaluated in different scenarios (different numbers of batches of data for training). The objective in this step is to understand the effects of samples in the stability/performance of each MoLE model with different penalties. In the second step, the mixture models that were trained on the full training data are compared to other predictive models by assessing the performance on the test dataset. The predictive performance is measured by the normalized root mean square error (NRMSE) and the coefficient of determination (R^2). The definitions of these metrics are as follows:

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}{\max(\mathbf{y}) - \min(\mathbf{y})}, \tag{23}$$

$$\text{R}^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}, \tag{24}$$

where $\hat{y}_i$ is the predicted output and $\overline{y}$ is the sample mean of output y: $\overline{y} = \frac{1}{n}\sum_{i}^{n} y_i$. Lower values of NRMSE mean better models, while higher values of R^2 mean better models. R^2 can be interpreted as the rate of explained variance of the output to be predicted by the model, with bounds $0 \leq R^2 \leq 1$, where 1 means perfect fit.

3.2. Comparison of a Regularized Mixture of Expert Models

In this section, the performances of different regularized mixture of experts, MoLE-Lasso, MoLE-EN, MoLE-RR, and Mix-PLS, are compared when using different numbers of samples for training. For this purpose, from the total of the 27 batches, 1, 5, 10, 15, and 20 batches were chosen randomly to compose the training dataset for training the MoLE models, while the original test set of six batches was kept fixed for evaluating the performance. The same procedure was repeated 10 times for each experiment, and then the R^2 metric was computed for each repetition. The average results for each of the 10 trials are shown in Table 2, for each model and for the different number of batches in the trainning dataset.

Table 2. Average R^2 values of different regularized MoLE models trained with different numbers of batches on acidity and viscosity datasets. The bold marked values indicate the best performance.

	Acidity				Viscosity			
	MoLE Lasso	**MoLE RR**	**MoLE EN**	**Mix-PLS**	**MoLE Lasso**	**MoLE RR**	**MoLE EN**	**Mix-PLS**
1 batch	**0.484**	0.345	0.365	0.325	0.614	0.463	**0.760**	0.385
5 batches	0.814	**0.843**	0.792	0.651	**0.928**	0.910	0.920	0.772
10 batches	**0.910**	0.883	0.891	0.817	**0.930**	0.920	0.924	0.821
15 batches	**0.924**	0.909	**0.924**	0.905	**0.944**	0.925	0.943	0.905
20 batches	0.927	0.914	**0.928**	0.924	0.945	0.923	0.941	**0.949**

To complement the analysis, the boxplots of R^2 between predictions and test data under a different number of batches for acidity and viscosity are presented in Figure 2.

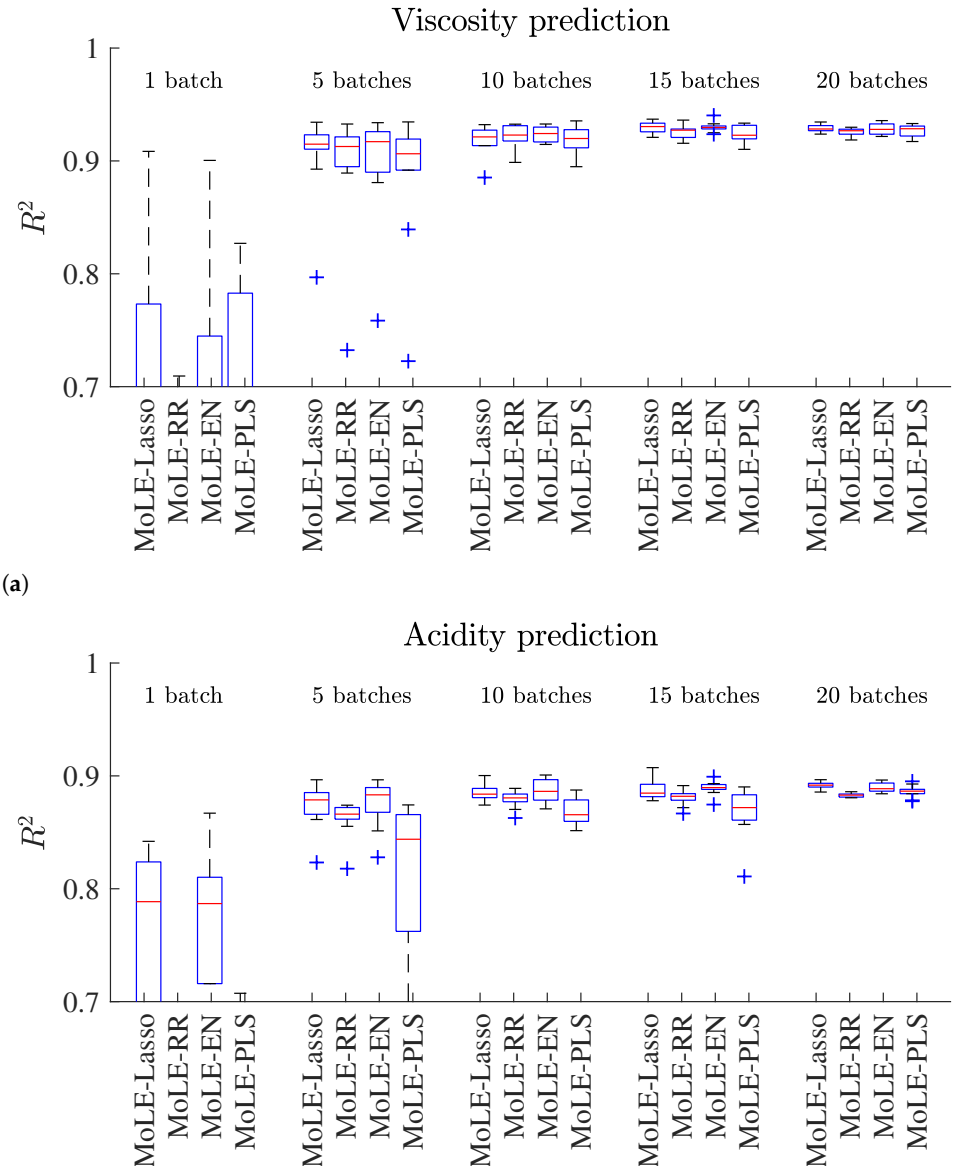

Figure 2. Boxplot for the different MoLE models trained with different numbers of batches for (**a**) resin viscosity prediction and (**b**) resin acidity number prediction. +: They are part of box plot, they are outliers in the residuals.

From Table 2, the results of training with 1 batch for resin acidity number prediction, show that MoLE-Lasso performs better with an average $R^2 = 0.44$. On the other hand, the performance of resin viscosity prediction with 1 batch has better results for MoLE-EN, with a value of $R^2 = 0.76$. This shows that the resin viscosity can be predicted even with few data for training. From the boxplot, Figure 2, the MoLE-Lasso and MoLE-EN have the lowest variance on resin viscosity prediction, while for resin acidity number the variance of prediction is higher. Figure 2, exhibits that the performance of all models increases with the number of batches used for training, as well the variance decreases. An interesting observation is related to the Mix-PLS performance, which shows a high variance of performance for small batches, which decreases as the number of training batches increases. For 20 batches it performs quite similar to, or even better than, Lasso regularization. This suggests that Mix-PLS can be a valid option when dealing with a large number of samples (in our experiment, more than 400 samples). Overall, RR regularization performs poorly among the experiments. This is reinforced by results in Table 2, where MoLE-RR provides the worst results in the majority of experiments.

Overall, all regularized MoLE performed well in the experiments. The MoLE-Lasso has shown to be more stable along the experiments, mainly when dealing with small number of batches, as concluded from the prediction performance and from inspecting the boxplot's results.

3.3. Comparison with Other Predictive Models

To compare the predictive performance of MoLE-Lasso, MoLE-EN, and MoLE-RR with other predictive algorithms, all models were run in the training dataset of 25 batches and tested on the test dataset. For comparison purposes, the following models were implemented: a multivariate linear regression model with Lasso, EN, and RR penalties, an artificial neural network (ANN), a support vector regression, with radial basis kernel (SVR), with hyper-parameters γ (kernel width) and C (Gain), a decision tree (DT), and partial least squares (PLS). The parameters of Lasso, EN, RR, and PLS were selected so as to minimize the BIC criterion. The number of hidden nodes N_h of the ANN and the regularization parameter $\gamma_{\text{LS-SVR}}$ and the Gaussian kernel parameter $\sigma_{\text{LS-SVR}}$ of the SVR were determined using a 10-fold cross validation. Table 3 shows the parameters obtained for the PLS, ANN and SVR models.

Table 3. Parameters selected for PLS, ANN, and SVR models for each data set.

Dataset	PLS	ANN	LS-SVR
Viscosity	LV = 10	$N_h = 3$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 10$
Acidity	LV = 17	$N_h = 3$	$\gamma_{\text{LS-SVR}} = 50$, $\sigma_{\text{LS-SVR}} = 25$

Table 4 shows the prediction performance on the resin acidity number and resin viscosity, for all models, on the test dataset. Table 4, shows that the regularized MoLE models perform better among all datasets. On the other hand, nonlinear models such as ANN, SVM and DT have the worst results. Indeed, according to the results, the performance results have shown that for viscosity this is essentially a linear modeling problem, while for the acidity, there is a clear benefit of using the regularize MoLE methods. As the MoLE models can capture the different phases of the process, it could increase the performance over the linear and nonlinear models.

Table 4. Performance results of all methods on the polymerization dataset, normalized root mean square error (NRMSE), and R^2.

Acidity											
	MoLE Lasso	**MoLE RR**	**MoLE EN**	**Mix-PLS**	**Lasso**	**RR**	**EN**	**PLS**	**ANN**	**SVR**	**DT**
NRMSE	3.64	4.60	6.82	3.62	6.81	6.95	6.91	7.01	4.25	5.94	6.85
R^2(%)	0.973	0.958	0.965	0.974	0.960	0.939	0.959	0.962	3.63	0.929	0.908
Viscosity											
	MoLE Lasso	**MoLE RR**	**MoLE EN**	**Mix-PLS**	**Lasso**	**RR**	**EN**	**PLS**	**ANN**	**SVR**	**DT**
NRMSE	7.41	8.62	8.31	8.90	7.57	8.38	7.41	7.60	9.95	12.38	10.01
R^2(%)	0.941	0.921	0.927	0.916	0.934	0.925	0.942	0.931	0.909	0.890	0.910

4. Conclusions

This paper derived different regularized MoLE models for multiphase-multimode modeling. For this purpose, a MoLE algorithm that integrates Lasso, EN, or RR penalties, was derived and used in the experiments. In the presented case study, the MoLE-Lasso has shown to provide the most stable performance, even when learning with few samples. The other regularized MoLE was shown to have less stability when learning with few samples—for instance, take the MoLE-RR and the Mix-PLS—but they have provided consistent results when increasing the number of training samples. The performance-regularized MoLE is problem dependent, and they all must be a valid option when dealing with multimode data.

Future work will check the performance of MoLE on feature selection on industrial data and new ways to improve the stability of MoLE learning. Future works will also address the comparison of regularized MoLE on different domain problems, with variety with respect to the sample size and the number of input variables and compare its performance against state-of-art-machine learning models.

Author Contributions: Conceptualization, methodology, formal analysis, and writing—original draft preparation, F.S.; conceptualization, software, validation, and editing, J.M. and R.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work was supported by project CONNECTA-X/2017/33354 co-financed by PT2020, in the framework of the COMPETE 2020 Programme, and by the European Union through the European Regional Development Fund (ERDF).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fortuna, L.; Graziani, S.; Xibilia, M.G. *Soft Sensors for Monitoring and Control of Industrial Processes*; Springer: Berlin/Heidelberg, Germany, 2007.
2. Kadlec, P.; Gabrys, B.; Strandt, S. Data-Driven Soft Sensors in the process industry. *Comput. Chem. Eng.* **2009**, *33*, 795–814. [CrossRef]
3. Souza, F.A.A.; Araújo, R.; Mendes, J. Review of Soft Sensors Methods for Regression Applications. *Chemom. Intell. Lab. Syst.* **2016**, *152*, 69–79. [CrossRef]

4. Offermans, T.; Szymańska, E.; Buydens, L.M.C.; Jansen, J.J. Synchronizing process variables in time for industrial process monitoring and control. *Comput. Chem. Eng.* **2020**, *140*, 106938. [CrossRef]
5. Curreri, F.; Graziani, S.; Xibilia, M.G. Input selection methods for data-driven Soft sensors design: Application to an industrial process. *Inf. Sci.* **2020**, *537*, 1–17. [CrossRef]
6. Kadlec, P.; Grbić, R.; Gabrys, B. Review of adaptation mechanisms for data-driven soft sensors. *Comput. Chem. Eng.* **2011**, *35*, 1–24. [CrossRef]
7. Zhao, L.; Zhao, C.; Gao, F. Between-Mode Quality Analysis Based Multimode Batch Process Quality Prediction. *Ind. Eng. Chem. Res.* **2014**, *53*, 15629–15638. [CrossRef]
8. Facco, P.; Bezzo, F.; Barolo, M. Nearest-Neighbor Method for the Automatic Maintenance of Multivariate Statistical Soft Sensors in Batch Processing. *Ind. Eng. Chem. Res.* **2010**, *49*, 2336–2347. [CrossRef]
9. Souza, F.A.A.; Araújo, R. Mixture of Partial Least Squares Experts and Application in Prediction Settings with Multiple Operating Modes. *Chemom. Intell. Lab. Syst.* **2014**, *130*, 192–202. [CrossRef]
10. Ge, Z.; Gao, F.; Song, Z. Mixture probabilistic PCR model for soft sensing of multimode processes. *Chemom. Intell. Lab. Syst.* **2011**, *105*, 91–105. [CrossRef]
11. Ge, Z.; Song, Z.; Zhao, L.; Gao, F. Two-level PLS model for quality prediction of multiphase batch processes. *Chemom. Intell. Lab. Syst.* **2014**, *130*, 29–36. [CrossRef]
12. Liu, Y.; Gao, Z. Real-time property prediction for an industrial rubber-mixing process with probabilistic ensemble Gaussian process regression models. *J. Appl. Polym. Sci.* **2015**, *132*. [CrossRef]
13. Yuan, X.; Ge, Z.; Zhang, H.; Song, Z.; Wang, P. Soft Sensor for Multiphase and Multimode Processes Based on Gaussian Mixture Regression. *IFAC Proc. Vol.* **2014**, *47*, 1067–1072. [CrossRef]
14. Peng, K.; Zhang, K.; You, B.; Dong, J. Quality-related prediction and monitoring of multi-mode processes using multiple PLS with application to an industrial hot strip mill. *Neurocomputing* **2015**, *168*, 1094–1103. [CrossRef]
15. Shao, W.; Ge, Z.; Song, Z. Soft-Sensor Development for Processes With Multiple Operating Modes Based on Semi-supervised Gaussian Mixture Regression. *IEEE Trans. Control Syst. Technol.* **2018**, *27*, 2169–2181. [CrossRef]
16. He, Y.; Zhu, B.; Liu, C.; Zeng, J. Quality-Related Locally Weighted Non-Gaussian Regression Based Soft Sensing for Multimode Processes. *Ind. Eng. Chem. Res.* **2018**, *57*, 17452–17461. [CrossRef]
17. Shi, X.; Kang, Q.; Zhou, M.; Abusorrah, A.; An, J. Soft Sensing of Nonlinear and Multimode Processes Based on Semi-Supervised Weighted Gaussian Regression. *IEEE Sens J.* **2020**, *20*, 12950–12960. [CrossRef]
18. Wang, J.; Shao, W.; Song, Z. Student's-t Mixture Regression-Based Robust Soft Sensor Development for Multimode Industrial Processes. *Sensors* **2018**, *18*, 3968. [CrossRef]
19. Jacobs, R.A.; Jordan, M.I.; Nowlan, S.J.; Hinton, G.E. Adaptive Mixtures of Local Experts. *Neural Comput.* **1991**, *3*, 79–87. [CrossRef]
20. Jacobs, R.A.; Peng, F.; Tanner, M.A. A Bayesian Approach to Model Selection in Hierarchical Mixtures-of-Experts Architectures. *Neural Netw.* **1997**, *10*, 231–241. [CrossRef]
21. Yuksel, S.E.; Wilson, J.N.; Gader, P.D. Twenty Years of Mixture of Experts. *IEEE Trans. Neural Netw. Learn. Syst.* **2012**, *23*, 1177–1193. [CrossRef]
22. Souza, F.; Araújo, R. Mixture of Elastic Net Experts and its Application to a Polymerization Batch Process. In Proceedings of the 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), Porto, Portugal, 18–20 July 2018; pp. 939–944.
23. Khalili, A. New estimation and feature selection methods in mixture-of-experts models. *Can. J. Stat.* **2010**, *38*, 519–539. [CrossRef]
24. Peralta, B.; Soto, A. Embedded local feature selection within mixture of experts. *Inf. Sci.* **2014**, *269*, 176–187. [CrossRef]
25. Tang, Q.; Karunamuni, R.J. Robust variable selection for finite mixture regression models. *Ann. Inst. Stat. Math.* **2018**, *70*, 489–521. [CrossRef]
26. Chamroukhi, F.; Huynh, B.T. Regularized Maximum-Likelihood Estimation of Mixture-of-Experts for Regression and Clustering. In Proceedings of the The International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018 .
27. Tibshirani, R. Regression Shrinkage and Selection via the Lasso. *J. R. Stat. Soc. Ser. B* **1994**, *58*, 267–288. [CrossRef]
28. Hoerl, A.E.; Kennard, R.W. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics* **1970**, *12*, 55–67. [CrossRef]
29. Zou, H.; Hastie, T. Regularization and variable selection via the elastic net. *J. R. Stat. Soc. Ser. B* **2005**, *67*, 301–320. [CrossRef]
30. Fan, J.; Li, R. Variable Selection via Nonconcave Penalized Likelihood and Its Oracle Properties. *J. Am. Stat. Assoc.* **2001**, *96*, 1348–1360. [CrossRef]
31. Huynh, B.T.; Chamroukhi, F. Estimation and Feature Selection in Mixtures of Generalized Linear Experts Models. *arXiv* **2019**, arXiv:1907.06994.
32. Nguyen, T.; Nguyen, H.D.; Chamroukhi, F.; McLachlan, G.J. An l_1-oracle inequality for the Lasso in mixture-of-experts regression models. *arXiv* **2020**, arXiv:2009.10622.
33. Jordan, M.I.; Jacobs, R.A. Hierarchical Mixtures of Experts and the EM Algorithm. *Neural Comput.* **1994**, *6*, 181–214. [CrossRef]
34. Friedman, J.; Hastie, T.; Tibshirani, R. Regularization Paths for Generalized Linear Models via Coordinate Descent. *J. Stat. Softw.* **2010**, *22*, 1–22. [CrossRef]

35. Friedman, J.; Hastie, T.; Höfling, H.; Tibshirani, R. Pathwise Coordinate Optimization. *Ann. Appl. Stat.* **2007**, *1*, 302–332. [CrossRef]
36. Chamroukhi, F. Robust mixture of experts modeling using the t distribution. *Neural Netw.* **2016**, *79*, 20–36. [CrossRef] [PubMed]