Biointerface Research in Applied Chemistry
Open-Access Journal (ISSN: 2069-5837)

Article
Volume 13, Issue 2, 2023, 184
https://doi.org/10.33263/BRIAC132.184

A Comparative Study of Different Kernel Functions Applied to LW-KPLS Model for Nonlinear Processes

Joyce Chen Yen Ngu [1], Wan Sieng Yeo [1,*]

[1] Department of Chemical Engineering, Curtin University Sarawak Campus, CDT 250, Miri 98009, Sarawak, Malaysia
* Correspondence: christineyeo@curtin.edu.my (W.S.Y.);

Scopus Author ID 57199053825
Received: 12.12.2021; Accepted: 3.01.2022; Published: 2.04.2022

Abstract: Soft sensors are inferential estimators when the employment of hardware sensors is inapplicable, expensive, or difficult in industrial plant processes. Currently, a simple soft sensor, namely locally weighted partial least squares (LW-PLS), which can cope with the nonlinearity of the process, has been developed. However, LW-PLS exhibits the disadvantages of handling strong nonlinear process data. To address this problem, Kernel functions are integrated into LW-PLS to form locally weighted Kernel partial least squares (LW-KPLS). Notice that a minimal study was carried out on the impact of different kernel functions that have not been integrated with the LW-KPLS, in which this model has the potential to be applied to different chemical-related nonlinear processes. Thus, this study investigates the predictive performance of LW-KPLS with several different Kernel functions using three nonlinear case studies. As the results, the predictive performances of LW-KPLS with Polynomial Kernel are better than other Kernel functions. The values of root-mean-square errors (RMSE) and error of approximation (E_a) for the training and testing dataset by utilizing this Kernel function are the lowest in their respective case studies, which are 34.60% to 95.39% lower for RMSEs values and 68.20% to 95.49% smaller for E_a values.

Keywords: soft sensors; locally weighted kernel partial least squares; kernel functions; linear kernel; polynomial kernel; nonlinear chemical processes.

Abbreviations: CPU: Central Processing Unit; CSTR: Continuous Stirred Tank Reactor; ITHS: Intelligent Tuned Harmony Search; GK: Gaussian Kernel; JIT: Just-In-Time; KPLS: Kernel Partial Least Squares; LW-KPLS: Locally Weighted Kernel Partial Least Squares; LW-PLS: Locally Weighted Partial Least Squares; MK: Multiquadric Kernel; MLR: Multiple Regression; PCR: Principle Component Regression; PLS: Partial Least Squares; RMSE: Root-Mean-Square Error; STHE: Shell and tube heat exchangers; x: Input variables; y: Output variables; n: Number of samples; L: Number of output variables; M: Number of input variables; T: Transpose matrix; X: Matrix of input; Y: Matrix of output; $\hat{y}_q$: Predicted output; x_q: Queried matrix of input; 1_n: Vector with length n; 1_{n_t}: Vector with length n_t; B: Dual representation of the scaling of projection direction; S: Latent variables number; Ω: Similarity matrix; ω_n: Index of similarity; φ: Localization parameter; σ_n: Standard deviation of d_n; t_s: s^{th} latent variable of X_s; w_s: Eigenvector of $X_s^T \Omega Y_s Y_s^T \Omega X_s$ that correlates to maximal eigenvalue; p_s: s^{th} loading vector of X_s; q_s: s^{th} regression coefficient vector; $t_{q,s}$: s^{th} latent variable of X_q; N_t: Total samples number; N_1: Number of training samples; N_2: Number of testing samples; y_i: Actual values of output; $\hat{y}_i$: Predicted values of output; $\bar{y}$: Actual output mean value: $RMSE_1$: Root-Mean-Square Error of Training Data; $RMSE_2$: Root-Mean-Square Error of Testing Data; b: Kernel Parameter; E_a: Error of Approximation; σ: Sigma.

1. Introduction

Hardware sensors are generally adapted in chemical plants for process sensing and control systems [1]. However, there are several existing drawbacks for the hardware sensors from the prospect of technical and economic. For instance, shortages of technology, the time delay from measurement, and many online variables contribute to costly devices [2]. Meanwhile, these sensors allow better process control and monitoring for optimization, which can lead to business profit targets and minimize costs while meeting the desired product quality [3]. Moreover, they also allow chemical plants to be in a more sustainable and profitable situation by achieving operational excellence goals.

Soft sensors, therefore, are being introduced to overcome the limitations of the hardware sensors. Soft sensors, which are inferential estimators, are preferable when the employment of hard sensors is inapplicable, expensive, or difficult in the industrial plant processes. Generally, soft sensors are assorted as inferential models, which require only a simple step on variables estimation. Moreover, soft sensors can easily predict hard-to-measure process variables and provide beneficial information from the view of fault detection using easy-to-measured variables from the hardware sensors [4]. In other words, soft sensors are ordinarily adopted for online predictions according to the analysis of measurement signals from hardware sensors with software executed mathematical models [5,6]. In industry, it is commonly used in industrial processes for providing predictions to the control system in achieving their operational excellence, which ultimately leads to more sustainable and profitable operations [7,8]. Besides, soft sensors with better measurements can generate better product quality and improve the system's performance through enhanced operation, fault detection, and process management [9,10]. Thus, they are commonly used in the industrial process to settle issues that correspond to cost, availability, quality, and reliability of measurements [11].

Soft sensors have been broadly developed to measure hard-to-measure variables such as product quality or other significant online or offline indexes, which cannot be estimated by hardware sensors in real-time [12]. The term soft measurement is advocated and acts as a parameter forecast tool subjected to online process measurement [13]. In an extremely impressive view, soft sensors are classified into two distinct classes: model-driven and data-driven based soft sensors. The model-driven or first principal models depict the process physically and chemically. However, they are sometimes difficult to use since they require the structure's complexity and insufficient professional knowledge of the chemical processes [12]. Therefore, data-driven soft sensors are preferable in this situation.

The data-driven-based soft sensors predict the connection between the primary and secondary variables extracted from the processes in the circumstance of real conditions [14]. They acquire a great amount of interest [12]. Data-driven modeling is superior for mapping out the model if the development of the modeling sample is enough to train the model [13]. Besides, multivariate regression models are usually used for data-driven soft sensors development. They include multiple regression (MLR), principal component regression (PCR), and partial least squares (PLS) regression [15]. However, the predictive performance of the MLR is disappointing and imprecise when it comes to handling a huge amount of collinear data and noise estimation [16,17]. Hence, PCR, and PLS are adopted to overcome these drawbacks.

PCR and PLS can overcome the constraints or regression when they set up a MLR model with the low capability to create a nonlinear connection between dependent and

independent variables [18,19]. PLS also generates the components in terms of modeling the relationship between input and output variables and retaining most of the information in input variables simultaneously, which is dissimilar from PCR which uses input variables in modeling [20]. Due to better predictive performance, the PLS model is proved to be more used as a dominant multivariate statistical tool when applied in the real world than PCR [21]. However, both PCR and PLS, which are latent-variable-based methods, cannot handle nonlinear process data [22].

Nonlinear models are postulated to generate better accuracy estimation in processing high nonlinearity data [23]. Nonlinear soft sensors usually employ artificial neural or Kernel-based methods such as Kernel-PLS (KPLS), support vector regression, and artificial neural network [15]. The assembling of Kernel functions in PLS supports KPLS to become better efficiency algorithms in conquering nonlinearity data since KPLS maps the data into a higher-dimensional space. Nonetheless, the accuracy of the non-adaptive and nonlinear soft sensors degrades due to the alternation in the process conditions.

To adapt the alternation in process characteristics and nonlinearity, Just-In-Time (JIT) modeling can be resorted to developing adaptive soft-sensors or virtual sensors. Locally weighted partial least squares (LW-PLS), a JIT model, has been studied and prosperity applied in various industries due to its simplicity and capability [24]. Nevertheless, LW-PLS fails to accomplish a successful estimation for nonlinear processes. Hence, the locally weighted Kernel partial least squares (LW-KPLS) model is established by assembling Kernel functions with LW-PLS to handle the highly nonlinear issue [25]. Due to the good predictive performance, the LW-KPLS was also studied by Yeo, Saptoro, and Kumar [25], Yeo and Lau [26], and Yeo, *et al.* [27]. However, the impact of other different Kernel functions on the predictive performance of LWKPLS has not been detailly carried out. This study investigates the effect of these different Kernel functions that have not been accomplished yet in the LW-KPLS model. Then, the performance of LW-KPLS, which was subjected to different Kernel functions, was evaluated and compared.

2. Materials and Methods

In this section, the description of LW-KPLS is expressed. Then, it is followed by optimizing parameters, data splitting and setting of parameters, measurement of quality prediction, kernel functions, and specifications of computer configuration.

2.1. Locally weighted Kernel partial least squares.

By applying a Kernel function, the LW-KPLS model was built under the LW-PLS procedures. As shown in (1) and (2), the input variables and output variables, x, and y, are expressed [25].

$$x_n = \left[x_{n1}, x_{n2}, \ldots, x_{nM}\right]^T \quad (1)$$

$$y_n = \left[y_{n1}, y_{n2}, \ldots, y_{nL}\right]^T \quad (2)$$

The symbols n, M, L, and T represent the number of samples, number of input variables, number of output variables, and a transpose of a matrix, respectively. The input matrix is symbolized by $X \in R^{n \times M,}$ while the output matrix is symbolized by $Y \in R^{n \times L}$. These input and output matrices are the databases saved x and y in MATLAB. To obtain the predicted output,

$\hat{y}_q$ for a query, x_q for a nonlinear process, and LW-KPLS algorithm was developed by Yeo, Saptoro, and Kumar [25], and it is shown as follows:

Step 1: To obtain the Kernel matrices for input variables, V, and query, V_q, they are mapped into a high dimensional feature space by applying a Kernel function.

Step 2: Using (3) and (4), mean centering is done on the mapped V and V_q.

$$\tilde{V} = \left(I - \frac{1}{n}1_n 1_n^T\right) V \left(I - \frac{1}{n}1_n 1_n^T\right) \quad (3)$$

$$\tilde{V}_q = \left(V_q - \frac{1}{n}1_{n_t} 1_n^T V\right)\left(I - \frac{1}{n}1_n 1_n^T\right) \quad (4)$$

where the length vector with n and the length vector with n_t are symbolized by 1_n and 1_{nt}, respectively.

Step 3: Dual KPLS discrimination is conducted by applying (5) to produce B.

$$B = YY'V\beta \text{ with normalization, } \beta = \frac{\beta}{\|\beta\|} \quad (5)$$

where B represented the dual representation of the scaling in the projection direction.

Step 4: The computation on re-scaled queried and input, V_q, and V variables are conducted using (6) and (7).

$$X_q = V_q B \quad (6)$$

$$X = VB \quad (7)$$

More details about the LW-KPLS model can be found in Yeo, Saptoro, and Kumar [25].

2.2. Optimisation of parameters.

By carrying the optimization of parameters in a model, the model can be presumed as a well-performing model. In the LW-KPLS algorithm, the parameters used are N_t, S, ∅ and b. N_t symbolizes the total number of samples; S symbolizes the amount of latent variable while the localization parameter is symbolized by ∅ , and b is a kernel parameter. N_t is the total number of samples used for the LW-KPLS model, and it is an insensitive parameter to the LW-KPLS algorithm. Moreover, according to the study carried out by Yeo, Saptoro, and Kumar [25], the first few latent variables of PLS-based models can indicate the predominant data feature; hence S is set at 1. LW-KPLS model uses ∅ to cope with the nonlinearity of the process data. The nonlinearity of the process can be handled by the model when the ∅ is small, thus ∅ is set as 0.1 [25]. In addition, the Kernel parameter, *b* is the most ingenious parameter of the LW-KPLS algorithm; hence it must be fine-tuned properly for each case study.

2.3. Data splitting and setting of parameters.

A total of 5,000 data sets is achieved and separated into training and testing sets. Hence, a dataset ratio of 90% to 10% is used for training and testing purposes, respectively. Then, N_t is 5,000 while N_1 and N_2 represent the number of training and testing data are 4,500 and 500, respectively. As mentioned earlier, S is allocated as 1, and an optimal value of 0.1 is allocated ∅. By referring to Table 1, the parameter values involved in each case study are summarized.

Table 1. Values of parameters in the algorithm.

Parameters	**N_t**	**N_1**	**N_2**	**S**	**∅**
Values	5,000	4,500	500	1	0.1

2.4. Measurement of quality prediction.

The implementation of indexes which are root-mean-square error (RMSE) and error of approximation (E_a), are carried out to evaluate the prediction performance of LW-KPLS quantitatively. The formula of RMSE is demonstrated in (8), while E_a is calculated using (9) [28-30].

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N_t}\left(\hat{y}_i - y_i\right)^2}{N_t}} \quad (8)$$

$$E_a = \left(\frac{N_1}{N_t}\right)RMSE_1 + \left(\frac{N_2}{N_t}\right)RMSE_2 + \left|RMSE_1 - RMSE_2\right| \quad (9)$$

where $\hat{y}_i$ and y_i are the predicted and actual values of the output variable, respectively, while $\bar{y}$ is the mean of the actual output. $RMSE_1$ and $RMSE_2$ are the RMSE of training and testing data, respectively. Sometimes, the overall predictive performance of the LW-KPLS model is difficult to be determined when two RMSE values ($RMSE_1$ and $RMSE_2$) for each case study are varied. For instance, there could be a case where the error of training data is the smallest, the testing set error could be high. To solve this issue, E_a is applied. The lower the E_a, the better the predictive performance of the model [25].

2.5. Kernel functions.

Kernel functions map the nonlinear data to a high dimension space [31]. In this study, several different Kernel functions are integrated into the LW-KPLS model; then, their results were accessed and compared. Table 2 shows the equations for Kernel functions employed in this study, where the majority of them have not been tested in the LW-KPLS model yet.

Table 2. General and modified form of Kernel functions applied.

Type of Kernel function	Kernel function
Linear Kernel (1)	$k(x, y) = x^T y + \sigma$
Linear Kernel (2)	$k(x, y) = bx^T y + \sigma$
Polynomial Kernel (1)	$k(x, y) = \left(x^T y + 1\right)^b$
Polynomial Kernel (2)	$k(x, y) = \left(bx^T y + b\right)^b$
Gaussian Kernel	$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$
Laplacian Kernel	$k(x, y) = \exp\left(-\frac{\|x - y\|}{\sigma}\right)$
Hyperbolic Tangent (Sigmoid) Kernel	$k(x, y) = \tanh\left(bx^T y + 1\right)$
Multiquadric Kernel	$k(x, y) = \sqrt{\|x - y\|^2 + b^2}$
Inverse Multiquadric Kernel	$k(x, y) = \frac{1}{\sqrt{\|x - y\|^2 + b^2}}$

Type of Kernel function	Kernel function
Power Kernel	$k(x, y) = -\|x - y\|^b$
Log Kernel	$k(x, y) = -\log\left(\|x - y\|^b + 1\right)$
Cauchy Kernel	$k(x, y) = \frac{1}{1 + \frac{\|x - y\|^2}{\sigma^2}}$

2.6. Specifications of computer configuration.

In this section, the computer configuration used to execute the LW-KPLS model is specified. The operating system (OS) is Windows 10 (64-bit). Besides, Intel(R) Core (TM) i7-4500U j(1.80 GHz) 2.40 GHz is the central processing unit or the main processor. Furthermore, the random-access memory installed is 4.00 GB, while the MATLAB version R2019b is used.

3. Results and Discussion

3.1. Case Studies.

The LW-KPLS model with different Kernel functions was applied to three different case studies. The first case study is a predictive control of a wastewater treatment process [32]. The second case study is re-scaled the Rosenbrock function, which was studied by Turgut *et al.* [33]. Lastly, the third case study is the process of polymerization of methyl methacrylate, which was adopted from Shafiee *et al.* [34].

3.1.1. Case study 1: Predictive control of a wastewater treatment process.

Case study 1 demonstrates the wastewater treatment process studied by Caraman, Sbarciog and Barbu [32]. One of the most significant environmental conservation processes in the industry is the wastewater treatment process. It is usually complicated, nonlinear, and multivariable as it has multiple inputs and outputs. In this wastewater treatment process, an aeration tank, a biological reactor, consists of a mixture of liquid and suspended solids. A microorganism population is raised to eliminate the organic substrate from the mixture. A settler acts as a clarifier tank to split the sludge and the clear effluent using gravity. An amount of sludge is recycled back to the aeration tank while the other amount is eliminated. The details of this case study, including the mass balance equations and their notations, can be found in Caraman, Sbarciog, and Barbu [32].

3.1.2. Case study 2: Rescaled Rosenbrock function.

This case study was carried out by Turgut, Turgut, and Coban [33] and Picheny, *et al.* [35], in which a nonlinear test function expressed in (10) is, the Rosenbrock function, is employed. This function is re-scaling to keep a mean of zero and one variance [35].

$$y(x) = \frac{1}{3.755 \times 10^5}\left[\sum_{j=1}^{3}\left(100\left(\bar{x}_{j+1} - \bar{x}_j^2\right)^2 + \left(1 - \bar{x}_i\right)^2\right) - 3.827 \times 10^5\right] \quad (10)$$

where $\bar{x}$=15x-5 and -2.048 $\leq x_i \leq$2.048.

3.1.3. Case study 3: Polymerization of methyl methacrylate.

In this case study, a continuous stirred tank reactor (CSTR) with a cooling jacket is employed to polymerize Methyl Methacrylate. Surplus heat is delivered throughout the exothermic polymerization. Hence, the cooling jacket of CSTR plays an important role in removing the heat. There are three feed streams for CSTR, which include the feed streams for solvent, monomer, and initiator. At the outlet stream of CSTR, the unreacted polymer, initiator, solvent, and polymer product are transmitted downstream to undergo the separation process. Table 3 tabulates the input variables for this case study. On the other hand, zeroth, first, and second molecular weight distribution moments are the output variables. More details of this case study can be obtained in [34].

Table 3. Input variables of polymerization of methyl methacrylate.

Input variable	x_1	x_2	x_3
Variable description	Initiator concentration (gmol/L)	Monomer concentration (gmol/L)	Reactor temperature (K)

3.2. Results and discussion.

3.2.1. Case study 1: Predictive control of a wastewater treatment process.

As shown in Table 4, the values of RMSE for training and testing datasets and E_a with the Kernel functions after tuning the b in Case study 1 were tabulated in ascending order. The b must be tuned with care to prevent overestimation and underestimation [36]. A lower RMSE indicates the closeness of the regression line to the data points and gives a better fit to the data. In this case study, Linear Kernel (1) and Linear Kernel (2), as well as Polynomial Kernel (1) and Polynomial Kernel (2), gave the same results, and they have the lowest RMSEs. From Table 2 above, it can be seen that Linear Kernel and Polynomial functions have slightly different characteristics in which Linear Kernel is equivalent to a Polynomial Kernel of degree one [37]; hence the result of RMSEs as well as E_a by using Linear Kernel (1) and Linear Kernel (2) were equivalent. However, Linear Kernel (1), Linear Kernel (2), Polynomial Kernel (1), and Polynomial Kernel (2) were chosen as the best Kernel functions since they gave the lowest RMSEs, comparable central processing unit (CPU) running time, and more accurate as compared to other Kernel functions in this case study.

Moreover, from Table 4, the range of central processing unit (CPU) running time for the training and testing data set, CPU_1 and CPU_2 of Linear Kernel (1), Linear Kernel (2), Polynomial Kernel (1), and Polynomial Kernel (2) were between 252s and 263s as well as 29s and 33s, respectively which are comparable to other Kernel functions. Moreover, they have similar results that gave the values of 0.6703, 0.6728, and 0.6731 for the RMSE of training and testing data and the E_a, respectively. Since the values of the output variable in this case study were small, between -1.9536 to 23.9143, thus the RMSE and E_a values were also small. In other words, the higher the values of the output variables, the bigger the values for RMSE and E_a. Besides, as the testing data set utilized training data to develop the model, hence it is observed that the $RMSE_1$ is smaller than $RMSE_2$ [38]. Other than that, among the Kernel functions, Multiquadric Kernel (MK) gave the largest value of RMSEs and E_a at 1.1922, 2.5505, and 2.6863, respectively, when its Kernel parameter, b was tuned at 2. These results indicate that MK, which transforms the scattered data into a very precise, appropriate model of a graph or surface [39], does not fit the data in this case study. By comparing the results from

MK, the Linear Kernel (1), Linear Kernel (2), Polynomial Kernel (1), and Polynomial Kernel (2) improved the results by 43.78% for $RMSE_1$, 73.62% for $RMSE_2$, and 74.94% for E_a.

Hence, it can be concluded that the predicted data from the LW-KPLS model with the Linear Kernel (1), Linear Kernel (2), Polynomial Kernel (1), and Polynomial Kernel (2) are said to be closer to the real nature of the data set as the E_a is lower as compared to the rest of the kernel functions. The training and testing data graphs on their actual output values against the predicted outputs from the LW-KPLS model with the Linear Kernel (1) were plotted and shown in Figures 1 and 2. Since these Kernel functions gave the best results, Linear Kernel (1) was chosen to plot Figures 1 and 2. From these figures, the predicted output of training and testing data is allocated more concentrated along with the actual output of training and testing data. These results indicate that the LW-KPLS model with Linear Kernel (1), Linear Kernel (2), Polynomial Kernel (1), and Polynomial Kernel (2) can fit the data in Case study 1 well.

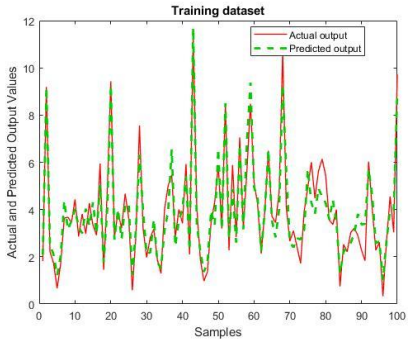

Figure 1. Graph of a training dataset of output variable of actual and predicted output values for output 1 from LW-KPLS model with Linear Kernel (1) in Case Study 1.

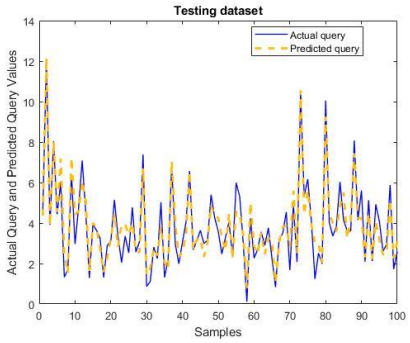

Figure 2. Graph of testing dataset of output variable of actual query and predicted query output values for output 1 from LW-KPLS model with Linear Kernel (1) in Case Study 1.

3.2.2. Case study 2: Rescaled Rosenbrock function.

The RMSE values for training and testing data and the value of E_a with Kernel functions for Case study 2 were organized in ascending order, as shown in Table 5. By comparing the results obtained from the Kernel functions, Polynomial Kernel (1) gave the lowest RMSEs and E_a. When b was tuned at a value of 4, the RMSE values for training and testing data sets and E_a obtained from Polynomial Kernel (1) were 55.6670, 59.5754, and 59.9662, respectively. From Table 2, the results indicate the Polynomial Kernel (1) had mapped the dataset in this case study to a high dimensional space, enabling the LW-KPLS model to predict better results than the rest of the kernel functions.

From Table 5, it can be found that the Polynomial Kernel (2) and Gaussian Kernel (GK) obtained the nearest values when their b were tuned at 10. Polynomial Kernel (2) gave the values of RMSEs and E_a as 57.3369, 61.4174, and 61.8255, respectively, while GK gave the

values of RMSEs and E_a 61.3635, 61.9157, and 61.9710, respectively. The Polynomial Kernel (2) can adapt to problems of normalized training data, while GK can estimate the anticipated value [40]. Thence, both Kernel functions can be considered to obtain the nearest values of RMSEs and E_a. Additionally, the results obtained using MK are the highest among the Kernel functions. This result indicates that the MK with a non-positive definite Kernel [41] does not work well with the data in this case study. MK gave the values of RMSEs and E_a, which are 85.1186, 224.8771, and 238.8530, respectively. As compared to MK, Polynomial Kernel (1) performed an improvement of 34.60%, 73.51%, and 74.89% for $RMSE_1$, $RMSE_2$, and E_a, respectively.

On the other hand, the average CPU running time for the training and testing data set, CPU_1 and CPU_2 of Polynomial Kernel (1), were 265.08s and 32.41s, respectively, which are not the lowest. However, it is still the best model since it gave the lowest RMSEs and E_a compared to other Kernel functions. In Figures 3 and 4, the comparison of the actual output values against the predicted outputs from the LW-KPLS model with Polynomial Kernel (1) for training and testing data were plotted and shown. Both predicted values of the training and testing data sets are to be seen within the data range of actual output. These results express the integration of nonlinear features in LW-KPLS with the Polynomial Kernel (1) function and have proved the effectiveness of LW-KPLS when dealing with the high nonlinearity of data [42].

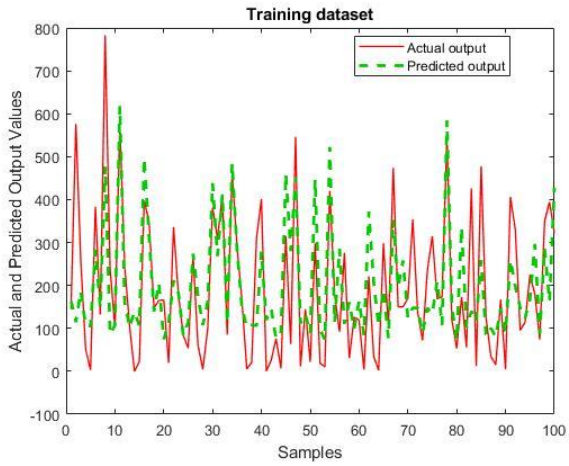

Figure 3. Graph of a training dataset of output variable of actual and predicted output values for output 1 from LW-KPLS model with polynomial Kernel (1) in Case Study 2.

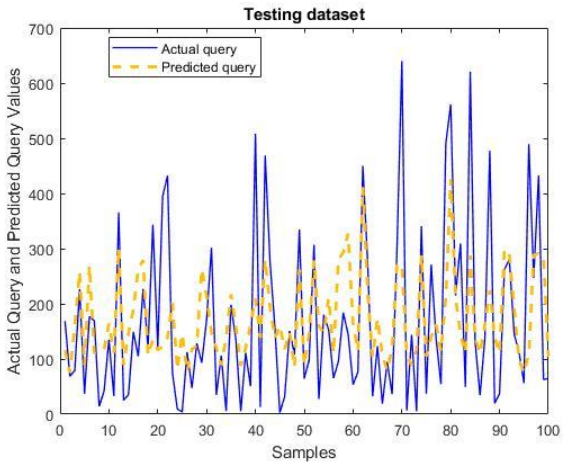

Figure 4. Graph of testing dataset of output variable of actual query and predicted query output values for output 1 from LW-KPLS model with polynomial Kernel (1) in Case Study 2.

3.2.3. Case study 3: Polymerization of methyl methacrylate.

This case study is a multi-input and multi-output case study with three outputs; hence three sets of RMSE values for the training and testing data sets and E_a in Kernel functions are being studied. Tables 6 and 7 show the values of RMSE for training and testing data, and E_a with the Kernel functions after tuning the Kernel parameter in this case study were demonstrated in ascending order. By carrying out the comparison, it can be concluded that the lowest RMSEs and E_a have been achieved by adopting Linear Kernel (1) and (2) as well as Polynomial Kernel (1) and (2). Hence, they are selected as the best models in this case study.

For output 1, Linear Kernel (1) and Linear Kernel (2), as well as Polynomial Kernel (1) and Polynomial Kernel (2), provided the values of 1.0117×10^{-5}, 5.9672×10^{-3}, and 6.2180 for the RMSE for training and testing data sets and E_a, respectively. For output 2, Linear Kernel (1) and Linear Kernel (2), as well as Polynomial Kernel (1) and Polynomial Kernel (2) gave the values of the RMSE for training and testing data sets and E_a, which are 3.2955×10^{-5}, 0.0197, and 21.3119, respectively. Moreover, output 3 gave the RMSEs for training and testing data and E_a of 3.5239×10^{-5}, 0.0211, and 22.8213, respectively. Furthermore, in Table 6, CPU_1 and CPU_2 of Linear Kernel (1) and Linear Kernel (2), as well as Polynomial Kernel (1) and Polynomial Kernel (2), were ranged from 252s to 260s for training data, and 26s to 32s for testing data. It can be said that they gave lower CPUs running times than the majority of other Kernel functions.

Referring to Table 6, MK provided the highest values of RMSEs and E_a for the three outputs. It gave values of 3.2006×10^{-5}, 0.0191, 20.0034 for $RMSE_1$ of outputs 1, 2 and 3, respectively, 4.7745×10^{-4}, 0.3383, 462.2747 for $RMSE_2$ of outputs 1, 2 and 3, respectively, 5.2200×10^{-4}, 0.3702, 506.5018 for E_a of outputs 1, 2 and 3, respectively. According to Drewnik and Pasternak-Winiarski [43], MK is a non-positive and definite Kernel function. However, MK does not cope well with the data in this case study. As a result, Linear Kernel (1) and Linear Kernel (2), as well as Polynomial Kernel (1) and Polynomial Kernel (2), are greater than MK as they can enhance the performance better by 68.39% to 68.92% for $RMSE_1$, 93.10% to 95.39% for $RMSE_2$, and 93.25% to 95.49% for E_a, respectively.

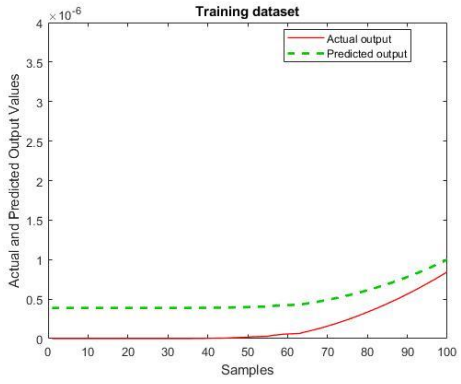

Figure 5. Graph of a training dataset of output variable of actual and predicted output values for output 1 from LW-KPLS model with Linear Kernel (1) in Case Study 3.

The graphs of training and testing data of the three output variables, which show the actual and predicted output values from LW-KPLS with Linear Kernel were also plotted, as illustrated in Figures 5 to 10. Similar to Case study 1, since these Kernel functions gave the best results, Linear Kernel (1) was chosen to plot Figures 5 to 10. Figures 5, 7, and 9 show that the predicted output values of training data sets for outputs 1, 2, and 3 are also close to their

actual output values. In contrast, the predicted query outputs of testing data for the 3 outputs are also closed to their actual query output values by observing Figures 6, 8, and 10. To conclude, LW-KPLS with Linear Kernel (1) and Linear Kernel (2) as well as Polynomial Kernel (1) and Polynomial Kernel (2) can fit the data in this study very well and accomplish superior performance when compared to other Kernel functions.

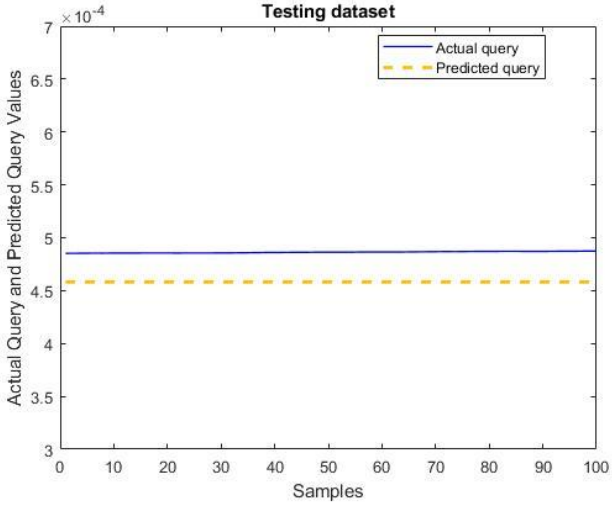

Figure 6. Graph of testing dataset of output variable of actual query and predicted query output values for output 1 from LW-KPLS model with Linear Kernel (1) in Case study 3.

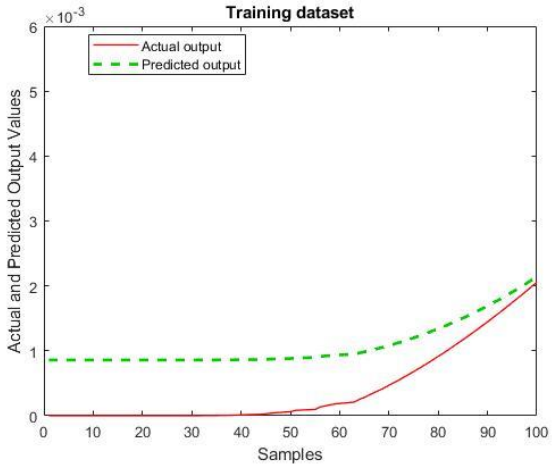

Figure 7. Graph of a training dataset of output variable of actual and predicted output values for output 2 from LW-KPLS model with Linear Kernel (1) in Case study 3.

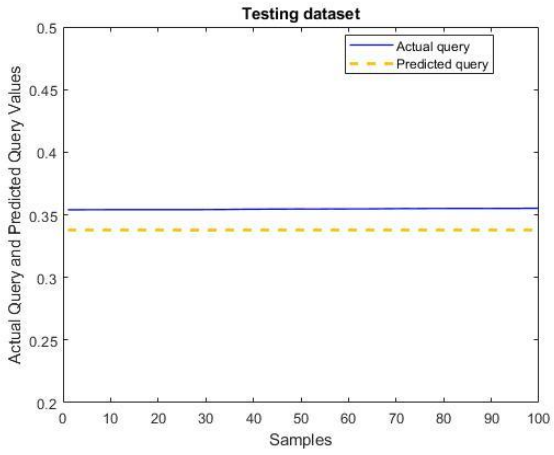

Figure 8. Graph of testing dataset of output variable of actual query and predicted query output values for output 2 from LW-KPLS model with Linear Kernel (1) in Case study 3.

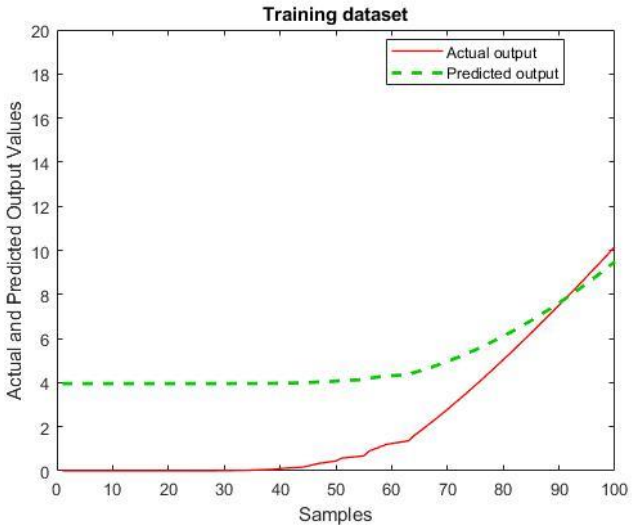

Figure 9. Graph of a training dataset of output variable of actual and predicted output values for output 3 from LW-KPLS model with Linear Kernel (1) in Case study 3.

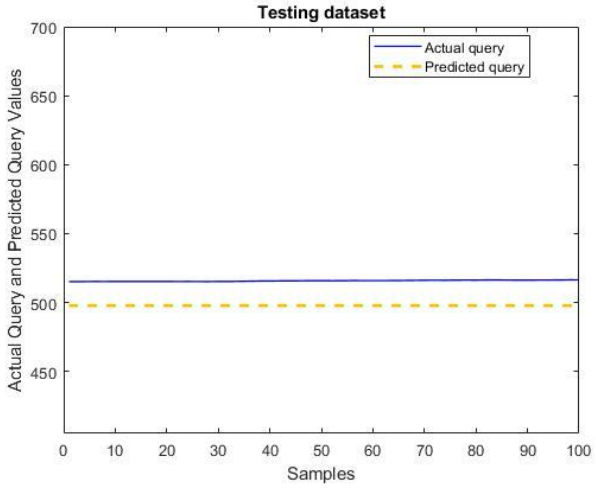

Figure 10. Graph of testing dataset of output variable of actual query and predicted query output values for output 3 from LW-KPLS model with Linear Kernel (1) in Case study 3.

4. Conclusions

In a nutshell, Kernel functions possess the ability to cope with the nonlinearity of the data and to map the data to the different high dimensions of space. In this study, the predictive performance of Kernel functions in the LW-KPLS model has been evaluated via 3 case studies, which are the predictive control of a wastewater treatment process denoted as Case study 1, re-scaled Rosenbrock function designated as Case study 2; thus the process of polymerization of methyl methacrylate denoted as Case study 3. It was found that Linear Kernel (1), Linear Kernel (2), Polynomial Kernel (1), and Polynomial Kernel (2) have provided the best results among Kernel functions in Case studies 1 and 3 while Polynomial Kernel (1) is the best model in Case study 2. Since Polynomial Kernel (1) performed well in these nonlinear case studies, it can conclude that it is the more suitable Kernel function for the LW-KPLS model. From the results in these case studies, it was found that Polynomial Kernel (1) gave 34.60% to 95.39% lower for RMSEs values and 68.20% to 95.49% smaller for E_a values, especially the comparison made with the MK, which gave the highest values of RMSEs and Ea values.

Table 4. $RMSE_1$, $RMSE_2$ and E_a using LW-KPLS model with different Kernel functions for Case Study 1.

Kernel	LNK (1)	LNK (2)	PK(1)	PK(2)	GK	PK	HTK	LK	CK	IMK	LPK	MK
b	**3**	3	1	1	1	7	7	7	10	1	2	2
$RMSE_1$	**0.6703**	**0.6703**	**0.6703**	**0.6703**	1.2412	1.5566	1.3468	1.3468	1.3571	1.3057	1.1201	1.1922
CPU_1 (s)	251.22	262.81	256.49	259.12	251.67	286.39	258.20	291.57	263.51	263.20	253.62	268.36
$RMSE_2$	**0.6728**	**0.6728**	**0.6728**	**0.6728**	1.1214	2.2697	2.4264	2.4264	2.4432	2.4672	2.4910	2.5505
CPU_2 (s)	30.41	32.42	32.44	28.62	27.78	68.95	32.66	67.39	47.18	46.90	40.20	48.88
E_a	**0.6731**	**0.6731**	**0.6731**	**0.6731**	1.3489	2.3410	2.5344	2.5344	2.5518	2.5833	2.6281	2.6863

Legends: LNK(1) – Linear Kernel (1); LNK(2) – Linear Kernel (2); PK(1) – Polynomial Kernel (1); PK(2) - Polynomial Kernel (2); GK - Gaussian Kernel; PK - Power Kernel; HTK - Hyperbolic Tangent (Sigmoid) Kernel; LK – Log Kernel; CK - Cauchy Kernel; IMK - Inverse Multiquadric Kernel; LPK – Laplacian Kernel; MK - Multiquadric Kernel;.

Table 5. $RMSE_1$, $RMSE_2$ and E_a using LW-KPLS model with different Kernel functions for Case Study 2.

Kernel	PK(1)	PK(2)	GK	EK	LNK (1)	LNK (2)	HTK	CK	IMK	PK	LK	MK
b	**4**	10	10	1	3	3	1	1	10	3	9	1
$RMSE_1$	**55.6670**	57.3369	61.3635	91.3526	124.9340	124.9340	131.2939	162.5037	157.9390	147.9563	140.5478	85.1186
CPU_1 (s)	263.23	262.04	255.24	258.67	258.66	257.97	252.84	263.50	271.94	264.69	286.57	273.10
$RMSE_2$	**59.5754**	61.4174	61.9157	92.7694	126.9206	126.9206	131.9680	177.4952	179.9464	189.9410	193.9760	224.8771
CPU_2 (s)	34.39	27.68	28.89	28.12	32.76	26.07	28.99	45.92	56.50	52.53	69.59	50.36
E_a	**59.9662**	61.8255	61.9710	92.9111	127.1193	127.1193	132.0354	178.9943	182.1471	194.1395	199.3189	238.8530

Legends: LNK(1) – Linear Kernel (1); LNK(2) – Linear Kernel (2); PK(1) – Polynomial Kernel (1); PK(2) - Polynomial Kernel (2); GK - Gaussian Kernel; PK - Power Kernel; HTK - Hyperbolic Tangent (Sigmoid) Kernel; LK – Log Kernel; CK - Cauchy Kernel; IMK - Inverse Multiquadric Kernel; MK - Multiquadric Kernel;

Table 6. $RMSE_1$, $RMSE_2$ and E_a using LW-KPLS model with different Kernel functions for Case Study 3

Kernel	LNK (1)	LNK (2)	PK(1)	PK(2)	GK	HTK	CK	IMK	LK	PK	MK
b	3	10	1	1	1	1	1	1	9	10	1
$RMSE_{1,1}$	$\mathbf{1.0117\times10^{-5}}$	$\mathbf{1.0117\times10^{-5}}$	$\mathbf{1.0117\times10^{-5}}$	$\mathbf{1.0117\times10^{-5}}$	2.3970×10^{-5}	2.2783×10^{-5}	4.8152×10^{-5}	4.6910×10^{-5}	4.9723×10^{-5}	4.7201×10^{-5}	3.2006×10^{-5}
$RMSE_{1,2}$	$\mathbf{5.9672\times10^{-3}}$	$\mathbf{5.9672\times10^{-3}}$	$\mathbf{5.9672\times10^{-3}}$	$\mathbf{5.9672\times10^{-3}}$	0.0168	0.0161	0.0303	0.0291	0.0303	0.0286	0.0191
$RMSE_{1,3}$	**6.2180**	**6.2180**	**6.2180**	**6.2180**	26.9303	27.0225	37.9383	35.2956	34.4207	30.3804	20.0034
CPU_1 (s)	259.84	261.38	253.95	251.74	265.81	251.81	311.74	315.63	301.61	340.23	267.44
$RMSE_{2,1}$	$\mathbf{3.2955\times10^{-5}}$	$\mathbf{3.2955\times10^{-5}}$	$\mathbf{3.2955\times10^{-5}}$	$\mathbf{3.2955\times10^{-5}}$	4.8407×10^{-5}	4.6463×10^{-4}	4.6578×10^{-4}	4.6979×10^{-4}	4.7655×10^{-4}	4.7754×10^{-4}	4.7745×10^{-4}
$RMSE_{2,2}$	**0.0197**	**0.0197**	**0.0197**	**0.0197**	0.0289	0.3237	0.3249	0.3291	0.3371	0.3383	0.3383
$RMSE_{2,3}$	**21.3119**	**21.3119**	**21.3119**	**21.3119**	30.8969	428.1452	430.9400	439.7454	458.7096	462.4150	462.2747
CPU_2 (s)	31.00	29.28	29.33	25.15	32.00	42.98	93.18	94.46	75.38	117.59	47.11
$E_{a,1}$	$\mathbf{3.5239\times10^{-5}}$	$\mathbf{3.5239\times10^{-5}}$	$\mathbf{3.5239\times10^{-5}}$	$\mathbf{3.5239\times10^{-5}}$	5.0851×10^{-5}	5.0881×10^{-4}	5.0755×10^{-4}	5.1208×10^{-4}	5.1923×10^{-4}	5.2057×10^{-4}	5.2200×10^{-4}
$E_{a,2}$	**0.0211**	**0.0211**	**0.0211**	**0.0211**	0.0301	0.3544	0.3544	0.3591	0.3678	0.3693	0.3702
$E_{a,3}$	**22.8213**	**22.8213**	**22.8213**	**22.8213**	31.2935	468.2575	470.2402	480.1904	501.1385	505.6184	506.5018

Legends: LNK(1) – Linear Kernel (1); LNK(2) – Linear Kernel (2); PK(1) – Polynomial Kernel (1); PK(2) - Polynomial Kernel (2); GK - Gaussian Kernel; PK - Power Kernel; HTK - Hyperbolic Tangent (Sigmoid) Kernel; LK – Log Kernel; CK - Cauchy Kernel; IMK - Inverse Multiquadric Kernel; MK - Multiquadric Kernel;

Therefore, it can also be concluded that the MK is not the right Kernel function for the LW-KPLS model. Future studies can further improve the LW-KPLS model to cope with missing data.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Acknowledgments

To the best of my knowledge and belief, this manuscript contains no material previously published by any other person except where due acknowledgment has been made. The authors would like to thank Curtin University Malaysia for providing financial support for this project through the final year funding scheme.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Jiang, Y.; Yin, S.; Dong, J.; Kaynak, O. A review on soft sensors for monitoring, control and optimization of industrial processes. *in IEEE Sensors Journal* **2020**, *21*, 12868-12881, https://doi.org/10.1109/JSEN.2020.3033153.
2. Ito, E.H.; Secchi, A.R.; Gomes, M.V.; Paiva, C.R. Development of a gas composition soft sensor for distillation columns: A simplified model based and robust approach. In: *Computer Aided Chemical Engineering*. Elsevier: Volume 44, **2018**; pp. 661-666; https://doi.org/10.1016/B978-0-444-64241-7.50105-1.
3. Ahmad, T.; Zhang, D. Using the internet of things in smart energy systems and networks. *Sustainable Cities and Society* **2021**, *68*, https://doi.org/10.1016/j.scs.2021.102783
4. Sun, K.; Liu, J.; Kang, J.-L.; Jang, S.-S.; Wong, D.S.-H.; Chen, D.-S. Soft Sensor Development with Nonlinear Variable Selection Using Nonnegative Garrote and Artificial Neural Network. In: *Computer Aided Chemical Engineering*; Elsevier: Volume 33, **2014**; pp. 883-888. https://doi.org/10.1016/B978-0-444-63456-6.50148-4.
5. Paulsson, D.; Gustavsson, R.; Mandenius, C.-F. A soft sensor for bioprocess control based on sequential filtering of metabolic heat signals. *Sensors* **2014**, *14*, 17864-17882, https://doi.org/10.3390/s141017864.
6. Nkulikiyinka, P.; Yan, Y.; Güleç, F.; Manovic, V.; Clough, P. Prediction of sorption enhanced steam methane reforming products from machine learning based soft-sensor models. *Energy and AI* **2020**, *2*, https://doi.org/10.1016/j.egyai.2020.100037.
7. Maddikunta, P.K.R.; Pham, Q.-V.; Prabadevi, B.; Deepa, N.; Dev, K.; Gadekallu, T.R.; Ruby, R.; Liyanage, M. Industry 5.0: a survey on enabling technologies and potential applications. *Journal of Industrial Information Integration* **2021**, https://doi.org/10.1016/j.jii.2021.100257.
8. Tseng, M.-L.; Tran, T.P.T.; Ha, H.M.; Bui, T.-D.; Lim, M.K. Sustainable industrial and operation engineering trends and challenges Toward Industry 4.0: A data driven analysis. *Journal of Industrial Production Engineering* **2021**, *38*, 581-598, https://doi.org/10.1080/21681015.2021.1950227.
9. Noga, R.; de Prada, C.; Ohtsuka, T.; Blanco, E.; Casas, J. Non-linear moving horizon state estimation and control for the superfluid helium cryogenic circuit at the large Hadron Collider. In: *Proceedings of the 53rd IEEE Conference on Decision and Control*. **2014**; pp. 3530-3535, https://doi.org/10.1109/CDC.2014.7039937.
10. Zhou, Y.; Xu, K.; He, F.; He, D. Nonlinear fault detection for batch processes via improved chordal kernel tensor locality preserving projections. *Control Engineering Practice* **2020**, *101*, https://doi.org/10.1016/j.conengprac.2020.104514.
11. Liu, Y.; Xie, M. Rebooting data-driven soft-sensors in process industries: A review of kernel methods. *Journal of Process Control* **2020**, *89*, 58-73, https://doi.org/10.1016/j.jprocont.2020.03.012.
12. Chen, K.; Ji, J.; Wang, H.; Liu, Y.; Song, Z.; Adaptive local kernel-based learning for soft sensor modeling of nonlinear processes. *Chemical Engineering Research and Design* **2011**, *89*, 2117-2124, https://doi.org/10.1016/j.cherd.2011.01.032.

13. Yang, H.; Csukás, B.; Varga, M.; Kucska, B.; Szabó, T.; Li, D. A quick condition adaptive soft sensor model with dual scale structure for dissolved oxygen simulation of recirculation aquaculture system. *Computers and Electronics in Agriculture* **2019**, *162*, 807-824, https://doi.org/10.1016/j.compag.2019.05.025.
14. Qin, P.; Zhao, L.; Liu, Z. State of health prediction for lithium-ion battery using a gradient boosting-based data-driven method. *Journal of Energy Storage* **2021**, https://doi.org/10.1016/j.est.2021.103644.
15. Shah, D.; Wang, J.; He, Q.P. A feature-based soft sensor for spectroscopic data analysis. *Journal of Process Control* **2019**, *78*, 98-107, https://doi.org/10.1016/j.jprocont.2019.03.016.
16. Bidar, B.; Sadeghi, J.; Shahraki, F.; Khalilipour, M.M. Data-driven soft sensor approach for online quality prediction using state dependent parameter models. *Chemometrics and Intelligent Laboratory Systems* **2017**, *162*, 130-141, https://doi.org/10.1016/j.chemolab.2017.01.004.
17. Ghosh, D.; Moreira, J.; Mhaskar, P. Application of data-driven modeling approaches to industrial hydroprocessing units. *Chemical Engineering Research and Design* **2021**, *177*, 123-135, https://doi.org/10.1016/j.cherd.2021.10.023.
18. Huang, G.; Ruan, X.; Chen, X.; Lin, D.; Liu, W. A segmented PLS method based on genetic algorithm. *Analytical Methods* **2014**, *6*, 2900-2908, https://doi.org/10.1039/C3AY41765D.
19. Wu, Z.; Zhu, M.; Kang, Y.; Leung, E.L.-H.; Lei, T.; Shen, C.; Jiang, D.; Wang, Z.; Cao, D.; Hou, T. Do we need different machine learning algorithms for QSAR modeling? A comprehensive assessment of 16 machine learning algorithms on 14 QSAR data sets. *Briefings in bioinformatics* **2021**, *22*, https://doi.org/10.1093/bib/bbaa321.
20. Yeo, W.S. Prediction of Yellowness Index Using Partial Least Square Regression Model. In: Proceedings of the 2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST). **2021**; pp. 1-5. https://doi.org/10.1109/GECOST52368.2021.9538723.
21. Lin, B.; Recke, B.; Knudsen, J.K.; Jørgensen, S.B. A systematic approach for soft sensor development. *Computers & Chemical Engineering* **2007**, *31*, 419-425, https://doi.org/10.1016/j.compchemeng.2006.05.030.
22. Debik, J.; Sangermani, M.; Wang, F.; Madssen, T.S.; Giskeødegård, G.F. Multivariate analysis of NMR-based metabolomic data. *NMR in Biomedicine* **2021**, https://doi.org/10.1002/nbm.4638.
23. Raposo, F.; Barceló, D. Assessment of goodness-of-fit for the main analytical calibration models: Guidelines and case studies. *Trac Trends in Analytical Chemistry* **2021**, *143*, https://doi.org/10.1016/j.trac.2021.116373.
24. Uchimaru, T.; Kano, M. Sparse sample regression based just-in-time modeling (SSR-JIT): Beyond locally weighted approach. *IFAC-PapersOnLine* **2016**, *49*, 502-507, https://doi.org/10.1016/j.ifacol.2016.07.392.
25. Yeo, W.S.; Saptoro, A.; Kumar, P. Development of adaptive soft sensor using locally weighted Kernel partial least square model. *Chemical Product and Process Modeling* **2017**, *12*, 1-13, https://doi.org/10.1515/cppm-2017-0022.
26. Yeo, W.S.; Lau, W.J. Predicting the whiteness index of cotton fabric with a least squares model. *Cellulose* **2021**, *28*, 8841–8854, https://doi.org/10.1007/s10570-021-04096-y.
27. Yeo, W.S.; Chan, M.K.; Bukhari, N.A. Prediction of Glucose Concentration Hydrolysed from Oil Palm Trunks Using a PLSR-Based Model. In: *Advances in Intelligent Systems and Computing*. Cham **2021**; pp. 927-937, https://doi.org/10.1007/978-3-030-93247-3_88.
28. Thien, T.F.; Yeo, W.S. A comparative study between PCR, PLSR, and LW-PLS on the predictive performance at different data splitting ratios. *Chemical Engineering Communications* **2021**, 1-18, https://doi.org/10.1080/00986445.2021.1957853.
29. Yeo, W.S.; Saptoro, A.; Kumar, P. Adaptive soft sensor development for non-Gaussian and nonlinear processes. *Industrial Engineering Chemistry Research* **2019**, *58*, 20680-20691, https://doi.org/10.1021/acs.iecr.9b03821.
30. Yeo, W.S.; Saptoro, A.; Kumar, P. Missing data treatment for locally weighted partial least square-based modelling: A comparative study. *Asia-Pacific Journal of Chemical Engineering* **2020**, *15*, https://doi.org/10.1002/apj.2422.
31. Harkat, M.-F.; Kouadri, A.; Fezai, R.; Mansouri, M.; Nounou, H.; Nounou, M.; Systems, E. Machine learning-based reduced kernel PCA model for nonlinear chemical process monitoring. *Journal of Control, Automation and Electrical Systems* **2020**, *31*, 1196-1209, https://doi.org/10.1007/s40313-020-00604-w.
32. Caraman, S.; Sbarciog, M.; Barbu, M. Predictive control of a wastewater treatment process. *IFAC Proceedings Volumes* **2006**, *39*, 155-160, https://doi.org/10.3182/20060830-2-SF-4903.00028.
33. Turgut, O.E.; Turgut, M.S.; Coban, M.T. Design and economic investigation of shell and tube heat exchangers using Improved Intelligent Tuned Harmony Search algorithm. *Ain Shams Engineering Journal* **2014**, *5*, 1215-1231, https://doi.org/10.1016/j.asej.2014.05.007.
34. Shafiee, G.; Arefi, M.; Jahed-Motlagh, M.; Jalali, A. Nonlinear predictive control of a polymerization reactor based on piecewise linear Wiener model. *Chemical Engineering Journal* **2008**, *143*, 282-292, https://doi.org/10.1016/j.cej.2008.05.013.
35. Picheny, V.; Wagner, T.; Ginsbourger, D. A benchmark of kriging-based infill criteria for noisy optimization. *Structural and Multidisciplinary Optimization* **2013**, *48*, 607-626, https://doi.org/10.1007/s00158-013-0919-4.

36. Tsamardinos, I.; Rakhshani, A.; Lagani, V. Performance-estimation properties of cross-validation-based protocols with simultaneous hyper-parameter optimization. *International Journal on Artificial Intelligence Tools* **2015**, *24*.
37. Howley, T.; Madden, M.G. The genetic kernel support vector machine: Description and evaluation. *Artificial Intelligence Review* **2005**, *24*, 379-395, https://doi.org/10.1007/s10462-005-9009-3.
38. Rashid, N.A.; Rosely, N.A.M.; Noor, M.A.M.; Shamsuddin, A.; Hamid, M.K.A.; Ibrahim, K.A. Forecasting of refined palm oil quality using principal component regression. *Energy Procedia* **2017**, *142*, 2977-2982, https://doi.org/10.1016/j.egypro.2017.12.364.
39. Hardy, R.L. Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968–1988. *Computers Mathematics with Applications* **1990**, *19*, 163-208, https://doi.org/10.1016/0898-1221(90)90272-L.
40. Mesquita, D.P.; Gomes, J.P.; Corona, F.; Junior, A.H.S.; Nobre, J.S. Gaussian kernels for incomplete data. *Applied Soft Computing* **2019**, *77*, 356-365, https://doi.org/10.1016/j.asoc.2019.01.022.
41. Zhang, X.; Song, Q. A multi-label learning based kernel automatic recommendation method for support vector machine. *PloS one* **2015**, *10*, https://doi.org/10.1371/journal.pone.0120455.
42. Zhang, X.; Kano, M.; Li, Y. Locally weighted kernel partial least squares regression based on sparse nonlinear features for virtual sensing of nonlinear time-varying processes. *Computers & Chemical Engineering* **2017**, *104*, 164-171, https://doi.org/10.1016/j.compchemeng.2017.04.014.
43. Drewnik, M.; Pasternak-Winiarski, Z. SVM kernel configuration and optimization for the handwritten digit recognition. In: Proceedings of the IFIP International Conference on Computer Information Systems and Industrial Management. **2017**; pp. 87-98, http://dx.doi.org/10.1007/978-3-319-59105-6_8.