Department of Chemical Engineering

Adaptive Soft Sensors for Non-Gaussian Chemical Process Plant Data Based on Locally Weighted Partial Least Square

Yeo Wan Sieng

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

July 2019

Declaration

"To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made. This thesis also contains no material which has been accepted for the award of any other degree or diploma in any university."

Signature:

Name **:** Yeo Wan Sieng

Date **:** 3rd July 2019

To my beloved parents, brother, sisters and friends

Acknowledgement

First, I would like to express my sincere gratitude to my main supervisor, A.Prof. Dr. Agus Saptoro. His counsel and support were helpful and much appreciated. I want to thank him for the countless hours he spent helping me during the research works and in the writing of this thesis. His guidance helped to keep me on track in completing my research.

I would also like to express my sincere appreciation to my co-supervisor, A.Prof. Dr. Perumal Kumar. He provided an abundance of valuable feedback on the preparation of this thesis. Without his contributions, this research would not have progressed as smoothly.

My sincere thanks also go to Dr. Hendra Gunawan Harno, Dr. Henry Foo Chee Yew and Dr. Raymond Chiong Choo Wee, my colleagues at Curtin University, Malaysia. During my research, they provided kind assistance in giving comments and suggestions.

Lastly, I would like to thank the people who have contributed directly and indirectly to the success of this research, they being my family members, other colleagues, and friends who were always there to support and believe in me.

Abstract

Since data in most industrial processes are non-Gaussian distributed and contains missing measurements, the existing algorithms for adaptive soft sensors that ignore these scenarios work poorly and limited research has been dedicated to address the above issues. Thus, in this study, a newly proposed algorithm, namely expectation maximization ensemble locally weighted independent component Kernel partial least square (EM-E-LW-IC-KPLS) was formulated. It is introduced through modification on locally weighted partial least square (LW-PLS) to include the ensemble method, the Kernel function, the independent component analysis and the expectation maximisation algorithms. It is hypothesised EM-E-LW-IC-KPLS can improve the capability of adaptive soft sensors to work well with non-Gaussian distributed data, nonlinear data, and incomplete measurements. The newly developed algorithms were tested using process data generated from six simulated plants where MATLAB software was utilised as a platform for modelling and simulation studies. From two case studies, it was found that the presence of an unsteady state in training data of the output variable cause higher prediction errors for all algorithms than the test data where training data usually has lower prediction errors than the test data. For nonlinear and Gaussian distributed process data, the ensemble locally weighted Kernel partial least square (E-LW-KPLS) performed better than the LW-PLS in which it's the error of approximation, E_a values improved approximately 7% to 54%. On the other hand, the ensemble locally weighted independent component Kernel partia least square (E-LW-IC-KPLS) also provided better prediction performances compared to the LW-PLS, and the E_a values were lower by roughly 8% to 94%. Based on case studies, the performance of the EM-E-LW-IC-KPLS in dealing with 5% to 60% of missing data in nonlinear and non-Gaussian process data was investigated. It was found the EM-E-LW-IC-KPLS model performs better than the integration of LW-PLS models with the missing data imputation methods including EM model, trimmed score regression and singular vector decomposition in handling up to 20% of missing data. Moreover, the inclusive of ensemble method in the newly proposed E-LW-KPLS, E-LW-IC-KPLS, and EM-E-LW-IC-KPLS algorithms had significantly minimised the penalty which is the computational burden. While, these algorithms showed more accurate predictive performance as compared to the existing LW-PLS model in dealing with nonlinear, non-Gaussian distributed and missing data. As a conclusion, the proposed EM-E-LW-IC-KPLS algorithm has successfully narrowed down the current research gap. The incorporation of other local models in locally weighted algorithms should be further investigated.

Publications arising from this thesis

Journal Papers

1. Yeo, Wan Sieng, Agus Saptoro and Perumal Kumar. 2017. "Development of Adaptive Soft Sensor Using Locally Weighted Kernel Partial Least Square Model." *Chemical Product and Process Modeling*. 12(4), doi: 10.1515/cppm-2017-0022.
2. Vivianna Maria Mickel, Wan Sieng Yeo, and Agus Saptoro. 2019. "Evaluating the performance of newly integrated model in nonlinear chemical process against missing measurements." *Chemical Product and Process Modeling*. 14(4), doi: 10.1515/cppm-2018-0066.
3. Yeo, Wan Sieng, Agus Saptoro and Perumal Kumar. 2019. "Adaptive soft sensor development for non-Gaussian and nonlinear processes." *Industrial and Engineering Chemistry Research*, doi: 10.1021/acs.iecr.9b03821.
4. Yeo, Wan Sieng, Agus Saptoro and Perumal Kumar. 2019. "Misssing data treatment for locally weighted partial least square based modelling." *Asia Pacific Journal of Chemical Engineering.* (Under review)
5. Wan Sieng Yeo, Agus Saptoro and Perumal Kumar. 2019. "An overview on just-in-time based soft sensors for processes industry." *Journal of Process Control.* (In preparation)

Conference Papers

1. Wan Sieng Yeo, Agus Saptoro and Perumal Kumar. 2016. "Development of Adaptive Soft Sensor using Locally Weighted Kernel Partial Least Square Model." *29th Symposium of Malaysian Chemical Engineers, Miri, Sarawak, Malaysia, 1st – 3rd December 2016.*
2. Yeo, Wan Sieng, Agus Saptoro and Perumal Kumar. 2017. "Adaptive Soft Sensor based on Modified Locally Weighted Partial Least Square Algorithm for non-

Gaussian and Nonlinear Processes." *Proceeding of One Curtin International Postgraduate Conference 2017, Miri, Sarawak, Malaysia, 10th – 12th December 2017.*

3. Wan Sieng Yeo, Agus Saptoro, and Perumal Kumar. 2018. "PLS based non-Adaptive and LW-PLS based Adaptive soft sensor development with missing data." *Proceeding of one Curtin international postgraduate conference (OCPC) 2018, Miri, Sarawak, Malaysia, 26th – 28th November 2018.*

4. Vivianna Maria Mickel, Wan Sieng Yeo, and Agus Saptoro. 2018. "The development of trimmed scores regression locally weighted Kernel partial least squares for nonlinear chemical process data with missing measurements." *Proceeding of one Curtin international postgraduate conference (OCPC) 2018, Miri, Sarawak, Malaysia, 26th – 28th November 2018.*

5. Wan Sieng Yeo, Agus Saptoro, and Perumal Kumar. 2019. "Non Gaussian Locally Weighted Kernel PLS based Adaptive soft sensor." *7th International Conference on Smart Computing and Communication 2019, Miri, Sarawak, Malaysia, 28th - 30th June 2019.*

Table of Contents

List of Tables

List of Figures

Nomenclatures

Abbreviations

ACS	Adjusted local cosine similarity
ALD	Approximate linearity dependence
ANN	Artificial neural network
CoJIT	Correlation-based just-in-time
CPU	Central processing unit
CSTR	Continuous stirred tank reactor
DLOER	Dual learning-based online ensemble regression
D-LWPCR	Double locally weighted principal component regression
D-LWKPCR	Double locally weighted Kernel principal component regression
EJITGPR	Ensemble just-in-time Gaussian process regression
E-JITL	Ensemble just-in-time learning
ELM	Extreme leaning machine
E-LW-IC-KPLS	Ensemble locally weighted independent component Kernel partial least square
E-LW-KPLS	Ensemble locally weighted Kernel partial least square
EM	Expectation maximization

EM-E-LW-IC-KPLS	Expectation maximization ensemble locally weighted independent component Kernel partial least square
FLOO	Fast leave-one-out
FLOO-CV	Fast leave-one-out cross validation
GB	Gigabyte
GMM	Gaussian mixture model
GPR	Gaussian process regression
Grey	Grey-box models
ICA	Independent component analysis
ICR	Independent component regression
JIT	Just-in-time
JIT-ELM	Just-in-time extreme learning machine
JITL	Just-in-time learning
JITL-bagging-PLS	Just-in-time learning bagging partial least square
JITL-MWGPR	Just-in-time learning moving window Gaussian process regression
JITL-RLSSVR	Just-in-time learning based recursive least square support vector regression
JSELM	Just-in-time semi-supervised extreme learning machine
KPLS	Kernel partial least square

LFA	Latent factor analysis
LSSVR	Least square support vector regression
LW	Locally weighted
LW-IC-KPLS	Locally weighted independent component Kernel partial least square
LWL	Locally weighted learning
LW-KPCR	Locally weighted Kernel principal component regression
LW-KPLS	Locally weighted Kernel partial least square
LW-PCR	Locally weighted principal component regression
LW-PLS	Locally weighted partial least square
LWR	Locally weighted regression
LWSLFA	Locally weighted supervised latent factor analysis
LWSSLFA	Locally weighted semi-supervised latent factor analysis
MACS	Modified adjusted cosine similarity
MI	Mutual information
MLR	Multiple linear regression
MPF	M-phase promoting factor
MW	Moving window
MWGPR	Moving window Gaussian process regression

MWJIT-LS-SVM	Moving window just-in-time learning least square support vector machine regression
MW-JITL-TD	Moving window locally weighted partial least square under time difference
NC	Nearest correlation
NGR	Non-Gaussian regression
Non-JIT	Non-just-in-time
NNPLS	Neural network partial least square
NPLS	Nonlinear partial least square
OS	Operating system
O.L.	Other linear regressions
OLLASS	Online local learning based adaptive soft sensor
OLPLS	Local partial least square based approach for online soft sensing
OSVR	Online support vector regression
PC	Principal Component
PCA	Principal component analysis
PCR	Principal component regression
Phys	Physical models
PLS	Partial least square

QPLS	Quadratic partial least square
RAM	Random access memory
RMSE	Root mean squared error
SC	Spectral clustering
SPLS	Spline partial least square
SSD	Solid-state drive
SSR-JIT	Sparse sample regression just-in-time
SVD	Singular value decomposition
SVDD	Support vector data description
SVD-E-LW-IC-KPLS	Singular value decomposition ensemble locally weighted independent component Kernel partial least square
SVD-LW-PLS	Singular value decomposition locally weighted partial least square
SVR	Support vector regression
TD	Time difference
TSR	Trimmed score regression
TSR-E-LW-IC-KPLS	Trimmed scores regression ensemble locally weighted independent component Kernel partial least square
TSR-LW-PLS	Trimmed scores regression locally weighted partial least square

WLDS	Weighted linear dynamic system
WPPCA	Locally weighted probabilistic principal component analysis

Symbols

A	Unknown mixing matrix in independent component analysis (-)
$A_1,\ A_2$	The reactant species (-)
ACS	Adjusted local cosine similarity index (-)
b	Kernel parameter (-)
b_k	The regression coefficient in Kernel partial least square (-)
B	A scaled version of projection direction (-)
B_c	Orthogonal matrix in independent component analysis (-)
$B_1,\ B_2$	The desired product (-)
c	Number of independent components (-)
C	Factor loading matrices of input data set (-)
C_2	Unwanted product (-)
CA	Concentration of the reactant species ($\frac{kmol}{m^3}$)
C_{A0}	Concentration of A in inlet stream ($\frac{mol}{m^3}$)

CB	Concentration of the desired product ($\frac{kmol}{m^3}$)
C_j	Features for just-in-time learning extreme learning machine (-)
C_{k_t}	The k_t-th Gaussian component (-)
CP	Product concentration ($\frac{mol}{l}$)
$C_{s,v}$	Double-weighted covariance matrix (-)
d	A d-dimensional variable (-)
D_2	By-product (-)
DO	Dissolved oxygen concentration ($\frac{mg}{l}$)
d_n	Distance between x_q and x_n (-)
d_t	Number of column in matrix X_t (-)
e_0	The predicted error vector for the newest labelled sample (-)
E_a	The error of approximation (-)
E_{N_F,λ_F}^{FLOO}	The fast leave-one-out (FLOO) based error index (-)
e_i	The predicted error vector for the i-th nearest sample around the query sample (-)
F	Feature space (-)

F_i	Inlet flow rate of CSTR ($\frac{m^3}{hr}$)
f_m, $f_m(\bullet)$	The localized online support vector regression models (-)
H	The output matrix of hidden-layer vector (h_i)
J	Similarity factor (-)
J^*	Anti-over-fitting criterion (-)
J_F	A matrix consists of a unit matrix for the graph Laplacian (-)
K, k	Number of latent variables (-)
k_1, k_2, k_3	The reaction rates (hr^{-1})
$K_i, \tilde{K}$	Kernel matrix for training data (-)
K_t, k_t	Number of Gaussian component in Gaussian mixture model (-)
K_{test}, $\tilde{K}_{test}$	Kernel matrix for test data (-)
$kurt(s)$	Kurtosis (-)
L	Number of output variable (-)
L_1	Level in non-isothermal CSTR (m)
l, $l_{\min}$, $l_{\max}$	The related samples which are chosen before constructing a just-in-time learning model (-)
LD_m	M local domains (-)

L_F	The number of training sample (-)
m	The number of iteration in Expectation maximization (-)
M	Number of input or observed variable (-)
$MACS$	Modified adjusted cosine similarity (-)
MAE	Mean absolute error (-)
MAE_1	Mean absolute error for training data (-)
MAE_2	Mean absolute error for test data (-)
m_{ij}	Element in matrix M_s (-)
$\overline{m_{ij}}$	Element in matrix $\overline{M_s}$ (-)
MSE	Mean square error (-)
MSE_1	Mean square error for training data (-)
MSE_2	Mean square error for test data (-)
M_s	The missing data indicator matrix (-)
$\overline{M_s}$	The complement of M_s (-)
m_t	Mean value of the observed data (-)
M_t	A vector that consists of m_t (-)
n	Number of row in matrix $\phi(x)$ (-)

N	The total number of samples (-)
N_1	The number of training data set (-)
N_2	The number of testing data set (-)
N_F	The number of hidden nodes (-)
N_G	The total number of data in matrix X_t (-)
n_m	Number of local modelling sampling (-)
P	Factor loading matrices of output data set (-)
$p(C_j \mid x_{new})$	Posterior probability for just-in-time learning extreme learning machine (-)
$p(LD_m \mid x_{new})$	The diverse local models (-)
$P(GPR_l \mid x_q)$	The posterior probability of new test sample, x_{new} (-)
p_k	The kth loading vector of X_k (-)
P_m	The loading matrix for Gaussian mixture model (-)
Q	Whitening matrix (-)
q_i	The loading vector of Y_i (-)
q_k	The kth regression coefficient vector (-)
$\dot{Q}_k$	Heat flow ($\frac{J}{hr}$)

q_m	Loading vector for Gaussian mixture model (-)
$Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)$	Posterior probability of the j-th sample data with the k-th non-Gaussian component at the m-th iteration (-)
Q_s	Q statistic (-)
$RMSE$	Root mean square error (-)
$RMSE_1$	Root mean square errors of the training data set (-)
$RMSE_2$	Root mean square errors of the testing data set (-)
s	Column vector for independent components (-)
s_c	Unknown independent components (-)
s_i	The weight of the respective sample (-)
s_l	Presented cumulative similarity factor (-)
s_{qi}	The similarity factor between the query sample and the sample in the data set (-)
S_s	Substract concentration ($\frac{mg}{l}$)
T$_1$	Temperature of the single chemical reactor (K)
T$_2$	Temperature of highly nonlinear CSTR (K)
T	The score matrix of $\phi(x_i)$ (-)

T^2	Hotelling's T squared statistic (-)
t_1	Central processing unit time for training data (sec)
t_2	Central processing unit time for testing data (sec)
t_i	The score vector of $\phi(x_i)$ (-)
t_k	The kth latent variable of X_k (-)
$t_{q,k}$	The kth latent variable of X_q (-)
u	Dimensionless concentration of active MPF (-)
U	The score matrix of Y_i (-)
u_i	The score vector of Y_i (-)
U_1, R_1	The undesired by-products (-)
v	Dimensionless concentration of total cyclin (-)
V	Kernel matrix for training data (-)
$\dot{V}$	Flow rate ($\frac{m^3}{hr}$)
V_q	Kernel matrix for query/ testing data (-)
$\frac{V}{V_R}$	Scaled volumetric inlet flow (-)
W	The demixing matrix in independent component analysis (-)

w_k	The eigenvector of $X_k^T \Omega Y_k Y_k^T \Omega X_k$ (-)
W_m	The weighting matrix for Gaussian mixture model (-)
W_s	Sample weighted data matrix (-)
W_v	Variable weighted data matrix (-)
X , x, x_n	Input/ observed variable/ nth historical input data (-)
$\overline{x}$	Mean values of the input variables (-)
x_1 , x_2	The input variables with a range from -25 to 25 (-)
X_B	Biomass concentration ($\frac{mg}{l}$)
x_i, X_i	The i-th test vector in Kernel partial least square (-)
x_{nM}	The missing data element in the matrix (-)
x_j, X_j	The j-th training vector in kernel partial least square (-)
X_k	The kth historical input data (-)
X_m	A matrix with missing data (-)
$\bar{X}_m$	Weighted mean for input data (-)
$x_{m,i}$	The local training input (-)

x_{new}	New test sample (-)
X_n	Historical data (-)
X_q, x_q	Query data (-)
$X_{q,k}$	Query data for *k*-th latent variable (-)
X_r	Recycled biomass concentration ($\frac{mg}{l}$)
$X_{s,v}$	Double-weighted data matrix (-)
X_t	A new reconstructed matrix for Gaussian mixture model (-)
x_{ti}	Element in matrix X_t (-)
y, y_n, Y	Output variable (-)
$y_{GPR_i,new}$	The local prediction mean from the *i*-th local model for Gaussian process regression, $GPR_i,\ i \in 1,2,..M_e$ (-)
Y_i	Output variable in Kernel partial least square (-)
y_i	The training data (-)
Y_k	The nth historical output data (-)
$\bar{Y}_l$	Weighted mean for output data (-)

$y_{m,new}$	The output variable from local model (-)
y_{new}	The final output (-)
$\hat{y}_q$	An output prediction (-)
$\hat{Y}_{test}$	The prediction on test data (-)
$\hat{Y}_{train}$	The prediction on training data (-)
$\bar{y}_w$	Weighted mean of output (-)
$\bar{y}$	Mean values of the output variables (-)
z	The over sphered zero-mean vectors in independent component analysis (-)
$\bar{z}_q$	Mean vector of posterior distribution of query point (-)
Λ	Diagonal matrix of eigenvalues (-)
β	Formulated regression coefficients (-)
β_q	Regression coefficient vector (-)
α, γ	Parameters for regression coefficient vector (-)
$\alpha_{m,i}$, $\alpha^*_{m,i}$, b_m	The online support vector regression model parameters (-)
γ_R	The regularization parameter (-)

ϕ_{test}	The matrix of the mapped test data (-)
$\phi(x_i)$, $\phi(x_j)$	A nonlinear mapping function that projects the input vectors from the original space to F (-)
Ω	A similarity matrix (-)
Ω_t	Parameter in expectation maximization (-)
λ	Setting parameter (-)
λ_F	A balance parameter (-)
ω_n	Similarity index (-)
π_{k_t}	The probabilistic weight of the *k*-th Gaussian component (-)
$\pi_{k_t}^{m+1}$	Prior probability of the k-th non-Gaussian component at the (m+1)-th iteration (-)
μ	Mean vector in Gaussian mixture model (-)
$\mu_{k_t}^{m+1}$	Mean of the k-th non-Gaussian component at the (m+1)-th iteration (-)
Σ	Covariance matrix in Gaussian mixture model (-)
$\sum_x$	Measured noise variance of input data set (-)
$\sum_{k_t}^{m+1}$	Covariance of the k-th non-Gaussian component at the (m+1)-th iteration (-)
φ	A localization parameter in similarity measurement (-)

σ	The width parameter of the Exponential Kernel function (-)
σ_d	Standard deviation of d_n in similarity measurement (-)
θ	Regression coefficient vector (-)
θ_t	Gaussian model parameters (-)
θ_{D}	Diagonal matrix with all positive elements (-)
ϑ_0	Initial temperature (K)

List of Appendices

Chapter 1

Introduction

1.1. Background and motivation

An unstable economy, stiff competition, increasing energy and material costs and stricter environmental regulations have forced process industries to improve their operational efficiency. Due to these challenges, process industries are required to achieve operational excellence in their processing plants to obtain optimal performance in productivity and quality of products. Hence, industries such as steel-making, pharmaceutical, food processing, semiconductor and petrochemical industries are looking for advanced technological tools to predict, monitor and control production processes. Soft sensors that use easy-to-measure variables to predict hard-to-measure variables are a technological tool to improve process monitoring and controls. Soft sensors have been used as alternative sensors when their hardware counterparts are not available or are cost-prohibitive or where variables are difficult to measure (Fortuna *et al.* 2007; Kaneko and Funatsu 2011b; Saptoro 2014).

In other words, soft sensors are used in industrial processing plants to predict important variables such as product quality when online measurements are associated with excessive delays, real-time measurements are not available, expensive and or difficult to procure. Industrial questionnaire surveys on soft sensor applications were conducted in chemical industrial process plants by Kano and Koichi (2013) and Kano and Ogawa (2010). These pharmaceutical, steel-making, semiconductor, refinery, and petrochemical processing industries reported soft sensors had been successfully applied in important chemical engineering unit operations such as distillation, polymerisation and reaction processes. Their survey results found the physical model (Phys), data-driven and hybrid (grey-box) based soft sensors had been used in major unit operations. The data-driven based soft sensors included partial least square (PLS), multiple linear regression (MLR), just-in-time

(JIT), artificial neural network (ANN), and other linear regressions (O.L.). Table 1.1 summarises the survey results of the industrial applications of these soft sensors. Phys or 'first principle models' are built by incorporating physicochemical knowledge of the process. The data-driven models are constructed from recorded historical operational data from the processing plants. Meanwhile, grey-box models are hybrid models combining some of the properties of both Phys and data-driven models.

Table 1.1 Survey on the industrial applications of soft sensor (Kano and Ogawa 2010)

Process	**Methodology**							
	Phys	**MLR**	**PLS**	**O.L.**	**ANN**	**JIT**	**Grey**	**Total**
Distillation	20	256	41	6	0	5	3	331
Reaction	5	32	43	0	0	5	1	86
Polymerisation	0	4	8	0	3	0	5	20
Others	0	1	1	0	0	0	0	2
Total	25	293	93	6	3	10	9	439

It can be seen from Table 1.1 that most industrial soft sensors are developed using data-driven approaches. Moreover, compared to other data-driven approaches, MLR models are more widely used due to their simplicity in model development (Jin *et al.* 2014; Kano and Koichi 2013; Kim *et al.* 2013a; Kim *et al.* 2013b; Kano and Ogawa 2010). Nonetheless, MLR models can be over-fitted and inaccurate when dealing with high dimensional and highly collinear data. Hence, PLS models that can address the limitations of the MLR models are preferable. Based on the existing comparative studies, the PLS models provide more accurate prediction capability than principal component regression

(PCR) and MLR models (Hazama and Kano 2015; Kano and Koichi 2013; Wentzell and Montoto 2003; Cramer III 1993).

Soft sensors have been widely used in industrial applications though there are some practical difficulties. Traditional data-driven soft sensors such as MLR, PLS, PCR, and ANN were constructed using the plants' historical data in offline mode before their application. Hence, the predictive performance of these soft sensors has slowly deteriorated over time. This condition is due to changes in the state of the processes and plant characteristics such as sensor and process drift caused by fouling, catalyst de-activation, changes in raw materials, equipment aging, clogging and wear, etc. (Saptoro 2014; Kano and Koichi 2013; Kadlec *et al.* 2011). According to most industrial engineers, the main problem with soft sensors is the deterioration of their accuracy as a result of changes in process conditions (Kano and Koichi 2013; Kano and Ogawa 2010).

Thus, to cope with process changes and to maintain the soft sensor's effective performance in practice, a soft sensor must be updated often since the process characteristics change over time. This new soft sensor, which can adapt to the current state of the plant, is called the 'adaptive soft sensor.' Earlier adaptive models used to develop the newer adaptive soft sensors can be found in the literature. The common adaptive soft sensors such as correlation based JIT (CoJIT) and locally weighted (LW) PLS (LW-PLS) have been successfully applied in industries (Hazama and Kano 2015; Jin *et al.* 2015a; Kano and Koichi 2013).

Most of the data driven approach for soft sensors are easy to develop and give reasonable predictive capability. Hence, these data driven soft sensors can be incorporated to the controller as part of the adaptive controller. However, a majority of the current adaptive data-driven models, which include CoJIT and LW-PLS assume that process data always follows a Gaussian distribution (Liu and Gao 2015) with no missing measurement (Kadlec *et al.* 2009). However, the majority of current industrial process data are non-Gaussian distributed (Ge *et al.* 2013) and sometimes consist of missing data (Yin *et al.* 2014). Therefore, the current adaptive data-driven models in soft sensor may not function well in these conditions. The development of an improved adaptive algorithm, which is capable of addressing non-Gaussian and incomplete data, has become necessary. Therefore, this research aims to respond to the fundamental needs as stated above where a new and improved adaptive algorithm provides a possible solution to the limitations of the existing algorithms.

1.2. Problem statement

The existing algorithms for developing adaptive soft sensors, which include a popular LW-PLS algorithm, do not take into account the fact that most of the process data are non-Gaussian distributed and contain incomplete measurements. Missing measurements are data that do not have a value in an observation. It is also observed that less effort had been considered regarding non-Gaussian distributed data in developing adaptive soft sensors, especially JIT-based soft sensors. In practice, the industrial process data clearly shows non-Gaussian behaviour (Qin 2012) predominantly in highly non-linear processes. Hence, the current data driven soft sensors including LW-PLS suffer when the data distribution is non-Gaussian since their models are unable to properly extract useful information from non-Gaussian data. Thus, an improved adaptive soft sensor can cope with non-Gaussian distributed data and it is required to provide more accurate prediction capabilities.

On the other hand, the presence of missing measurement in the process data is inevitable. Incomplete data are commonly caused by routine sensor maintenance, sensor failure, and sensors with different sampling rates (Folch-Fortuny *et al.* 2015; Arteaga and Ferrer 2002). When the algorithms for developing adaptive soft sensors are subject to incomplete data conditions, the datasets containing the missing values are usually omitted (Folch-Fortuny *et al.* 2016) so only the complete datasets are utilized. In some cases, the removal of this data is undesirable and impractical since the omitted data sets may contain significant information about the process and ultimately cause data loss when the amount of missing data is large. Furthermore, the negative impact of the incomplete data on regression methods used to develop soft sensors, including principal component analysis and PLS have been demonstrated by Nelson *et al.* (2006). Hence, to address the significant issues of non-Gaussianity and incomplete process data in the development and application of adaptive soft sensors, a new and more robust algorithm that can manage non-Gaussian distributed data and may contain incomplete measurements is needed.

1.3. Research questions

As stated above, certain limitations and other issues faced by the current adaptive algorithms used to develop new adaptive soft sensors have created interest in pushing for additional research in this area. Some concerns and questions have arisen (as shown in the following section) and they will be examined in this research. Subsequently, the aim and scopes of this research are identified.

a. How many simultaneous integration and modification of different data analysis and modelling theories be effectively performed to formulate a robust, non-Gaussian adaptive soft sensor algorithm?

b. Will the proposed algorithm has any limitations in terms of the maximum degree of robustness against missing data?

c. Are there any consequences associated with the proposed algorithm especially in regard to computational efficiency? If there are, what are the approaches to minimise them?

1.4. Aim and objectives

This research aims to introduce a novel improved algorithm to develop an adaptive soft sensor, which can deal with nonlinear and non-Gaussian distributed data as well as missing measurements. To achieve this aim, the following objectives are proposed:

a. To formulate a new algorithm, namely expectation maximization (EM) ensemble LW independent component Kernel partial least square (EM-E-LW-IC-KPLS) algorithm as a robust against missing data, nonlinear and non-Gaussian approach to developing adaptive soft sensors.

b. To investigate the predictive performance of the newly introduced adaptive algorithms using simulated data and then compares the results with the bench-marking algorithm (widely used LW-PLS algorithm).

c. To minimise the computational efficiency of the newly improved adaptive algorithms while evaluating their predictive performance.

1.5. Scopes of research

In this section, the research scope and methodologies employed in this current research work are briefly explained. Firstly, the traditional and commonly used LW-PLS model is adopted as a base case since it is simple and able to cope with outliers, nonlinearity, and collinearity in the process data set. In this study, a new and improved algorithm is formulated and proposed through a modification on LW-PLS to incorporate the ensemble method, the Kernel function, the independent component analysis (ICA) and the expectation maximisation (EM) algorithms.

ICA is integrated into the LW-PLS model to accommodate non-Gaussian data while Kernel function is also incorporated to enhance further the algorithm's ability to cope with nonlinear processes. Moreover, the inclusion of the EM algorithm is established to deal with missing data. The newly developed algorithm is called expectation maximization ensemble locally weighted independent component Kernel partial least square (EM-E-LW-IC-KPLS) algorithm. Hypothetically, this newly developed algorithm will be able to improve the adaptive soft sensors concerning their capability to work well with non-Gaussian distributed data, nonlinear data, and incomplete measurements.

Also, other newly modified Gaussian-based and non-Gaussian-based LW algorithms are also formulated in this study. Gaussian-based LW algorithms are formulated by incorporating Kernel function into the LW-PLS algorithm to obtain the ensemble LW Kernel partial least square (E-LW-KPLS) algorithms. Several Kernel functions are employed in E-LW-KPLS algorithm as an internal model within PLS instead of a linear regression to cope with nonlinear process data. Kernel functions such as polynomial Kernel, inverse multi-quadric Kernel, power Kernel, and log Kernel are used. This newly developed algorithm is expected to be more accurate in dealing with non-linear process data. The predictive performance of this recently developed algorithms are then analysed using the simulated Gaussian distributed data in case studies 1 to 3, and the results are compared to the LW-PLS, and LW-KPLS algorithms.

On the other hand**,** the new Gaussian-based LW algorithms developed are further improved to cope with nonlinear, non-Gaussian distributed and missing data. In this step, ICA is used to generate independent components from non-Gaussian distributed data and incorporated into the improved Gaussian-based LW algorithms. This newly improved non-Gaussian-based LW algorithm is called the 'ensemble LW independent component Kernel partial least square' (E-LW-IC-KPLS). Then, the EM algorithm is integrated into the E-LW-IC-KPLS to analyse the missing data problem. Hence, another new algorithm, the 'EM ensemble LW independent component Kernel partial least square' (EM-E-LW-

IC-KPLS) has been developed. This new algorithm is expected to be not only robust against missing measurements but also more accurate in dealing with non-linear and non-Gaussian distributed process data. These algorithms have been tested using data in case studies 4 to 6. Their results are then compared to the existing LW-PLS and LW-KPLS and their integrated algorithms.

The newly proposed algorithms are tested using process data generated from six simulated plants. They are numerical example 1 (case study 1), a single chemical reactor (case study 2), wastewater treatment (case study 3), numerical example 2, a static approximation of two sine waves (case study 4), Eukaryotic cell cycle regulation (case study 5) and a highly nonlinear continuously stirred tank reactor (case study 6). Historical data of these simulated plants are used to validate the developed models. These data are produced from virtual plants using the MATLAB Simulink. The simulated data in case studies 1 to 3 are nonlinear, and Gaussian distributed data while case studies 4 to 6 are nonlinear and non-Gaussian distributed data. Then, the simulated data in case studies 4 to 6 are further treated to generate different levels of random missing measurements ranging from 0% to 60%.

In addition**,** from the practical point of view, the new algorithms should not only be evaluated regarding predictive performances, but their computational efficiencies also have to be assessed by calculating the central processing unit (CPU) running time to predict the targeted process variable(s). Hence, computational times of the developed and current adaptive algorithms are analysed and compared. These comparisons are accomplished by running the improved and the existing algorithms in a software platform using a computer to determine their CPU times for data processing and predictive modelling. In this research, to minimise these loads, the ensemble method is adopted in the proposed Gaussian- and non-Gaussian-based LW algorithms.

1.6. Novelty, contribution and significance

The novelty of, contributions to and significance of this current research is indicated in the following:-

Scientific contributions

a. A new adaptive algorithm utilised to formulate an adaptive soft sensor, able to analyse nonlinear and non-Gaussian distributed data as well as possessing robustness against missing data has been developed to address the research gaps of the existing algorithms.

b. This new algorithm is an additional approach to develop an adaptive soft sensor and is a more accurate alternative to the existing algorithms to address the nonlinear and non-Gaussian distributed data as well as with the presence of missing data.

c. Other associated outcomes of the EM-E-LW-IC-KPLS are E-LW-KPLS and E-LW-IC-KPLS are newly developed algorithms and methods for improving adaptive soft sensors.

Practical applications

a. The new EM-E-LW-IC-KPLS algorithm for adaptive soft sensors has the potentials to be implemented in any industrial process that has nonlinear and non-Gaussian distributed operational data in addition to missing measurements.

1.7. Structure of the thesis

In this section, a brief description of the structure of this thesis is provided. This thesis is organised and divided into seven chapters. Chapter 2 describes an overview of the existing adaptive soft sensors. Including a critical review of their limits and finally, the research gaps in current research on adaptive soft sensors are identified.

Chapter 3 describes the basic concept of the newly integrated algorithm formulation to address problems that have been identified earlier. Then, it is followed by the main stage of the research, the framework of the newly proposed Gaussian and non-Gaussian LW algorithms formulations. Next, case studies, predictive performance measurement and the specifications of computing facilities are briefly explained.

Chapter 4 presents the new Gaussian-based LW algorithm formulation for adaptive soft sensors for process plants, the E-LW-KPLS followed by an explanation of similarity measurement, Kernel partial least square and LW-PLS algorithms. Later in Chapter 5, there is a discussion of the formulation of non-Gaussian-based LW algorithms, including the E-LW-IC-KPLS, EM-E-LW-IC-KPLS, and EM.

In Chapter 6, the formulated E-LW-KPLS, E-LW-IC-KPLS, EM-E-LW-KPLS and EM-E-LW-IC-KPLS algorithms described in Chapters 4 and 5 are applied to the generated Gaussian and non-Gaussian data for virtual plants. The results obtained from these algorithms are analysed, discussed and compared with the LW-PLS, LW-KPLS and their integrated algorithms. Besides, computational times used for running these algorithms are assessed.

Finally, Chapter 7 presents the research summary, conclusions of this current research and recommendations for further studies on adaptive soft sensors. Figure 1.1 illustrates the overview of the thesis structure in a flow diagram.

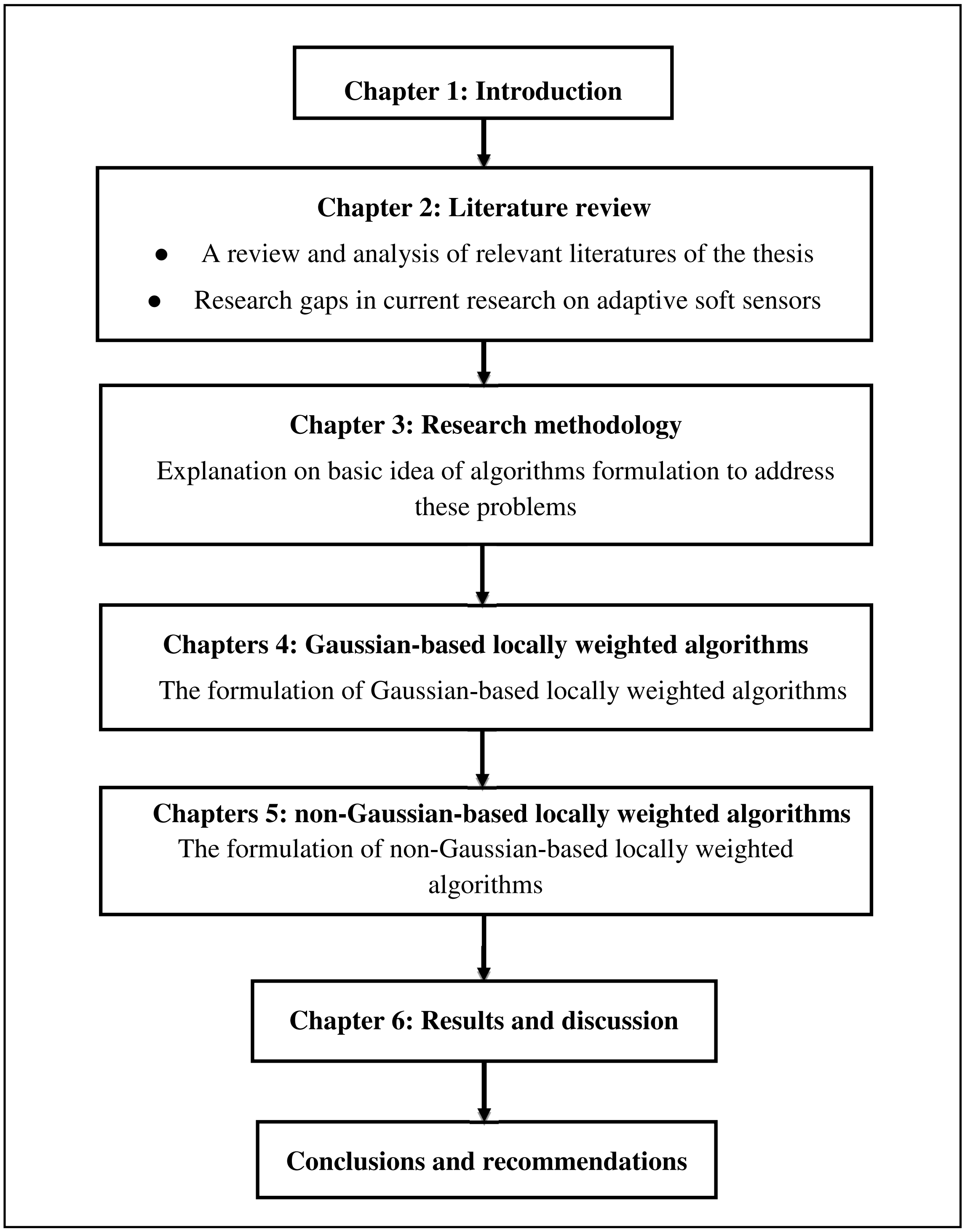
Chapter 1: Introduction
Chapter 2: Literature review
● A review and analysis of relevant literatures of the thesis
● Research gaps in current research on adaptive soft sensors
Chapter 3: Research methodology
Explanation on basic idea of algorithms formulation to address these problems
Chapters 4: Gaussian-based locally weighted algorithms
The formulation of Gaussian-based locally weighted algorithms
Chapters 5: non-Gaussian-based locally weighted algorithms
The formulation of non-Gaussian-based locally weighted algorithms
Chapter 6: Results and discussion
Conclusions and recommendations

Figure 1.1 Flow diagram of the thesis

Chapter 2

Literature review

2.1. Introduction

In this chapter, the existing algorithms used to develop adaptive soft sensors are discussed and critically reviewed. Firstly, the background of soft sensors used to reduce the drawbacks of hardware sensors is provided and then followed by a discussion of adaptive algorithms. The adaptive algorithms are broadly classified into non-just-in-time (non-JIT) and just-in-time (JIT) based algorithms. The JIT based algorithms, categorised into either Gaussian-based or non-Gaussian-based are further illustrated. Moreover, there is a brief discussion on algorithms that analyse nonlinear data and missing data. The pros and cons of these algorithms are outlined and discussed and ultimately identifying the research gaps to be addressed in this research project.

2.2. Soft sensors

Hardware sensors are commonly used in industrial processing plants for fault detection, process monitoring, and control to ensure safe, optimal and environmentally friendly operations. These sensors are required to observe and manage the processes, then undertake any necessary responses to address any abnormal process performance to achieve process optimisation with regards to efficiency and operating costs (Sharma and Tambe 2014). Despite their successful implementation, however, current applications of hardware sensors have encountered issues summarised in Table 2.1. These tabulated data were collected from an industry survey conducted by the JSPS PSE 143 Committee (2004).

Table 2.1 Problems with hardware sensors (JSPS PSE 143 Committee 2004)

Raising issues	**Percentage**
Time-consuming maintenance	27%
Need for calibration	21%
Aged deterioration	15%
Insufficient accuracy	13%
Long dead-time, slow dynamics	10%
Large noise	8%
Low reproducibility	2%
Others	4%

These commonly cited issues may lead to safety, and environmental concerns increased production costs and lower final product quality. To address these concerns associated with hardware sensors, virtual sensing technology, often referred as soft sensors or inferential sensors, have been proposed and developed by many researchers (Zhang *et al.* 2017; Kano and Koichi 2013; Kadlec *et al.* 2009). Past research and industrial applications have indicated soft sensors have become a popular alternative to hardware sensors when the hardware sensors are unavailable, or their drawbacks have become overwhelming.

Table 2.2 summarises and illustrates the general classifications of soft sensors. Soft sensors can be categorised as either ‘first principle’ or ‘white-box’ or ‘physical model-based’ (or simply model-based), historical data-driven or ‘black-box’ and a combination of model-based and data-driven or ‘grey-box’ based soft sensors. Developing a model-based soft sensor requires accurate fundamental models, which are formed based on detailed knowledge about physicochemical phenomena of processes such as kinetics, fluid flow, mass transfer, heat transfer, and thermodynamics. Since most of the industrial processes are usually very complex, obtaining this knowledge is time-consuming and expensive (Parish and Duraisamy 2016; Braunschweig and Joulia 2008). For example, predicting the properties of a polymer (e.g., viscosity, densities) via a nonlinear first principle model requires a product molecular property analysis in a lab that is usually expensive and time-consuming (Pantelides and Renfro 2013). Sometimes, the model-based method is impractical due to insufficient accurate fundamental model parameters. As a result, data-driven approaches have gained popularity and have been widely employed in the last two decades in developing predictive models from process data (Ge 2014; Kadlec *et al.* 2009).

Table 2.2 Overview of soft sensor types

Soft sensors		
Model-based approaches (White-box models)	Data-driven approaches (Black-box models)	Hybrid approaches (Grey-box models)

Moreover, the results of the industrial survey mentioned earlier have indicated data-driven soft sensors are more widely employed than model-based soft sensors, as shown in Table 1.1 in Chapter 1. Additionally, model-based and data-driven models have also been combined to overcome the limitations of a model-based model (Ahmad *et al.* 2014; Ukai *et al.* 2011; Nakabayashi *et al.* 2010; Chen *et al.* 2010). Such integrated models are known as grey-box models because they are constructed using both white-box and black-box models.

Data-driven based soft sensors have been developed based on historical data collected from industrial processing plants using empirical modelling techniques such as multiple linear regression (MLR), principal component regression (PCR), partial least square (PLS), artificial neural networks (ANN), support vector regression (SVR), least square support vector regression (LSSVR), Gaussian process regression (GPR) and independent component regression (ICR) (Kano and Koichi 2013; Stulp and Sigaud 2015). Table 1.1 reveals that among the existing techniques, data-driven-based soft sensors are the most popular followed by model-based soft sensors. Detailed discussions on the above modelling techniques can be found in review papers written by Kadlec *et al.* (2009), Qin (2012), Ge *et al.* (2013), Stulp and Sigaud (2015), Souza *et al.* (2016) and Funatsu (2018).

Traditional data-driven soft sensors use historical plant data in offline mode before their applications. Although the historical data are rich (Dong and McAvoy 1996), they do not contain all possible future plant operating conditions. Hence, the predictive performance abilities of these soft sensors will gradually deteriorate over time. This condition is due to changes in the state of the processing plants and process characteristics such as process feed materials, process fouling, catalyst deactivation, equipment aging, and changes in the external environment, etc. These newer states or characteristics are not captured in the historical data. For sensors to respond to any possible new conditions, non-adaptive soft sensors must be regularly maintained and updated, and therefore the incorporation of self-adapting mechanisms into soft sensors has attracted considerable interest.

2.3. Adaptive algorithms

In the literature, adaptive methods such as moving window (MW), recursive, time difference (TD) and ensemble approaches have been proposed to develop adaptive data-driven soft sensors. Despite their ability to automatically update models, these approaches have been associated with some drawbacks in their practice. MW models are built with the most recently adopted samples by adjusting a window to include the newest data and then eliminating the oldest data from the model. However, this makes MW perform poorly in processes with abrupt changes. Furthermore, according to Saptoro (2014) and Kadlec *et al.* (2011), MW must keep all data within the window and thus its application is limited to the memory size. Moreover, it is difficult to set the optimal size of the window and the adaption interval between updates (step size). Furthermore, MW losses the old data which may contain beneficial information (Yang *et al.* 2016).

On the other hand, rather than storing all the data in the memory like MW, recursive methods utilise a ‘forgetting factor’ to decrease the influence of old information. Besides, the recursive method allows the gathering of new information for constructing a predictive model. These methods can perform model adaptation and maintain their stable model structure. Nevertheless, Saptoro (2014), Ge *et al.* (2013) and Kadlec *et al.* (2011) have reported these methods to perform poorly in processes with abrupt changes. Kaneko and Funatsu (2013) have also indicated due to its sequential updating, and the recursive method is sensitive to sudden changes. Furthermore, when the process operation is carried out within a narrow range for certain duration, the recursive techniques will blindly update the model and a significant period is required to adapt into a new process operating condition.

TD approaches have been used by Kaneko and Funatsu (2011a), Kaneko and Funatsu (2011b), Kaneko and Funatsu (2011c) as well as Kaneko and Funatsu (2013b) to develop adaptive soft sensors. According to these authors, the TD approaches utilise time difference of explanatory variables, x and that of target variable(s), y to build models to decrease the impact of deterioration over time. However, since the TD approaches do not observe process conditions such as changes of a state for a chemical process, its predictive performance can be lower than that of other models (Kaneko and Funatsu, 2013b, 2011b; Okada1 *et al.*, 2010). To address this limitation, updated models which include the MW and JIT models have been combined with the TD algorithm by Kaneko and Funatsu (2015b). Research on the deterioration of adaptive methods against changes in the state of chemical plants was done by Kaneko and Funatsu (2013a).

Ensemble algorithms are complicated when a set of different internal models is employed. Ensemble methods have been demonstrated by Krogh and Vedelsby (1995) that the average construction of the various internal models is not an optimal integration approach and the weighted integration method such as the JIT locally weighted approach is often preferred (Kadlec *et al.* 2011). On the other hand, the JIT modelling algorithm only uses a single model to estimate target variables in modelling. Hence, it has a lower computational load than the ensemble methods. Moreover, JIT methods can be applied to nonlinear processes and processes with abrupt changes. As a result, JIT modelling was introduced as an attractive solution to build adaptive soft sensors; it addresses the limitations of the methods mentioned above are shown in Table 2.3.

Generally, MW, recursive, TD, ensemble and JIT methods have been used to develop adaptive data-driven soft sensors. Reviews of MW, recursive and ensemble methods have been published by Saptoro (2014), Kano and Koichi (2013), Ge *et al.* (2013), Kadlec *et al.* (2011), and Kadlec *et al.* (2009). Reviews of JIT methods were briefly discussed by Saptoro (2014), and Kano and Koichi (2013). While the following sections present an extensive overview of JIT based algorithms for adaptive data-driven soft sensors.

2.4. Just-in-time based algorithms

The JIT modelling algorithm was formulated based on ideas from database technology and local modelling and thus, it is also known as 'instance-based learning,' locally weighted model, lazy learning or model-on-demand (Jin *et al.* 2015b; Yuan *et al.* 2014a; Saptoro 2014; Kano and Koichi 2013). For the JIT modelling approach, there is an assumption that all useful observations are stored in the historical database. And then, a predictive model is built dynamically upon a query. Later, these models are used to predict the output of the query sample.

Table 2.3 Limitations of various adaptive methods for data-driven soft sensors (Agus 2014; Kadlec *et al.* 2011)

Methods	**Limitations**
Moving window	Perform poorly in the process with abrupt changes. Is not suitable for large window and memory limited applications. Is difficult to set the optimal size of the adaption window (window size) and adaption intervals between the updates (step sizes). Losses the eliminated old data which may consist of beneficial information.
Recursive methods	Perform poorly in the process with abrupt changes. When process operation is carried out within a narrow range for certain duration, recursive methods will blindly update the model. A period is required to adapt to a new process operating condition.
Time difference methods	Do not observe process conditions such as changes of state for the chemical process.
Ensemble methods	The average constructing of different internal models is not an optimal integration approach. Have high computational load since these methods consist of a set of different local models.

Unlike traditional methods that refer to global modelling, a JIT model is a local model that is developed from a historical dataset around a query data sample when an estimated value of this point is required. Moreover, the global model is constructed offline while the JIT model is built online (Liu and Yoo 2016; Jin *et al.* 2015a; Zhang *et al.* 2015; Yuan *et al.* 2014a; Jin *et al.* 2014; Saptoro 2014; Ge and Song 2010). For this reason, the JIT model is able to trace the current state of the process more efficient. It not only can deal with slow-varying operations but also abrupt process changes (Liu and Yoo 2016; Jin *et al.* 2015a; Jin *et al.* 2015b; Zhang *et al.* 2015; Liu and Gao 2015; Jin *et al.* 2014; Fan *et al.* 2014).

As mentioned previously, the JIT based method builds a local model using historical data having a high similarity with the query point. Therefore, some versions of JIT modelling algorithms have been proposed based on the various similarity measures (Hazama and Kano 2015; Jin *et al.* 2015a; Saptoro 2014; Kano and Koichi 2013). These versions include linear, combined linear and angle distance and correlation-based JIT (CoJIT) algorithms. Among these various JIT algorithms, the distance based similarity measure algorithms have been more widely used in the literature since it is easier to implement (Zhang *et al.* 2015; Saptoro 2014; Jin *et al.* 2014; Xie *et al.* 2014). Nevertheless, the CoJIT proposed by Fujiwara *et al.* (2012), Fujiwara *et al.* (2009) and Fujiwara *et al.* (2008) has been found to be more effective than the distance-based JIT algorithms c predictive performance.

Other classifications of the JIT modelling algorithm may also be based on the type of local model used and the characteristics of the process data. It has been reported by Hazama and Kano (2015), Yuan *et al.* (2014a) and Kano and Koichi (2013) that the adopted local models may be either MLR, PLS, SVR, LSSVR, GPR, locally weighted regression (LWR) and locally weighted PLS (LW-PLS) models. Therefore, JIT algorithms have been proposed based on these local models such as JIT PLS, JIT SVR, JIT LSSVR, JIT GPR, LWR, and LW-PLS. Meanwhile, characteristics of the process data include the following:

Gaussian versus non-Gaussian distributed data, linear versus nonlinear data and complete versus incomplete data. The sub-sections below summarise the literature survey of the classification based on these process data characteristics.

2.5. Gaussian-based just-in-time algorithms

In this section, those JIT modelling algorithms that recognise Gaussian distributed data are discussed in the following sub-sections. These sub-sections review previous studies which were based on CoJIT, locally weighted learning (LWL) based JIT, sparse sample regression based JIT (SSR-JIT), SVR and LSSVR based JIT methods as well as online local learning based and GPR-based JIT methods.

2.5.1. Correlation based just-in-time algorithms

As previously stated, the CoJIT algorithm was proposed to develop adaptive soft sensors. In the CoJIT modelling, samples for local modelling are chosen from the correlation between process variables (Fujiwara *et al*. 2008). Fujiwara *et al.* (2008) and Fujiwara *et al.* (2009) introduced CoJIT modelling for soft sensors. In their studies, Q statistic, Q_s which was derived by principal component analysis (PCA), measured the dissimilarity between the samples and the modelling data based on the correlation between process variables. Meanwhile, Hotelling's T squared statistic, T^2 is used to prevent extrapolation and ensure the sample is situated in the modelling data. The similarity factor, J for CoJIT, can be obtained from the following equation:

$$J = \lambda T^2 + (1-\lambda) Q_s \tag{2.1}$$

where λ is the setting parameter from trial-and-error and its range is from 0 to 1.

Later, Fujiwara *et al.* (2012) incorporated the nearest correlation (NC) into CoJIT to further improve its accuracy in dealing with individual differences in production units as these units have different characteristics even though they have the same catalogue specifications. In their proposed algorithm, NC, a correlation-based pattern recognition technique was adopted to determine the samples used for local modelling having a strong correlation with the query. In addition, this developed NC-CoJIT algorithm is able also to perform effectively with smaller operation data sets. The effectiveness of this relatively recently developed algorithm has been established using a parallelized chemical reaction process.

Subsequently, Fujiwara *et al.* (2010) later established an improved version of NC-CoJIT where NC was integrated with spectral clustering (SC), a graph partitioning approach. In this proposed NC-SC CoJIT, NC was used to build a weighted graph that showed correlation-based similarities between the samples and the query. Then, the weighted graph is later split using SC. Fujiwara *et al.* (2010) showed that their proposed CoJIT based algorithm could provide improved accuracy compared to conventional distance-based methods. Kano and Koichi (2013) stated that the CoJIT (with PLS as its internal model) is widely used in current industrial applications due to its simplicity and superior performance. However, all of the abovementioned CoJIT based approaches perform poorly with outliers and have difficulty in determining the parameter (λ), which is usually carried out by trial-and-error (Saptoro 2014).

To cope with outliers and time-varying processes, Liu *et al.* (2012b) have adopted robust nearest correlation to develop CoJIT based soft sensors. Moreover, these authors combined their CoJIT algorithm with an inductive confidence predictor to form an interval soft sensor to enhance its predictive performance. Since the computational load of this proposed algorithm is high, multi-model ensemble learning is utilized to reduce its computational burden. Furthermore, this algorithm performs well in processes with and or without the presence of outliers. However, all of the abovementioned CoJIT algorithms are only applicable to Gaussian distributed process data without the presence of missing data.

2.5.2. Locally weighted learning based just-in-time algorithms

Apart from CoJIT algorithms, linear distance-based algorithms based on LWL such as LWR, LW-PLS, and ensemble LW-PLS have also been formulated for developing adaptive soft sensors and process monitoring. Shigemori *et al.* (2011) built a LWR based quality design system for the steelmaking industry which can identify optimal manufacturing conditions. In their study, the target quality was monitored and ensured using this LWR model. Later, Shigemori (2015) also applied the LWR model to control cooling temperatures in a steel plate production process. Toshiya and Kano (2015), however, reported that the LWR algorithm becomes problematic when the number of new input variables is more than their nearest neighbours' samples in the historical data.

In this regard, the LW-PLS algorithm has been found to be a more effective alternative due to its ability to deal with a higher dimensionality of variables (a large number of input variables) and outliers (Hazama and Kano 2015; Kaneko and Funatsu 2015a) as well as collinearity among the process variables (Kano and Koichi 2013). Applications of the LW-PLS based soft sensors in various industries such as semiconductor, petrochemical, biochemical and pharmaceutical processing have been reported in publications and analysis indicates the LW-PLS algorithm has become a popular algorithm in developing soft sensors (Hazama and Kano 2015; Toshiya and Kano 2015; Kano and Koichi 2013; Kim *et al.* 2013b; Kim *et al.* 2013a; Nakagawa *et al.* 2012).

On the other hand, to enhance the predictive performance of the LW-PLS algorithm, a hybrid of the LW-PLS algorithm with different adaptive approaches such as the MW, TD and ensemble methods have also been developed for adaptive soft sensors. Yuan *et al.* (2015a) introduced MW LW-PLS algorithm under the TD (MW-JITL-TD) method with temporal and spatial adaptive techniques for nonlinear time-varying and variable drifting processes. In the MW-JITL-TD model, the MW has been adopted to adapt to the new process state while the TD model has been used to tackle the process change of variable drifts. Meanwhile, spatial adaptive LW-PLS algorithms and temporal adaptive MW are utilised to improve the performance of the predictive model. Nonetheless, this proposed approach did not take into consideration multiple process states which operate separately.

Consequently, Kaneko and Funatsu (2016) proposed the ensemble LW-PLS algorithm where a traditional LW-PLS algorithm is combined with ensemble learning. This newly developed method is claimed to be more effective compared to the LW-PLS since LW-PLS model has lower prediction accuracy when it is applied in multiple process states that perform individually. Hence, the LW-PLS models are formulated for each multiple sub-datasets and are weighted with ensemble learning and Bayes' theorem. This recent method can cope with time-varying, and multiple processes state concurrently. However, this ensemble LW-PLS model did not take into account strongly nonlinear data.

More recently, Zhang *et al.* (2017) integrated Kernel functions into the LW-PLS algorithm, the LW-KPLS for nonlinear time-varying processes. In their paper, a sparse Kernel features a characterization factor that considers the strength of nonlinear dependency between a query and training samples. In the high dimensional Hilbert feature, space is utilized to weight their training dataset. Similar to the SVR and LSSVR, the Kernel-based models, which include the LW-KPLS model, have increased their computation loads (Goldberg and Elhadad 2008). However, an investigation into the computational efficiency of these newly developed models was not carried out.

In addition to the LW-PLS based methods, another type of LWL method based on principal component regression (PCR), the PCA, or latent factor analysis (LFA) has also been established for soft sensing. These other methods include locally weighted Kernel principal component regression (LW-KPCR), locally weighted probabilistic principal component analysis (WPPCA), double locally weighted principal component regression (D-LWPCR), locally weighted supervised latent factor analysis (LWSLFA), locally weighted semi-supervised latent factor analysis (LWSSLFA) and weighted linear dynamic system (WLDS) approaches.

Since linear PCR in a locally weighted PCR (LW-PCR) has issues when dealing with nonlinear processes, Yuan *et al.* (2014a) incorporated Kernel functions into the LW-PCR model to form the LW-KPCR algorithm to cope with nonlinear time-varying processes. The nonlinear features of Kernel functions such as linear Kernel, polynomial Kernel, sigmoid Kernel, and Gaussian Kernel functions enable the LW-KPCR model to capture nonlinear data. Hence, the LW-KPCR algorithm has a higher prediction ability for highly nonlinear processes compared to the LW-PCR, Kernel PCR, and PCR.

Later, Chen *et al.* (2018) proposed D-LWKPCR based on approximate linearity dependence (ALD) for time-varying and strong nonlinear processes. This D-LWKPCR extracts more accurate output-related nonlinear features than LW-KPCR since it considers the variable relevance with the quality output. Euclidean distance is used to obtain the sample and variable weight coefficients, and then different weights are assigned for variables during local modeling. To decrease the computational time of D-LWPCR, ALD is used to decide the requirement for updating the local model. Hence, it has lower computational burden than the ordinary LW-KPCR. The ability of this D-LWKPCR was only evaluated using an industrial roller kiln. Thus, its applicability in other processes is uncertain.

Instead of using the Kernel function, Yuan *et al.* (2015b) proposed a WPPCA that uses a nonlinear dimensional reduction method to obtain nonlinear features from process data to build the soft sensor model. JIT learning (JITL) is utilised by WPPCA to select the most relevant samples for each query sample for local modelling. Then, each relevant sample is weighted separately based on their similarities to the testing sample under the probabilistic framework. Next, the WPPCA model is formulated to extract nonlinear features to predict the output of the query sample, $\hat{y}_q$, via the following equation:

$$\hat{y}_q = \bar{y}_w + \theta^T \bar{z}_q \tag{2.2}$$

where $\bar{y}_w$, θ and $\bar{z}_q$ are the weighted mean of the output, the regression coefficient vector and the mean vector of the posterior distribution of the query point, respectively. However, the LW-KPCR and WPPCA only consider the input information without any reference to the output information which could lead to imperfect sample selection.

To consider including the output information into the relevant sample selection, Yuan *et al.* (2016b) proposed the D-LWPCR which utilizes an improved sample selection method in the supervised latent structure where the extracted latent variables are extremely related to the output variables. By considering sample importance and variable importance, which are obtained from the sample-wise and variable-wise weighting methods, the D-LWPCR can extract nonlinear and output-related features for prediction. The double-weighted data matrix, $X_{s,v}$ and the double-weighted covariance matrix, $C_{s,v}$ is shown below:

$$X_{s,v} = W_s X W_v \tag{2.3}$$

$$C_{s,v} = \frac{1}{N} W_v^T X^T W_s^T W_s X W_v \tag{2.4}$$

where W_s, W_v, X, and N are the sample weighted data matrix, the variable weighted data matrix, the historical input matrix and the number of samples, respectively.

Additionally, the abovementioned LW-PLS based and the PCA locally weighted based approaches did not consider both supervised and semi-supervised process data. Hence, Yao and Ge (2017) created two probabilistic locally weighted prediction approaches, the LWSLFA, and LWSSLFA models for the data which employs the expectation maximization (EM) algorithm. Unlike traditional global modelling approaches, a range of historical samples is extracted by integrating the similarity index into the noise variance of the process variables that provide reliable information about nonlinearity and abrupt changes in the process. After underlying factors are determined, the predicted output variables are calculated through an Equation (2.5). The authors showed the effectiveness and flexibility of both newly developed methods using industrial processes. The computational load of these approaches has not been examined since the EM algorithm can be time consuming (Lan *et al.* 2016; Karanja *et al.* 2013).

$$\hat{y}_q = CP^T\left(\sum_x + PP^T\right)^{-1}\left(x_q - \bar{x}\right) + \bar{y} \tag{2.5}$$

where C and P are the factor loading matrices of the input and output data set, respectively while $\bar{x}$ and $\bar{y}$ are the mean values of the input and output variables. Additionally, $\sum_x$ is the measured noise variance of the input dataset.

Since most of the abovementioned models use a static linear approach, they are not suitable for processes that are dynamic with nonlinearity. Hence, recently, Yuan *et al.* (2018) have developed a new WLDS model for dynamic and nonlinear processes. In their study, a WLDS-based probabilistic latent variable model is introduced for nonlinear feature representation. Two types of weights in the WLDS were constructed for local linearization of the nonlinear state evolution and state emission approximation. Moreover, a weighted log-likelihood function has been built, and the EM is utilised for parameter estimation. The effectiveness of the improved model is illustrated using numerical examples and industrial process data.

2.5.3. Sparse sample regression based just-in-time algorithms

Subsequently, Uchimaru and Kano (2016) introduced the SSR-JIT that is an improved approach for the conventional distance based locally weighted learning methods. These distanced based approaches do not often perform well since correlation among the process variables is not considered. Even though the CoJIT can obtain high prediction accuracy, it is very dependent on fine-tuning the parameters. To address these limitations, the SSR-JIT utilizes an elastic net to locate useful past samples to build an accurate local model. A sparse regression model has been constructed to estimate the output of a query when the SSR-JIT solves the optimization problem (Zou and Hastie 2005) as shown in Equations (2.6) and (2.7) where the formulated regression coefficient, β is employed as a weighted sample. However, this method has not considered highly nonlinear processes data.

$$\beta_q = \arg\min_{\beta} \left\| x_q - X^T \beta \right\|^2 \tag{2.6}$$

$$\text{s.t.} \left(1 - \alpha\right)\left\|\beta\right\|_1 + \alpha\left\|\beta\right\|^2 \leq \gamma \tag{2.7}$$

where β_q is the regression coefficient vector, while α and γ are parameters.

Later, Wei *et al.* (2019) proposed three algorithms namely neighbourhood preserving embedding regression (NPER), sparse neighbourhood preserving embedding regression (SNPER), and locally weighted SNPER (LW-SNPER) for nonlinear, high-dimensional and time-varying processes. NPER can make sure of the intrinsic local topology structure and proximity relations among data samples which cannot be done by the global-based methods in estimating the quality output variables. However, when the number of process variables is more than the number of samples, SNPER is more suitable than NPER. SNPER uses elastic net regulation that is also utilised in SSR-JIT and it can spontaneously consider the local geometry of data and the connection between the input and output variables. To update this predictive model, the locally weighted method is integrated with SNPER to deal with time-varying characteristics in the process. Nevertheless, the ability of these newly developed algorithms to deal with outliers and non-Gaussian distributed data has not been examined.

2.5.4. Support vector regression and least square support vector regression based just-in-time algorithms

Furthermore, the nonlinear JIT algorithms, JIT SVR and JIT LSSVR are also used to develop adaptive soft sensors for highly nonlinear processes. A comparative study of the JIT PLS, JIT SVR and JIT LSSVR conducted by Ge and Song (2010) showed the JIT LSSVR provided more effective prediction capabilities for highly nonlinear processes due to its nonlinear model and a higher value of correlation coefficient compared to the JIT SVR. In contrast to the standard JIT PLS, which has a linear internal model, the JIT SVR and JIT LSSVR have a greater ability to capture strongly nonlinear process data. On the other hand, Liu and Yoo (2016) have applied the JIT LSSVR to predict and monitor the indoor air quality for hazardous pollutants and have shown the superior prediction performance of the JIT LSSVR compared to the LSSVR. The SVR and LSSVR based algorithms which include the JIT SVR and JIT LSSVR have been acknowledged to support a much higher computational load than PLS-based algorithms (Liu *et al.* 2012a; Ge and Song 2010).

Different versions of the JIT LSSVR have been established to reduce the computational load. Liu *et al.* (2012a) proposed a JITL based recursive LSSVR (JITL-RLSSVR) for nonlinear batch processes. In this study, they presented a cumulative similarity factor, S_l, which can be calculated via Equation (2.8) is used to determine the similarity between a query and relevant samples which includes the weight of similarity and the size of the corresponding set. After that, a fast leave-one-out cross-validation (FLOO-CV) strategy with a low computational load is employed to tune the kernel parameters in the JITL-

RLSSVR model adaptively. Hence, this proposed approach performs better than the JIT SVR and JIT LSSVR since it is not equipped for online optimization of model parameters. However, the disadvantage of this new approach is it does not consider the multistage, multiphase or multi-grade of the batch processes.

$$s_l = \frac{\sum_{l=1}^{l} s_{qi}}{\sum_{l=1}^{l_{\max}} s_{qi}}, \qquad l_{\min} \le l \le l_{\max} \tag{2.8}$$

where S_{qi} is the similarity factor between the query sample and the sample in the data set, while l, l_{min} and l_{max} are the related samples, which are chosen before constructing a JITL model.

Later, Liu and Chen (2013) developed another type of JIT LSSVR algorithm combining the JIT LSSVR with probabilistic analysis to address the multi-grade processes problem that often occurs in nonlinear fine chemical and polymer processes. First, the LSSVR models are derived for each product steady-state grade. And then, a probabilistic analysis approach utilizing the statistical property of steady-state classes is used for online prediction of a query sample. When the probability of the query sample for a particular steady-state grade is high enough, the prediction is made based on the respective LSSVR model. If the query sample is not similar to any steady-state category, it is categorised as in a transitional mode. Then, the JIT LSSVR model is built utilizing the sample most identical to the query sample via the FLOO-CV strategy with the low computational load. Still, this innovative approach has not yet investigated the non-Gaussian distributed data.

Other than the JIT SVR and JIT LSSVR, another type of nonlinear JIT algorithm, the MW JITL least square-support vector machine regression (MWJIT-LS-SVM) was also proposed by Li *et al.* (2015) for nonlinear and time-varying chemical distillation processes. The prediction accuracy of the JITL least square-support vector machine regression can be enhanced using MW with a particular moving window size algorithm. This technique is to ease proper samples selection when constructing a local model for a query. Additionally, incremental and decremental algorithms are adopted to reduce the computational complexity of the MWJIT-LS-SVM algorithm. Nevertheless, the computational load of the MWJIT-LS-SVM is still considered high, and its applicability to other processes has not been examined.

2.5.5. Online local learning based algorithms

Another variant of the JIT method commonly referred to as online local online learning has also been adopted to develop adaptive soft sensors. Jin *et al.* (2015b) proposed a multi-model online support vector regression (OSVR) for nonlinear time-variant batch processes. This method has been developed according to the local learning framework and OSVR. Initially, a batch process is described by a set of local operating domains via a t-test and the moving window strategy. Then, the localized OSVR models, f_m are constructed for each local operating domain via Equation (2.9). Based on Bayesian ensemble learning, these localized models demonstrate the best prediction result of similar samples to the query data and are then adaptively merged to predict the target output. This output is further corrected by implementing offset compensation. However, the application of this approach has been limited to batch processes only and has not considered multi-output processes.

$$f_m(x) = \sum_{i=1}^{n_m} \left(\alpha_{m,i} - \alpha_{m,i}^{*}\right) \ K\left(x_{m,i}, x\right) + b_m \tag{2.9}$$

where n_m and $f_m(x)$ are the number of local modelling sampling and the local OSVR model, respectively; While $x_{m,i}$ is the local training input, and $\alpha_{m,i}$, $\alpha_{m,i}^{*}$ and b_m are the OSVR model parameters.

Other than the SVR based algorithm, Shao *et al.* (2014) also introduced local learning based soft sensor, the local PLS based approach for online soft sensing (OLPLS) for multi-output processes. Two steps, the operation states division, and the local model adaptation are utilized to develop the OLPLS. Firstly, the unique local time regions are extracted using the adaptive process states division based on an *F*-test. Then, for local model adaptation, an anti-over-fitting criterion, J^* determined via Equation (2.10) is demonstrated by considering the correlation between process variables and the information is labeled and unlabeled samples. The authors showed the prediction performance of the OLPLS was superior to the CoJIT methods. The OLPLS has the same limitation as of the CoJIT, the ability to deal with outliers.

$$J^* = (1-\gamma_R)e_0\theta_D e_0^T + \gamma_R \sum_{i=1}^{K} s_i e_i \theta_D e_i^T / \sum_{i=1}^{K} s_i \tag{2.10}$$

where e_0 designates the predicted error vector for the newest labeled sample, e_i denotes the predicted error vector for the *i*-th nearest sample of the query sample, Θ_D is the diagonal matrix with all positive elements, S_i is the weight of the respective sample, and γ_R is the regularization parameter.

Jin *et al.* (2015a) also developed online local learning based adaptive soft sensor (OLLASS) for the industrial fed-batch chlortetracycline fermentation process. In this study, the neighbouring sample based on the mutual information (MI) weighted similarity measure is used to choose the samples for local modelling. Next, the maximum local modelling sizes for situations with and without the neighbour output information are determined by self-validation and neighbour-validation. Meanwhile, an online dual updating strategy is used to update the local model. Then, the model output offset is also updated to prevent unwanted local model regeneration to reduce computational load. Moreover, the maximal similarity replacement rule based on the MI weighted similarity measure is used to update the data set. However, the effectiveness and applicability of the OLLASS have not been tested in other nonlinear chemical and biological processes.

Moreover, Jin *et al.* (2016) developed an adaptive soft sensor using dual learning-based online ensemble regression (DLOER) for nonlinear time-varying processes. In this study, an adaptive localization approach utilizes JITL to build M local domains, LD_m and local models to handle process nonlinearity. Besides, statistical hypothesis testing is used to eliminate unnecessary local models. Afterward, the posterior probabilities of each test dataset, x_{new} corresponding to the diverse local models, p(LD_m|x_{new}) are determined via Bayesian inference. Then, they are applied as an adaptive weight to unite local predictions into a final output, $\hat{y}_q$ which can be calculated via Equation (2.11). Furthermore, the DLOER framework consists of a dual learning-based adaption mechanism, incremental local learning, and JITL, to allow recursive adaption and the online inclusion of local models to achieve high prediction performance. However, DLOER utilizes PLS which is a linear model for local modelling and hence does not work well in strongly nonlinear processes.

$$\hat{y}_q = \sum_{m=1}^{M} p(LD_m \mid x_{new}) y_{m,new} \tag{2.11}$$

where $y_{m,new}$ is the output variable from the local PLS model.

2.5.6. Gaussian process regression based just-in-time algorithms

Furthermore, Yang *et al.* (2016) developed ensemble JIT GPR (EJITGPR) based adaptive soft sensors for an industrial batch rubber mixing nonlinear process. Since a single local model of JIT GPR has limited capability for capturing all the relevant process features, the EJITGPR approach constructs a series of input variable datasets using the concept of partial MI and random re-sampling of historical data sets for multiple local modelling.

Meanwhile, a PLS analysis procedure is employed to eliminate unnecessary or redundant local variable sets. Moreover, the finite mixture and Bayesian inference techniques are utilised to unite the best local prediction outcomes into the final output, y_{new} as shown in Equation (2.12). However, the applicability and effectiveness of this recent approach have not been investigated in other nonlinear and multiple operation mode processes.

$$y_{new} = \sum_{i=1}^{M_e} y_{GPR_i,new} P(GPR_i \mid x_{new}) \tag{2.12}$$

where $y_{GPRi,new}$ and $P(GPR_i|x_{new})$ are the local prediction mean from the i-th local model, GPR_i, $i \in 1,2,\ldots M_e$, and the posterior probability of a new test sample, x_{new} for different local GPR models.

Besides, the JIT based MW GPR (MWGPR) approach with dual updating strategy was proposed by Xiong *et al.* (2016) to deal with multi-mode and nonlinear processes. Moreover, this approach has the capability of capturing switching dynamics or process nonlinearity. Firstly, the Gaussian mixture model (GMM) is adapted to distribute the data into separate operating modes. For each operating mode, the MWGPR utilises JITL to select relevant data in the specific window via specific nearest neighbourhood criterion to construct a local model for query sample. The predicted output is then improved using bias updating. However, the GPR in the MWGPR and EJITGPR has a higher modelling burden compared to the PLS and KPLS models (Jin *et al.* 2015a).

After that, Mei *et al.* (2018a) introduced a new JITL MW GPR for time-varying, nonlinear and multi-parameter characteristics of an industrial Erythromycin fermentation process. JITL is utilised to allow the construction of the local GPR model based on the sample kept in the last given MW. Different from the traditional soft sensors, this JITL MW GPR considers spatial characteristic of a query data point and local temporal characteristic of real-time process conditions. However, this proposed approach only tested on one case study and its predictions on other processes are not carried out.

2.5.7. Extreme learning machine based just-in-time methods

In addition, Li *et al.* (2018) proposed JIT extreme learning machine (JIT-ELM) for Polyethylene terephthalate (PET) production process to predict its viscosity. This polymerisation has multi-mode operating and multi-standard production conditions. This JIT-ELM is mainly designed to address the poor accuracy of the online instrument and time-consuming laboratory analysis problems. The local model for JIT-ELM is built to predict the output after the relevant samples are selected from the Euclidean distance and angle distance method. JIT-ELM is superior to LWR since it has fast learning speed and good generalization performance. Nevertheless, JIT-ELM has not been examined in other types of process.

Meanwhile, Zheng *et al.* (2018) suggested JIT semi-supervised ELM (JSELM) to predict the Mooney viscosity for industrial rubber mixers. This JSELM utilises a fast leave-one-out (FLOO) to train its regression model. The FLOO-based error index, E_{N_F,λ_F}^{FLOO} is shown in Equation (2.13) consisting of JSELM's parameters (the number of hidden nodes, N_F and a balance parameter, λ_F). These parameters need to be obtained using the FLOO strategy to train a JSELM regression model for an adopted similar dataset. Besides, the k-means clustering technique is also used to select the similar sample for semi-supervised learning of a local regression model. Nevertheless, JSELM has only tested on rubber mixers only.

$$E_{N_F,\lambda_F}^{FLOO} = \frac{1}{L_F}\sum_{i=1}^{L_F}\left(\frac{y_i - h_i[(J_F + \lambda_F L_F{}^T)H]^+ J_F Y}{1-[h_i[(J_F + \lambda_F L_F{}^T)H]^+ J_F]_i}\right)^2 \qquad (2.13)$$

where y_i is the training data, L_F is the number of training sample, J_F is a matrix consists of a unit matrix for the graph Laplacian, H is the output matrix of hidden-layer vector (h_i), and Y is the output data sample.

2.5.8. Just-in-time based methods

Besides, Liu *et al.* (2018) introduced a JITL strategy with common feature extraction algorithm for the complicated nonlinear characteristics of multi-grade processes. In their method, JITL is employed to select the relevant sample from different grades with regards to the query. Then, the common features shared by different grades are extracted based on the selected samples. Next, a PLS model is utilised to the special features for each grade. Hence, quality prediction can be done by integrating the common and special features of each grade. However, when the number of samples in each grade is sufficient, a simple PLS regression can obtain the similar prediction as this technique.

Meanwhile, Yuan *et al.* (2018) demonstrated a new ensemble JITL (E-JITL) method using multiple similarity measurements for nonlinear processes. In this method, different local models are built using different groups of relevant samples respective to the similarity measurements. The similarity measurements include Euclidean distance measurement, distance, and angle measurement, and distance measurement in the low-dimensional supervised latent space using PLS. Then, the final prediction is found using the ensemble method on each local model. The effectiveness of E-JITL is illustrated using two industrial applications. Nevertheless, the type of similarity measurements for the group of the sample is predefined. Hence, the optimal results using this E-JITL are not guaranteed.

At the same time, Mei *et al.* (2018b) suggested a novel JITL-bagging-PLS for fermentation processes. Different from the traditional similarity measurement, the authors use the Gaussian Kernel function to find the most relevant samples. The bagging method is utilised to reselect the obtained samples to prevent setting the size of the selected samples. Then, the local model is built by PLS for prediction purpose. The prediction performance of his JITL-bagging-PLS algorithm is evaluated using an industrial Penicillin fermentation. However, since Kernel function is utilised, the computation time for the proposed algorithm can be high, and it should be investigated.

Table 2.4 summarises the JIT based adaptive soft sensors for Gaussian distributed data in recent years. Table 2.4 shows that LW-PLS using Euclidean distance measure has been widely adopted. Among these JIT based approaches for Gaussian distributed data, LW-PLS model may still be preferable due to its simplicity and practicability. Since the LW-PLS model is simple and has been used in industrial applications, an improved LW-PLS model to deal with non-Gaussian distributed data is needed.

Table 2.4 Summary of the Just-in-time (JIT) based adaptive soft sensors for Gaussian distributed data

Methods	**Similarity factor**	**Process type [B, C]***	**Advantages**	**Process description**	**Publication**
Correlation-based JIT (CoJIT)	Q and Hotelling's T^2 statistics of PCA (Correlation measure)	C	Abrupt changes, time-varying, nonlinearity	CSTR, A cracked gasoline fractionator	(Fujiwara *et al.* 2008; Fujiwara *et al.* 2009)
Nearest correlation correlation-based JIT (NC CoJIT)	Q and Hotelling's T^2 statistics of PCA (Correlation measure)	C	Abrupt changes, time-varying, nonlinearity	Numerical example, a parallelized chemical reaction process	(Fujiwara *et al.* 2012)
Nearest correlation spectral clustering correlation-based JIT (NC-SC CoJIT)	Nearest correlation correlation spectral clustering method (Correlation measure)	B	Abrupt changes, time-varying, nonlinearity	Numerical example, a case study of parallelized batch process	(Fujiwara *et al.* 2010)

Ensemble Correlation-based JIT (CoJIT) with inductive confidence predictor	Q and Hotelling's T^2 statistics of PCA (Correlation measure)	C	Abrupt changes, time-varying, highly nonlinearity, Outlier	Wastewater treatment process	(Liu *et al.* 2012b)
JIT Locally Weighted Regression (JIT LWR)	Euclidean distance measure	C	Abrupt changes, time-varying, nonlinearity	Steel products plant	(Shigemori *et al.* 2011; Shigemori 2015)
Locally Weighted Partial Least Square (LW-PLS)	Euclidean distance measure	C	Abrupt changes, time-varying, nonlinearity, multiple mode, collinearity	Industrial petrochemical process, Industrial Pharmaceutical process, Industrial chemical plant, Industrial semiconductor	(Hazama and Kano 2015; Toshiya and Kano 2015; Kim *et al.* 2013b; Kim *et al.* 2013a; Nakagawa *et al.* 2012)

				process, industrial food processing plant	
Moving window Locally Weighted Partial Least Square under TD (MW-JITL-TD)	Euclidean distance measure with moving window and time difference strategies	C	Abrupt changes, time-varying, nonlinearity, multiple mode, collinearity	Numerical example, sulfur recovery unit, blast furnace iron making process	(Yuan *et al.* 2015a)
Ensemble Locally Weighted Partial Least Square (ELWPLS)	New distance measure together with data density and the nearest data	C	Abrupt changes, time-varying, nonlinearity, multiple mode, collinearity	Numerical example, a debutanizer column, an IPA production process	(Kaneko and Funatsu 2016)
JIT Locally Weighted Kernel Principal Component Regression (LW-KPCR)	Euclidean distance measure	B, C	Abrupt changes, time-varying, highly nonlinearity, collinearity	Debutanizer column, Fermentation process for Penicillin production	(Yuan *et al.* 2014a)

Double locally weighted Kernel principle component regression (D-LWKPCR)	Euclidean distance measure with weighted technique	C	Abrupt changes, time-varying, highly nonlinearity, collinearity	Industrial roller kiln	Chen *et al.* (2018)
Locally weighted Kernel partial least square regression (LW-KPLSR)	Euclidean distance measure	C	Abrupt changes, time-varying, nonlinearity	Numerical example, Penicillin fermentation process, Magnesium stearate concentration estimation in cleaning processes	(Zhang *et al.* 2017)
Locally weighted probabilistic principal component analysis (WPPCA)	Euclidean distance measure	C	Abrupt changes, time-varying, nonlinearity, dimensional reduction,	Numerical example, industrial debutanizer column,	(Yuan *et al.* 2015b)

			feature extraction		
Double locally weighted Kernel principle component regression (D-LWPCR)	Euclidean distance measure	C	Abrupt changes, time-varying, feature extraction, reference to output information	Numerical example, industrial blast furnace iron making process	(Yuan *et al.* 2016b)
Locally weighted supervised latent factor analysis (LWSLFA)	Euclidean distance measure	C	Abrupt changes, time-varying, highly nonlinear	Industrial debutanizer column, industrial of CO_2 column	(Yao and Ge 2017)
Weighted linear dynamic system (WLDS)	Euclidean distriance measure	C	Abrupt changes, time-varying, highly nonlinear	Numerical example, industrial debutanizer column	(Yuan *et al*. 2018)

Sparse sample regression based JIT (SSR-JIT)	Elastic net	C	Abrupt changes, time-varying, nonlinearity,	Industrial cleaning process, industrial drugs productions	(Uchimaru and Kano 2016)
Multiphase JIT - Kernel Partial Least Square (MJIT-KPLS)	Euclidean distance measure	B	Abrupt changes, time-varying, highly nonlinearity, multiple phases	Industrial chlortetracycline fed-batch fermentation process	(Jin *et al.* 2014)
JIT Least Square Support Vector Regression (JIT- LSSVR)	Euclidean distance measure	C	Abrupt changes, time-varying, highly nonlinearity	Tennessee Eastman Process, Debutanizer column	(Ge and Song 2010)
JIT Least Square Support Vector Regression (JIT- LSSVR)	Distance measure	C	Abrupt changes, time-varying, highly nonlinearity, Outlier	Indoor air quality of hazardous pollutants	(Liu and Yoo 2016)

JIT-Recursive Least Square Support Vector Regression (JIT-RLSSVR)	Distance and angle measure	B	Abrupt changes, time-varying, highly nonlinearity	Streptokinase fed-batch fermentation process	(Liu *et al.* 2012a)
JIT-Least Square Support Vector Regression (JIT-LSSVR)	Distance and angle measure	C	Abrupt changes, time-varying, highly nonlinearity, multi-grade	A simulated CSTR process, industrial polyethylene process	(Liu and Chen 2013)
MW JITL least square-support vector machine regression (MW JIT LS-SVM)	Distance and angle measure with moving window strategy	C	Abrupt changes, time-varying, nonlinearity	Distillation process	(Li *et al.* 2015)
Multi-model online SVR (MOSVR)	t-test and moving window strategy	B	Nonlinearity, multi-phases, batch to batch variations	Simulated fed-batch penicillin fermentation process, Industrial fed-batch	(Jin *et al.* 2015b)

				chlortetracycline fermentation process	
Adaptive/online local PLS (OLPLS)	F-test	C	Time-varying, nonlinearity, multi-output, process state division, local model adaption, reduce the memory cost	Debutanizer distillation column, Sulfur recovery unit	(Shao *et al.* 2014)
Online local learning based adaptive soft sensor (OLLASS)	Mutual information (MI) weighted/ neighbor sample based similarity measure	B	Abrupt changes, time-varying, nonlinearity	Industrial fed-batch chlortetracycline fermentation process	(Jin *et al.* 2015a)

Dual learning-based online ensemble regression (DLOER)	Euclidean distance measure	B	Abrupt changes, time-varying, nonlinearity	Fed-batch penicillin fermentation process	(Jin *et al.* 2016)
Ensemble JIT GPR (EJITGPR)	Euclidean distance measure	B	Abrupt changes, time-varying, nonlinearity	Industrial batch rubber mixing process	(Yang *et al.* 2016)
JIT based MW GPR (MWGPR)	Euclidean distance measure with moving window strategy	B, C	Abrupt changes, time-varying, nonlinearity	A continuous fermentation process, a pilot scale experiment	(Xiong *et al.* 2016)
JITL MW GPR (JIT-MWGPR)	Gaussian function-based similarity criterion	B	Abrupt changes, time-varying, nonlinearity	Industrial Erythromycin fermentation process	Mei *et al.* (2018)
JIT Extreme Learning Machine (JIT ELM)	Distance and angle measure	C	Abrupt changes, time-varying,	Industrial Polyethylene	Li *et al.* (2018)

			nonlinearity, multi-mode operating, multi-standard production condition	terephthalate polymerization process	
JIT semi-supervised Extreme Learning Machine (JSELM)	K-mean clustering method	B	Abrupt changes, time-varying, nonlinearity	Industrial rubber mixer	Zheng *et al.* (2018)
JITL strategy	Euclidean distance measure	C	Abrupt changes, time-varying, nonlinearity, multi-grade	A numerical example, industrial polyethylene process	Liu *et al.* (2018)
Ensemble JITL (E-JITL)	Euclidean distance measure, distance and angel measure,	C	Abrupt changes, time-varying, nonlinearity	Industrial hydro cracking process,	Yuan *et al.* (2018)

	distance measure in the low dimensional supervised latent space using PLS			blast furnace iron making process	
JIT-bagging-PLS	Gaussian function-based similarity criterion	B	Abrupt changes, time-varying, nonlinearity	A numerical example, Penicillin fermentation process	Mei *et al.* (2018)

*B = Batch processes; C = Continuous processes

2.6. Non-Gaussian based just-in-time algorithms

Several different approaches have been developed to reveal meaningful information in non-Gaussian data. Fan *et al.* (2014) have introduced a new GMM based similarity criterion which is based on distance measure for JIT based soft sensors to handle time-varying and non-Gaussian behaviour processes. In their study, the GMM has been built from a set of training samples to obtain the non-Gaussian information in the process data set. After considering the non-Gaussianity of the process data and the behaviour of the query sample, a more accurate similarity criterion for sample selection to *m* query, $\hat{y}_q$ can be calculated via Equation (2.14). However, Ge *et al.* (2013) have reported that GMMs are unable to model all types of non-Gaussian data and their model training is complicated.

$$\hat{y}_q = x_q W_m \left(P_m^{\ T} W_m\right)^{-1} q_m \tag{2.14}$$

where W_m, P_m, and q_m are the weighting matrix, loading matrix, and loading vector, respectively.

Xie *et al.* (2014) have integrated non-Gaussian regression (NGR) into support vector data description (SVDD) based JIT soft sensors on coping with non-Gaussian distributed process data. This integrated algorithm utilizes NGR to extract non-Gaussian information from process data while the SVDD is used to perform distance-based similarity measurements for constructing a local model. However, Xie *et al.* (2014) have reported this method can lead to suboptimal results due to the tuning parameter, which is used to ensure the best response of this method is obtained via a heuristic technique. Moreover, Ge *et al.* (2013) have declared that setting the Kernel parameter for the SVDD based model is difficult and the SVDD has a tighter control limit which may cause more false alarms. Also, Zhang *et al.* (2015) have reported that these GMM-based and SVDD based JIT algorithms are restricted to continuous processes.

On the other hand, Zhang *et al.* (2015) and Ge and Song (2008) have integrated independent component analysis (ICA) into their advanced JIT based algorithm for batch processes as an ICA can capture meaningful information on higher order statistics of non-Gaussian distributed data. Ge and Song (2008) have proposed a two-step ICA – PCA that combines with a JIT LSSVR model to perform online monitoring of nonlinear multiple mode processes with non-Gaussian information. However, Ge and Song (2008) did not consider multi-phases in their advanced algorithm, and their algorithm was limited to continuous processes.

As mentioned earlier, the JIT LSSVR suffers from a high computational load (higher than the JIT PLS). Zhang *et al.* (2015) have integrated ICA – PLS and non-Gaussian dissimilarity measures into a JITL soft sensor to estimate the quality variable of batch processes that are time-varying, nonlinear, non-Gaussian, with multi-phases and batch-to-batch variations. However, in their study, the proposed approach showed a much higher computational load than a PLS-based algorithm and became problematic when dealing with outliers since correlation-based measures were used.

Besides, Liu and Gao (2015) have proposed to integrate support vector clustering which is a non-Gaussian outlier detection method into the JIT GPR to control for non-Gaussian data and outliers for online prediction in nonlinear processes. Liu and Gao (2015) have adopted the JIT GPR which is a nonlinear model since it supplies probabilistic information for prediction and can maximise its parameters spontaneously which cannot be done by the JIT SVR and JIT LSSVR models. Nevertheless, Jin *et al.* (2015a) have reported that GPR methods have a heavier modelling burden compared to the PLS and KPLS methods. Hence, the computational load for the JIT GPR methods is high.

Peng *et al.* (2017) have created a JITL extreme learning machine (JITL ELM) for non-Gaussian chemical processes with multimode operating conditions. In their study, a speedy Kernel ELM approach, which uses Fast food Kernel (an approximate kernel expansion) has been adopted to extract information from high nonlinear process data. Additionally, a new similarity index, the modified adjusted cosine similarity (*MACS*) is proposed for local modelling and can be measured using Equation (2.15). With the help of a Bayesian classifier, the *MACS* index can control for *M* multiple modes, and C_j features in the JITL ELM model. Nevertheless, the developed approach did not consider the transition data, which may lead a failure to classify transition data from fault data.

$$MACS\,(x_{new}, x_i) = \sum_{j=1}^{M} p(C_j \mid x_{new}) ACS\,(x_{new}, x_i) \tag{2.15}$$

where $p(C_j|x_{new})$ and ACS are the posterior probability and the adjusted local cosine similarity index respectively.

From the review above, it is obvious there is a need to create a non-Gaussian algorithm that is simpler and more practical than the non-Gaussian algorithms discussed earlier in this chapter. Table 2.5 shows the summary of the JIT based adaptive soft sensors for non-Gaussian distributed data. Much less JIT based algorithms have been established for non-Gaussian distributed data as compared to the Gaussian based algorithms in Table 2.4. Nevertheless, the performance of the LW-PLS model in handling nonlinear processes data should be improved. In the literature, the incorporation of the Kernel function into the PCR and PLS models has resulted in Kernel PCR and KPLS models, respectively, and have shown improvements in their predictive performances (Jin *et al.* 2014; Hu *et al.* 2013). Thus, hypothetically, the inclusion of the Kernel function into the LW-PLS algorithm will be able to enhance the performance of LW-PLS models under nonlinear conditions. A brief review of the nonlinear issue is included in the following section.

Table 2.5 Summary of the Just-in-time (JIT) based adaptive soft sensors for Non-Gaussian distributed data

Methods	**Similarity factor**	**Process type [B, C]***	**Advantages**	**Process description**	**Publication**
JIT Gaussian mixture model (JIT-GMM)	Mahalanobis distance measure	C	Abrupt changes, time-varying, nonlinearity	Numerical study, Debutanizer column	(Fan *et al.* 2014)
Non-Gaussian Regression JIT Support Vector Data Description (NGR-JIT-SVDD)	Q and Hotelling's T^2 statistics of PCA (Correlation measure)	C	Abrupt changes, time-varying, nonlinearity	Numerical study, Sulfur recovery unit	(Xie *et al.* 2014)
JIT Independent component analysis Principal Component analysis Least Square Support Vector Regression (JIT-ICA-PCA-LSSVR)	Distance and angle measure	C	Abrupt changes, time-varying, highly nonlinearity, multiple mode	Numerical example, simulation of the Tennessee Eastman benchmark process	(Ge and Song 2008)

Support Vector Clustering JIT Gaussian Process Regression (SVC-JGPR)	Euclidean distance measure	B	Abrupt changes, time-varying, nonlinearity, non-Gaussian, outliers	Industrial polyethylene production process	(Liu and Gao 2015)
Non-Gaussian JITL (NG-JITL)	Non-Gaussian (ICA-PLS) dissimilarity measure (Correlation measure)	B	Abrupt changes, time-varying, nonlinearity, non-Gaussian, multi-phases, batch-to-batch variation	Fed-batch penicillin fermentation process	(Zhang *et al.* 2015)
JITL and Extreme Leaning Machine (JITL ELM)	Cosine similarity and posterior probability	C	Abrupt changes, time-varying, non-Gaussian	Numerical example, distillation column system	(Peng *et al.* 2017)

*B = Batch processes; C = Continuous processes

2.7. Nonlinear data issues

Although the LW-PLS model has been widely used to develop adaptive soft sensors, a linear PLS regression in the LW-PLS model may not work well when processes have nonlinear characteristics (Gao *et al.* 2015; Shan *et al.* 2015; Hu *et al.* 2013; Zhang and Zhang 2009). Thus, an improved algorithm for LW-PLS based adaptive soft sensors, which is capable of dealing with nonlinear data is required. Since developing an improved LW-PLS model is the primary objective of this study, only nonlinear types of PLS able to cope with nonlinear data is discussed in this subsection.

Several nonlinear versions of a PLS such as quadratic PLS (QPLS), spline PLS (SPLS), neural network PLS (NNPLS) and Kernel PLS (KPLS) models have been proposed for nonlinear issues. The internal nonlinear PLS (NPLS) models, QPLS, SPLS and NNPLS make use of nonlinear functions (e.g., a simple polynomial transformation of observed data) to define a relationship between latent variables (not directly observed variables) (Wang *et al.* 2015b; Rosipal 2010; Frank 1990). However, the predefined form of the quadratic function in QPLS algorithms has restricted its flexibility to develop a nonlinear model (Wang *et al*. 2015b; Shah *et al*. 2015; Shah *et al*. 2014; Zhou *et al*. 2007).

Although SPLS and NNPLS algorithms provide flexibility to capture nonlinearity relationships for variables, these methods may lead to local minima or obtain over-fitted model (Wang *et al.* 2015b; Shan *et al.* 2014; Wold 1992). Unlike the abovementioned internal NPLS models, the KPLS model, first introduced by Rosipal and Trejo (2002), maps observed variables into a high-dimensional feature space using the Kernel function before the linear PLS models are built in this new feature space. Compared to other NPLS models, Kernel-based techniques including the KPLS model, do not involve the nonlinear optimization strategy (Hu *et al.* 2013) and the KPLS model only requires linear algebra which is as simple as a linear PLS regression (Jin *et al.* 2014; Zhang and Zhang 2009).

Moreover, different types of Kernel functions like polynomial Kernel, Gaussian Kernel, Sigmoid Kernel, the exponential Kernel and the Fourier Kernel (Wang *et al.* 2015b) allow the KPLS model to control for different forms of nonlinearities (Hu *et al.* 2011). Due to these advantages, the KPLS model has been used to capture nonlinear information from chemical processes by Hu *et al.* (2013), Zhang *et al.* (2012), Zhang and Hu (2011), and Zhang and Zhang (2009) for process monitoring as well as by Jia and Zhang (2016) for fault detection. In addition, Wang *et al.* (2016), Jin *et al.* (2014), Zhang *et al.* (2010) and García-Reiriz *et al.* (2010) have adopted the KPLS model to develop soft sensors for nonlinear processes while Gao *et al.* (2015) used a modified KPLS for general modelling. On the other hand, Kernel functions were incorporated into a Locally Weighted (LW) based algorithm to control for nonlinear problems by Yuan *et al.* (2014a) for creating soft sensing and Jiang and Yan (2013) for process monitoring.

It is to be noted that researchers have not paid enough attention to applying the KPLS model for soft sensor development compared to other applications such as process monitoring and fault detection. However, the LW-PLS model is unable to analyse nonlinear process data, and an improved approach for the LW-KPLS algorithm has not been found yet. Moreover, locally weighted learning based JIT methods, including the LW-KPLS accept the dataset as complete without the missing data.

2.8. Missing data issues

The issues of missing data can be addressed by imputation methods such as deletion, replacement, and Expectation-Maximization (EM) methods. Karanja *et al.* (2013) ranked deletion, replacement, and EM methods as first, second and third generation methods, respectively. These imputation methods are commonly used for soft sensors, process monitoring and fault diagnosis to address incomplete observation problems. However, a more comprehensive review of other missing data imputation techniques were carried out by Miao *et al.* (2018), Liu and Gopalakrishnana (2017), Folch-Fortuny *et al.* (2016), Karanja *et al.* (2013), Schafer and Graham (2002), Arteaga and Ferrer (2002) and Nelson *et al.* (1996). Additional comparative studies of different imputation approaches have done by Askarian *et al.* (2016), Riggi *et al.* (2015), Folch-Fortuny *et al.* (2015), and Gómez-Carracedo *et al.* (2014).

Deletion methods such as list-wise and pairwise deletions eliminate dataset containing missing values. These deletion methods are usually performed in the pre-processing data stage before developing the soft sensors and where complete data sets are assumed (Souza *et al.* 2015; Pani and Mohanta 2011). These methods are straightforward, but a large amount of data may be lost (Xu 2016; Folch-Fortuny *et al.* 2015; Karanja *et al.* 2013) which will affect the accuracy of the predictive model (Yin *et al.* 2014; Nelson *et al.* 2006; Lopes and Menezes 2005). Moreover, the negative impact of missing measurements on the predictive models for soft sensors and process monitoring has been reported by Nelson *et al.* (2006).

In addition, replacement methods including regression imputation, interpolation replacement and mean substitution have been proposed to estimate the missing measurements using other observed values in the dataset. These replacement methods have been studied for soft sensors and process monitoring applications by many researchers : Yuan *et al.* (2016a), Folch-Fortuny *et al.* (2015), Lin *et al.* (2007), Lopes and Menezes (2005), Arteaga and Ferrer (2005), Arteaga and Ferrer (2002), and Nelson *et al.* (1996). However, Karanja *et al.* (2013) and Peugh and Enders (2004) pointed out that replacing the missing values decreases variability in the hypothetically complete dataset and leads to biased estimates. Hence, EM is recommended by Karanja *et al.* (2013) since it can reduce the impact of the lack of variability in the imputed dataset.

To date, the EM has been utilised by numerous researchers; (Xiong *et al.* 2015; Li *et al.* 2015; Junger and de Leon 2015; Gómez-Carracedo *et al.* 2014; Yin *et al.* 2014; Jin *et al.* 2012) for process sensing and monitoring to control for missing data problems. Nevertheless, limited research has been carried out to consider the missing data issue in developing adaptive soft sensors including locally weighted learning based adaptive soft sensors. Integrating the locally weighted learning based JIT method with EM to address the missing data problem has not been explored.

2.9. Research gaps

From the above review, it is observed the JIT approach has been more commonly used to develop adaptive soft sensors compared to non-JIT approaches such as the MW, recursive, TD, and ensemble methods. Among the existing JIT based adaptive soft sensors, the LW-PLS has been found to be a unique algorithm in both the literature and in industrial applications as it exhibits some advantages such as simplicity, ability to cope with high dimensionality and collinearity among the variables as well as the ability to handle outliers. However, most JIT algorithm based adaptive soft sensors, including the LW-PLS model, assume all of the observed data are complete without any missing data and have a Gaussian distribution. Missing data are inevitably present in the observed data due to transmission errors, the failure of hardware sensors and problems in accessing the database.

On the other hand, not all data behave as a Gaussian distribution. And yet, JIT algorithm based adaptive soft sensors do not simultaneously take into account the presence of non-Gaussian distributed data and incomplete measurements in their model development stage and applications. Furthermore, the existing approaches for dealing with non-Gaussianity data such as the LSSVR based, GMM based, SVDD based, the non-Gaussian dissimilarity measure based and the GPR based JIT algorithms have higher complexities and computational loads compared to the PLS based algorithm. Moreover, all the algorithms mentioned above were developed to treat complete data only. Thus, it is imperative that a new simpler algorithm, which can deal with non-Gaussian data and robust against missing data is available. This new algorithm is also expected to have low computational time as compared to its counterpart and it can solve the hardware sensors issues stated in Table 2.1.

2.10. Summary

This chapter covers the historical background of applying soft sensors, adaptive algorithms, and reviews the developed JIT-based adaptive soft sensors to handle Gaussian and non-Gaussian distributed data as well as nonlinear and missing data. Research gaps are then identified based on the limitations of the existing adaptive soft sensors. It can be seen that majority of the JIT based algorithms for adaptive soft sensors are proposed to address nonlinear and non-Gaussian distributed chemical process data. Moreover, limited work has focused on missing data issues. In reality, the process data can be highly nonlinearity, non-Gaussianity and consists of missing measurement. Less work for adaptive soft sensors has simultaneously considered these characteristics of data. Hence, these research gaps have opened the challenges for researchers to develop a new and improved algorithm for adaptive soft sensors.

Chapter 3

Research methodology

3.1. Introduction

This chapter provides an overview of the research methodology for developing new improved adaptive soft sensors, which involves Gaussian-based and non-Gaussian-based locally weighted (LW) algorithms. Firstly, the framework of algorithm formulation used to address the problem statement explained in Chapter 1 and the challenges encountered are presented. Then, the main stage of the research is described. The formulation of the novel developed Gaussian based, and non-Gaussian based LW algorithms are then briefly described. Then, case studies, predictive performance measurement, specification of computer facilities, the measurement of prediction quality, computational times and Kernel functions are also presented.

3.2. Basic idea of algorithms formulation

In view of practical applications and theoretical perspectives, there is still an opportunity for improving the existing locally weighted partial least square (LW-PLS) algorithms. As mentioned earlier, current LW-PLS algorithms have been formulated based on two main assumptions. They must contain Gaussian distributed data and a complete dataset. Therefore, it is necessary to improve the prediction performance of the LW-PLS algorithm to handle the missing data and non-Gaussian distributed data.

On the other hand, among the existing algorithms, such as the principal component analysis (PCA), partial least square (PLS) and LW-PLS, Qin (2012) has mentioned the PCA can be used to generate the principal components, the latent variables required for process modelling. However, the PCA algorithm can only model up to second-order statistics of process data that describes Gaussian distribution information. Hence, an independent component analysis (ICA), Gaussian mixture models and a support vector data description are the most commonly used methods to deal with non-Gaussian distributed data (Ge *et al.* 2013). Among these methods, the ICA is more appropriate for incorporating into the LW-PLS model since it is simple, easy to understand and can extract high-order data information.

Moreover, the ICA model, which is an extension of the PCA model, has been used by Ge and Song (2008), and Zhang *et al.* (2015) to develop JIT-based soft sensors. In their studies, two-step ICA-PCA and ICA-PLS models have been proposed to capture both Gaussian and non-Gaussian distributed data since the ICA can include higher order statistics of data in algorithms modelling. Hence, for the LW-PLS model to handle non-Gaussian distributed data, it has to be combined with an ICA model in the improvement of adaptive soft sensors. This ICA integrated algorithm should also consider its robustness against missing data since it has yet to be developed.

To cope with missing data, Karanja *et al.* (2013) have suggested using a third generation missing data imputation method, expectation maximization (EM). First generation based algorithms such as the deletion method are more likely to eliminate too many data sets and may cause data loss and lead to a larger standard of errors in the estimate if the amount of missing data is not minimized. Second-generation imputation methods such as mean and multiple linear regression-based substitutions are more robust than first-generation techniques. However, these replacement methods also reduce the accuracy of the predictive performance of the model as they reduce the variability present in the data set.

Third generation techniques, especially EM are preferable as the variability in the imputed data are optimized. Recently, EM has been used as a missing data imputation method for developing soft sensors by Li *et al.* (2015) and for producing a robust version of PLS-based soft sensors by Yin *et al.* (2014). A modified PLS based soft sensor was developed by Nelson *et al.* (1996) using a second generation missing data imputation method. Nonetheless, an integration of the EM method with the LW-PLS model can be a solution to handle the missing data problem for adaptive soft sensors.

The primary objective of this work is to propose and develop a non-Gaussian algorithm for the LW-PLS based adaptive soft sensor, which is sufficiently robust against missing measurements. This algorithm will be formulated by a simultaneous integration of EM, LW, the ensemble method and ICA algorithms into Kernel PLS (KPLS) algorithms to form the EM ensemble locally weighted independent component Kernel partial least square (EM-E-LW-IC-KPLS).

The integration of the ICA model (instead of using the PCA only in the existing algorithms) is to deal with non-Gaussian data, and the inclusion of the EM algorithm is to handle missing data. Whereas, the Kernel function is incorporated into the PLS algorithm to enhance the algorithm ability to further deal with nonlinear dynamic processes. The proposed algorithm for creating adaptive soft sensors is expected to improve process sensing and control and optimisation as well as enhancing the operational efficiency of process plants which ultimately leads to more sustainable and profitable operations. The framework used to construct a new algorithm is presented in Figure 3.1.

3.3. Main stages of the research

The main stages of this research are described in the following:

a. Developing virtual plants to generate data

Historical data used to develop and validate the soft sensor model is generated from virtual plants simulated using MATLAB Simulink or MATLAB. The virtual plants are six different case studies with nonlinear data as well as Gaussian and non-Gaussian data. Besides, different sets of data with various levels of random missing measurements ranging from 5% to 60% are generated.

b. Developing and evaluating a robust, Gaussian LW algorithm

The newly proposed Gaussian LW algorithms are formulated by incorporating Kernel functions and ensemble model into the LW-PLS algorithm to obtain the ensemble LW

Kernel PLS (E-LW-KPLS) algorithm. Different Kernel functions are employed as an internal model within PLS instead of linear regression.

c. Developing and evaluating a robust, non-Gaussian LW algorithm

The newly developed Gaussian-based LW algorithms developed is extended to handle non-Gaussian distributed data. Independent components (ICs) generated from ICA is employed in KPLS modelling to address non-Gaussianity of the data while EM algorithm is used to deal with missing measurements. The obtained algorithms are called ensemble LW independent component Kernel PLS (E-LW-IC-KPLS) and EM-E-LW-IC-KPLS.

d. Assessing and minimising potential penalty in computational time of the new algorithm

The new algorithms should not only be evaluated in term of predictive performances but also their computational efficiency by determining the central processing unit (CPU) running time of each algorithm for predicting the targeted process variable(s). Therefore, computational loads of the newly proposed and existing algorithms are assessed and compared. These comparisons are conducted by running these algorithms in MATLAB platform using the same computer to trace CPU time for data processing and predictive modelling.

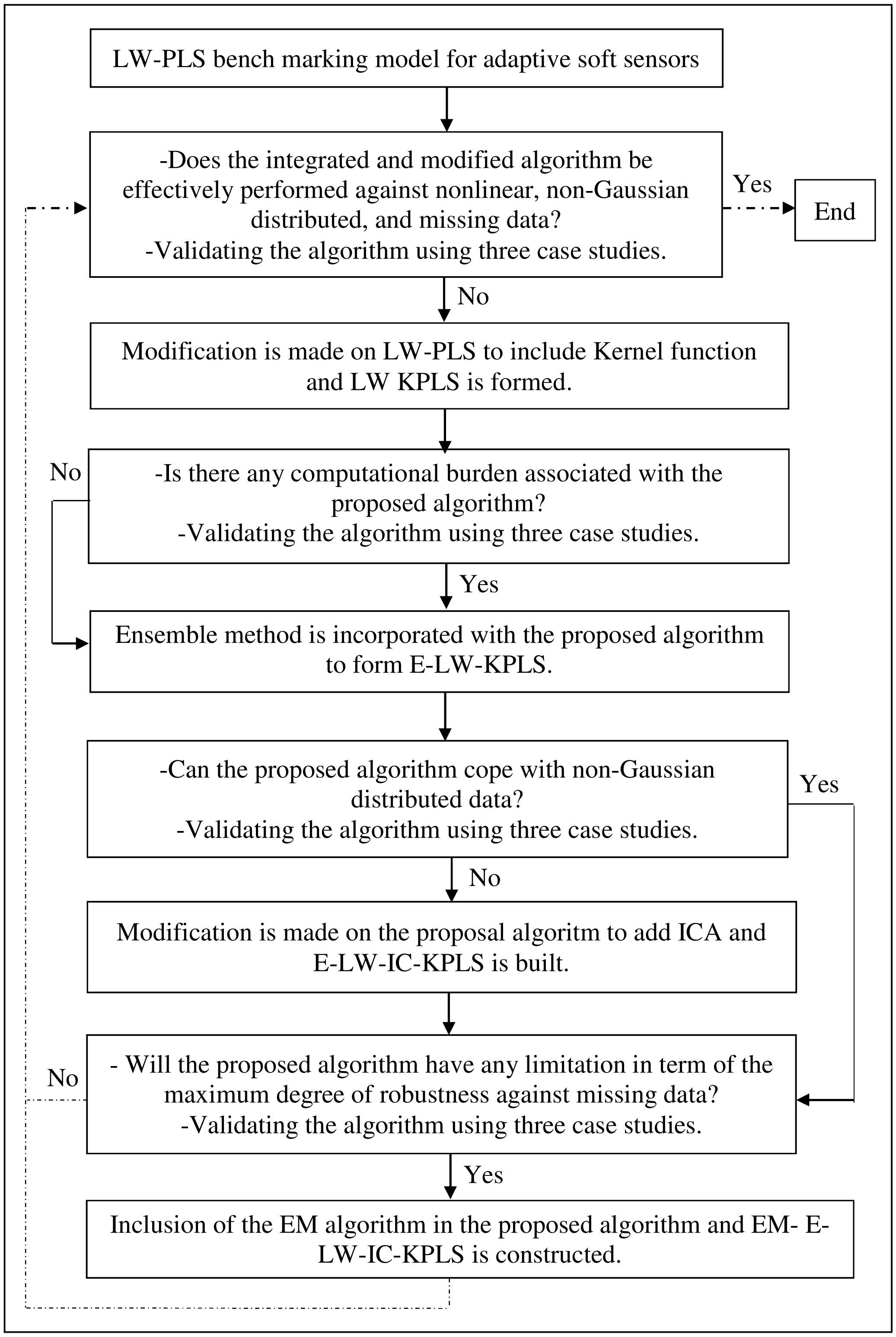
LW-PLS bench marking model for adaptive soft sensors
-Does the integrated and modified algorithm be effectively performed against nonlinear, non-Gaussian distributed, and missing data?
-Validating the algorithm using three case studies.
Yes
End
No
Modification is made on LW-PLS to include Kernel function and LW KPLS is formed.
No
-Is there any computational burden associated with the proposed algorithm?
-Validating the algorithm using three case studies.
Yes
Ensemble method is incorporated with the proposed algorithm to form E-LW-KPLS.
-Can the proposed algorithm cope with non-Gaussian distributed data?
-Validating the algorithm using three case studies.
Yes
No
Modification is made on the proposal algoritm to add ICA and E-LW-IC-KPLS is built.
No
- Will the proposed algorithm have any limitation in term of the maximum degree of robustness against missing data?
-Validating the algorithm using three case studies.
Yes
Inclusion of the EM algorithm in the proposed algorithm and EM- E-LW-IC-KPLS is constructed.

Figure 3.1 A flow chart represents the framework used to construct a new algorithm

3.4. Description of algorithms formulation

A Gaussian-based LW algorithm is developed to deal with missing data and nonlinear data while a non-Gaussian-based LW algorithm is constructed to cope with non-Gaussian data. These LW algorithms are briefly explained in the following subsections.

3.4.1. Gaussian based locally weighted algorithms

A Gaussian-based LW algorithm is a combination of the LW-PLS model, the ensemble method and the Kernel function, where the Kernel function is utilized to deal with nonlinear data. In the first instance, the integration of Kernel function, the ensemble method, and the LW-PLS are done to form the E-LW-KPLS model. This new and modified algorithm is an improved algorithm for nonlinear processes. Meanwhile, it has lower computational time compared to LW Kernel PLS (LW-KPLS) model. Next, the predictive performance of this new Gaussian-based LW algorithm, the E-LW-KPLS, is evaluated using various case studies. The E-LW-KPLS model is applied to the nonlinear process, and the results are compared with the LW-PLS and LW-KPLS models. The detailed formulation about these Gaussian-based LW algorithms is shown in Chapter 4 while the results and discussion about its predictive performance in different case studies are given in Chapter 6.

3.4.2. Non-Gaussian based locally weighted algorithms

This E-LW-KPLS model is then integrated with the ICA algorithm to form a non-Gaussian-based LW algorithm, called the EM-E-LW-IC-KPLS. In this integrated algorithm, the ICA model is employed to handle non-Gaussian distributed data. Initially, the ICA is incorporated with the E-LW-KPLS model to create the locally weighted independent component Kernel partial least square (LW-IC-KPLS) model and the ensemble LW independent component Kernel PLS (E-LW-IC-KPLS) model. After that, the EM is combined with the E-LW-IC-KPLS model to build the EM-E-LW-IC-KPLS model.

The E-LW-IC-KPLS is the associated outcomes of the EM-E-LW-IC-KPLS, and it is a new and improved algorithm. This new adaptive algorithm utilised to formulate an adaptive soft sensor, able to analyse nonlinear and non-Gaussian distributed data. On the other hand, the newly developed algorithms including E-LW-IC-KPLS and EM-E-LW-IC-KPLS are additional approaches to develop an adaptive soft sensor. They are the more accurate alternative to the existing algorithms since it can address the nonlinear and non-Gaussian distributed data as well as with the presence of missing data simultaneously. Meanwhile, EM-E-LW-IC-KPLS has been developed to address the research gaps of the existing algorithms.

In the later stage of the study, these three newly formulated algorithms, the LW-IC-KPLS, E-LW-IC-KPLS, and EM-E-LW-IC-KPLS models have been applied to various case studies to investigate their predictive accuracy. Notably, the predictive performance of the E-LW-IC-KPLS model is distinguished from other algorithms such as the LW-PLS, LW-KPLS, and the LW-IC-KPLS. Moreover, the EM-E-LW-IC-KPLS model is applied to process data consisting of missing data. More information about the development of these developed non-Gaussian based LW algorithms is illustrated in Chapter 5.

In addition, these algorithms are applied in different case studies, and their results and discussion are demonstrated in Chapter 6. Besides, for case studies with nonlinear and non-Gaussianity, results of LW-PLS, LW-KPLS, LW-IC-KPLS, and E-LW-IC-KPLS are evaluated and compared. Furthermore, for the case studies with missing data, the results of EM-E-LW-IC-KPLS are compared to another popular missing data imputation method, the trimmed score regression (TSR) and singular value decomposition (SVD) that are also integrated with the E-LW-IC-KPLS model to form TSR E-LW-IC-KPLS (TSR-E-LW-IC-KPLS) and SVD E-LW-IC-KPLS (SVD-E-LW-IC-KPLS) models. The TSR has a good agreement in the results as demonstrated by Folch-Fortuny *et al.* (2016a), Folch-Fortuny *et al.* (2016b), Folch-Fortuny *et al.* (2015) and Arteaga and Ferrer (2002) while SVD is also a popular missing data imputation method. Besides, SVD and TSR models are also integrated with the benchmark LW-PLS model, and the results are compared with EM-E-LW-IC-KPLS.

3.5. Case studies

To evaluate the predictive performance of the developed Gaussian- and Non-Gaussian-based LW algorithms, process data generated from six simulated plants are used. These simulated plants including numerical example 1 (case study 1), a single chemical reactor (case study 2), wastewater treatment (case study 3), numerical example 2, a static approximation of two sine waves (case study 4), Eukaryotic cell cycle regulation (case study 5) and a highly nonlinear continuously stirred tank reactor (CSTR) (case study 6) were produced using the MATLAB Simulink. Historical data of these simulated plants are utilised to validate the developed models. The simulated data in case studies 1 to 3 are nonlinear, and Gaussian distributed data while case studies 4 to 6 are nonlinear, and non-Gaussian distributed data. The non-Gaussian distributions in these case studies include bimodal distribution, multimodal distribution, plateau distribution and exponential distribution (Weaver *et al.* 2018). Then, the simulated data in case studies 4 to 6 are further treated to generate different levels of random missing measurements ranging from 5% to 60%. Table 3.1 summaries the characteristics of data in the case studies.

3.6. Predictive performance measurement

To assess how well the developed LW algorithms predict an output in each case study, the standard measure of prediction errors including root mean square error (*RMSE*), mean absolute error (*MAE*), and the error of approximation (E_a) are considered. These predictive errors are applied to both training and test data. The lower values of *RMSE*, *MAE*, and E_a indicate that the predictions of algorithms are better while higher values denote more significant prediction errors. Meanwhile, the amount of time used to execute the algorithms determines their computational complexity. Hence, the CPU times used for an algorithm in each case study are calculated. The overall predictive performance of an algorithm is evaluated using its *RMSE*, *MAE*, E_a and t (computational time). However, the main performance measurement used to compare the different algorithms is E_a. Then, the results of each developed algorithm are evaluated and compared to distinguish their effectiveness.

To measure and compare the prediction accuracy of the models, several predictive errors are used in this work. There is a possible network where the error of the training set is smallest, however, the error of the test set is large (Dou *et al.* 2005). This form of network is unsteady when it is utilized to predict an unknown sample.

To prevent this type of situation, a new evaluation criterion of the network, the error of approximation, E_a is used. The description of this criterion, taken from Saptoro *et al.* (2006) is given in Equations (3.1), and (3.2) as below:

$$E_a = \left(\frac{N_1}{N}\right) RMSE_1 + \left(\frac{N_2}{N}\right) RMSE_2 + \left| RMSE_1 - RMSE_2 \right| \tag{3.1}$$

where $RMSE_1$ and $RMSE_2$ are the root mean square errors of the training and the testing datasets. $RMSE_1$ and $RMSE_2$ are used to evaluate the ability of a model to fit data and its predictive power (Zhang *et al.* 2010). The equation of root mean square error, *RMSE* is shown as below:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \tag{3.2}$$

where N, N_1, and N_2 are the total number of samples, the number of training and testing datasets, respectively. In addition, y_i and $\hat{y}_i$ are the real and estimated values of the output. According to Dou *et al.* (2005) and Saptoro *et al.* (2006), a smaller E_a indicates more models are close to the real nature of the dataset. Hence, the effects of both training and test sets are examined using this evaluation criterion (Dou *et al.* 2005).

Moreover, the mean absolute error (MAE) and mean square error (MSE) are also used to evaluate the performance of predictive models. The MAE and MSE for both training and test data are also calculated. They are shown as below:

$$MAE = \frac{| y_i - \hat{y}_i |}{N} \tag{3.3}$$

$$MSE = \frac{| y_i - \hat{y}_i |^2}{N} \tag{3.4}$$

E_a, *RMSE*, *MAE* and *MSE* are utilized to evaluate the accuracy of the model. The smaller their values, the higher the accuracy of the model will be.

To calculate the centre processing unit (CPU) or the computational time used to run the developed model, *t*, a simple formula as below is used. The computation times of the training and test data, which are t_1 and t_2, respectively, are measured through the determination of the length of time required to perform a computational process.

$$t = \text{stop time - start time} \tag{3.5}$$

Table 3.1 Summary of the charactheristics of data and Kernel functions used in all case studies

Type of data	Nonlinear and Gaussian distributed data			Nonlinear and non-Gaussian distributed data		
Case study	Case study 1	Case study 2	Case study 3	Case study 4	Case study 5	Case study 6
Name of the case study	Numerical example 1	A single chemical reactor	Wastewater treatment	Numerical example 2	Eukaryotic cell cycle regulation	A highly nonlinear CSTR
Description	A nonlinear test problem	A well-mixed, non-isothermal continuous stirred tank reactor (CSTR)	An aerated bioreactor with a settler	A static function approximation of the sum of two sine waves	A biochemical reaction in a cell cycle to control frog egg development	An acid-catalyzed electrophilic hydration in aqueous using CSTR with a cooling jacket
Used Kernel functions	Inverse multi-quadric	Polynomial	Inverse multi-quadric	Log	Log	Log
Input 1 [Min, Max, Mean, Std]*	x_1 Min: 1.0994×10^{-4} Max: 87.6862 Mean: 18.0303 Std: 13.5182	Concentration of the reactant species, CA Min: 1.7770 Max: 4	Recycled biomass concentration, X_r Min: 320	x Min: 0 Max: 10 Mean: 4.9950 Std: 2.8870	Dimensionless concentration of active M-phase promoting factor, u	Scaled volumetric inlet flow, $\frac{V}{V_R}$ Min: 3.0044 Max: 34.9950

		Mean: 1.7975 Std: 0.0624	Max: 576.4784 Mean: 547.1739 Std: 49.8538		Min: 0 Max: 0.2393 Mean: 0.0579 Std: 0.0682	Mean: 19.0443 Std: 9.2830
Input 2 [Min, Max, Mean, Std]*	x_2 Min: 1.0994x10^{-4} Max: 87.6862 Mean: 18.0303 Std: 13.5182	Temperature of the reactor, T_1 Min: 300 Max: 458.2479 Mean: 455.1939 Std: 2.9283	Biomass concentration, X_B Min: 200 Max: 290.0542 Mean: 276.2885 Std: 19.4681	-	-	Heat flow, $\dot{Q}_k$ Min: -9x10^3 Max: -0.2777 Mean: -4.5018 x10^3 Std: 2.6228 x10^3
Input 3 [Min, Max, Mean, Std]*	-	-	Substrate concentration, S_s Min: 10.2489 Max: 88 Mean: 15.2548 Std: 4.1475	-	-	-

Output 1 [Min, Max, Mean, Std]*	y Min: -1.3407 Max: 1.3112 Mean: -5.2603x10^{-4} Std: 0.7160	Concentration of the desired product, CB Min: 0 Max: 1.8126 Mean: 1.7924 Std: 0.0528	Dissolved oxygen concentration, DO Min: 1.5971 Max: 7.8571 Mean: 6.0636 Std: 0.8113	y Min: -1.1518 Max: 1.1462 Mean: -3.0194E-4 Std: 0.5097	Dimensionless concentration of total cyclin, v Min: 0 Max: 0.4874 Mean: 0.3673 Std: 0.0857	Product concentration, CP Min: 1.0033 Max: 1.0925 Mean: 1.0591 Std: 0.0140
Output 2 [Min, Max, Mean, Std]*	-	-	-	-	-	Reactor temperature, T_2 Min: 386.0555 Max: 387.35 Mean: 386.0578 Std: 0.0132
Type of non-Gaussian distribution	-	-	-	Multimodal	Exponential, bimodal	Multimodal

* Min = minimum values; Max = Maximum values; Std = Standard deviation

3.7. Specifications of computing facilities

The configuration of the computer software used in this study is explained in this section. The software and hardware specification are stated as below:

Operating system (OS): Windows 10 (64 bit), CPU: 2.20 GHz Intel Core M3-6Y30 CPU processor (it has the same performance as Intel Core i5 6200U), 4.0 Gigabyte (GB) of Random Access Memory (RAM) and 128GB solid-state drive (SSD) storage. The version of MATLAB used is 2015a.

3.8. Kernel functions

As mentioned earlier in Chapter 4, the Kernel method is used for nonlinear problems to map data into higher dimensional spaces *F*. Then, the mapped data can be more effectively structured before developing local models. The nonlinear mapping can be done via the following inner product in feature space *F*, which is called the Kernel function (Jia and Zhang 2016).

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{3.6}$$

where $\phi(x_i)$ and ϕ (x_j) are *i*-th and *j*-th row vectors respectively. In this study, several different Kernel functions, obtained from Kamath *et al.* (2010) are used in the case studies.

Inverse multi-quadricKernel function:

An inverse multi-quadric Kernel function with a Kernel parameter squared, b^2 greater than zero is shown in Equation (3.7). This function allows the creation of non-singular matrices (De Marchi 2013).

$$k(x_i, x_j) = \frac{1}{\sqrt{\left\|x_i - x_j\right\|^2 + b^2}} \tag{3.7}$$

Polynomial Kernel function:

A polynomial Kernel is a non-stationary Kernel often used in Kernelized nonlinear models such as a support vector and least square support vector machines. It represents the data features as polynomial expansions up to an index *b*. Equation (3.8) is the polynomial Kernel function taken from Wang *et al.* (2015b).

$$k(x_i, x_j) = \left(x_i^T x_j + 1\right)^b \tag{3.8}$$

where *b* is the order of the polynomial and is also a Kernel parameter.

Log Kernel function:

A Log Kernel is usually used in an image database and is only conditionally positive definite. The Log Kernel function is displayed as below:

$$k(x_i, x_j) = -\log\left(\left\|x_i - x_j\right\|^b + 1\right) \tag{3.9}$$

where *b* is called the Kernel parameter.

3.9. Summary

This chapter describes the framework of new algorithms that can cope with nonlinear, non-Gaussian distributed and missing data. Moreover, the main stages of research that were used to address the research questions while achieving the research objectives are also illustrated. Then, brief descriptions on the newly developed algorithms for both Gaussian-based and non-Gaussian-based locally LW algorithms are also carried out. Besides, case studies, the measurement used to evaluate the predictive performance of these developed algorithms, the computing facilities, the measurement of prediction quality, computational times and Kernel functions are presented too. In the next chapter, details of the formulation of Gaussian-based LW algorithms are presented.

Chapter 4

Gaussian based locally weighted algorithms

4.1. Introduction

In this chapter, the newly developed Gaussian-based locally weighted (LW) algorithm which is an ensemble locally weighted Kernel partial least square (E-LW-KPLS) model is described. Firstly, the similarity measurement used in the Gaussian-based LW algorithm is presented. Next, the locally weighted partial least square (LW-PLS) algorithm is also described since the novel Gaussian-based LW algorithm is an extension of the LW-PLS. Illustrations of the Kernel partial least square (KPLS) model flow and lastly, this novel E-LW-KPLS model is presented.

4.2. Similarity measurement

The original Gaussian-based LW algorithm consists of the LW-PLS; hence a similarity measurement is also required in the proposed algorithm. The predictive performance of the LW-PLS is strongly influenced by the selection of the similarity measurement (Hazama and Kano 2015). Therefore, the similarity measurement must be well nominated. In this research, the common similarity measurement, Euclidean distance is used due to its simplicity (Zhang *et al.* 2015; Saptoro 2014; Jin *et al.* 2014; Xie *et al.* 2014). The Euclidean distance-based similarity index, ω_n is determined based on the distance between query, X_q and historical data, X_n. The distance-based similarity index obtained from Ma *et al.* (2015) is shown in the following equations:

$$\omega_n = \exp(-\frac{d_n}{\varphi\sigma_n}) \tag{4.1}$$

$$d_n = \sqrt{(x_n - x_q)^T (x_n - x_q)} \tag{4.2}$$

where σ_d is the standard deviation of d_n (n= 1, 2,…,N) and φ is a localization parameter. To handle the nonlinear relationship between input and output variables, the value of φ should be low. However, when it has a shallow value of φ, the sensitivity of the LW-PLS towards noise is going to be increased (Kim *et al.* 2013b). In addition, the predictive performance of the LW-PLS model is similar to the partial least square (PLS) model when φ equals to infinite (Toshiya and Kano 2015; Kim *et al.* 2011). Therefore, φ must be appropriately adjusted to ensure the prediction accuracy of the LW-PLS model is superior to the PLS. Usually, the value of φ is within the range of 0 to 10 (Uchimaru and Kano 2016).

4.3. Locally weighted partial least square algorithm

When the number of input variables is larger than the neighbouring samples, the LW regression becomes problematic (Toshiya and Kano 2015). In this case, a just-in-time based algorithm, the LW-PLS, can be a suitable alternative. The LW-PLS not only can deal with a large number of input variables but also with outliers (Hazama and Kano 2015; Kaneko and Funatsu 2015a) and collinearity among the input and output variables (Kano and Koichi 2013). Due to its advantages, the LW-PLS based soft sensors have become popular and have been utilized in biochemical, semiconductor, petrochemical, and pharmaceutical processing industries (Kano and Koichi 2013; Hazama and Kano 2015; Toshiya and Kano 2015; Kim *et al.* 2013b; Kim *et al.* 2013a; Nakagawa *et al.* 2012).

In creating the LW-PLS model, a local PLS model is built by using the similarity between a query which is new data and historical data (Kim *et al.* 2011; Kano and Koichi 2013). The following description of the LW-PLS algorithm is adopted from Hazama and Kano (2015). The input and output variables, x and y respectively, for n number of samples can be expressed as:

$$x_n = [x_{n1}, x_{n2}, \ldots, x_{nM}]^T \tag{4.3}$$

$$y_n = [y_{n1}, y_{n2}, ..., y_{nL}]^T \tag{4.4}$$

where *M* and *L* are numbers of *x* and *y*, respectively. These *x* and *y* variables are kept in a historical database. The similarity index, ω_n between a query, X_q and the *k*-th historical data, X_k are measured to develop a local PLS model when the prediction of output variable, $\hat{y}_q$ is required for X_q. Detailed explanations of the LW-PLS model can be found in Nakagawa *et al.* (2012), Kim *et al.* (2013b) and Nakagawa *et al.* (2014).

The predicted output $\hat{y}_q$ is determined by using the LW-PLS algorithm through the following steps:

1. Fix the number of latent variables *K* and set *k* = 1.
2. Determine a similarity matrix Ω using Equations (4.1), (4.2) and (4.5).

$$\Omega = dig\{\omega_1, \omega_2, ..., \omega_N\} \tag{4.5}$$

3. Calculate X_k, Y_k, and $Xq_{,k}$ where 1_N is a vector of ones using Equations (4.6) to (4.10).

$$X_k = X - 1_N \left[\bar{X}_1, \bar{X}_2, ..., \bar{X}_M \right] \tag{4.6}$$

$$Y_k = Y - 1_N \left[\bar{Y}_1, \bar{Y}_2, ..., \bar{Y}_L \right] \tag{4.7}$$

$$X_{q,k} = X_q - 1_N \left[\bar{X}_1, \bar{X}_2, ..., \bar{X}_M \right]^T \tag{4.8}$$

$$\bar{X}_m = \frac{\sum_{n=1}^{N} \omega_n X_{nm}}{\sum_{n=1}^{N} \omega_n} \tag{4.9}$$

$$\bar{Y}_l = \frac{\sum_{n=1}^{N} \omega_n Y_{nl}}{\sum_{n=1}^{N} \omega_n} \tag{4.10}$$

4. Set $\hat{y}_q = [\bar{y}_1, \bar{y}_2, ..., \bar{y}_L]^T$.

5. Derive the *k*-th latent variable of X_k using Equations (4.11) and (4.12).

$$t_k = X_k w_k \tag{4.11}$$

where w_k is the Eigenvector of $X_k^T \Omega Y_k Y_k^T \Omega X_k$, which corresponds to the maximum Eigenvalue, and is derived by:

$$w_k = \frac{X_k^T \Omega Y_k}{\left\| X_k^T \Omega Y_k \right\|} \tag{4.12}$$

6. Derive the *k*-th loading vector of X_k and the *k*-th regression coefficient vector.

$$p_k = \frac{X_k^T \Omega t_k}{t_k^T \Omega t_k} \tag{4.13}$$

$$q_k = \frac{Y_k^T \Omega t_k}{t_k^T \Omega t_k} \tag{4.14}$$

7. Derive the *k*-th latent variable of X_q.

$$t_{q,k} = X_{q,k}^T w_k \tag{4.15}$$

8. Replace $\hat{y}_q$ with $\hat{y}_q + t_{q,k} q_k$.

9. If $k = K$, then finish prediction. Otherwise, set

$$X_{k+1} = X_k - t_k p_k^T \tag{4.16}$$

$$Y_{k+1} = Y_k - t_k q_k^T \tag{4.17}$$

$$X_{q,k+1} = X_{q,k} - t_{q,k} p_k \tag{4.18}$$

10. Set $k = k + 1$ and go to Step 5.

4.4. Kernel partial least square algorithm

Kernel approaches are commonly used to solve nonlinear problems since they provide good performance in a majority of real-world applications (Zhang and Hu 2011). Thus, for PLS based soft sensors, Kernel functions were incorporated into the PLS model to enhance their predictive performance in controlling for nonlinear data. With the help of Kernel functions, the KPLS maps original observed data into a high dimensional feature space, *F* and then the linear PLS models are constructed in the feature space, *F*. It is assumed there is a nonlinear transformation of independent variables, x_i, $i = 1, 2, \ldots, n$ into feature space *F* and this can be represented by Equation (4.19).

$$\phi : x_i \in R^n \rightarrow \phi(x_i) \in F \tag{4.19}$$

where $\sum_{i=1}^{n} \phi(x_i) = 0$ is assumed and $\phi(x_i)$ is a nonlinear mapping function that projects the input vectors from the original space to the feature space *F*. Regardless, the dimensionality of the feature space *F* is arbitrarily large and can be infinite. The order of matrix $\phi(x_i)$ is $n \times M$ where the *n*-th row is the vector in the *M*-dimensional feature space *F*. Using the introduction of the Kernel trick, $\phi(x_i)^{\mathrm{T}}\phi(x_j) = \mathrm{K}(x_i, x_j)$, one can prevent both performing explicit nonlinear mapping and computing dot products in the feature space. Steps for the KPLS model from Gao *et al.* (2015) are shown as follows:

K is the number of latent variables ($i = 1, 2, \ldots, K$) and the below steps are repeated.

1. Initialise, set $K_i = K$, $Y_i = Y$, set u_i equal to any column of Y_i.
2. Compute the score vector of $\phi(X)$ using Equation (4.20).

$$t_i = \frac{K u_i}{\sqrt{u_i^{T} K_i u_i}} \tag{4.20}$$

3. Compute the loading vector of Y_i using Equation (4.21).

$$q_i = \frac{Y_i t_i}{\left\| t_i^{T} t_i \right\|} \tag{4.21}$$

4. Compute the score vector of Y_i using Equation (4.22).

$$u_i = \frac{Y_i q_i}{q_i^T q_i} \tag{4.22}$$

5. If u_i converges, then go to Step 6, otherwise return to Step 2.
6. Calculate the residual of $\phi(X)$ and Y using Equations (4.23) and (4.24).

$$K_{i+1} = (I - t_i t_i^T / t_i^T t_i) K_i (I - t_i t_i^T / t_i^T t_i) \tag{4.23}$$

$$Y_{i+1} = (I - t_i t_i^T / t_i^T t_i) Y_i \tag{4.24}$$

7. Then, $i = i + 1$, and go to Step 2.

The regression coefficient, b_k in the KPLS model is determined from Equation (4.25) after all the K latent variables are extracted.

$$b_k = \phi^T U \left(T^T K U\right)^{-1} T^T Y \tag{4.25}$$

where $T = [t_1, t_2, \ldots, t_H]$ and $U = [u_1, u_2, \ldots, u_H]$ are the score matrices. Therefore, when the number of test data is N_2, the prediction on training data and test data can be made as below, respectively:

$$\hat{Y}_{train} = \phi b_k = KU \left(T^T KU\right)^{-1} TY \tag{4.26}$$

$$\hat{Y}_{test} = \phi_{test} b_k = K_{test} U \left(T^T KU\right)^{-1} T^T Y \tag{4.27}$$

where ϕ_{test} is the matrix of the mapped test data and K_{test} is the $N_2 \times M$ test Kernel matrix whose elements are $K_{test}(i, j) = K(x_i, x_j)$, where x_i is the i-th test vector and x_j is the j-th training vector.

The mean centering of the data should be done before performing the KPLS model (Schölkopf *et al.* 1998). The Kernel matrices K_i and K_{test} are substituted with $\tilde{K}_i$ and $\tilde{K}_{test}$, where

$$\tilde{K} = \left(I - \frac{1}{M} 1_M 1_M^T \right) K \left(I - \frac{1}{M} 1_M 1_M^T \right) \tag{4.28}$$

$$\tilde{K}_{test} = (K_{test} - \frac{1}{M} 1_{N_2} 1_M^T K)(I - \frac{1}{M} 1_M 1_M^T) \tag{4.29}$$

where I is an M-dimensional identity matrix, 1_M and 1_{N_2}, are vectors whose elements are all ones, with length M and N_2, respectively.

4.5. Ensemble locally weighted Kernel partial least square algorithm

As mentioned earlier, the LW-PLS can achieve adequate performance in industrial applications. However, it still may not fulfil the prediction accuracy requirement for nonlinear processes. In this research, the E-LW-KPLS is introduced to handle this issue. First, the modification is made on the LW-PLS model to have the ability to map the original variables into high dimensional space using the Kernel function. And, locally weighted models are still built using the same procedure as the LW-PLS model. Similar to the method of the LW-PLS model, before constructing a locally weighted model, this new high dimension feature variable vectors, $\phi(x)$ should also be weighted-mean centered. Besides, the second modification is made using ensemble method to slipt the data into numerous of LW-KPLS models that can run simultaneously. So that, the computational time for the LW-KPLS model can be minimised. It is assumed that simplicity is fulfilled and therefore the E-LW-KPLS model can be formulated as shown in this section.

Since original variables are mapped into a Kernel-defined feature space, correlations between allocated input variables and predicted output variables must be determined. This mapping can be done by performing a dual KPLS model discrimination to measure the projection direction in the Kernel-defined feature space containing the maximum amount of variance in the data. This discrimination is done by obtaining a dual representation of a scaled version of projection direction, where *B* moved a Kernel matrix, *V*. According to Shawe-Taylor and Cristianini (2004), this can be calculated by Equation (4.30). A detailed explanation and MATLAB coding for Equation (4.30) can be found in Shawe-Taylor and Cristianini (2004).

$$B = YY'V\beta \text{ with the normalisation, } \beta = \frac{\beta}{\|\beta\|} \tag{4.30}$$

The presence of Kernel function increases the computational complexity of the LW-KPLS dramatically. The ensemble method is utilised to assign and divide the original input dataset into different sub-datasets with the same sample size to overcome the computational burden problem. Ensemble method is to form local models which are a number of LW-IC-KPLS models in this work that are built from the corresponding original variables subsets (Jin *et al.* 2016). These sub-datasets are then sent to the LW-KPLS models, and these models are calculated simultaneously. Then, the multiple output data from these models are combined. Each of the predicted process output data is later combined to calculate the overall root mean square error of prediction. Additionally, the ensemble method also can enhance the generalization performance of each learning machine (He *et al.* 2016). Figure 4.1 shows the flow diagram of the E-LW-KPLS model proposed in this work.

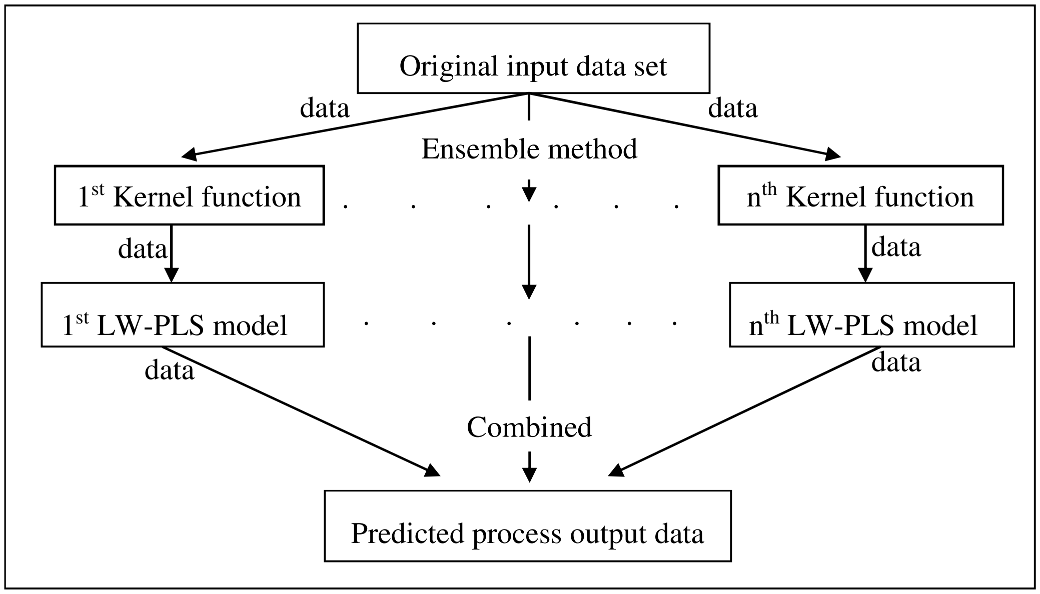

Figure 4.1 Flow diagram of the proposed E-LW-KPLS model

The predicted output, $\hat{y}_q$ which is calculated through each of the LW-KPLS models, is shown as the following procedure:

1. Set $V = x$ as the input data for the training data for Kernel matrix and $V_q = X_q$ as the input data for the query/ testing data where they are mapped into a higher dimensional feature space using a selected Kernel function.
2. Conduct mean centering on the V and V_q using Equations (4.28) and (4.29), respectively.
3. Perform dual kernel partial least square discrimination to obtain B using Equation (4.30).
4. Compute the rescaled query, and input variable matrices, V_q, and V using Equations (4.31) and (4.32), respectively.

$$X_q = V_q B \tag{4.31}$$

$$X = VB \tag{4.32}$$

5. Determine the number of latent variables K and set $k = 1$.

6. Calculate a similarity matrix Ω using Equations (4.1), (4.2) and (4.5).

7. Calculate X_k, Y_k, and $X_{q,k}$ using Equations (4.6) to (4.10).

8. Set $\hat{y}_q = [\bar{y}_1, \bar{y}_2, ..., \bar{y}_L]^T$.

9. Derive the k-th latent variable of X_k using Equations (4.11) and (4.12).

10. Derive the k-th loading vector of X_k and the k-th regression coefficient vector using Equations (4.13) and (4.14).

11. Derive the k-th latent variable of X_q using Equation (4.15).

12. Replace $\hat{y}_q$ with $\hat{y}_q + t_{q,j} q_j$ where $t_{q,k}$ is the k-th latent variable of X_q.

13. If $k = K$, then finish prediction. Otherwise, set Equations (4.16) to (4.18).

14. Set $k = k + 1$ and go to Step 9.

4.6. Summary

This chapter mainly provides the details explanation of the newly modified Gaussian-based LW algorithm, which is called E-LW-KPLS. Besides, the existing Euclidean distance based similarity measure, as well as the current algorithms LW-PLS, and KPLS are also presented since they are preliminary studies adopted as base cases to the newly proposed algorithm. These initial studies provided some useful frameworks for the development of the E-LW-KPLS algorithm. In the next chapter, Chapter 5, the extended Gaussian-based LW algorithms which are non-Gaussian LW algorithms are illustrated.

Chapter 5

Non-Gaussian based locally weighted algorithms

5.1. Introduction

In this chapter, the two non-Gaussian-based locally weighted (LW) algorithms, which are extended algorithms for Gaussian-based LW algorithms and are described in Chapter 4 are demonstrated. They are the ensemble LW independent component Kernel partial least square (E-LW-IC-KPLS) model and the expectation maximization ensemble LW independent component kernel partial least square (EM-E-LW-IC-KPLS) model. The independent component analysis (ICA) is incorporated into Gaussian-based LW algorithms to cope with non-Gaussian distributed data. The similarity measurement used in the non-Gaussian-based LW algorithms is also a Euclidean distance since it is simple and easy to use. In the following sections, the existing ICA is discussed first then followed by the proposed E-LW-IC-KPLS, expectation maximization (EM), and EM-E-LW-IC-KPLS algorithms.

5.2. Independent component analysis

The proposed Gaussian-based LW algorithms utilize principal component analysis (PCA) to extract latent variables in which PCA only imposes principal components up to second order statistics information (Ge *et al.* 2013). That means PCA only considers the mean and variance-covariance of data (Lee *et al.* 2006). When the process data involves higher order statistical information that is non-Gaussian distributed data, these Gaussian-based LW algorithms perform poorly. On the other hand, ICA breaks down the observed data into linear combinations of statistically independent components (ICs), which are decorrelated from each other and exhibit decreased high-order statistical dependencies (Jiang and Yan 2013; Lee *et al.* 2006; Lee 2000). Compared to PCA, ICA may reveal more meaningful information in the non-Gaussian distributed data (Ge *et al.* 2013; Shao *et al.* 2006). More detailed descriptions of ICA can be found in Hyvärinen (2013), Naik and Kumar (2011), Hyvärinen *et al.* (2004), Hyvärinen and Oja (2000), and Lee (2000).

To handle the non-Gaussianity of data, ICA has been employed by Tong *et al.* (2016), Peng *et al.* (2016), Chen *et al.* (2016), Wang *et al.* (2015a), Jiang *et al.* (2014), Ge *et al.* (2014), Song *et al.* (2012) and Lee *et al.* (2006) for use in soft sensors, process monitoring, fault detection, and diagnosis methods. In this study, a fixed point ICA is utilised since it is simple and provides high efficiency (Peng *et al.* 2016; Lee *et al.* 2006; Hyvärinen and Oja 2000). The below description of the ICA algorithm is taken from Hyvärinen and Oja (2000).

In the ICA algorithm, the observed variables $x_1, x_2, \ldots, x_n$ can be expressed as linear combinations of c ICs $S_1, S_2, \ldots, S_c$ where $c<n$. It is assumed the observed variables and ICs have zero mean. Let designate the observed variables and ICs in column vectors, which are $x = [x_1, x_2,\ldots, x_n]^T$ and $S = [S_1, S_2, \ldots, S_c]^T$. Their relationship can be represented by the ICA model as follows:

$$x = As \tag{5.1}$$

where $A= [a_1, a_2,\ldots, a_m] \in R^{nxc}$ is the mixing matrix. Equation (5.1) describes a process of mixing the ICs generates the observed variables. In other words, A and S can only be estimated based on the observed variable(s), x. To simplify the estimation, A is assumed to be an unknown square matrix where $n = c$ and S has unit variance: $E(SS^T) = I$. Then, S can be estimated from the inverse of A, W which is the so-called 'demixing matrix' (Equation (5.2)).

$$s = Wx \tag{5.2}$$

In the ICs, S is estimated through the following procedure:

1. Apply whiten transformation to remove all the cross-correlation between random variables using Equation (5.3) in which the over-sphered zero-mean vector, z is calculated.

$$z = \Lambda^{-\frac{1}{2}} U^T x = Qx \tag{5.3}$$

where $Q = \Lambda^{-\frac{1}{2}} U^T$ is the whitening matrix, and Λ (the diagonal matrix of its Eigenvalues) and U (the orthogonal matrix of eigenvectors) are obtained from the Eigenvalue decomposition of the covariance matrix $E(xx^T) = U\Lambda U^T$. After the transformation, Equations (5.4) and (5.5) can be obtained.

$$z = Qx = QAs = B_c s \tag{5.4}$$

$$s = B_c^T z = B_c^T Qx \tag{5.5}$$

where $B_c = QA$ is the orthogonal matrix and is given as $E(zz^T) = B_c(E(ss^T))B_c^T = B_c B_c^T = I$. From Equations (5.2) and (5.5), the relationship of W and B_c can be expressed as:

$$W = B_c^T Q \tag{5.6}$$

2. Determine B_c, subject to Equation (5.5).
3. Calculate W using Equation (5.6) in order to estimate s via Equation (5.5).

Since each column vector in B_c is randomly initialed, they must be updated in Step 2 until the ICs have maximum non-Gaussianity. Commonly, Kurtosis or the fourth-order cumulant and negentropy can be used to measure non-Gaussianity (Hyvarinen 1999). Since negentropy is computationally very difficult, Kurtosis is the chosen method in this study and can be defined as (Hyvärinen and Oja 2000):

$$kurt(s) = E\{s^4\} - 3(E\{s^4\})^2 \tag{5.7}$$

5.3. Ensemble locally weighted independent component Kernel partial least square algorithm

To further enhance the predictive performance of the developed Gaussian-based LW algorithm, the ensemble locally weighted Kernel partial least square (E-LW-KPLS) is integrated with ICA to form the E-LW-IC-KPLS algorithm. In this regard, ICA is used to deal with non-Gaussian distributed data; Using ICA, informative data is extracted from non-Gaussian data before they are mapped to higher dimensional space using Kernel function. In the E-LW-IC-KPLS model, original input variables are divided into separate sub-datasets with similar sample size via the ensemble method and these data are executed in the multiple proposed models, which are LW-IC-KPLS models concurrently.

Each of these sub-datasets is first mapped into high dimensional space using the Kernel function. Subsequently, these mapped sub-datasets are sent to ICA models to extract the ICs consisting of non-Gaussian information. Next, the ICs are used to predict process output data using LW-PLS models. Lastly, each of the anticipated process output data from the sub-models is combined to calculate the overall root mean square error of prediction. Figure 5.1 illustrates the flow diagram of the proposed E-LW-IC-KPLS model. Similar to E-LW-KPLS model, the number of models to be ensemble, n^th are pre-defined.

The predicted output $\hat{y}_q$ is calculated through the following procedure:

1. Set $V = x$ as training data for Kernel matrix and $V_q = X_q$ as the query or testing data where they are mapped into higher dimensional feature space using selected Kernel function.
2. Conduct mean centering on the V and V_q using Equations (4.28) and (4.29).

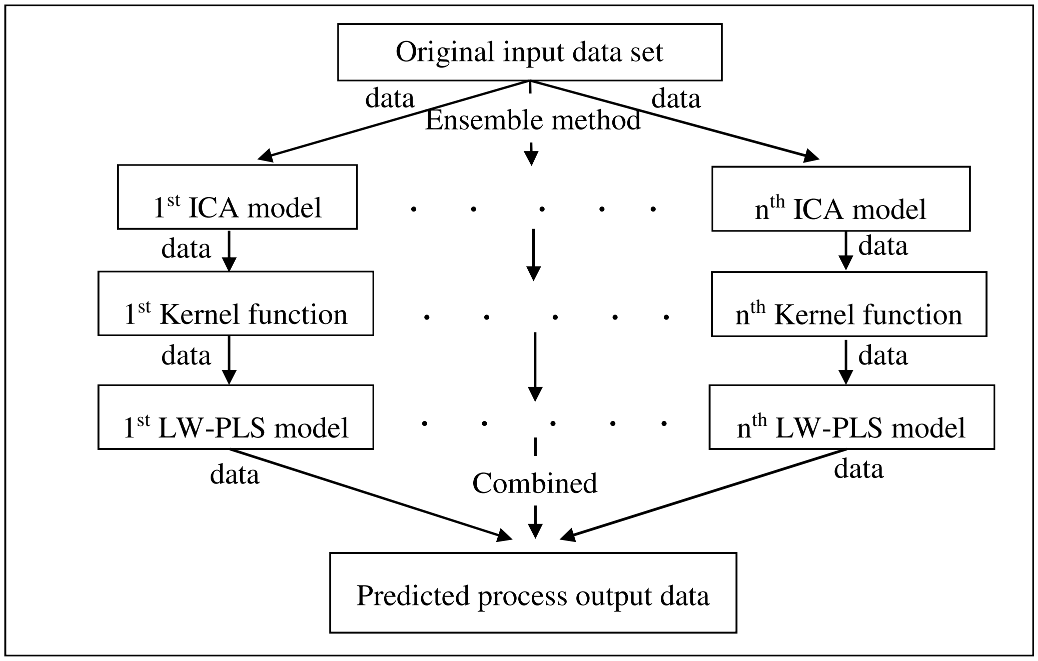

Figure 5.1 Flow diagram of the proposed E-LW-IC-KPLS model

3.Perform dual Kernel partial least square discrimination to obtain *B* using Equation (4.30).

4. Compute the rescaled query, and input variable matrices, V_q, and V using Equations (4.31) and (4.32).

5.Determine the number of ICs, *c* which is similar to the number of input variables and then X and X_q are set as the observed variables in the ICA models.

6. Apply whiten transformation to remove all of the cross-correlations between random variables using Equation (5.3) in which the over-sphered zero-mean vector, *z* is calculated.

7. Determine B_c, subject to Equation (5.5).

8. Calculate *W* using Equation (5.6) to estimate *s* via Equation (5.5).

9. The estimated *s* for both X and X_q are set as input variables and a query for the LW-PLS model is conducted.

10. Determine the number of latent variables *K* and set $k = 1$.

11. Calculate a similarity matrix Ω using Equations (4.1), (4.2) and (4.5).

12. Calculate X_k, Y_k, and $X_{q,k}$ using Equations (4.6) to (4.10).

13. Set $\hat{y}_q = [\bar{y}_1, \bar{y}_2, ..., \bar{y}_L]^T$.

14. Derive the k-th latent variable of X_k using Equations (4.11) and (4.12).

15. Derive the k-th loading vector of X_k and the k-th regression coefficient vector using Equations (4.13) and (4.14).

16. Derive the k-th latent variable of X_q using Equation (4.15).

17. Replace $\hat{y}_q$ with $\hat{y}_q + t_{q,j} q_j$ where $t_{q,k}$ is the k-th latent variable of X_q.

18. If $k = K$, then finish prediction. Otherwise, set Equations (4.16) to (4.18).

19. Set $k = k + 1$ and go to Step 9.

5.4. Expectation maximization algorithm

In this section, an overview of the EM algorithm is explained. EM can be defined as a statistical approach for conducting likelihood estimation with missing data (Pangborn *et al.* 2011). This approach preferable when the pattern of missing data is at random (Karanja *et al.* 2013). This method has been used as a missing data imputation method by Xiong *et al.* (2015), Li *et al.* (2015), Junger and de Leon (2015), Yin *et al.* (2014), Jin *et al.* (2012) and Schön (2009). More information about EM can be found in Lan *et al.* (2016), Gómez-Carracedo *et al.* (2014), Karanja *et al.* (2013) and Schön (2009).

When the data to the model are not unimodal or do not have symmetric shape, these data are more likely referring to non-Gaussian data, and a mixture model could be used to describe the data distribution (Ma 2011). Moreover, the mixture model has shown success in handling non-Gaussian data (Yuan *et al.* 2014b). On the other hand, EM for the mixture model has been employed by Zhou and Lim (2014) to replace missing data. Hence, in this study, EM for the mixture model is used to handle missing data problem.

In this study, it is assumed that the missing data is present in the input variables (x-variables) since missing data rarely happens in the output variables (y-variables). In most cases, the output variables that are related to the final product including its concentration is tested using the hardware sensors manually. For instance, the concentration of gasoline and butane in the distillation column are measured using gas chromatography in the laboratory. Hence, the majority of the previous studies do not consider missing data in the output variables. These studies include Basir and Wei (2018), Severson *et al.* (2017), Yuan *et al.* (2016), Junger and De Leon (2015), Ringgi *et al.* (2015), Schmitt *et al.* (2015), De la Fuente *et al.* (2010), Schön (2009), Sentas and Angelis (2006), Lopes and Menezes (2005), Arteaga and Ferrer (2005), Arteaga and Ferrer (2002), Walczak and Massart (2001), and Nelson *et al.* (1996). Therefore, the EM is developed with the consideration that missing data happens in the *x*-variables.

Before performing EM for the mixture model, the initial value that is the mean value of the observed data, m_t is needed to replace the missing data to generate a complete mixture model. By substituting the original missing data with m_t, a new reconstructed matrix, X_t for the mixture model is obtained. Let X_m be a n x M matrix with missing data where $x = [x_{n1}, x_{n2}, \ldots, x_{nM}]$ is the *n*-th row containing the information for observation *n* and *x* is the value of the *M*-th variable for observation *n*. Moreover, the missing data indicator matrix, M_s is the binary $n \times M$ matrix in which the element in M_s, $m_{ij} = 1$ if x_{nM} is missing, and $m_{ij} = 0$ if x_{nM} is presence. In addition, the complement of M_s, $\overline{M}_s$ has the element, $\overline{m_{ij}} = 1 - m_{ij}$. At the beginning stage, missing data in X_m are filled with zeroes. Then, X_t is built using the below equation which is a modified equation from Folch-Fortuny *et al.* (2015):

$$X_t = \overline{M}_s X_m + M_t M_s \tag{5.8}$$

where M_t is a $n \times M$ matrix that consists of m_t.

Then, EM for the mixture model is used to generate a new optimum M_t by using X_t. Similar to X_m, X_t be a $n \times M$ matrix where $x = [x_{n1}, x_{n2}, \ldots, x_{nM}]$. The probability density function for X_t which has a M-dimensional sample can be written as:

$$f(X_t \mid \theta) = \frac{1}{\sqrt{(2\pi)^{d_t} \Sigma}} \exp\left\{ \frac{-1}{2} (X_t - \mu)^T \Sigma^{-1} (X_t - \mu) \right\} \tag{5.9}$$

where μ is the mean vector, and Σ is the covariance matrix. $\theta = \{\mu, \Sigma\}$ is the mixture model parameters that are used to obtain a distribution. Then, the probability density function for the X_t, which is from a mixture model can be derived as shown (Bishop 2006; Yuan *et al.* 2014b):

$$p(X_t \mid \Omega_t) = \sum_{k_t=1}^{K_t} \pi_{k_t} f(X_{\mathrm{t}} \mid \theta_{k_t}) \tag{5.10}$$

where K_t is the number of non-Gaussian components, and θ_{k_t} in mixture model is pre-defined. π_{k_t} is the probabilistic weight of the k_t -th non-Gaussian component where π_{k_t} should be a positive value and $\sum_{k_t=1}^{K_t} \pi_{k_t} = 1$. Ω_t consists of the parameters in the mixture model with K_t components and it can be defined as $\Omega_t = (\{\pi_1, \mu_1, \Sigma_1\}, \{\pi_1, \mu_1, \Sigma_1\}, \ldots, \{\pi_{K_t}, \mu_{K_t}, \Sigma_{K_t}\})$.

The likelihood and log-likelihood functions of X_t in EM for the mixture model are shown as below:

$$L(X_t,\Omega_t) = \prod_{n_t=1}^{N} \sum_{k_t=1}^{K_t} \pi_{k_t} f(\mathrm{X}_t^{n_t} \mid \theta_{k_t}) \qquad (5.11)$$

$$\log L(X_t,\Omega_t) = \sum_{n_t=1}^{N} \log\left[\sum_{k_t=1}^{K_t} \pi_{k_t} f(\mathrm{X}_t^{n_t} \mid \theta_{k_t})\right] \qquad (5.12)$$

Initially, a log sum of the exponential is computed to avoid potential overflow and underflow of the exponential function in EM for the mixture model (Pangborn *et al.* 2011). This model is shown as Equation (5.13).

$$\log(\sum_i \exp(x_i)) \equiv \max(X_t) + \log\left(\sum_i \exp(x_i - \max(X_t))\right) \qquad (5.13)$$

Next, the unknown parameters in the set Ω_t must be determined. In this study, EM for the mixture model is used to obtain the parameters that maximise the log-likelihood function and can be written as:

$$\Omega_t = \underset{\Omega_t}{\arg\max} \log L(X_t,\Omega_t) \qquad (5.14)$$

The EM algorithm is comprised of two steps, which are an expectation step (E-step), and a maximization step (M-step). The E-step estimates the parameters, and then the M-step updates the estimated parameters. The E-step and M-step are repeated iteratively, then the Ω_t will converge to optimum values. As a result, EM generates a sequence of parameters which are $\{\Omega_t^1, \Omega_t^2, \ldots, \Omega_t^m\}$. In this study, EM for the mixture model is taken from Yuan *et al.* (2014b) and Yu and Qin (2008).

The missing data, X_t is calculated using the EM algorithm through the following steps:

1. Set m = 1 and start with an initial guess $\Omega_t^1 = -\infty$.
2. Expectation step: Compute

$$Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right) = \frac{\pi_{k_t}^m f(X_{tj} \mid \mu_{k_t}^m, \Sigma_{k_t}^m)}{\sum_{k_t=1}^{K_t} \pi_{k_t}^m f(X_{tj} \mid \mu_{k_t}^m, \Sigma_{k_t}^m)} \tag{5.15}$$

   where $Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)$ is the posterior probability of the j-th sample data within the k-th non-Gaussian component at the m-th iteration.

3. Maximization step: Compute

$$\pi_{k_t}^{m+1} = \frac{\sum_{j=1}^{N} Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)}{N} \tag{5.16}$$

$$\mu_{k_t}^{m+1} = \frac{\sum_{j=1}^{N} Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right) X_{tj}}{\sum_{j=1}^{N} Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)} \tag{5.17}$$

$$\Sigma_{k_t}^{m+1} = \frac{\sum_{j=1}^{N} Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)\left(X_{tj} - \mu_{k_t}^{m+1}\right)\left(X_{tj} - \mu_{k_t}^{m+1}\right)^T}{\sum_{j=1}^{N} Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)} \tag{5.18}$$

   where $\mu_{k_t}^{m+1}$, $\Sigma_{k_t}^{m+1}$ and $\pi_{k_t}^{m+1}$ are the mean, covariance, and prior probability of the k-th non-Gaussian component at the (m+1)-th iteration, respectively.

4. If not converged, update $m = m + 1$ and return to Step 2.

5. If converged, then the newly generated $m_t = \mu_{k_t}$ is obtained. Next, M_t is substituted into Equation (5.8) to obtain a new X_t where missing data are replaced with this new m_t.

5.5. Expectation maximization ensemble locally weighted independent component Kernel partial least square algorithm

The EM-E-LW-IC-KPLS algorithm is an extended algorithm of the E-LW-IC-KPLS model in which EM are used to handle missing data. At first, EM model is integrated to the newly developed E-LW-IC-KPLS algorithm. As mentioned in Section 5.4, EM uses an expectation step to estimates the value for the missing data, and then the estimated data is updated using a maximization step to get an optimum value. The original data including the missing data are initially executed using the EM algorithm. Subsequently, the output data from the EM algorithm are sent to the E-LW-IC-KPLS model. Figure 5.2 displays the flow diagram of the original EM-E-LW-IC-KPLS model, which is designed for processes that contain missing data, nonlinear and non-Gaussianity distributed data.

In this study, the range of missing data levels in the input data used to test the developed algorithm is from 5% to 60%. In most cases, the output variable(s) that is related to the final product quality is tested using hardware sensors such as laboratory assays and gas chromatography. Hence, the majority of previous studies did not consider missing data in the output variables (Yuan et al. 2017, Folch-Fortuny et al. 2015, Junger and De Leon 2015, Riggi et al. 2015, De la Fuente et al. 2010, Schön 2009, Arteaga and Ferrer 2005, Lopes and Menezes 2005, Arteaga and Ferrer 2002, Walczak and Massart 2001, Nelson et al. 1996).

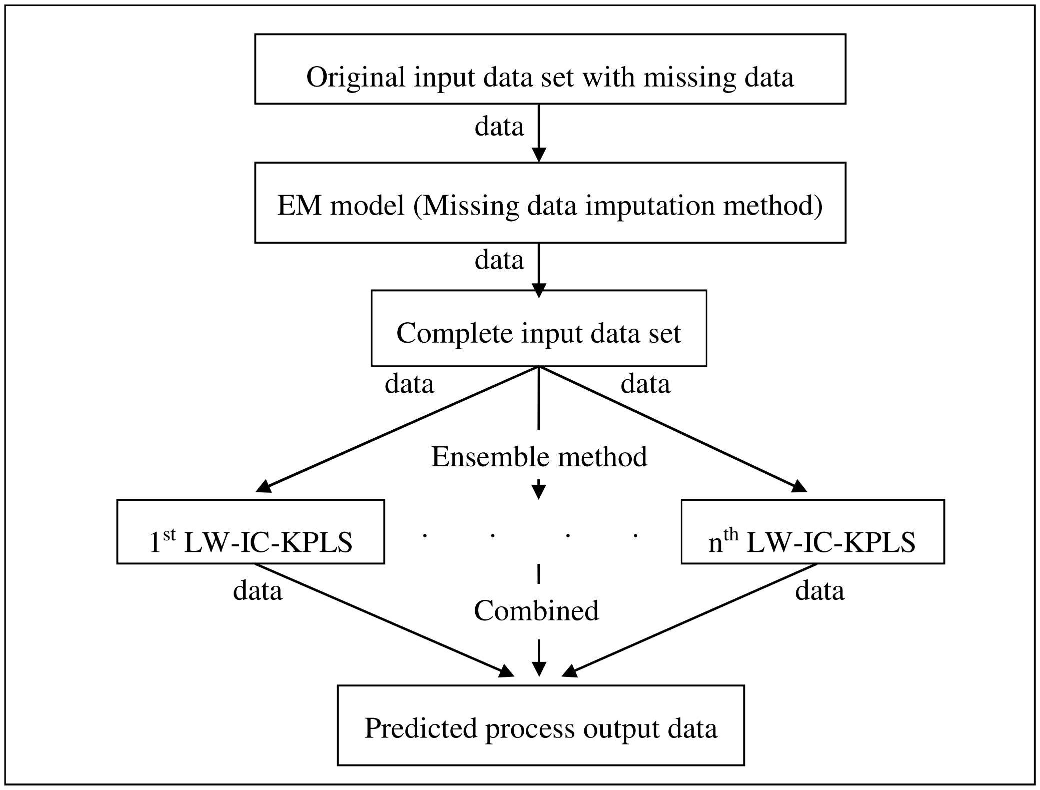
Original input data set with missing data
data
EM model (Missing data imputation method)
data
Complete input data set
data
data
Ensemble method
1st LW-IC-KPLS
nth LW-IC-KPLS
data
Combined
data
Predicted process output data

Figure 5.2 Flow diagram of the EM-E-LW-IC-KPLS model

The predicted output, $\hat{y}_q$ is determined via:

1. Missing data in X_m is filled with zeroes. Then, the initial value, the mean value of the observed data, m_t is calculated.
2. The calculated M_t is substituted into Equation (4.19) to build X_t .
3. Next, set m = 1 and start with an initial guess $\Omega_t^1 = -\infty$.
4. Expectation step: Compute $Q^m\left(c_{k_t} \mid X_{tj}, \Omega^m\right)$ using Equation (4.26).
5. Maximization step: Compute $\mu_{k_t}^{m+1}$, $\Sigma_{k_t}^{m+1}$ and $\pi_{k_t}^{m+1}$ using Equations (4.27), (4.28) and (4.29).
6. If the imputed value for the missing data is not converged, update $m = m + 1$ and return to Step 4 (Threshold of converge is $1\text{x}10^{-6}$).

7. If converged, then the newly generated $m_t = \mu_{k_t}$ is obtained. Next, M_t is substituted into Equation (4.19) to obtain a new X_t where missing data are replaced with this new m_t.
8. Set V as training data of X_t for the Kernel matrix and V_q as the query or testing data of X_t where they are mapped into a higher dimensional feature space using the selected Kernel function.
9. Conduct mean centering on the V and V_q using Equations (4.28) and (4.29).
10. Perform dual Kernel partial least square discrimination to obtain B using Equation (4.30).
11. Compute the rescaled query, and input variable matrices, V_q, and V using Equations (4.31) and (4.32).
12. Determine the number of ICs, c which is similar to the number of input variables and then X and X_q are set as the observed variables in the ICA models.
13. Use the whiten transformation to remove all the cross-correlation between random variables using Equation (5.3) in which the over sphered zero-mean vectors, z is calculated.
14. Determine B_c, subject to Equation (5.5).
15. Calculate W using Equation (5.6) in order to estimate s via Equation (5.5).
16. The estimated s for both X and X_q are set as input variable and query for LW-PLS model.
17. Determine the number of latent variables K and set $k = 1$.
18. Calculate a similarity matrix Ω using Equations (4.1), (4.2) and (4.5).
19. Calculate X_k, Y_k, and $X_{q,k}$ using Equations (4.6) to (4.10).
20. Set $\hat{y}_q = [\bar{y}_1, \bar{y}_2, ..., \bar{y}_L]^T$.
21. Derive the k-th latent variable of X_k using Equations (4.11) and (4.12).
22. Derive the k-th loading vector of X_k and the k-th regression coefficient vector using Equations (4.13) and (4.14).

23. Derive the k-th latent variable of X_q using Equation (4.15).

24. Replace $\hat{y}_q$ with $\hat{y}_q + t_{q,j} q_j$ where $t_{q,k}$ is the k-th latent variable of X_q.

25. If $k = K$, then finish prediction. Otherwise, set Equations (4.16) to (4.18).

26. Set $k = k + 1$ and go to Step 16.

5.6. Summary

The details formulation of non-Gaussian-based LW algorithms including the E-LW-IC-KPLS and the EM-E-LW-IC-KPLS models are given in this chapter. These novel algorithms are extended algorithms for Gaussian-based LW algorithm, which is E-LW-KPLS presented in Chapter 4. Due to the presence of ICA in the E-LW-IC-KPLS algorithm, it provides better predictive performance in handling non-Gaussian distributed data as compared to the existing locally weighted partial least square (LW-PLS) and locally weighted Kernel partial least square (LW-KPLS) as well as the E-LW-KPLS. However, neither E-LW-IC-KPLS nor LW-PLS nor LW-KPLS can deal with missing data. Hence, the EM-E-LW-IC-KPLS algorithm is proposed to cope with nonlinear, non-Gaussian distributed and missing data simultaneously. In the following chapter, Chapter 6, the effectiveness of these Gaussian-based and non-Gaussian-based LW algorithms is demonstrated using numerous case studies.

Chapter 6

Results and discussions

6.1. Introduction

In this chapter, the improved Gaussian-based and non-Gaussian-based locally weighted (LW) algorithms described in Chapters 4 and 5, respectively were applied to six different case studies. The effectiveness and capabilities of the Gaussian-based LW algorithm, an ensemble locally weighted Kernel partial least square (E-LW-KPLS), and the two non-Gaussian-based LW algorithms, the ensemble LW independent component Kernel partial least square (E-LW-IC-KPLS) and the expectation maximization ensemble locally weighted independent component Kernel partial least square (EM-E-LW-IC-KPLS) were demonstrated using these case studies.

Comparative studies were also carried out between the E-LW-KPLS and E-LW-IC-KPLS algorithms with locally weighted partial least square (LW-PLS) and other integrated algorithms as well as their associated models the locally weighted Kernel partial least square (LW-KPLS) and the locally weighted independent component Kernel partial least square (LW-IC-KPLS). The explanations of the six case studies and the configuration of the computer used are described first. Next, the parameter optimisations used in all LW algorithms, the splitting data, parameter settings, and missing data are discussed. Lastly, the results obtained from the case studies are presented and discussed.

6.2. Case studies

Three case studies with Gaussian distributed data, identified as case studies 1, 2 and 3 were used to demonstrate the performance of the E-LW-KPLS algorithm and another three case studies with non-Gaussian distributed data, are identified as case studies 4, 5 and 6 were employed to evaluate the effectiveness of the E-LW-IC-KPLS and EM-E-LW-IC-KPLS models. In this section, the details of these case studies are briefly described.

6.2.1. Case study 1: Numerical example 1

This numerical example is a nonlinear dynamic test problem with Gaussian distributed data adopted from Li *et al.* (2010). The following equation can represent the function:

$$y = \cos\left(\sqrt{x_1 + x_2}\right) \tag{6.1}$$

where x_1 and x_2 are the input variables and y is the output variable. These input variables are generated from a normal distribution with zero mean and unit variance. Moreover, a Gaussian noise whose mean is zero, and the variance is 0.1 is added to the above function (Equation 6.1).

6.2.2. Case study 2: A single chemical reactor

This simulation study which involves a single chemical reactor was obtained from Christofides and El-Farra (2005). This reactor is a well-mixed, non-isothermal continuous stirred tank reactor (CSTR) where three parallel irreversible elementary exothermic reactions of the form $A_1 \xrightarrow{k_1} B_1$, $A_1 \xrightarrow{k_2} U_1$ and $A_1 \xrightarrow{k_3} R_1$ occur. A_1 is the reactant species, B_1 is the desired product, U_1 and R_1 are the undesired by-products while k_1, k_2 and k_3 are the reaction rates.

In this study, the rate of heat input to the CSTR is set at 0 KJ/hr since an adiabatic process is assumed. Meanwhile, the feed flow rate of pure A_1 to the CSTR is kept constant in the range of 4.998 to 6 m^3/hr to achieve a nonlinear dynamic process. The differential equations that describe the process of this case study are shown in Appendix A.1 in Appendix A. The detailed information about this CSTR can be found in Christofides and El-Farra (2005) and additionally this study consisted of Gaussian distributed process data. Table 6.1 shows the input and output variables for the case study 2.

Table 6.1 Input and output variables for case study 2

Output variable	**Variable description**
CB	Concentration of the desired product
Input variables	**Variable description**
T_1	Temperature of the reactor
CA	Concentration of the reactant species

6.2.3. Case study 3: Wastewater treatment

Wastewater treatment processes are very complex and nonlinear. Figure H.1 in Appendix H shows a wastewater treatment plant with an aerated bioreactor containing a mixture of liquid and suspended solids where microorganisms are grown to eliminate the organic substrate from the mixture. Moreover, a settler is used to separate the sludge and the clear effluent. Some of the settled sludge is recycled back into the bioreactor, and the remaining sludge from the settler is discharged as waste (Caraman *et al*. 2007).

In this treatment process, substrate, S_s, biomass, X_B, and dissolved oxygen concentration, DO in the effluent are sent to the bioreactor first and then to the settler. From the settler, the recycled biomass, X_r in the sludge is recycled back to the bioreactor. In the bioreactor, an appropriate DO level enables the optimal growth of microorganisms used in the wastewater treatment process (Ingildsen 2002). Hence, in this study, the predictive output variable is the DO while there are three input variables which include S_s, X_B, and X_r. The differential equations that explain this wastewater treatment are give in Appendix A.2 in Appendix A. These variables involve Gaussian distributed data, and they are summarised in Table 6.2.

Table 6.2 Input and output variables for case study 3

Output variable	Variable description
DO	Dissolved oxygen concentration
Input variables	**Variable description**
S_s	Substrate concentration
X_B	Biomass concentration
X_r	Recycled biomass concentration

6.2.4. Case study 4: Numerical example 2

This nonlinear dynamic numerical example is a static function approximation of the sum of two sine waves adapted from Roffel and Betlem (2007). The sine function can be represented by Equation (6.2):

$$y = 0.5\sin(\pi x) + 0.5\sin(2\pi x) \tag{6.2}$$

where x is the input variable with a range of 0 to 10 and y is the output variable. These variables are generated using non-Gaussian distributed data, and moreover, the Gaussian noise whose mean is zero and variance 0.1 has been added to the above function (Equation 6.2). After plotting the histogram plot of the generated data using MATLAB, it was found that the non-Gaussian data in case study 1 followed multi-model distributions having more than one peak or mode.

6.2.5. Case study 5: Eukaryotic cell cycle regulation

This process is a biochemical reaction involving cyclin-dependent kinase and their associated proteins which are included in a cell cycle to control frog egg development. There are three key species in this biochemical reaction: the free cyclin, the M-phase promoting factor (MPF) and the other regulatory enzymes. MPF is a heterodimer formed by the proteins Cdc2 and cyclin. It controls the transition from the interphase to mitosis of the cell cycle. This Cdc2-cyclin regulatory system is assumed to operate as a spontaneous oscillator in the early embryos. The detailed description of this nonlinear dynamic process can be found in Novak and Tyson (1993). In this study, the output variable used to represent the product quality is a dimensionless concentration of total cyclin, *v*. Meanwhile, the predictor is a dimensionless concentration of active MPF, *u*. The differential equations which illustrate the eukaryotic cell cycle regulation are given in Appendix A.3 in Appendix A. In case study 5, the non-Gaussian distributed data follows exponential and bimodal distributions. Table 6.3 shows the input and output variables that are non-normal data for this case study.

Table 6.3 Input and output variables for case study 5

Output variable	**Variable description**
v	Dimensionless concentration of total cyclin
Input variable	**Variable description**
u	Dimensionless concentration of active MPF

6.2.6. Case study 6: A highly nonlinear continuous stirred tank reactor

This case study is adapted from Chen *et al.* (1995) where cyclopentenol, B_2 is generated from cyclopentadiene, A_2 by acid-catalyzed electrophylic hydration in aqueous using CSTR with a cooling jacket (as shown in Figure H.2 in Appendix H). This process is called a 'van der Vusse reaction,' and the following reaction scheme can describe it: $A_2 \xrightarrow{k_1} B_2 \xrightarrow{k_2} C_2$ and $2A_2 \xrightarrow{k_3} D_2$. The reactant, A_2 is transformed to B_2 and then B_2 is consecutively reacted to form the unwanted product cyclopentanediol, C_2.

Moreover, A_2 also undergoes another undesired parallel reaction to produce the by-product dicyclopentadiene, D_2. In Figure H.2, the flow rate $\dot{V}$ containing only cyclopentadiene, A_2 with an initial concentration C_{A0} and a temperature ϑ_0 is fed to a CSTR. The heat load, $\dot{Q}_k$ is removed from the coolant using an external heat exchanger. The differential equations described this van der Vusse reaction are provided in Appendix A.4 in the Appendix A. Additional descriptions about this nonlinear dynamic process can be found in Chen *et al.* (1995) and Helbig *et al.* (2000). The non-Gaussian distributed process data in this case study was found to be multimodel distribution. Table 6.4 shows the input and output variables containing non-Gaussian distributed data for case study 6.

Table 6.4 Input and output variables for case study 6

Output variable	**Variable description**
CP	Product concentration
T_2	Reactor temperature
Input variable	**Variable description**
$\frac{V}{V_R}$	Scaled volumetric inlet flow
$\dot{Q}_k$	Heat flow

6.3. Parameter optimization

An optimized parameter is required to be tuned to find a well-performing model. Since the LW-PLS algorithm has been integrated into the improved LW-KPLS, E-LW-KPLS, LW-IC-KPLS, E-LW-IC-KPLS and EM-E-LW-IC-KPLS models, these LW algorithms require similar optimized parameters such as the localisation parameter used to find similarity of the data, φ , the number of latent variables in the local regression model, K as well as the number of samples, training and test sets N, N_1 and N_2, respectively used to perform prediction. Furthermore, to handle nonlinear data, the improved Gaussian- and non-Gaussian-based LW algorithms require an additional optimised parameter in the Kernel function, which is the Kernel parameter, b.

In this study, K, N, N_1, N_2, and φ are fixed to proper values while b is tuned by trial and error experiments. Since N, N_1, and N_2 are not sensitive parameters for the LW algorithms, they are specified as an equal adequate value in all the case studies. Typically, the first few latent variables commonly referred to as principal components of PLS-based algorithms, are sufficient to describe the main features of the data (Yuan *et al.* 2014a). Furthermore, an equal value of K is applied in all the case studies to compare the algorithms fairly. The number of latent variables K is predefined. A set of latent variable explained the relationship between two variables, hence they are used for dimensionality reduction of data. Since less variables are involved in the case studies and higher number of K may increase the computational times of the integrated models, the number of latent variables K is fixed as 1.

Also, applying different values of φ in the LW algorithms does not change the results as φ is only useful when choosing related samples for query samples in the LW algorithms (Yuan *et al.* 2014a). Hence, φ is also fixed as equal value for all LW algorithms in the case studies. The optimal value for φ was studied by Yeo et al. (2017) and the publication can be found in Yeo *et al.* (2017). Moreover, in the expectation maximization (EM) algorithm for a Gaussian mixture model, the number of Gaussian components, K_t is specified as a small value for all the case studies since a higher value of K_t may increase the computational time of LW algorithms.

On the other hand, Kernel parameter, b in a reasonable space range varies from 0.01 to 100 (Wang *et al*. 2015b). Furthermore, the studies carried out by Mongillo (2011) and Orr (1996) have shown the minimum error is usually found when the value of b is small in the range of 0.01 to 10. In most cases, they also found when *b* equals 1 this provides the lowest predicted error. Hence, *b* is tuned in this range to obtain the smallest error of the dataset including the training and test dataset. Moreover, the number of the independent component, *c* used in the LW-IC-KPLS, E-LW-IC-KPLS and EM-E-LW-IC-KPLS models is different in each case study. Table 6.5 summarises all the parameters used in the algorithms.

Table 6.5 Parameters used in the algorithms

Conventional algorithm	**Used parameters**
LW-PLS	φ , K, N, N_1, N_2
New algorithms	**Used parameters**
LW-KPLS	φ , K, N, N_1, N_2, b
E-LW-KPLS	φ , K, N, N_1, N_2, b
LW-IC-KPLS	φ , K, N, N_1, N_2, b, c
E-LW-IC-KPLS	φ , K, N, N_1, N_2, b, c
EM-E-LW-IC-KPLS	φ , K, N, N_1, N_2, b, c , K_t

6.4. Splitting data and parameters setting

For each case study, data sets are generated using MATLAB Simulink. Large number of dataset is adopted to resemble the real scenarios in industrial processing plants in which abundant data are recorded. Furthermore, for historical database modelling, the higher the number of samples, the better the accuracy of the developed model will be. Thus, 10,000 datasets are generated in each case study and they are split into training and test datasets. A big number of datasets is used since the computational times used by each model to execute the dataset are measured and compared. If only few hundred or thousand dataset is used, the calculated computational time for each model is only few seconds. Besides, for all of the case studies, 75% of the datasets is randomly selected as training data. And, the remaining 25% of the datasets are used as test data

(Saptoro *et al.* 2006; Saptoro *et al.* 2008). Therefore, N, N_1, and N_2 are set as 10,000, 7,500 and 2,500, respectively.

K and φ for the Gaussian-based and the non-Gaussian-based LW algorithms are adjusted to 1 and 0.1, respectively, in the case studies. Table 6.6 illustrates the values of these used parameters for the developed LW algorithms. In this study, the Kernel functions used are shown in Section 6.3. The optimal value of Kernel parameter, b is tuned from 0.01 to 10 and the value of b is different in each case study. The value of c is equal to the number of input variables while the value of K_t in the EM algorithm is set as 1 for all the case studies to minimize the computational time.

Table 6.6 Values of used parameters for the algorithms

Used parameters	Values
N	10000
N_1	7500
N_2	2500
K	1
φ	0.1
b	0.01 to 10
c	Same as the number of input variables
K_t	1

6.5. Missing data

Since the missing at random (MAR) mechanism usually occurs in process industries (Schafer and Graham 2002), hence missing data imputation methods that can cope with MAR are used in this research study. MAR refers to the probability of missingness and may rely on observed data only and not on any missing data. Since the distributions of missing data are often related to the output variables, the MAR assumption may be more realistic (Junger and Ponce de Leon, 2015). During the stage of constructing the model for chemo-metric industries data, practitioners commonly must handle 5% to 20% of the missing data. Likewise, for complex processes including nonlinear processes, percentages of missing data varying from 30% to 60% can appear in the historical dataset (A Folch-Fortuny *et al*. 2015). Since the process data for case studies are nonlinear, 5% to 60% of missing data are used in this research work.

For each case study, different percentages of data are removed from the generated data set and the removed data is considered as missing data. In this study, missing data levels of 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55% and 60% are all considered in order to exam the ability of the EM-E-LW-IC-KPLS models in handling missing data problems. These different missing data levels are present in both training and test input data. To test these missing data in the relevant case studies, several missing data imputation methods are used to obtain their imputed data to replace them in the data set. Then, the complete data set with the imputed data are run using different integrated models. These models are EM model integrated with LW-PLS (EM-LW-PLS), the singular vector decomposition (SVD) method integrated with the E-LW-IC-KPLS (SVD-E-LW-IC-KPLS) and LW-PLS (SVD-LW-PLS) algorithms and the trimmed score regression (TSR) combined with the E-LW-IC-KPLS (TSR-E-LW-IC-KPLS) and LW-PLS (TSR-LW-PLS) models. The E_a, and *RMSE* are also measured and compared with the EM-E-LW-IC-KPLS algorithm.

6.6. Current results and discussions

The results of the improved E-LW-KPLS, E-LW-IC-KPLS and EM-E-LW-IC-KPLS models were applied to six different case studies. As mentioned in Chapters 4 and 5, the E-LW-KPLS, E-LW-IC-KPLS, and EM-E-LW-IC-KPLS algorithms involve numerous subsets of LW-KPLS and LW-IC-KPLS models. The number of data in each subset is the same, and the nonlinear relationship between input and output variables in each sub-dataset is consistent. In this study, 1 set to 30 subsets of the LW-KPLS and LW-IC-KPLS models were tested and evaluated. According to Kaneko and

Funatsu (2016), a model with a small dataset changes correlation and over-fitting. Hence, only up to 30 sets were examined and there has not been much improvement in most case studies after more than 20 subsets. Furthermore, in each case study, the value of E_a versus the number of graph sets was used to choose the optimal set of the E-LW-IC-KPLS algorithm.

The optimal set of the E-LW-IC-KPLS model was integrated with the EM to form the EM-E-LW-IC-KPLS algorithm and is used to solve the missing data problem. Next, its predictive performance was investigated under various levels of missing data and compared with other missing data imputation methods including the SVD-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS models. Also, different Kernel functions are utilised in separate case studies. The inverse multi-quadric and polynomial Kernel functions were found to work well with nonlinear and Gaussian distributed data. Furthermore, the Log Kernel function performed well on nonlinear and non-Gaussian distributed data.

All of the obtained results from the abovementioned algorithms were analysed, compared and discussed in this section. The predictive performance of the algorithms was evaluated and distinguished based on their $RMSE_1$, $RMSE_2$, E_a, both of the CPU running times for training data, t_1 and test data, t_2 as well as the MAE and MSE for training (MAE_1 and MSE_1) and the testing data (MAE_1 and MSE_2). For each case study, the computational time or load (t_1 or t_2) is measured using Equation (6.5) and it is obtained from the time taken from the start of executing the data using a model until the end as determined by an ordinary clock.

6.6.1. Gaussian distributed data

As mentioned earlier, case studies 1, 2, and 3 consisted of Gaussian distributed and nonlinear data. The Gaussian distributed data in these case studies are divided into training and testing data. These training and testing data of the case studies were used to evaluate the capabilities and effectiveness of the original Gaussian-based LW algorithm, the E-LW-KPLS. Furthermore, a sample of the MATLAB's coding for the

E-LW-KPLS model can be found in Appendix B. Based on the objective stated in Chapter 1, the obtained results from the E-LW-KPLS have to compare with the benchmarking LW-PLS and the developed LW-KPLS algorithms to investigate the predictive performance of E-LW-KPLS model.

Besides, to reduce the computation burden of the LW-KPLS which is also the objective of this research, the ensemble method was incorporated into LW-KPLS to form E-LW-KPLS. E-LW-KPLS has several LW-KPLS models that are run simultaneously. Generally, ensemble method is a learning method where multiple models are trained to solve the same problem and combined to shorten the processing time. Hence, E-LW-KPLS should have lower computational time than LW-KPLS. Figures 6.1 to 6.3 for case studies 1 to 3, respectively show the computational times for training and test data concerning the number of sets in E-LW-KPLS models. These number of sets or subsets are the multiple of LW-KPLS models. These figures have proven the effectiveness of the ensemble method in minimising computational times of the E-LW-KPLS models, especially from 1 set to 6 sets of LW-KPLS models.

Case study 1: Numerical example 1

In case study 1, an inverse multi-quadric Kernel which as shown in Equation (3.7) in Chapter 3 with b equal to 0.3 was utilized in both E-LW-KPLS and LW-KPLS models since it showed a more accurate predictive performance. Figure 6.4 displays the E_a values for all the E-LW-KPLS models while Table 6.7 shows the results of the prediction performance of LW-PLS, LW-KPLS and E-LW-KPLS algorithms Likewise, additional results for an E-LW-KPLS algorithm with other sets are listed in Appendix C.1.1.

As shown in Figure 6.4, the LW-KPLS model gave the best prediction results as it had the lowest value of E_a compared to other E-LW-KPLS algorithms. However, its computation load is the highest among all other algorithms. Hence, an E-LW-KPLS model with 3 subsets which has a slightly lower value of E_a than the LW-KPLS was chosen as the optimal set of E-LW-KPLS algorithm in this case study. As noted in the above Table 6.7 and Figure 6.1, the E-LW-KPLS model had 8 to 10 times lower computational loads than the LW-KPLS algorithm. As can be seen from Table 6.7 the E_a for both E-LW-KPLS and LW-KPLS models decreased by about 53.4% and 53.6% respectively, in comparison with the LW-PLS algorithm. This result has shown that the inverse multi-quadric Kernel in both LW-KPLS and E-LW-KPLS have significantly improved the predictive performance when dealing with nonlinear process data.

Furthermore, the $RMSE_1$, $RMSE_2$, MAE_1, MAE_2, MSE_1 and MSE_2 for E-LW-KPLS and LW-KPLS models were also much lower than LW-PLS. Again, with the help of the inverse multi-quadratic Kernel function, the E-LW-KPLS and LW-KPLS algorithms performed much better than the LW-PLS model. However, the t_1 and t_2 for the E-LW-KPLS and LW-KPLS algorithms were higher than the LW-PLS. By using the ensemble method, the computational times of E-LW-KPLS are lowered by than LW-KPLS. By using ensemble method, E-LW-KPLS allows a number of local LW-KPLS models to run spontaneously where each of the models has less number of data compared to a LW-KPLS model. Since the E-LW-KPLS has a lower computational time than the LW-KPLS model and all of its predictive errors are much smaller than the LW-PLS, it is still the most effective algorithm in this case study.

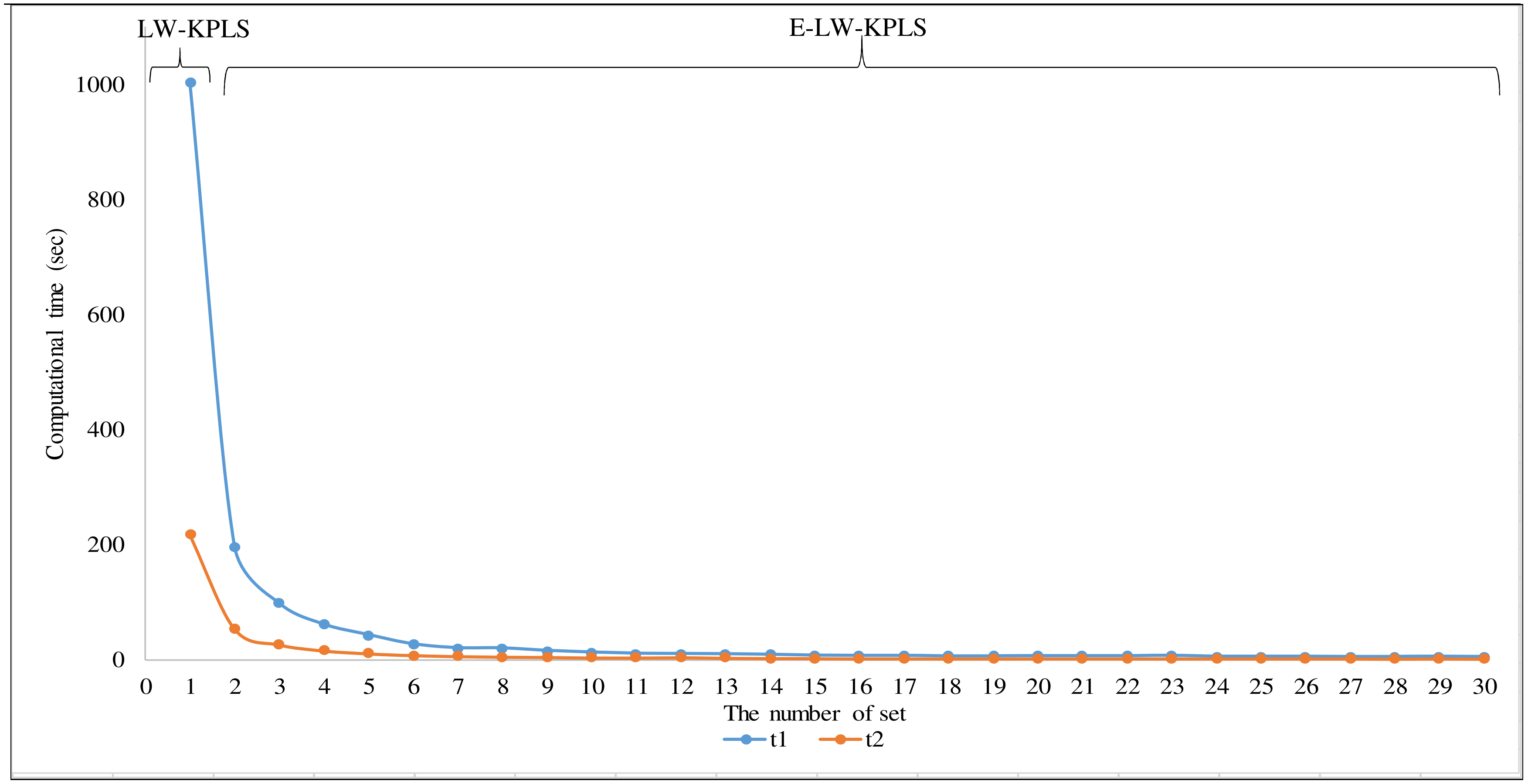

Figure 6.1 Computational times for LW-KPLS and E-LW-KPLS in case study 1

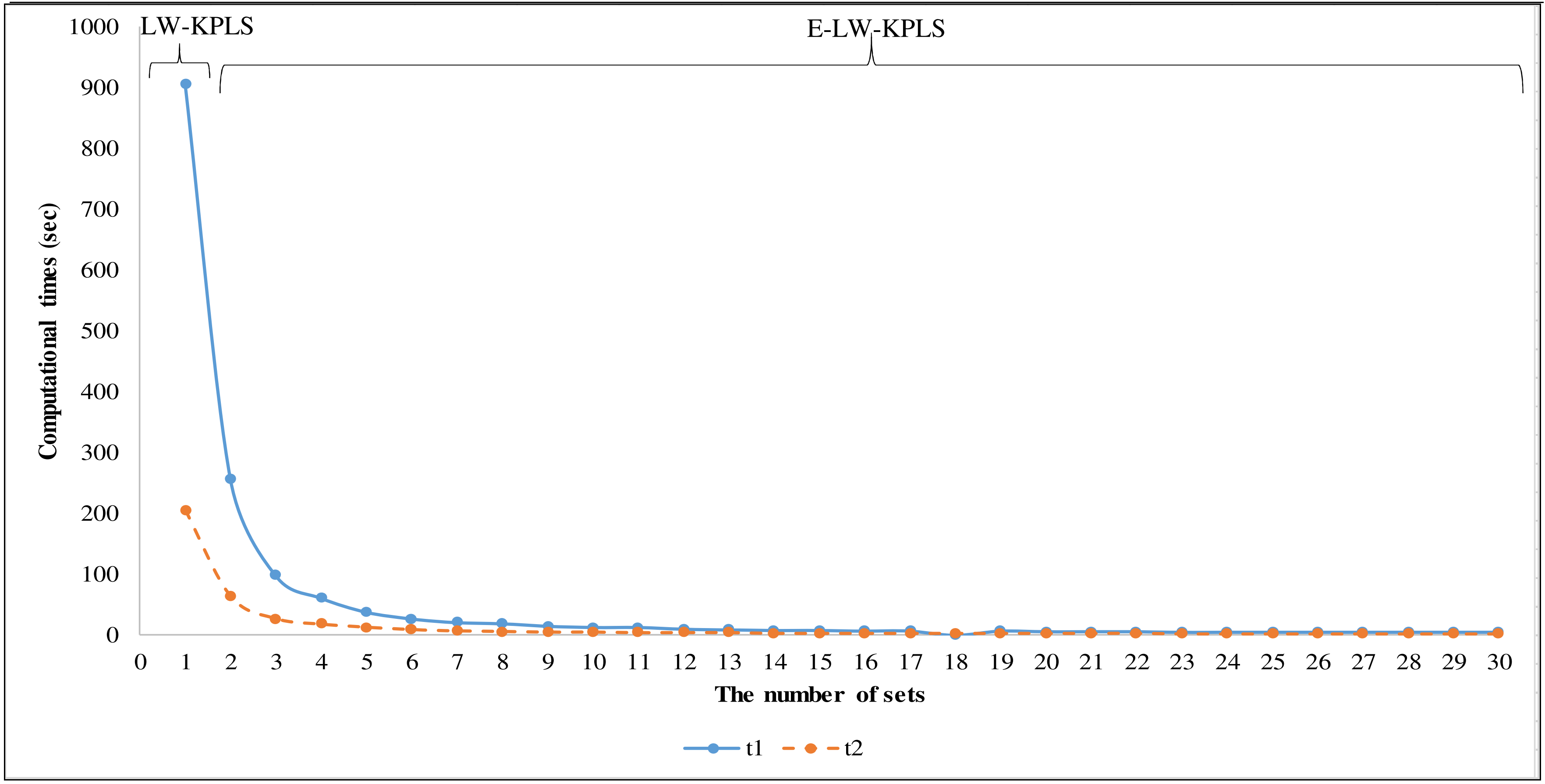

Figure 6.2 Computational times for LW-KPLS and E-LW-KPLS in case study 2

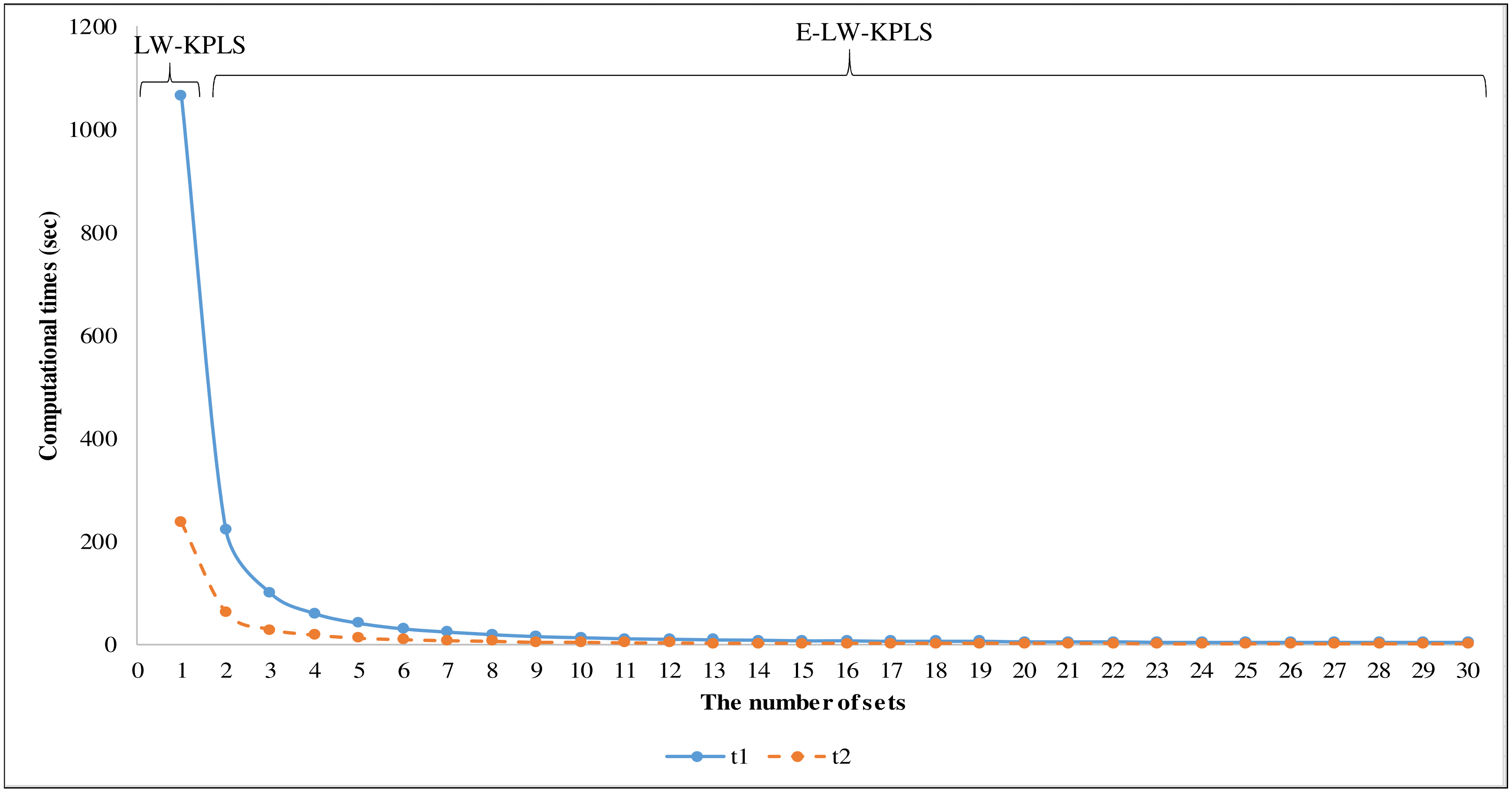

Figure 6.3 Computational times for LW-KPLS and E-LW-KPLS in case study 3

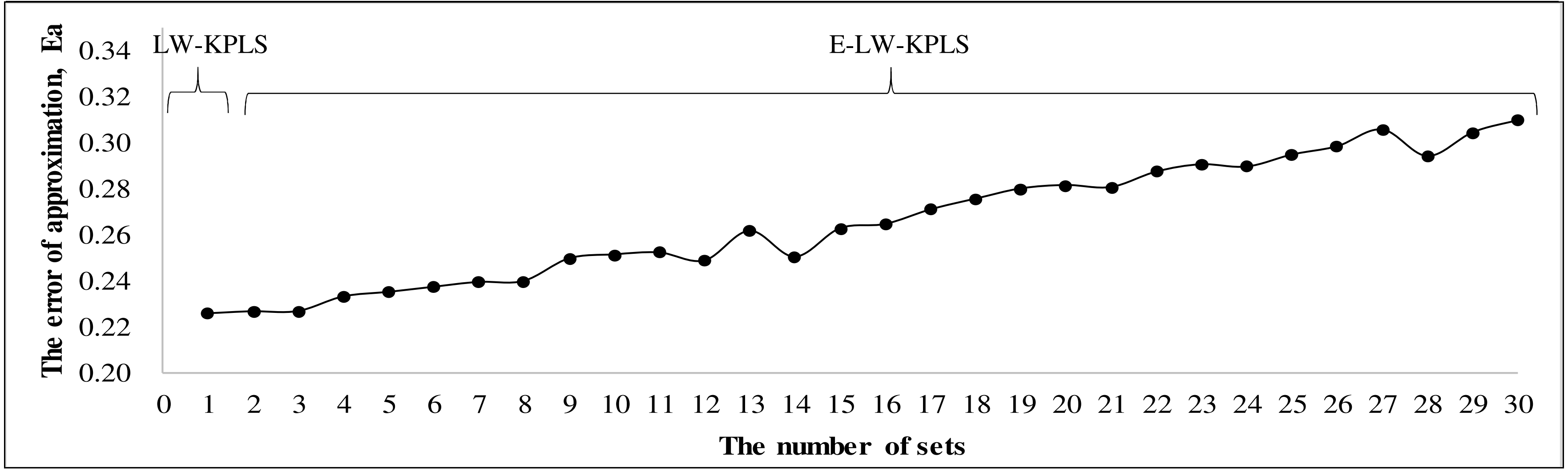

Figure 6.4 The error of approximation for separate subsets of the E-LW-KPLS in case study 1

Table 6.7 Prediction performances of E-LW-KPLS, LW-KPLS and LW-PLS models for case study 1

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**E-LW-KPLS**	**%**
Number of subsets	1	1		3	
Type of Kernel function	-	Inverse multi-quadric Kernel		Inverse multi-quadric Kernel	
Kernel parameter, *b*	-	0.3		0.3	
$RMSE_1$	0.4822	0.2244	53.5	0.2229	53.8
MAE_1	0.4264	0.1721	59.6	0.1717	59.7
MSE_1	0.2325	0.0504	78.3	0.0497	78.6
t_1 (sec)	14	1003		98	
$RMSE_2$	0.4859	0.2227	54.2	0.2259	53.5
MAE_2	0.4287	0.1706	60.2	0.1725	59.8
MSE_2	0.2325	0.0496	78.7	0.0510	78.1
t_2(sec)	4	218		26	
E_a	0.4868	0.2257	53.6	0.2267	53.4

On the other hand, the current prediction results from LW-PLS, LW-KPLS, and E-LW-KPLS models for training and test data in case study 1 are shown in Figures H.3 and H.4 in Appendix H. As can be seen in these figures, the predicted output from both E-LW-KPLS and LW-KPLS algorithms cope better with true output value than the LW-PLS. Also, LW-PLS was totally not able to predict the output values for both training and testing data since LW-PLS cannot cope with nonlinear dynamic data**.** As a conclusion, this newly developed E-LW-KPLS algorithm has achieved the objectives stated in Chapter 1 since it has better predictive performance than the benchmarking LW-PLS algorithm and its computational times are lower than LW-KPLS model.

Case study 2: A single chemical reactor

The polynomial Kernel shown in Equation (3.8) in Chapter 3 was used in the LW-KPLS, and E-LW-KPLS algorithms in case study 2 and the optimal value of *b* used were 0.01. The prediction results of the E-LW-KPLS, LW-KPLS and LW-PLS models for case study 2 are displayed in Table 6.8. Figure 6.5 illustrates the E_a for different subsets of the E-LW-KPLS algorithm for case study 2. From Figure 6.5, the E-LW-KPLS with 13 subsets has the lowest value of E_a and hence it was selected as the optimal set of E-LW-KPLS models. The current prediction results of other E-LW-KPLS algorithms can be seen in Appendix C.1.2.

Table 6.8 shows the E_a for the E-LW-KPLS and LW-KPLS models are lower by 58.3% and 17.6% respectively, compared to the LW-PLS. Due to the presence of the polynomial Kernel function in the E-LW-KPLS and LW-KPLS algorithms, they have slightly different results and their $RMSE_1$, MAE_1 and MSE_1 were less than the LW-PLS for nonlinear process data. Nevertheless, their $RMSE_2$, MAE_2 and MSE_2 were somewhat higher than the LW-PLS. The prediction errors of the LW-KPLS and LW-PLS models for training data ($RMSE_2$, MAE_2 and MSE_2) were higher than the test data. This result is due to the presence of an unsteady state condition in the first 30 samples of the output in the training data. The inclusion of these data in the local model constructions has caused the LW-KPLS and LW-PLS algorithms to demonstrate poorer predictive performance than the E-LW-KPLS. The ensemble method enables the E-LW-KPLS to exhibit more accurate generalization capabilities than by combining all data to be run in local models (Wang *et al*. 2016). Besides, ensemble method has the ability to reduce the variance of the error (Opitz and Maclin 1999). Generally, the smaller the value of variance will result in more reliable prediction (Liu and Gao 2015). Thus, ensemble method enables E-LW-KPLS model to reduce the variance of the unsteady state dataset in this case study.

Table 6.8 Prediction performances of E-LW-KPLS, LW-KPLS and LW-PLS algorithms for case study 2

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**E-LW-KPLS**	**%**
Number of subsets	1	1		13	
Type of Kernel function	-	Polynomial Kernel		Polynomial Kernel	
Kernel parameter, *b*	-	0.01		0.01	
$RMSE_1$	0.0075	0.0065	13.3	0.0040	46.7
MAE_1	0.0028	0.0028	0	0.0026	7.1
MSE_1	5.7×10^{-5}	4.2×10^{-5}	26	1.6×10^{-5}	72.1
t_1 (sec)	10	907		9	
$RMSE_2$	0.0032	0.0033	-3.1	0.0033	-3.1
MAE_2	0.0026	0.0027	-3.8	0.0027	-3.8
MSE_2	1×10^{-5}	1.1×10^{-5}	-3.8	1.1×10^{-5}	-6.7
t_2(sec)	3	205		4	
E_a	0.0108	0.0089	17.6	0.0045	58.3

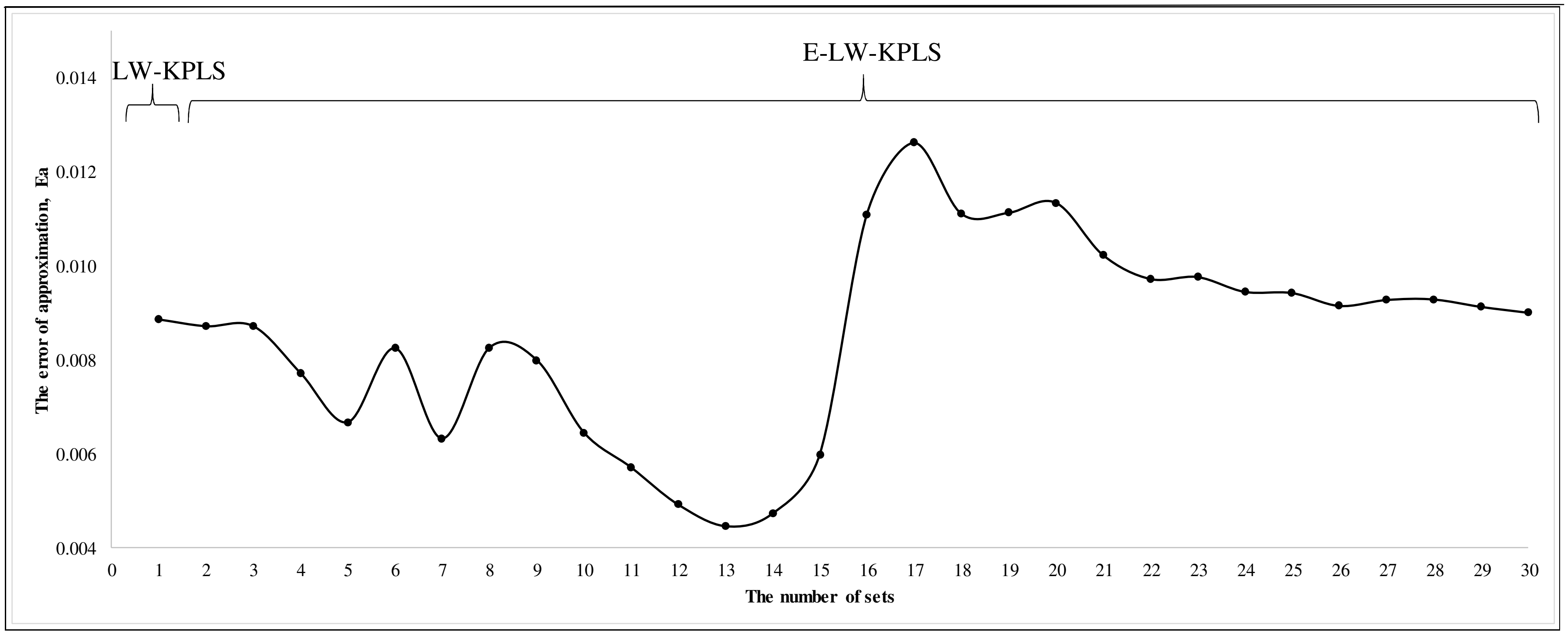

Figure 6.5 The error of approximation for different subsets of the E-LW-KPLS algorithm in case study 2

On the other hand, the t_1 and t_2 for LW-KPLS algorithms are much higher than for E-LW-KPLS and LW-PLS models. Moreover, as shown in the Table 6.8 and Figure 6.2, the computational times for the E-LW-KPLS are up to 100 times smaller than the LW-KPLS while its computation loads are almost the same as the LW-PLS. These results indicate the ensemble method which enables numerous of local LW-KPLS model run spontaneously is a useful tool to reduce the computational burden of LW-KPLS algorithms while providing the E-LW-KPLS better prediction results. From the overall results, E-LW-KPLS model is still preferable compared to the other two algorithms. Figures H.5 and H.6 in Appendix H show the prediction results for training and test data using LW-PLS, LW-KPLS, and E-LW-KPLS models. In Figure H.5, the predicted output from the E-LW-KPLS is slightly closer to real output than LW-KPLS and LW-PLS algorithms. While, in Figure H.6, the predicted output from E-LW-KPLS, LW-KPLS, and LW-PLS models are almost similar and closer to the actual output since the range of the output for testing data is small. Again, in case study 2, the results from E-LW-KPLS model have shown that the objectives of this research were attained.

Case study 3: Wastewater treatment

In case study 3, the inverse multi-quadric Kernel with the value of *b* set at 8 was used as a Kernel function in E-LW-KPLS and LW-KPLS algorithms. Table 6.9 displays current results of the prediction performance for LW-PLS, LW-KPLS and E-LW-KPLS models in case study 3. As shown in Figure 6.6, the E-LW-KPLS algorithm with 12 subsets of the E-LW-KPLS exhibited the lowest value of E_a compared to other E-LW-KPLS models and was therefore selected as the optimal set of E-LW-KPLS algorithms. Additionally, the results for the E-LW-KPLS model with 1 set to 30 subsets are presented in Appendix C.1.3.

As seen in Table 6.9, the E_a for the E-LW-KPLS and LW-KPLS algorithms are 6% and 6.9% lower than the LW-PLS respectively. This result shows the E-LW-KPLS, and LW-KPLS models with inverse multi-quadric Kernel performed better than the LW-PLS. Furthermore, the $RMSE_1$, MAE_1 and MSE_1 for the E-LW-KPLS are smaller than the LW-KPLS algorithm. In addition, the $RMSE_1$ and MSE_1 for the E-LW-KPLS models are slightly more than the LW-PLS and its MAE_1 is 2% lower than the LW-PLS due to the presence of Kernel function. For the test data set, the $RMSE_2$, MAE_1 and MSE_2 for the E-LW-KPLS and LW-KPLS algorithms are smaller than the LW-PLS.

Table 6.9 Prediction performances of the E-LW-KPLS, LW-KPLS and LW-PLS models for case study 3

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**E-LW-KPLS**	**%**
Number of subsets	1	1		12	
Type of Kernel function	-	Inverse multi-quadric Kernel		Inverse multi-quadric Kernel	
Kernel parameter, b	-	8		8	
$RMSE_1$	0.7103	0.7330	-3.2	0.7139	-0.5
MAE_1	0.5811	0.5859	-0.8	0.5693	2
MSE_1	0.5045	0.5373	-6.5	0.5097	-1
t_1 (sec)	11	1067		11	
$RMSE_2$	0.7744	0.7413	4.3	0.7314	5.6
MAE_2	0.6284	0.6053	3.7	0.5966	5.1
MSE_2	0.5997	0.5496	8.4	0.5350	10.8
t_2(sec)	4	238		4	
E_a	0.7905	0.7434	6	0.7358	6.9

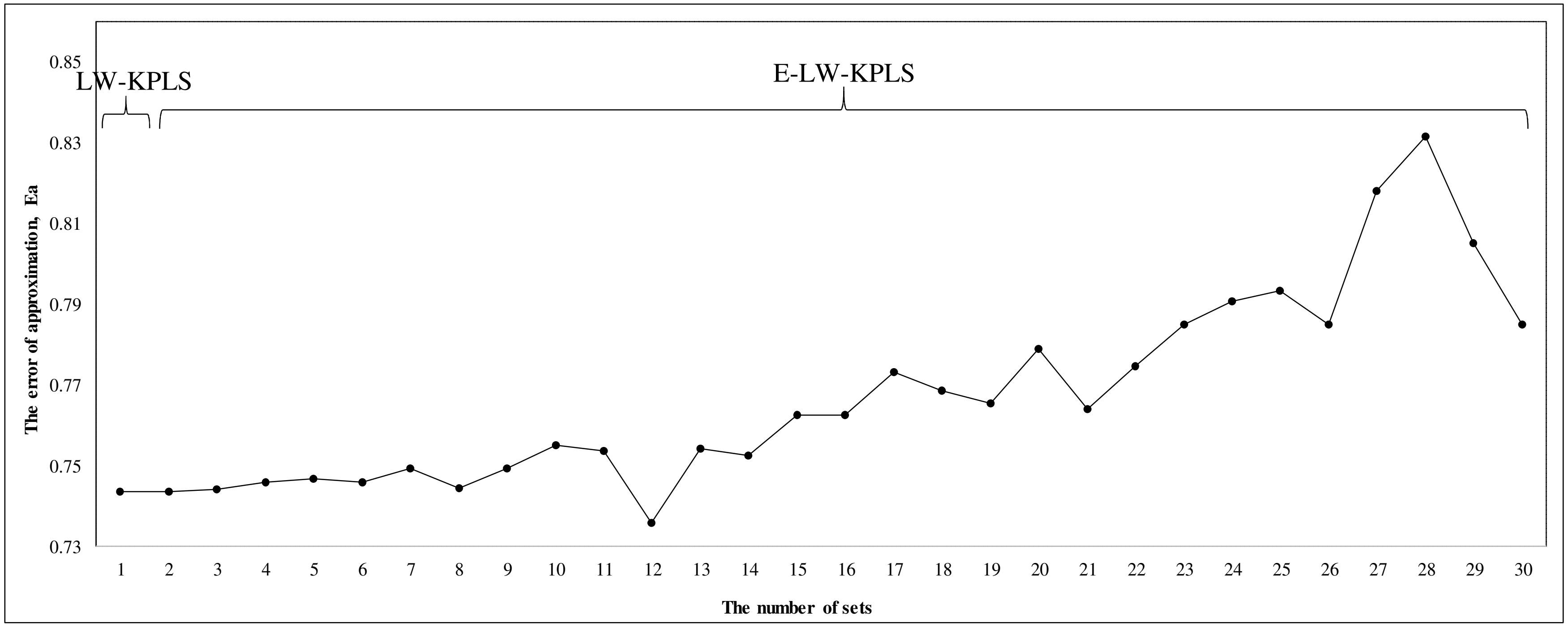

Figure 6.6 The error of approximation for different subsets of the E-LW-KPLS in case study 3

Regarding computational time, as seen in Table 6.9 and Figure 6.3, the t_1 and t_2 for the E-LW-KPLS are up to 100 times smaller than the LW-KPLS. Again, the ensemble method has significantly increased the computational efficiency of the E-LW-KPLS model while providing lower prediction errors than the LW-KPLS. Generally, the E-LW-KPLS offers the best prediction results among the other algorithms.

The current prediction results of output from the LW-PLS, LW-KPLS, and E-LW-KPLS models for training and test data of case study 3 are depicted in Figures H.7 and H.8 in the Appendix H. As illustrated in these figures, the predicted output from the E-LW-KPLS is nearer to the real output than the LW-PLS and LW-KPLS algorithms. Similar to the previous case studies, from the results in case study 3, the objectives of this research were met.

6.6.2. Non-Gaussian distributed data

Non-Gaussian distributed data in case studies 4, 5 and 6 were used to examine the capabilities and effectiveness of this study's newly created non-Gaussian-based LW algorithm, the E-LW-IC-KPLS. Appendix D shows an example of the MATLAB's coding for the E-LW-IC-KPLS. Besides, for comparison, the prediction results from the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms for the case studies 4, 5 and 6 are illustrated, evaluated and discussed in this section. Comparisons of results from these models are carried out to examine the prediction capability of the proposed E-LW-KPLS model, which is also an objective of this research.

In this case study, LW-IC-KPLS model which has a better prediction than LW-PLS in dealing with nonlinear and non-Gassianity data was developed. However, the penalty for this newly proposed model is its high computation loads. This penalty was expected and thus minimising this penalty was included as an objective in this research. Hence, the ensemble method was used to improve the computational efficiency of this proposed LW-IC-KPLS model. Numerous of LW-IC-KPLS models are executed spontaneously. The computational times of training and test data for each set of E-LW-IC-KPLS models in case studies 4 to 6 are illustrated in Figures 6.7 to

6.9. It can be seen that the computation times were reduced dramatically from 1 set to 6 sets in Figures 6.7 and 6.8 and from 1 set to 13 sets in Figures 6.9.

Case study 4: Numerical example 2

The Log Kernel function as shown in Equation (3.9) in Chapter 3 was chosen to be incorporated into the E-LW-IC-KPLS, LW-IC-KPLS, and LW-KPLS models in case study 4. The selected value of *b* in the Log Kernel function was 2 as it gave the best predictive results. The prediction results of the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms in case study 4 for the training and test data are summarised in Table 6.10. Figure 6.10 illustrates the error of approximation for the E-LW-IC-KPLS with unique subsets in case study 4. This figure shows that the E-LW-IC-KPLS models with 2, 3 and 13 subsets have the lowest E_a value and their values are almost identical. However, the E-LW-IC-KPLS with 13 subsets was picked as the optimal set for the E-LW-IC-KPLS algorithm since its $RMSE_1$, MAE_1 and MSE_1 are lower than the other two subsets. Also, the prediction results for other E-LW-IC-KPLS models with a variety of subsets can be seen in the Appendix E.1.1.

Table 6.10 shows both the LW-KPLS and LW-PLS algorithms have quite similar E_a value and prediction errors. Nevertheless, the LW-KPLS has a higher t_1 and t_2 than the LW-PLS. It can be concluded that the LW-KPLS and LW-PLS models have poor predictive qualities in this case study consisting of non-Gaussian distributed data. With the presence of independent component analysis (ICA) in both the E-LW-IC-KPLS and LW-IC-KPLS algorithms, all their prediction errors including E_a are more accurate than the LW-KPLS and LW-PLS models. It means that the LW-IC-KPLS and LW-IC-KPLS algorithms have greater success when dealing with non-Gaussian distributed data than the LW-PLS and LW-KPLS models.

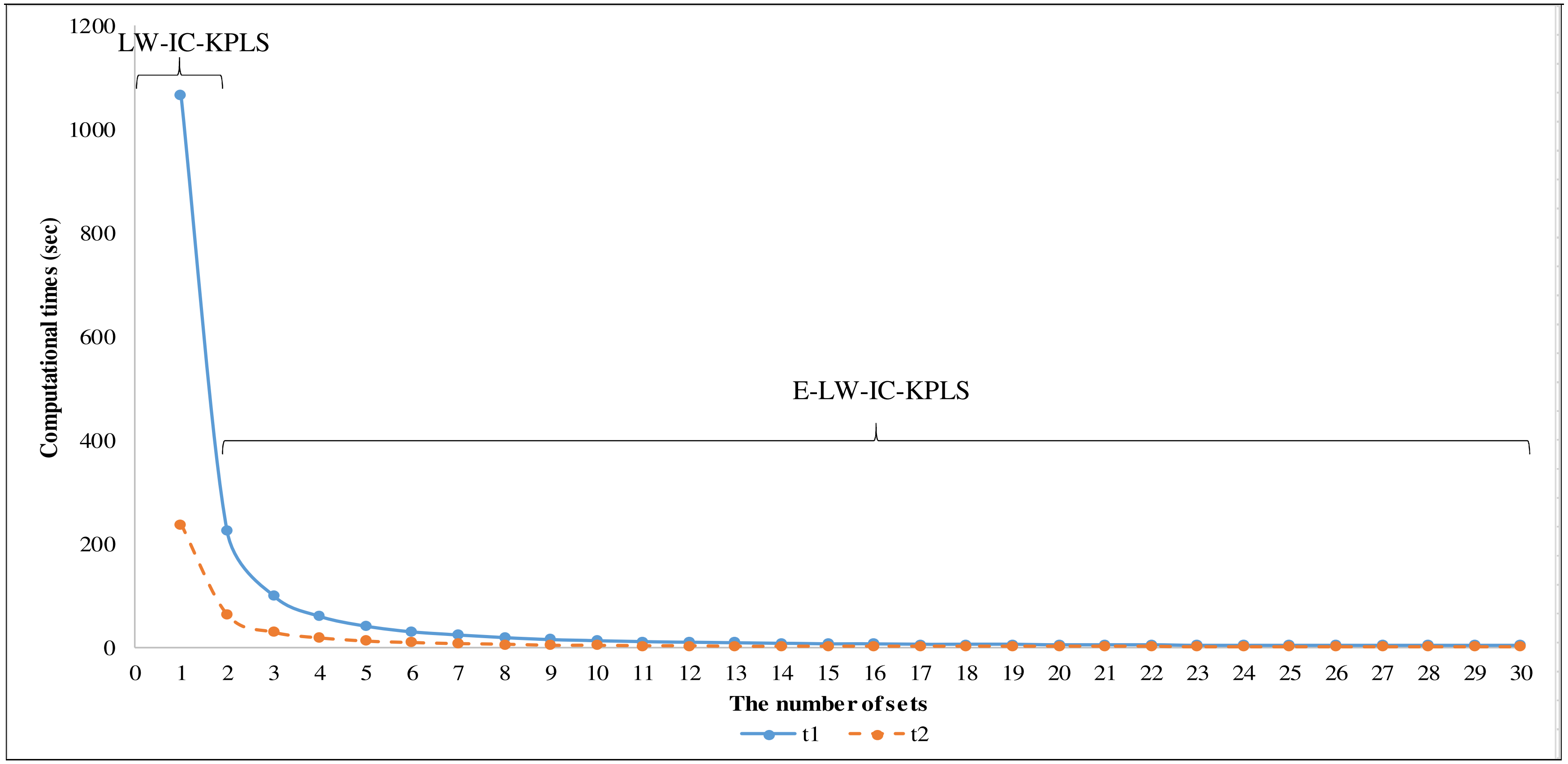

Figure 6.7 Computational times for LW-IC-KPLS and E-LW-IC-KPLS in case study 4

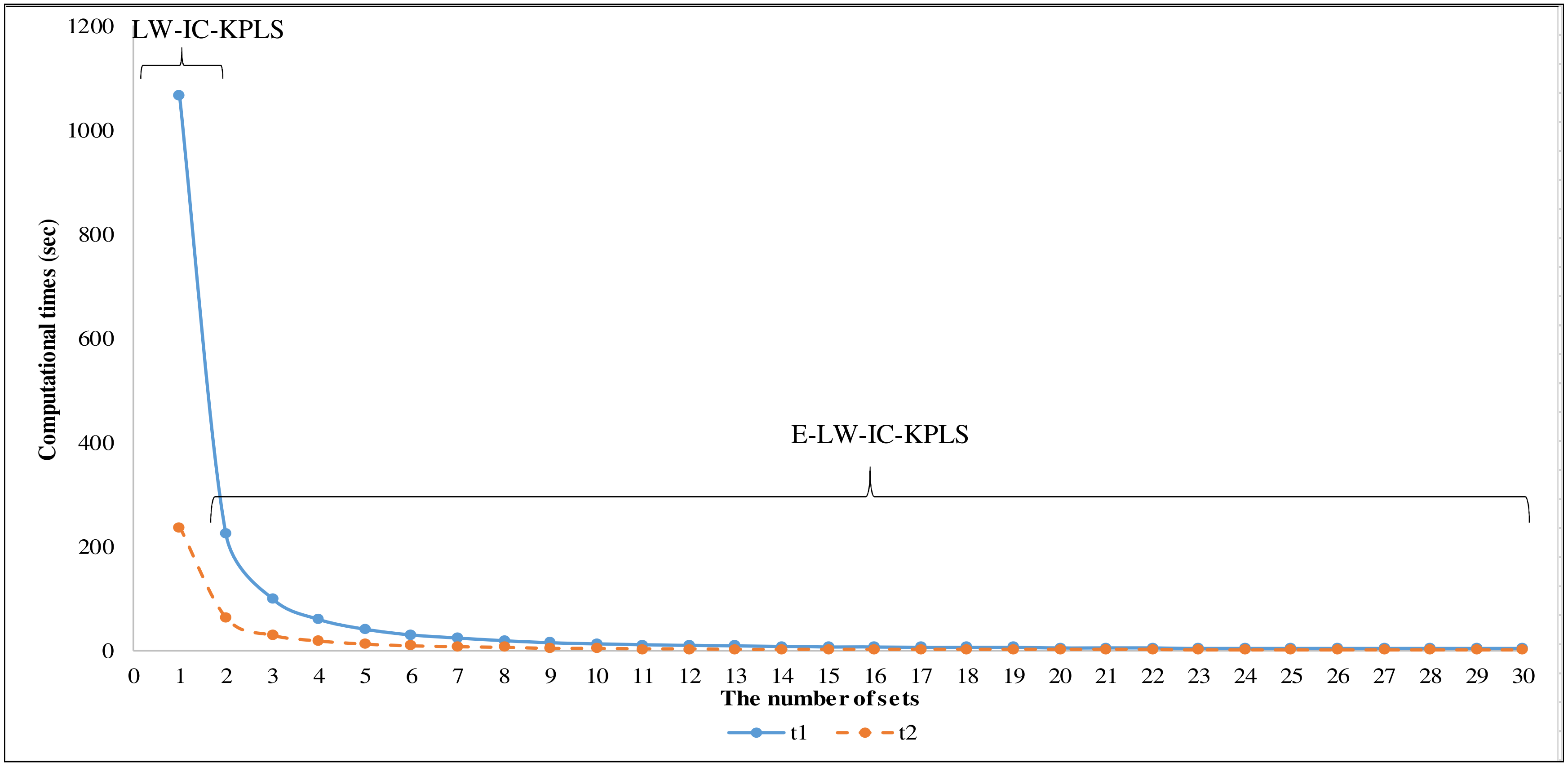

Figure 6.8 Computational times for LW-IC-KPLS and E-LW-IC-KPLS in case study 5

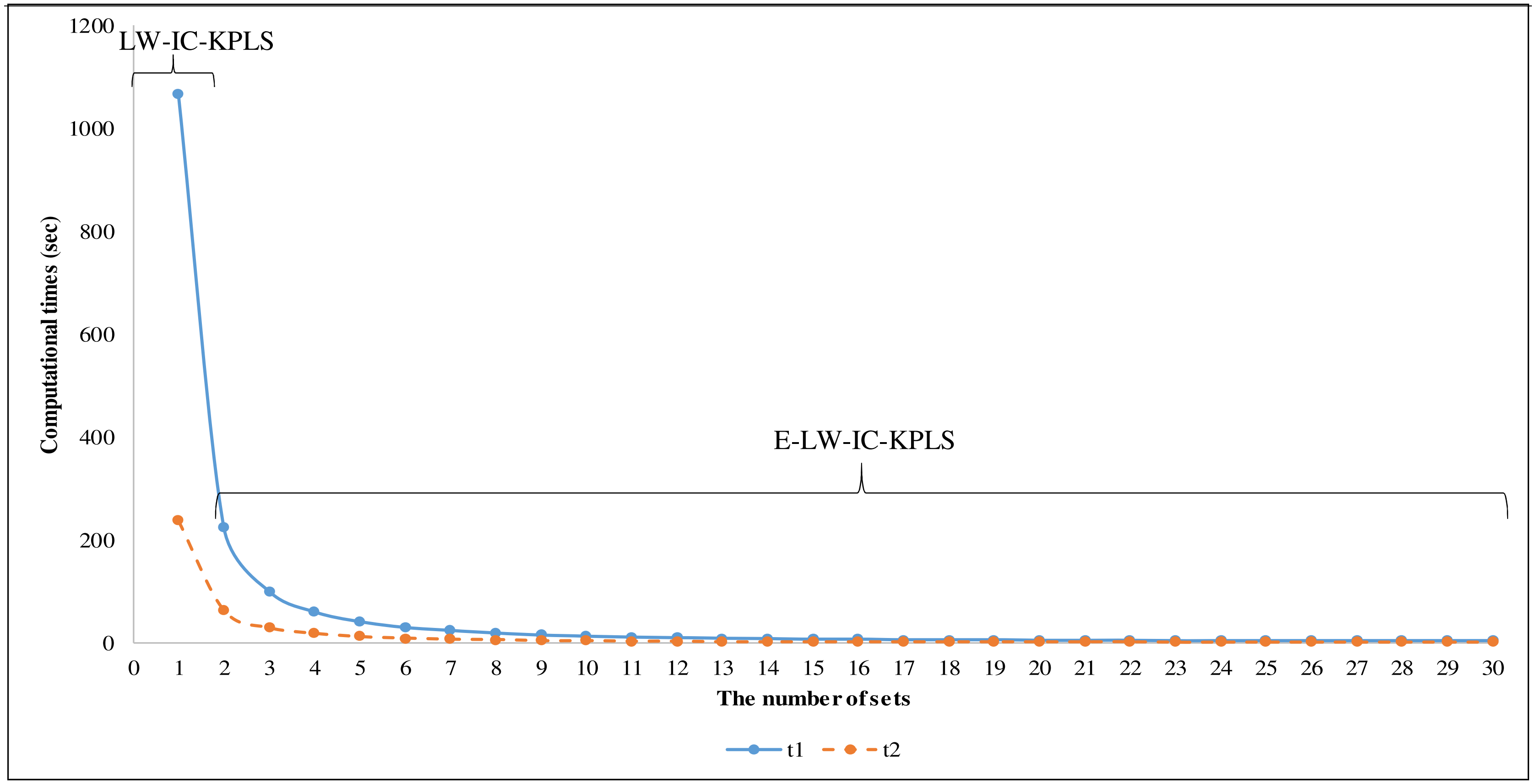

Figure 6.9 Computational times for LW-IC-KPLS and E-LW-IC-KPLS in case study 6

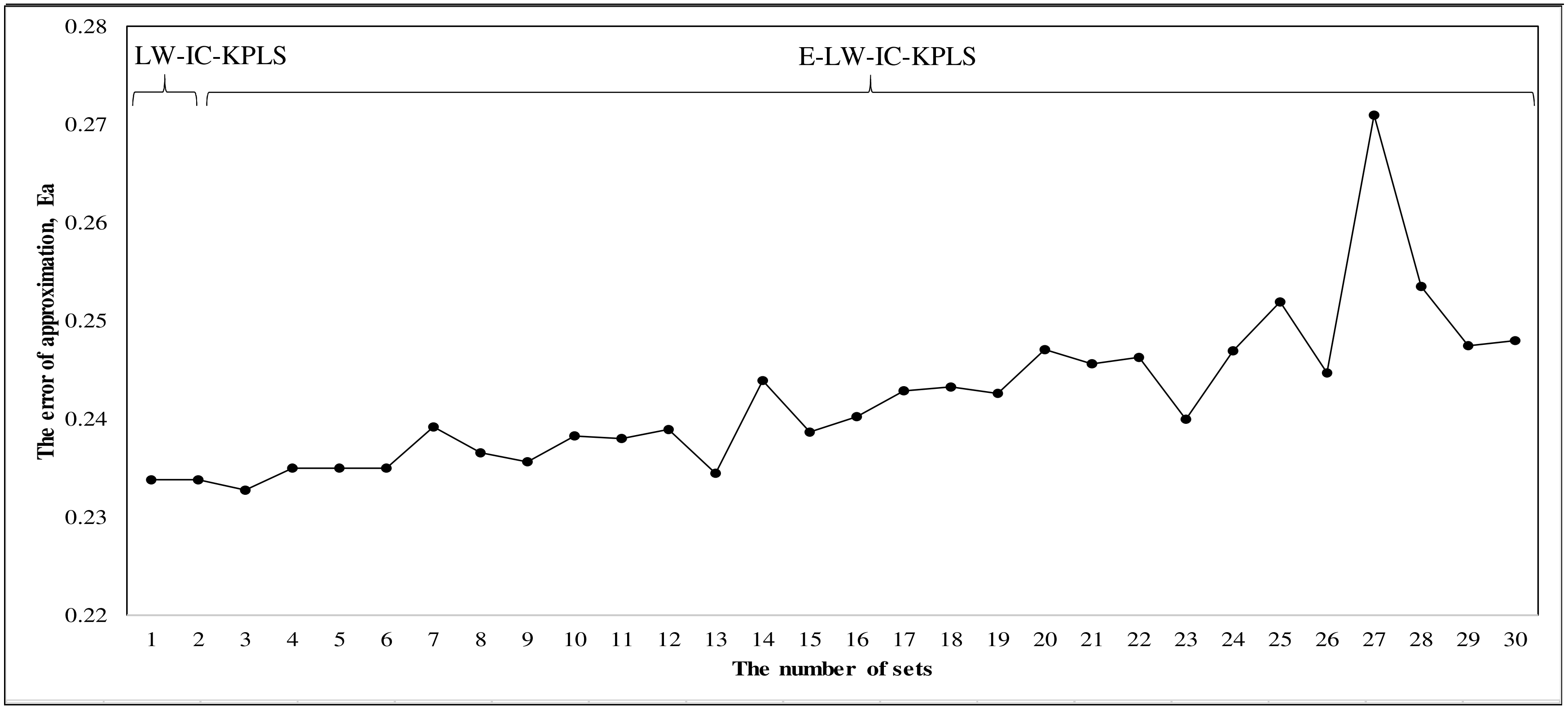

Figure 6.10 The error of approximation for different subsets of the E-LW-IC-KPLS model in case study 4

Table 6.10 Prediction performances of the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 4

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**LW-IC-KPLS**	**%**	**E-LW-IC-KPLS**	**%**
Number of subsets	1	1		1		13	
Type of Kernel function	-	Log Kernel		Log Kernel		Log Kernel	
Kernel parameter, b	-	2		2		2	
$RMSE_1$	0.2574	0.2593	-0.7	0.2269	11.8	0.2187	15
MAE_1	0.2140	0.2149	-0.4	0.1911	10.7	0.1843	13.9
MSE_1	0.0663	0.0672	-1.4	0.0515	22.3	0.0478	27.9
t_1 (sec)	9	449		7146		45	
$RMSE_2$	0.2591	0.2643	-2	0.2322	10.4	0.2313	10.7
MAE_2	0.2170	0.2206	-1.7	0.1971	9.2	0.1945	10.4
MSE_2	0.0671	0.0699	-4.2	0.0539	19.7	0.0535	20.3
t_2(sec)	3	112		1289		11	
E_a	0.2595	0.2656	-2.4	0.2335	10	0.2345	9.6

As shown in Table 6.10, the E-LW-IC-KPLS and LW-IC-KPLS algorithms have E_a values 9.6% and 10% lower than the LW-PLS, respectively. This improvement has resulted in trade-offs where their t_1 and t_2, especially for the LW-IC-KPLS are relatively greater than the LW-PLS. Compared to the LW-IC-KPLS, the E-LW-IC-KPLS model has a slightly higher E_a value but smaller prediction errors. From Table 6.10 and Figure 6.7, the ensemble method has reduced the computational complexity of the LW-IC-KPLS in which the t_1 and t_2 of the E-LW-IC-KPLS are 117 to 159 times lower than the LW-IC-KPLS algorithm. The ensemble method has significantly reduced the computational times of E-LW-IC-KPLS algorithm since it has multiple local LW-IC-KPLS models that are run in parallel and combined to make prediction rather than making prediction from a single LW-IC-KPLS regression model. Hence, it can be concluded the E-LW-IC-KPLS is superior to the LW-IC-KPLS, LW-KPLS and LW-PLS models.

Figures H.9 and H.10 in the Appendix H shows the comparisons of the real output values for training and test data against the predicted outputs from the LW-PLS, LW-KPLS, LW-IC-KPLS, and E-LW-IC-KPLS algorithms. In both figures, the LW-KPLS exhibited unimpressive results, whereas the prediction accuracy of the LW-PLS, LW-IC-KPLSand E-LW-IC-KPLS models were significantly improved. In comparison, the LW-IC-KPLS and E-LW-IC-KPLS algorithms delivered better results than the LW-PLS. From the results in case study 4, it can be concluded that the objectives of this research are accomplished.

Case study 5: Eukaryotic cell cycle regulation

For case study 5, the Log Kernel function was adopted in the E-LW-IC-KPLS, LW-IC-KPLS, and LW-KPLS models while the optimal value for b was 0.8. Figure 6.11 depicts the error of approximation for the E-LW-IC-KPLS with different subsets in case study 5. The E-LW-IC-KPLS with 4 and 13 subsets had the lowest and similar value of E_a. Since $RMSE_1$, MAE_1 and MSE_1 for 13 subsets were lower than 4 subsets, the E-LW-IC-KPLS with 13 subsets was chosen as the optimal set for E-LW-IC-KPLS. Appendix E.1.2 presents the prediction results for the E-LW-IC-KPLS with different subsets. Their predictive results for the training and test data using the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS, and LW-PLS algorithms are tabulated in Table 6.11.

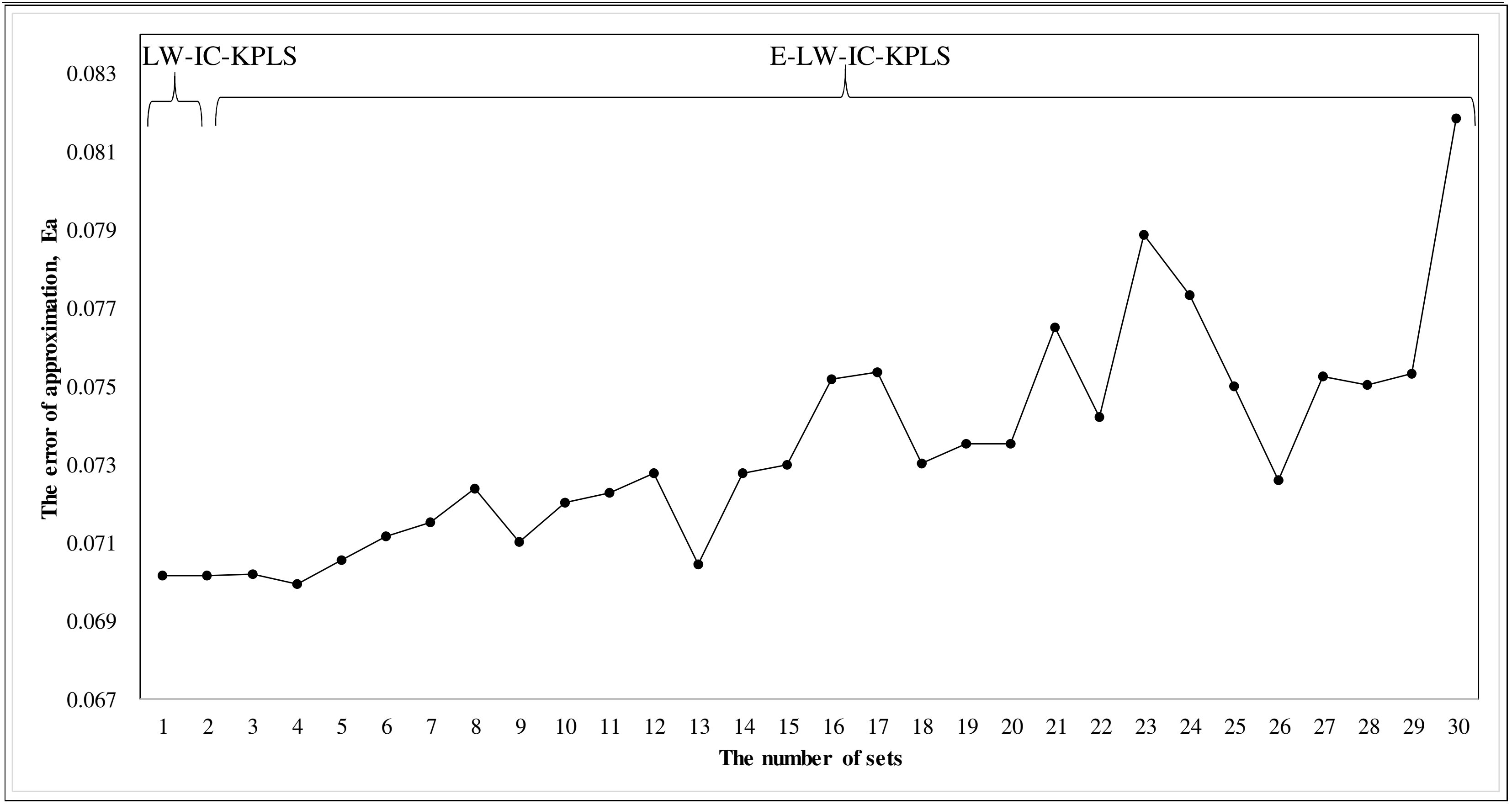

Figure 6.11 The error of approximation for different subsets of the E-LW-IC-KPLS in case study 5

Table 6.11 Prediction performances of the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS and LW-PLS algorithms for case study 5

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**LW-IC-KPLS**	**%**	**E-LW-IC-KPLS**	**%**
Number of subsets	1	1		1		13	
Type of Kernel function		Log Kernel		Log Kernel		Log Kernel	
Kernel parameter, *b*	-	0.8		0.8		0.8	
$RMSE_1$	0.0755	0.0702	7	0.0702	7	0.0675	10.6
MAE_1	0.0641	0.0556	13.3	0.0556	13	0.0533	16.8
MSE_1	0.0057	0.0049	14	0.0049	13	0.0046	19.3
t_1 (sec)	9	521		6850		54	
$RMSE_2$	0.0746	0.0697	6.6	0.0697	6.6	0.0699	6.3
MAE_2	0.0639	0.0554	13.3	0.0553	13.5	0.0549	14.1
MSE_2	0.0056	0.0049	12.5	0.0049	12.5	0.0049	12.5
t_2(sec)	3	140		1310		13	
E_a	0.0761	0.0706	7.2	0.0706	7.2	0.0704	7.5

It is apparent that the LW-KPLS and LW-IC-KPLS models have approximately the same value of prediction errors and E_a as well as their E_a values are 7.2% lower than the LW-PLS. As can be seen in Table 6.11, all their prediction errors are also smaller than the LW-PLS. It can be said that all of the developed models in this research performed better than the benchmarking LW-PLS. When the LW-PLS and E-LW-IC-KPLS algorithms are compared, all prediction errors of the E-LW-IC-KPLS were relatively lower than the LW-PLS. These results demonstrate that the Log Kernel has increased the predictive ability of the LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS models in handling the nonlinear process data. The ICA has allowed the LW-IC-KPLS to perform slightly better than the LW-KPLS since its MAE_2 is lower.

However, the disadvantages of these Kernel-based models are their t_1 and t_2 are much higher than the LW-PLS. In contrast to the LW-KPLS and LW-IC-KPLS algorithms, the ensemble method has reduced the computational loads of the LW-IC-KPLS from 10 to 100 times lower with better predictive results. This is because ensemble method builds a set of local LW-IC-KPLS models and run them simultaneously. Based on the overall application results in Table 6.11, the superiority of the E-LW-IC-KPLS over the other models in estimating the output of nonlinear and non-Gaussian processes has been confirmed. The comparison of the actual output value against the predicted output by the LW-PLS, LW-KPLS, LW-IC-KPLS, and E-LW-IC-KPLS models for training and test data, can be seen in Figures H.11 and H.12 in the Appendix H. In both figures, the predicted outputs from the E-LW-IC-KPLS are closer to the real output compared to other models. Hence, the results in this case study show that the objectives of this research are successfully achieved.

Case study 6: A highly nonlinear CSTR

In case study 6, a Log Kernel function was chosen to be incorporated into the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS algorithms. The value of b in this Kernel function was tuned to 9 since this gave the best predictive results. This case study is different from other case studies as it has two outputs, product concentration, and reactor temperature. Hence, there are three E_a values in Figure 6.12. The E_{a1}, and E_{a2} are E_a values for product concentration, reactor temperature, respectively while E_{a3} is the average value of the E_{a1}, and E_{a2}. In Figure 6.12, it can be noticed that the E_{a2}

values for the E-LW-IC-KPLS with 14 and 27 subsets are smaller than other subsets and quite similar. Nevertheless, E_{a1} for the E-LW-IC-KPLS with 27 subsets was lower than 14 subsets. Furthermore, the E_{a3} for the E-LW-IC-KPLS with 27 subsets has the lowest values among all of the subsets. Thus, the E-LW-IC-KPLS with 27 subsets was adopted as the optimal set for the E-LW-IC-KPLS model.

Table 6.12 Prediction performances of product concentration using the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 6

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**LW-IC-KPLS**	**%**	**E-LW-IC-KPLS**	**%**
Number of subsets	1	1		1		27	
Type of Kernel function	-	Log Kernel		Log Kernel		Log Kernel	
Kernel parameter, b	-	9		9		9	
$RMSE_1$	0.0155	0.0153	1.3	0.0153	1.3	0.0127	18.1
MAE_1	0.0111	0.0110	0.9	0.0110	0.9	0.0092	17.1
MSE_1	$2.4\text{x}10^{-4}$	$2.4\text{x}10^{-4}$	2.5	$2.4\text{x}10^{-4}$	2.5	$1.6\text{x}10^{-4}$	32.8
t_1 (sec)	17	1978		18307		106	
$RMSE_2$	0.0074	0.0076	-2.7	0.0079	-6.8	0.0106	-43.2
MAE_2	0.0061	0.0063	-3.3	0.0065	-6.6	0.0083	-36.1
MSE_2	$5.5\text{x}10^{-5}$	$5.8\text{x}10^{-5}$	-4.7	$6.3\text{x}10^{-5}$	-13.2	$1.1\text{x}10^{-4}$	-102
t_2(sec)	6	446		3243		25	
E_{a1} (1st output)	0.0216	0.0211	2.3	0.0209	3.2	0.0143	33.8

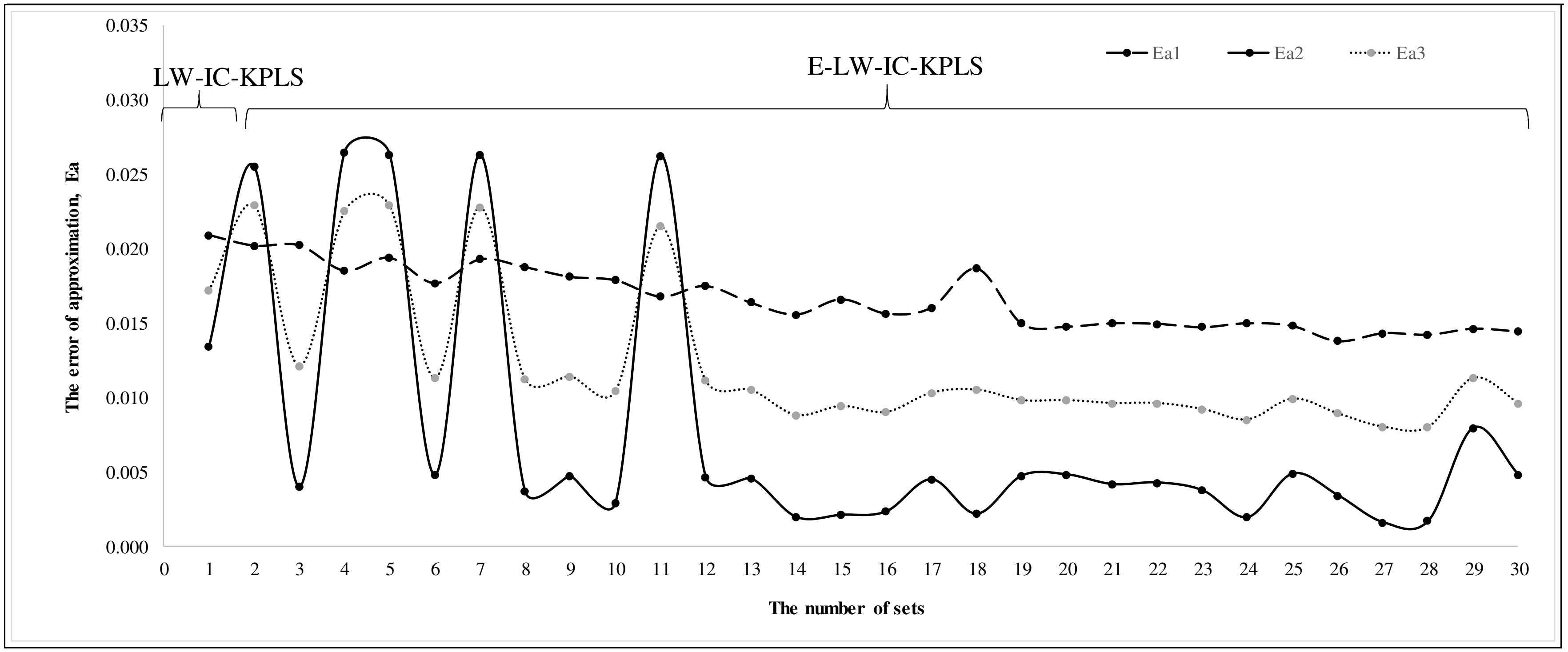

Figure 6.12 The error of approximation for different subsets of the E-LW-IC-KPLS in case study 6

The prediction results for the E-LW-IC-KPLS with different subsets are given in Appendix E.1.3. Next, the prediction results of the product concentration and reactor temperature for the new E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms in case study 6 for the training and test data are presented in Tables 6.12 and 6.13 respectively. As these two tables show, the three E_a values for the E-LW-IC-KPLS show significant improvement compared to the other algorithms. In Table 6.12, the E_{a1} values for the LW-KPLS and LW-IC-KPLS algorithms were lower by 2.3% and 3.2% as compared to the LW-PLS respectively. Irrespectively, from Table 6.12, the E-LW-IC-KPLS which utilises ensemble method has a 33.8% smaller value of E_{a1} than the LW-PLS.

It can be observed from Table 6.12 that the prediction errors of training data $RMSE_1$, MAE_1 and MSE_1 for the LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS models were lower than the LW-PLS. The integrated Log Kernel function enhanced the predictive capability of these algorithms concerning training data for product concentration. However, their prediction errors for test data ($RMSE_2$, MAE_2 and MSE_2) are higher than the LW-PLS. Similar to case study 2, this was due to the presence of unsteady state conditions in the first 2000 data sets of product concentration (output) in training data which affect the accuracy of models to capture the nonlinearity and non-Gausianty of data.

Moreover, due to this condition, the prediction errors of the training data for the LW-PLS, LW-KPLS, and LW-IC-KPLS algorithms are also much higher than the prediction errors of their test data. Moreover, due to the presence of the Log Kernel function, the t_1 and t_2 for the LW-KPLS and LW-IC-KPLS models are also much greater than for the LW-PLS and E-LW-IC-KPLS algorithms. From Table 6.12 and Figure 6.9, the ensemble method is used in the E-LW-IC-KPLS to lower the computational times, and it has significantly reduced the computational times of E-LW-IC-KPLS algorithm.

Table 6.13 Prediction performances of reactor temperature using the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS models for case study 6

Algorithms	**LW-PLS**	**LW-KPLS**	**%**	**LW-IC-KPLS**	**%**	**E-LW-IC-KPLS**	**%**
Number of subsets	1	1		1		27	
Type of Kernel function	-	Log Kernel		Log Kernel		Log Kernel	
Kernel parameter, *b*	-	9		9		9	
$RMSE_1$	0.0152	0.0086	43.4	0.0086	43.4	0.0011	92.8
MAE_1	$5.9\text{x}10^{-4}$	$3.9\text{x}10^{-4}$	33.6	$3.9\text{x}10^{-4}$	33.6	$2.5\text{x}10^{-4}$	57.8
MSE_1	$2.3\text{x}10^{-4}$	$7.4\text{x}10^{-5}$	68.1	$7.4\text{x}10^{-5}$	68.1	$1.2\text{x}10^{-6}$	99.5
t_1 (sec)	17	1978		18307		106	
$RMSE_2$	$5.4\text{x}10^{-4}$	$2.7\text{x}10^{-4}$	50.3	0.0022	-308.2	$4.8\text{x}10^{-4}$	11.1
MAE_2	$3.1\text{x}10^{-4}$	$1.9\text{x}10^{-4}$	37.4	$4.1\text{x}10^{-4}$	-30.6	$2.4\text{x}10^{-4}$	23.2
MSE_2	$2.9\text{x}10^{-7}$	$7.2\text{x}10^{-8}$	75.3	$4.8\text{x}10^{-6}$	-1549.5	$2.3\text{x}10^{-7}$	21.3
t_2(sec)	6	446		3243		25	
E_{a2} (2nd output)	0.0262	0.0148	43.5	0.0134	48.9	0.0016	93.9
E_{a3} (average)	0.0239	0.0180	24.7	0.0172	28.0	0.0080	66.5

From Table 6.13, the E_{a2} values for the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS models were improved by about 93.9%, 48.9% and 43.5% in comparison with LW-PLS, respectively. Moreover, the E_{a3} value, which is the average value for both E_{a1} and E_{a2} was lower by 66.5% compared to LW-PLS model and it is the lowest E_{a3} value among others. Furthermore, all their prediction errors are also lower than the LW-PLS. Nonetheless, the t_1 and t_2 for the LW-IC-KPLS and LW-KPLS are relatively higher than for the E-LW-IC-KPLS and LW-PLS algorithms. The ensemble method was utilised in the E-LW-IC-KPLS to minimise the computational times of the LW-IC-KPLS. It resulted in a reduction of approximately 173 times lower than the LW-IC-KPLS since the execution time for several local LW-IC-KPLS models is lesser than a LW-IC-KPLS model with the same amount of data. Furthermore, the addition of the ensemble approach and the ICA in the E-LW-IC-KPLS has improved its prediction errors. In general, the E-LW-IC-KPLS is superior to the LW-IC-KPLS, LW-KPLS and LW-PLS models concerning predictive performance.

The comparisons of the real output value with the predicted outputs of product concentration from the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms for training and test data are plotted in Figures H.13 and H.15 respectively in Appendix H. The true output value of the reactor temperature compared to the predicted output from the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS models for training and test data are displayed in Figures H.14 and H.16 respectively in the Appendix H. In these figures, the majority of the predicted outputs from the E-LW-IC-KPLS are closer to the right output value as compared to other algorithms. On the other hand, even the predictive errors of the E-LW-IC-KPLS for training data ($RMSE_2$, MAE_2 and MSE_2) shown in Table 6.13 are not as good as LW-KPLS. However, its overall performance is better since its E_{a3} value is smaller than LW-KPLS.

6.6.3. Missing data

Different missing data levels - 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55% and 60% - were considered in case studies 4, 5 and 6. The simulated data in these case studies were used to evaluate the capabilities and effectiveness of the current EM-E-LW-IC-KPLS algorithm. Also, the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS, and SVD-LW-PLS models were also tested using case studies 4, 5 and 6 with the presence of missing data. The Kernel functions and Kernel parameters that were used in the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, and SVD-E-LW-IC-KPLS algorithms are the same in Section 6.8.2.

Computation times of these models have not been discussed since they are the same. The results of the abovementioned algorithms are summarised, compared and discussed in this section. Only E_a values are evaluated and considered as they present the predictive errors for both training and test data**.** Additionally, a sample of the MATLAB's coding for the EM-E-LW-IC-KPLS is listed in the Appendix F. Moreover, for case studies 4 to 6, the results of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS, and SVD-LW-PLS algorithms in different missing data levels are summarised in the Appendix G.

Case study 4: Numerical example 2 (Non-Gaussian and missing data)

Figure 6.13 shows the E_a values for different missing data ratio of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms in case study 4. Besides, Figure 6.13 also shows that the EM-E-LW-IC-KPLS and TSR-E-LW-IC-KPLS models are statistically superior to the other models when dealing with 5% to 20% of the missing data. This result is due to both EM, and TSR models imputed more suitable values than the SVD method. Moreover, the E-LW-IC-KPLS in EM-E-LW-IC-KPLS and TSR-E-LW-IC-KPLS models have better predictive performance than LW-PLS model in dealing with non-Gaussian distributed data. However, the EM-E-LW-IC-KPLS and TSR-E-LW-IC-KPLS models appear to be unstable with 35% and 55% of the missing data. As seen in Tables G.1.1.7 and G.1.1.11 in the Appendix G, their test data performed poorly in this volume of missing data since the prediction errors for test data suddenly increased.

The SVD-E-LW-IC-KPLS and SVD-LW-PLS model overestimated the association, and their E_a values tended to be higher and more unreliable as compared to other models when the percentage of missing data increased. This result indicates that the precision of the imputed data from SVD model is lesser than the one from EM and TSR methods. Notice that SVD-E-LW-IC-KPLS algorithm became unstable at 40% of missing data. From Table G.1.1.8, it can say that the imputed data from SVD-E-LW-IC-KPLS algorithm has resulted in the predictive errors for test data became high. Anyhow, it can be concluded that EM-E-LW-IC-KPLS model performed better at up to 20% of missing data in case study 4 as compared to EM-LW-PLS, TSR-LW-PLS, SVD-LW-PLS, and SVD-E-LW-IC-KPLS models. This result has shown that the objectives of this research were attained since the performance of newly developed EM-E-LW-IC-KPLS model is better than the integration of the benchmarking LW-PLS with EM, TSR, and SVD methods.

Case study 5: Eukaryotic cell cycle regulation (Non-Gaussian and missing data)

In case study 5, the E_a values for various missing data ratios of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS, and SVD-LW-PLS algorithms are summarised in Figure 6.14. As shown in Figure 6.14, the EM-E-LW-IC-KPLS and TSR-E-LW-IC-KPLS models performed much better than the EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms in which their E_a values were lower at all of the percentages of missing data (5% to 60%). Similar to case study 4, it is due to the E-LW-IC-KPLS model performed better than LW-PLS model in data that is non-Gaussian distributed. Figure 6.14 also indicates the EM and TSR models provided more realistic imputed data for missing data than the SVD method.

On the other hand, it can be seen that the SVD-E-LW-IC-KPLS model has the highest E_a value at 20% of missing data. From Table G.1.2.4, the estimates from the SVD-E-LW-IC-KPLS model for test data with 20% of missing data are less precise. The overall results exhibit the capability of the EM-E-LW-IC-KPLS model is better than the existing LW-PLS models that were incorporated with EM, TSR and SVD models. Therefore, it can be said that the objectives of this research were met.

Case study 6: A highly nonlinear CSTR (Non-Gaussian and missing data)

As mentioned earlier in Section 6.8.2, there are two output variables, production concentration and reactor temperature in case study 6. The E_a values of these output variables for the missing data ratios of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS and SVD-E-LW-IC-KPLS models in case study 6 can be found in Figures 6.15 and 6.16. It is obvious that the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS and SVD-LW-IC-KPLS models worked much better than the LW-PLS models with EM, TSR and SVD models. This result denotes that their E-LW-IC-KPLS models have more accurate prediction performance than the LW-PLS model in nonlinear and non-Gaussian distributed data.

Also, it can be seen from Figure 6.15 that all of the E_a values for the EM-E-LW-IC-KPLS were less frustrated with missing proportions from 5% to 60%. This result shows the EM-E-LW-IC-KPLS was more stable compared to the TSR-E-LW-IC-KPLS and SVD-E-LW-IC-KPLS algorithms. Notice that these E-LW-IC-KPLS based models did not significantly outperform than any other since they have very similar imputed values for the missing data. Overall, the EM-E-LW-IC-KPLS model is more stable as compared to the TSR-E-LW-IC-KPLS and SVD-E-LW-IC-KPLS algorithms while it performed much better than EM-LW-PLS, TSR-LW-PLS, and SVD-LW-PLS algorithms.

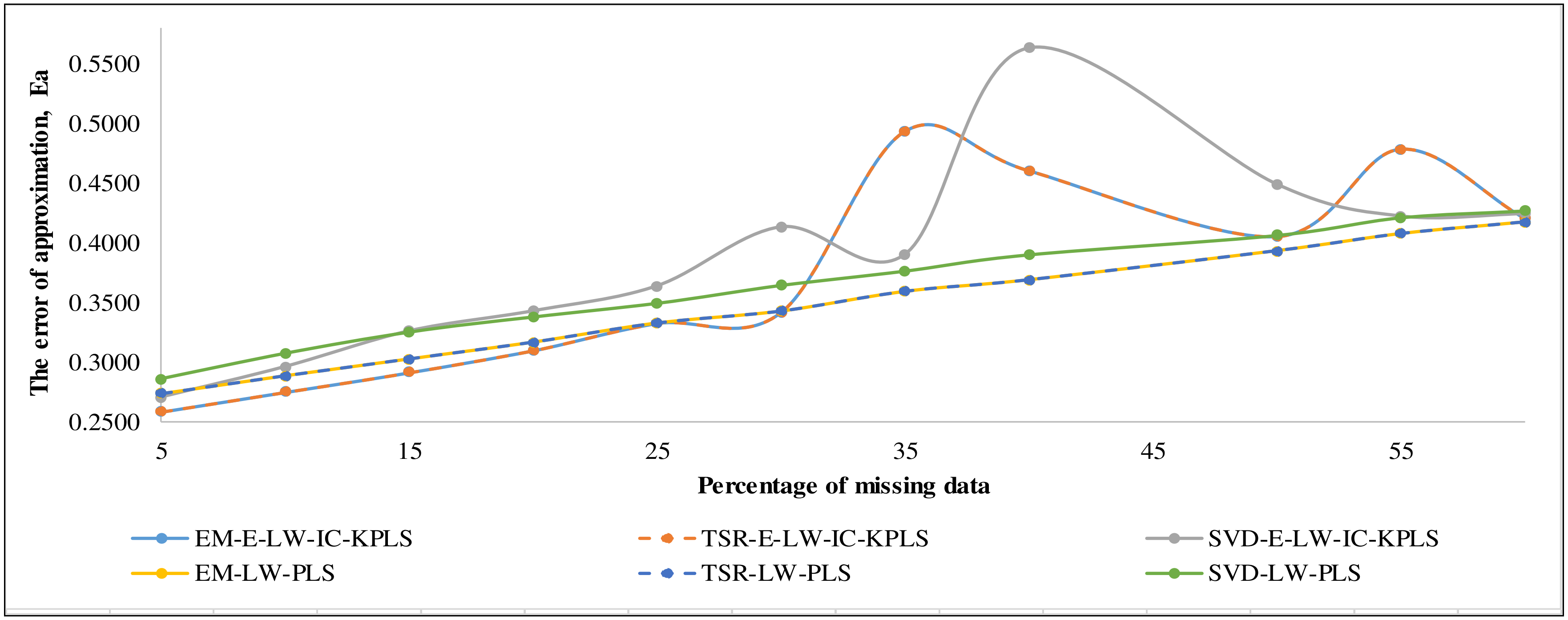

Figure 6.13 The error of approximation for assorted missing data ratios of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms in case study 4

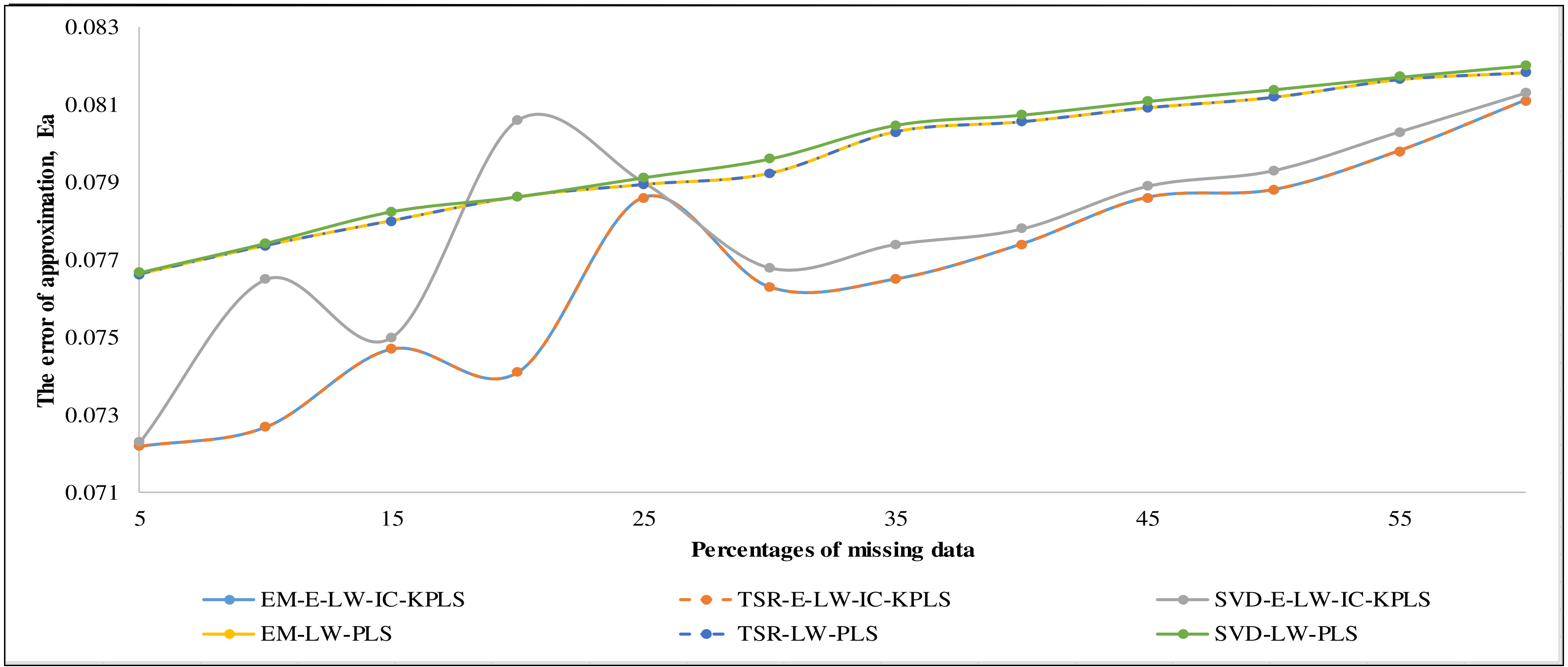

Figure 6.14 The error of approximation for assorted missing data ratios of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms in case study 5

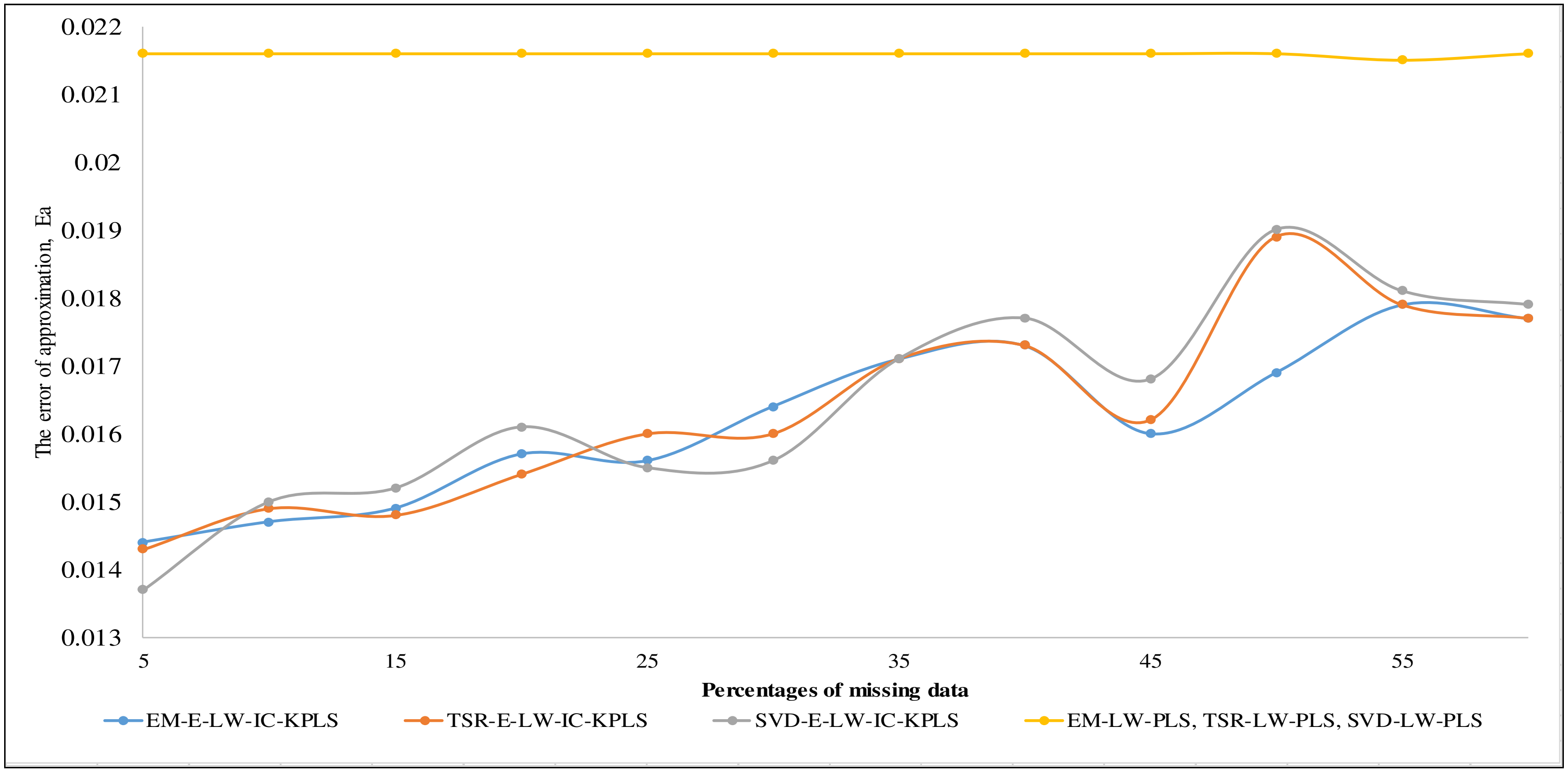

Figure 6.15 The error of approximation of product concentration for various missing data ratios of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms in case study 6

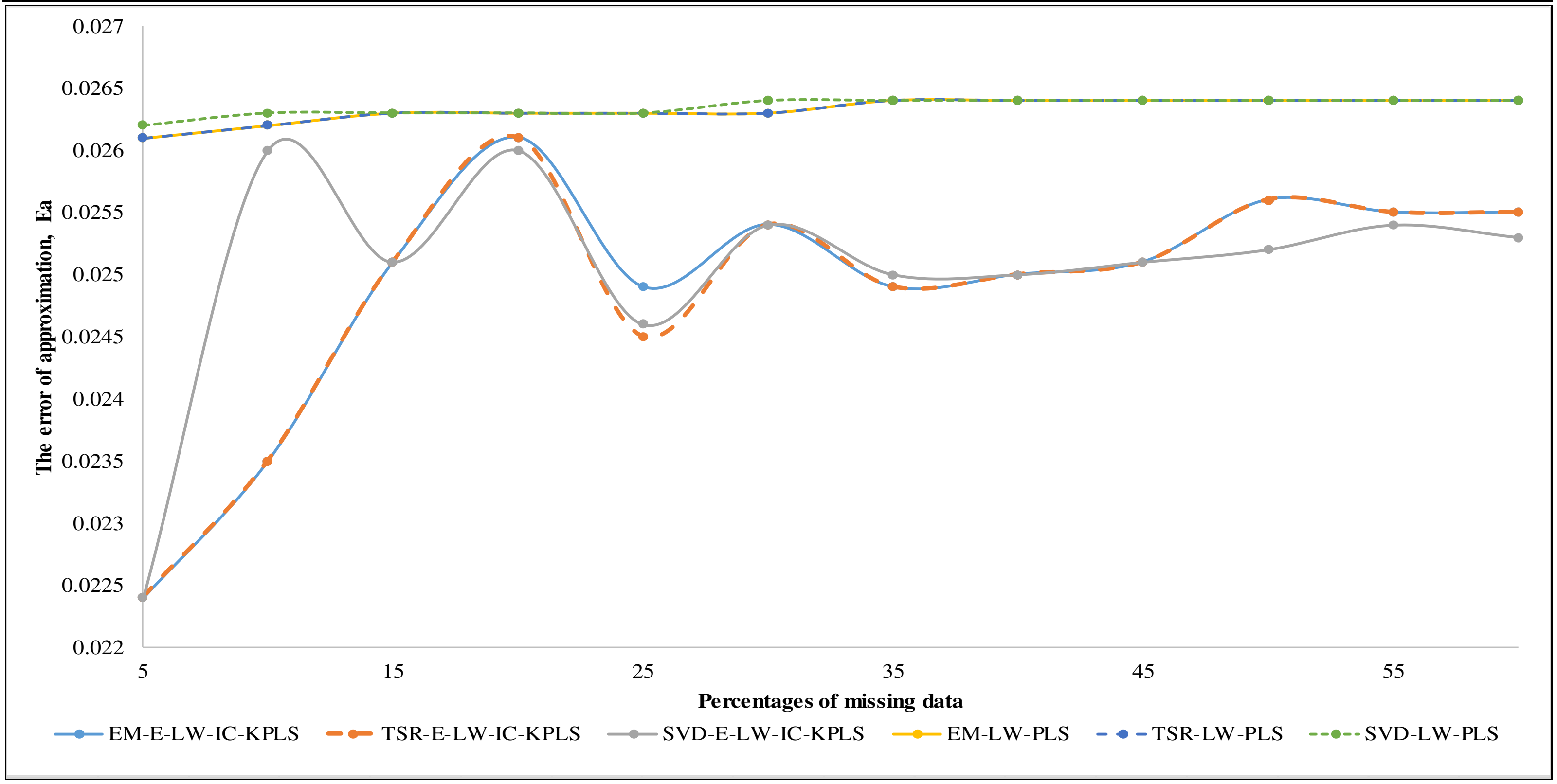

Figure 6.16 The error of approximation of reactor temperature for missing data ratios of the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS models in case study 6

As shown in Figure 6.16, the E_a values of the reactor temperature for the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS and SVD-E-LW-IC-KPLS models were smaller than the integration of LW-PLS models with missing data imputation methods. This result shows that E-LW-IC-KPLS based models outperformed than the LW-PLS based models in case study 6. Notice that the E_a values for these E-LW-IC-KPLS based models are relatively similar when handling 15% to 60% of missing data. This result is due to the range of the output variables, which is the reactor temperature is small, and the imputed values from these models are quite similar. Anyway, the newly developed EM-E-LW-IC-KPLS model achieved high prediction accuracy than the existing integrated LW-PLS models with EM, TSR and SVD models. Thus, the objectives of this research were accomplished.

6.7. Conclusion

The predictive performance of the LW-KPLS is much better than for the LW-PLS in case study 1. Nevertheless, it has a much higher computational load, and this is the penalty of the improved algorithm. The ensemble method was employed in the E-LW-KPLS to reduce the computational load of the LW-KPLS while it has almost the same predictive performance as the LW-KPLS. In case study 1, as compared to the LW-PLS, the newly developed E-LW-KPLS still has a higher computational time since it consists of only three subsets. Again, its penalty, the computational time could be further reduced by having more subsets, though its prediction performance might become less impressive but still better than the LW-PLS. In case studies 2 and 3, the results showed the E-LW-KPLS provided better prediction capabilities than of the LW-PLS and LW-KPLS models. On the other hand, the computational load for the E-LW-KPLS is the same as for the LW-PLS as it had more than ten subsets.

In addition, in case study 2, the presence of an unsteady state condition in the training data for the output variable makes the E-LW-KPLS and LW-KPLS algorithms produce a slightly poorer predictive performance on test data when compared to the LW-PLS. Moreover, this condition also results in all these models having higher prediction errors on the training data than the test data where it is usually the opposite in most cases. Regardless, in general, case studies 1, 2 and 3 have shown that the E-LW-KPLS performs better than the LW-PLS in dealing with nonlinear and Gaussian distributed process data where the E_a values of the E-LW-KPLS can be improved by approximately 7% to 54%. Based on the results in case studies 1 to 3, it can be said that the objectives stated in Chapter 1 were achieved.

For case studies 4 to 6, the LW-IC-KPLS and E-LW-IC-KPLS models performed better than the LW-PLS algorithm in dealing with nonlinear and non-Gaussian distributed data. This result is due to the presence of independent component analysis in the model which works well with non-Gaussian distributed data. Similar to the idea of the E-LW-KPLS, to minimize the computation load of the LW-IC-KPLS, the ensemble method was applied in which numerous sets of the LW-IC-KPLS were run simultaneously. In addition, notice that Log Kernel function is preferable for non-Gaussian distributed data since it attains good prediction performance.

Case study 6 also had an unsteady state condition in the training data of the product concentration as in case study 2. Therefore, the prediction errors for all models on the training data of production concentration were higher than the test data. Additionally, this condition has resulted in the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS models performing more poorly on test data compared to the LW-PLS. However, in case study 6, the general predictive performance of the E-LW-IC-KPLS was better than the LW-IC-KPLS, LW-KPLS and LW-PLS algorithms. For case studies 4 to 6, which showed nonlinear and non-Gaussianity conditions, the E-LW-IC-KPLS showed better predictive results as compared to the other algorithms. The E_a values for the E-LW-IC-KPLS improved 7.5% to 93.9% as compared to the LW-PLS in coping with nonlinear and non-Gaussian distributed data. Hence, this result has shown that the objectives of this research were attained.

In addition, five out of the six case studies (case studies 2 to 6) had good predictive performances at 12, 13 or 14 subsets of the improved E-LW-KPLS and E-LW-IC-KPLS algorithms. Hence, these subsets are the optimal subsets for the newly developed algorithms. Additionally, as mentioned earlier, the inverse multi-quadric Kernel function is more suitable for processes that behave nonlinearly and Gaussianity and process data with a large range of input variables such as data in case studies 1 and 3 (refer to Table 3.1 in Chapter 3). Besides, case study 2 which also behaves nonlinear and Gaussianity has the presence of an unsteady state in the training data of the output and small range of input variables. It was found that the polynomial Kernel to be preferable rather than the inverse multi-quadric Kernel function. Likewise, the Log Kernel function was found to be more appropriate for nonlinear and non-Gaussian distributed process data such as data in case studies 4 to 6. This data follows the non-Gaussianity types including plateau, multimodal, exponential and bimodal.

As mentioned earlier in Chapter 2, the LW-PLS based algorithms including the LW-KPLS, E-LW-KPLS and E-LW-IC-KPLS models can cope with outliers. This advantage is one of the reasons the LW-PLS-based algorithms were chosen to be studied in this research. As shown from the above results, the *RMSE* of the training and test data for all the case studies were more than *MAE*. This result indicates the presence of outliers in the process data in all the case studies (Bratu 2013; Legates and Gregory 1999). It can also be seen the newly developed LW algorithms can perform with the existence of outliers.

Based on the basic idea mentioned in Chapter 3, the EM has been incorporated into the newly developed E-LW-IC-KPLS model to address missing data problems as it has less sampling variability. This newly developed just-in-time based algorithm is referred to as the EM-E-LW-IC-KPLS model. In case study 4, it can be seen that this model performed better than the LW-PLS model integrated with missing data imputation methods (EM, TSR, and SVD models) when dealing with up 20% of missing data.

From case studies 5 and 6, EM-E-LW-IC-KPLS model was found to have better performance as compared to the EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS models in the presence of 5% to 60% of missing data. Moreover, the EM-E-LW-IC-

KPLS and TSR-E-LW-IC-KPLS models have practically equivalent ability since they yielded similar predictive results. Generally, the results showed the EM-E-LW-IC-KPLS model is superior to the LW-PLS model integrated with EM, TSR and SVD models when the proportions of missing values were not exceeded 20%. The results have also demonstrated the effectiveness of the EM-E-LW-IC-KPLS against missing data which means that the objectives of this research were successfully attained.

As reported earlier in Chapter 1, the possible performance trade-off of applying the newly improved and integrated adaptive algorithms is their high computational efficiency. Integration of different algorithms increases the computational complexity of the developed algorithms and results in high computational load. By using six different case studies, it had been proven that the ensemble method in the newly proposed E-LW-KPLS, E-LW-IC-KPLS and EM-E-LW-IC-KPLS algorithms could significantly reduce their computational burden as compared to their single regression model. In the meanwhile, these algorithms showed more accurate predictive performance as compared to the popular LW-PLS model which become problematic when handling with nonlinear, non-Gaussian distributed and missing data.

Chapter 7

Conclusions and recommendations

7.1. Introduction

This research work aimed to create a novel adaptive algorithm that can control for nonlinear and non-Gaussian distributed data and with the presence of missing data. In this research work, the formulated algorithm is called the expectation maximization ensemble locally weighted (LW) independent component kernel partial least square (EM-E-LW-IC-KPLS). In this chapter, the research summary, and conclusions, as well as the contributions to the development of an adaptive soft sensor, are described. Finally, recommendations for future research on adaptive LW algorithm modelling are suggested.

7.2. Research summary and conclusions

Soft sensing technologies, which refer to data-driven and model-based soft sensors, are becoming more popular since they can solve problems associated with hardware counterparts. As stated earlier, issues with hardware include time-consuming maintenance, the need for calibration, aged deterioration, insufficient accuracy, etc. Soft sensors offer solutions to some of these such as low cost, minimal time required, and more efficient use of resources. However, as also mentioned earlier, model-based soft sensors are sometimes inefficient when accurate details of model parameters are incomplete. Hence, data-driven soft sensors, which include historical data to improve prediction capabilities, are far more preferable. These soft sensors have to be adaptive to avoid inefficiencies due to changes in the state of plants and process characteristics.

In this research work, just-in-time (JIT) based adaptive algorithms including LW algorithms are the focus. JIT-based algorithms address the limitations of other adaptive models that these particular algorithms can be applied to processes that require abrupt changes, and they have a low computation load. A widely used JIT-based adaptive soft sensor the so-called LW partial least square (LW-PLS) was the benchmark technology in this research. This research aimed to develop an improved LW-PLS algorithm that could deal with nonlinear and non-Gaussian distributed data with the presence of missing measurements.

This research introduced a newly integrated EM-E-LW-IC-KPLS algorithm for nonlinear and non-Gaussian distributed process data with missing measurements. Besides, other associated outcomes are E-LW-KPLS and E-LW-IC-KPLS. They are also newly developed algorithms and methods for improving adaptive soft sensors. As part of this study, the capabilities and effectiveness of these newly developed algorithms had been tested in six different case studies. Based on the results, it was found the proposed algorithm was superior to the LW-PLS and LW Kernel partial least square (LW-KPLS) algorithm in controlling for nonlinear, and non-Gaussian distributed data. Moreover, for a missing data problem, the proposed algorithm is also deal to address a certain percentage of the missing data.

The main findings and conclusions of this study are presented in the following. These findings and conclusions have answered the research questions stated in Chapter 1.

a. In nonlinear and Gaussian distributed processes, the developed ensemble LW-KPLS (E-LW-KPLS) outperformed compared to the LW-PLS since its E_a values improved about 7% to 54%.

b. Another developed algorithm which is the ensemble LW independent component Kernel partial least square (E-LW-IC-KPLS) are also more accurate than the LW-PLS in which the E_a values were lower by approximately 8% to 94%.

c. The performance of the EM-E-LW-IC-KPLS in dealing with 5% to 60% of missing data in nonlinear and non-Gaussian process data was investigated.

d. By using case studies, it was found the EM-E-LW-IC-KPLS performs better than the existing integrated benchmarking LW-PLS with missing data imputation methods including expectation maximisation, trimmed score regression, and singular vector decomposition methods in handling up to 20% of missing data.

e. Besides having a more accurate prediction as compared to LW-PLS, the inclusive of ensemble method in the newly proposed E-LW-KPLS, E-LW-IC-KPLS, and EM-E-LW-IC-KPLS algorithms had successfully minimised their penalty which is the computational burden.

7.3. Contributions

This research work offers some contributions to the field of soft sensing technologies. The contributions of this research are outlined below:

a. A novel improved algorithm, EM-E-LW-IC-KPLS used to develop the adaptive soft sensor that is robust against incomplete data and capable of dealing with non-Gaussian distributed data has been formulated and proposed to address the limitations of the existing algorithms.

b. This new algorithm, EM-E-LW-IC-KPLS is a more accurate approach to develop adaptive soft sensors and a better alternative than the LW-PLS in dealing with non-Gaussian distributed data and missing data.

c. Though not as effective as the EM-E-LW-IC-KPLS, the E-LW-KPLS, EM-E-LW-KPLS and E-LW-IC-KPLS models are also newly developed algorithms and can be additional methods for developing adaptive soft sensors. These algorithms are adaptive due to the presence of locally weighted method, which is an adaptive model in their algorithms.

d. The proposed EM-E-LW-IC-KPLS algorithm has the potential to be applied in any industrial processing plant in which the process is nonlinear, and the operational data follows non-Gaussian distribution and contains missing measurements.

e. The results and new findings in this study can be used as a platform to develop better adaptive soft sensing and control in any chemical industrial processes.

7.4. Recommendations for future research

Although the proposed EM-E-LW-IC-KPLS algorithm for adaptive soft sensors, which can deal with non-Gaussian data and robust against missing data has successfully narrowed down the research gaps mentioned in Chapter 2, extensive studies still can be carried out. There are some recommendations for future studies which are listed below:

a. In addition, other local models including the support vector regression, the least square support vector regression, the Gaussian process regression, and the Gaussian mixture model may be considered as local modeling in LW algorithms in future studies.

b. Validating the newly proposed models using online data is recommended.

c. Further research on the control system using the developed adaptive soft sensors can be carried out.

Reference

Ahmad, Iftikhar, Manabu Kano, Shinji Hasebe, Hiroshi Kitada, and Noboru Murata. 2014. "Gray-box modeling for prediction and control of molten steel temperature in tundish." *Journal of Process Control* 24(4)**:** 375-382.

Arteaga, Francisco, and Alberto Ferrer. 2002. "Dealing with missing data in MSPC: several methods, different interpretations, some examples." *Journal of chemometrics* 16(8-10)**:** 408-418.

Arteaga, Francisco, and Alberto Ferrer. 2005. "Framework for regression-based missing data imputation methods in on-line MSPC." *Journal of chemometrics* 19(8)**:** 439-447.

Askarian, Mahdieh, Gerard Escudero, Moisès Graells, Reza Zarghami, Farhang Jalalli-Farahani, and Navid Mostoufi. 2016. "Fault diagnosis of chemical processes with incomplete observations: A comparative study." *Computers & chemical engineering* 84**:** 104-116.

Bashir, Faraj, and Hua-Liang Wei. 2018. "Handling missing data in multivariate time series using a vector autoregressive model-imputation (VAR-IM) algorithm." *Neurocomputing* 276: 23-30.

Bishop, Christopher M. 2006. "Pattern recognition." *Machine Learning* 128**:** 1-58.

Bratu, Mihaela. 2013. "MAE - Alternative method of measuring the global average uncertainty of inflation forecast in Romania." *The USV Annals of Economics and Public Administration* 12(1(15)): 230 - 236.

Braunschweig, Bertrand, and Xavier Joulia. 2008. *18th European symposium on computer aided process engineering*.France: Elsevier.

Caraman, Sergiu, Mihaela Sbarciog, and Marian Barbu. 2007. "Predictive Control of aWastewater Treatment Process." *International Journal of Computers Communications & Control* 2(2): 132-142.

Chen, Dong, Xinchun Li, Kai Xiang, Dong Wang, Akio Nakabayashi, Makoto Nakaya, and Tetsuya Ohtani. "Hybrid plant model of physical and statistical model with robust updating method." *Intelligent Control and Information Processing (ICICIP), 2010 International Conference,* Dalian, China, IAugust 13 - 15, 2010.

Chen, Hong, Andreas Kremling, and Frank Allgöwer. 1995. "Nonlinear predictive control of a benchmark CSTR." *Proceedings of 3rd European control conference*, Rome, Italy, September 5 - 8, 1995.

Chen, Mu-Chen, Chun-Chin Hsu, Bharat Malhotra, and Manoj Kumar Tiwari. 2016. "An efficient ICA-DW-SVDD fault detection and diagnosis method for non-Gaussian processes." *International Journal of Production Research* 54(17): 5208-5218.

Chen, Ning, Jiayang Dai, Xiaofeng Yuan, Weihua Gui, Wenting Ren, and Heikki N. Koivo. 2018. "Temperature Prediction Model for Roller Kiln by ALD-Based Double Locally Weighted Kernel Principal Component Regression." *IEEE Transactions on Instrumentation and Measurement* 67(8): 2001-2010.

Christofides, Panagiotis D., and Nael El-Farra. 2005. *Control of nonlinear and hybrid process systems: Designs for uncertainty, constraints and time-delays*.Berlin, Germany: Springer Science & Business Media.

Cramer III, Richard D. 1993. "Partial least squares (PLS): its strengths and limitations." *Perspectives in Drug Discovery and Design* 1(2)**:** 269-278.

De Marchi, Stefano. 2013. "Four lectures on Radial Basis Functions." www.math.unipd.it/~demarchi/RBF/LectureNotes.pdf.

De la Fuente, Rodrigo López-Negrete, Salvador García-Muñoz, and Lorenz T. Biegler. 2010. "An efficient nonlinear programming strategy for PCA models with incomplete data sets." *Journal of Chemometrics* 24 (6): 301-311.

Dong, Dong, and Thomas J. McAvoy.1996. "Nonlinear principal component analysis—based on principal curves and neural networks." *Computers & Chemical Engineering* 20(1)**:** 65-78.

Dou, Ying, Ying Sun, Yuqiu Ren, and Yulin Ren. 2005. "Artificial neural network for simultaneous determination of two components of compound paracetamol and diphenhydramine hydrochloride powder on NIR spectroscopy." *Analytica Chimica Acta* 528(1)**:** 55-61.

Fan, Miao, Zhiqiang Ge, and Zhihuan Song. 2014. "Adaptive Gaussian mixture model-based relevant sample selection for JITL soft sensor development." *Industrial & Engineering Chemistry Research* 53(51)**:** 19979-19986.

Folch-Fortuny, Abel, Francisco Arteaga, and Alberto Ferrer. 2015. "PCA model building with missing data: New proposals and a comparative study." *Chemometrics and Intelligent Laboratory Systems* 146**:** 77-88.

Folch-Fortuny, Abel, Francisco Arteaga, and Alberto Ferrer. 2016a. "Missing Data Imputation Toolbox for MATLAB." *Chemometrics and Intelligent Laboratory Systems* 154**:** 93-100.

Folch-Fortuny, Abel, Francisco Arteaga, and Alberto Ferrer. 2016b. "Assessment of maximum likelihood PCA missing data imputation." *Journal of Chemometrics* 30(7): 386-393.

Fortuna, Luigi, Salvatore Graziani, Alessandro Rizzo, and Maria Gabriella Xibilia. 2007. *Soft sensors for monitoring and control of industrial processes*.Berlin, Germany: Springer Science & Business Media.

Frank, Iidiko E. 1990. "A nonlinear PLS model". *Chemometrics and intelligent laboratory systems* 8(2)**:** 109-119.

Fujiwara, Koichi, Kano, Manabu, and Hasebe, Shinji. 2008. "Correlation-based Just-In-Time modeling for soft-sensor design." *Transactions of the Society of Instrument and Control Engineers* 44(4)**:** 317-324.

Fujiwara, Koichi, Manabu Kano, and Shinji Hasebe. 2010. "Development of correlation-based clustering method and its application to software sensing." *Chemometrics and Intelligent Laboratory Systems* 101(2)**:** 130-138.

Fujiwara, Koichi, Manabu Kano, and Shinji Hasebe. 2012. "Development of correlation-based pattern recognition algorithm and adaptive soft-sensor design." *Control Engineering Practice* 20(4)**:** 371-378.

Fujiwara, Koichi, Manabu Kano, Shinji Hasebe, and Akitoshi Takinami. 2009. "Soft-sensor development using correlation-based just-in-time modeling." *AIChE Journal* 55(7)**:** 1754-1765.

Funatsu, Kimito. 2018 "Process Control and Soft Sensors." *Applied Chemoinformatics: Achievements and Future Opportunities* 571-584.

Gao, Yingbin, Xiangyu Kong, Changhua Hu, Zhengxin Zhang, Hongzeng Li, and Li'an Hou. 2015. "Multivariate data modeling using modified kernel partial least squares." *Chemical Engineering Research and Design* 94**:** 466-474.

García-Reiriz, Alejandro, Patricia Cecilia Damiani, and Alejandro C. Olivieri. 2010. "Residual bilinearization combined with kernel-unfolded partial least-squares: A new technique for processing non-linear second-order data achieving the second-order advantage." *Chemometrics and Intelligent Laboratory Systems* 100(2)**:** 127-135.

Ge, Zhiqiang. 2014. "Active learning strategy for smart soft sensor development under a small number of labeled data samples." *Journal of Process Control* 24(9)**:** 1454-1461.

Ge, Zhiqiang, and Zhihuan Song. 2008. "Online monitoring of nonlinear multiple mode processes based on adaptive local model approach." *Control Engineering Practice* 16(12)**:** 1427-1437.

Ge, Zhiqiang, and Zhihuan Song. 2010. "A comparative study of just-in-time-learning based methods for online soft sensor modeling." *Chemometrics and Intelligent Laboratory Systems* 104(2)**:** 306-317.

Ge, Zhiqiang, Zhihuan Song, and Furong Gao. 2013. "Review of recent research on data-based process monitoring." *Industrial & Engineering Chemistry Research* 52(10)**:** 3543-3562.

Ge, Zhiqiang, Zhihuan Song, and Peiliang Wang. 2014. "Probabilistic combination of local independent component regression model for multimode quality prediction in chemical processes." *Chemical Engineering Research and Design* 92(3)**:** 509-521.

Goldberg, Yoav, and Micheal Elhadad. 2008. "splitSVM: fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications."*Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, Columbus, Ohio, USA, June 16 - 17, 2008.

Gómez-Carracedo, María Paz, José Andrade, Purificación López-Mahía, S. Muniategui, and Daniela Prada. 2014. "A practical comparison of single and multiple imputation methods to handle complex missing data in air quality datasets." *Chemometrics and Intelligent Laboratory Systems* 134**:** 23-33.

Hazama, Koji, and Manabu Kano. 2015. "Covariance-based locally weighted partial least squares for high-performance adaptive modeling." *Chemometrics and Intelligent Laboratory Systems* 146**:** 55-62.

He, YanLin, ZhiQiang Geng, and QunXiong Zhu. 2016. "Soft sensor development for the key variables of complex chemical processes using a novel robust bagging nonlinear model integrating improved extreme learning machine with partial least square." *Chemometrics and Intelligent Laboratory Systems* 151**:** 78-88.

Helbig, Achim, Wolfgang Marquardt, and Frank Allgöwer. 2000. "Nonlinearity measures: definition, computation and applications." *Journal of Process Control* 10(2-3): 113-123.

Hu, Yi, Hehe Ma, and Hongbo Shi. 2013. "Enhanced batch process monitoring using just-in-time-learning based kernel partial least squares." *Chemometrics and Intelligent Laboratory Systems* 123**:** 15-27.

Hu, Yi, Li Wang, Hehe Ma, and Hongbo Shi. 2011. "Online nonlinear process monitoring using kernel partial least squares." *CIESC Journal,* 9**:** 25.

Hyvarinen, Aapo. 1999. "Fast and robust fixed-point algorithms for independent component analysis." *IEEE transactions on Neural Networks* 10(3)**:** 626-634.

Hyvärinen, Aapo. 2013. "Independent component analysis: recent advances." *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 371(1984)**:** 20110534.

Hyvärinen, Aapo, and Erkki Oja. 2000. "Independent component analysis: algorithms and applications." *Neural networks* 13(4)**:** 411-430.

Hyvärinen, Aapo, Juha Karhunen, and Erkki Oja. 2004. *Independent component analysis*.New Jersey, United States: John Wiley & Sons.

Ingildsen, Pernille. 2002."*Realising full-scale control in wastewater treatment systems using in situ nutrient sensors*." PhD diss. Lund University. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.455.4749&rep=rep1&type=pdf

Jia, Qilong, and Yingwei Zhang. 2016. "Quality-related fault detection approach based on dynamic kernel partial least squares." *Chemical Engineering Research and Design* 106**:** 242-252.

Jiang, Qingchao, and Xuefeng Yan. 2013. "Weighted kernel principal component analysis based on probability density estimation and moving window and its application in nonlinear chemical process monitoring." *Chemometrics and Intelligent Laboratory Systems* 127**:** 121-131.

Jiang, Qingchao, Xuefeng Yan, Zhaomin Lv, and Meijin Guo. 2014. "Independent component analysis-based non-Gaussian process monitoring with preselecting optimal components and support vector data description." *International Journal of Production Research* 52(11)**:** 3273-3286.

Jin, Huaiping, Xiangguang Chen, Jianwei Yang, Hua Zhang, Li Wang, and Lei Wu. 2015b. "Multi-model adaptive soft sensor modeling method using local learning and online support vector regression for nonlinear time-variant batch processes." *Chemical Engineering Science* 131**:** 282-303.

Jin, Huaiping, Xiangguang Chen, Jianwei Yang, and Lei Wu. 2014. "Adaptive soft sensor modeling framework based on just-in-time learning and kernel partial least squares regression for nonlinear multiphase batch processes." *Computers & Chemical Engineering* 71**:** 77-93.

Jin, Huaiping, Xiangguang Chen, Jianwen Yang, Li Wang, and Lei Wu. 2015a. "Online local learning based adaptive soft sensor and its application to an industrial fed-batch chlortetracycline fermentation process." *Chemometrics and Intelligent Laboratory Systems* 143**:** 58-78.

Jin, Huaiping, Xiangguang Chen, Li Wang, Kai Yang, and Lei Wu. 2016. "Dual learning-based online ensemble regression approach for adaptive soft sensor modeling of nonlinear time-varying processes." *Chemometrics and Intelligent Laboratory Systems* 151**:** 228-244.

Jin, Xing, Siyun Wang, Biao Huang, and Fraser Forbes. 2012. "Multiple model based LPV soft sensor development with irregular/missing process output measurement." *Control Engineering Practice* 20(2)**:** 165-172.

JSPS PSE 143 (The Japan Society for the Promotion of Science, Process Systems Engineering 143) committee. 2004. *Annual Report 2003*.Japan.

Junger, Washington, and Antonio Ponce de Leon. 2015. "Imputation of missing data in time series for air pollutants." *Atmospheric Environment* 102**:** 96-104.

Kadlec, Petr, Bogdan Gabrys, and Sibylle Strandt. 2009. "Data-driven soft sensors in the process industry." *Computers & Chemical Engineering* 33(4)**:** 795-814.

Kadlec, Petr, Ratko Grbić, and Bogdan Gabrys. 2011. "Review of adaptation mechanisms for data-driven soft sensors." *Computers and chemical engineering* 35(1)**:** 1-24.

Kamath, Uday, Amarda Shehu, and Kenneth A. De Jong. 2010 "Feature and kernel evolution for recognition of hypersensitive sites in DNA sequences. International Conference on Bio-Inspired Models of Network, Information, and Computing Systems." *Springer*87: 213-228.

Kaneko, Hiromasa, and Kimito Funatsu. 2011a. "Development of soft sensor models based on time difference of process variables with accounting for nonlinear relationship." *Industrial & Engineering Chemistry Research* 50(18)**:** 10643-10651.

Kaneko, Hiromasa, and Kimito Funatsu. 2011b. "Maintenance-free soft sensor models with time difference of process variables." *Chemometrics and Intelligent Laboratory Systems* 107(2)**:** 312-317.

Kaneko, Hiromasa, and Kimito Funatsu. 2011c. "A soft sensor method based on values predicted from multiple intervals of time difference for improvement and estimation of prediction accuracy." *Chemometrics and Intelligent Laboratory Systems* 109(2)**:** 197-206.

Kaneko, Hiromasa, and Kimito Funatsu. 2013a. "Classification of the degradation of soft sensor models and discussion on adaptive models." *AIChE Journal* 59(7)**:** 2339-2347.

Kaneko, Hiromasa, and Kimito Funatsu. 2013b. "Discussion on time difference models and intervals of time difference for application of soft sensors." *Industrial & Engineering Chemistry Research* 52(3)**:** 1322-1334.

Kaneko, Hiromasa, and Kimito Funatsu. 2015a. "Adaptive Database Management Based on the Database Monitoring Index for Long-term Use of Adaptive Soft Sensors." *Chemometrics and Intelligent Laboratory Systems* 146**:** 179-185.

Kaneko, Hiromasa, and Kimito Funatsu. 2015b. "Moving window and just-in-time soft sensor model based on time differences considering a small number of measurements." *Industrial & Engineering Chemistry Research* 54(2)**:** 700-704.

Kaneko, Hiromasa, and Kimito Funatsu. 2016. "Ensemble locally weighted partial least squares as a just-in-time modeling method." *AIChE Journal* 62(3)**:** 717-725.

Kano, Manabu and Fujiwara Koichi. 2013. "Virtual sensing technology in process industries: Trends and challenges revealed by recent industrial applications." *Journal of Chemical Engineering of Japan* 46(1)**:** 1-17.

Kano, Manabu and Morimasa Ogawa. 2010. "The state of the art in chemical process control in Japan: Good practice and questionnaire survey." *Journal of Process Control* 20(9)**:** 969-982.

Karanja, Erastus, Jigish Zaveri, and Ashraf Ahmed. 2013. "How do MIS researchers handle missing data in survey-based research: A content analysis approach." *International Journal of Information Management* 33(5)**:** 734-751.

Kim, Sanghong, Manabu Kano, Hiroshi Nakagawa, and Shinji Hasebe. 2011. "Estimation of active pharmaceutical ingredients content using locally weighted partial least squares and statistical wavelength selection." *International journal of pharmaceutics* 421(2)**:** 269-274.

Kim, Sanghong, Manabu Kano, Shinji Hasebe, Akitoshi Takinami, and Takeshi Seki. 2013a. "Long-term industrial applications of inferential control based on just-in-time soft-sensors: economical impact and challenges." *Industrial & Engineering Chemistry Research* 52(35)**:** 12346-12356.

Kim, Sanghong, Ryota Okajima, Manabu Kano, and Shinji Hasebe. 2013b. "Development of soft-sensor using locally weighted PLS with adaptive similarity measure." *Chemometrics and Intelligent Laboratory Systems* 124**:** 43-49.

Krogh, Anders, and Jesper Vedelsby. 1995. "Neural network ensembles, cross validation, and active learning." *Advances in neural information processing systems*, 231-238.

Lan, Hua, Xuezhi Wang, Quan Pan, Feng Yang, Zengfu Wang, and Yan Liang. 2016. "A survey on joint tracking using expectation–maximization based techniques." *Information Fusion* 30**:** 52-68.

Lee, Jong Min., S. Joe Qin, and In Beum Lee. 2006. " Fault detection and diagnosis based on modified independent component analysis." *AIChE journal* 52(10)**:** 3501-3514.

Lee, Te Won. 2000. "*Independent component analysis: Theory and Applications*." Springer.

Legates, David R., and Geogory J. McCabe Jr. 1999. "Evaluating the use of "goodness-of-fit" measures in hydrologic and hydroclimatic model validation. *Water resources research*
35(1): 233 - 241.

Li, Genzi, Vikrant Aute, and Shapour Azarm, S. 2010. "An accumulative error based adaptive design of experiments for offline metamodeling." *Structural and Multidisciplinary Optimization* 40(1)**:** 137-155.

Li, Qi, Liping Xing, Wenya Liu, and Wei Ba. 2015. "Adaptive Soft Sensor Based on a Moving Window Just-in-time Learning LS-SVM for Distillation Processes." *IFAC-PapersOnLine* 48(28)**:** 51-56.

Li, Zhenxing, Kuangrong Hao, Lei Chen, Yongsheng Ding, and Biao Huang. 2018. "PET Viscosity Prediction Using JIT-based Extreme Learning Machine." *IFAC-Papers on Line* 51(18): 608-613.

Lin, Bao, Bodil Recke, Jørgen KH Knudsen, and Sten Bay Jørgensen. 2007. "A systematic approach for soft sensor development." *Computers & chemical engineering* 31(5)**:** 419-425.

Liu, Hongbin, and ChangKyoo Yoo. 2016. "A robust localized soft sensor for particulate matter modeling in Seoul metro systems." *Journal of Hazardous Materials* 305**:** 209-218.

Liu, Jingxiang, Tao Liu, and Junghui Chen. 2018. "Quality prediction for multi-grade processes by just-in-time latent variable modeling with integration of common and special features." *Chemical Engineering Science* 191: 31 - 41.

Liu, Yi, and Junghui Chen. 2013. "Integrated soft sensor using just-in-time support vector regression and probabilistic analysis for quality prediction of multi-grade processes." *Journal of Process Control* 23(6)**:** 793-804.

Liu, Yi, and Zengliang Gao. 2015. "Industrial melt index prediction with the ensemble anti-outlier just-in-time Gaussian process regression modeling method." *Journal of Applied Polymer Science* 132(22): 41958.http://doi:10.1002/app.41958.

Liu, Yi, Zengliang Gao, Ping Li, and Haiqing Wang. 2012a. "Just-in-time kernel learning with adaptive parameter selection for soft sensor modeling of batch processes." *Industrial & Engineering Chemistry Research* 51(11)**:** 4313-4327.

Liu, Yiqi, DaopingHuang, and Yan Li. 2012b. "Development of interval soft sensors using enhanced just-in-time learning and inductive confidence predictor." *Industrial & Engineering Chemistry Research* 51(8)**:** 3356-3367.

Liu, Yuzhe, and Vanathi Gopalakrishnan. 2017 "An Overview and Evaluation of Recent Machine Learning Imputation Methods Using Cardiac Imaging Data." *Data* 2(1): 8.

Lopes, Vitor V., and José C. Menezes. 2005. "Inferential sensor design in the presence of missing data: a case study." *Chemometrics and intelligent laboratory systems* 78(1)**:** 1-10.

Ma, Ming, Shima Khatibisepehr, and Biao Huang. 2015. "A Bayesian framework for real-time identification of locally weighted partial least squares." *AIChE Journal* 61(2)**:** 518-529.

Ma, Zhanyu. 2011. "Non-Gaussian statistical modelsand their applications." PhD diss, KTH Royal Institute of Technology.

Mei, Congli, Xu Chen, Yuhang Ding, Yao Chen, Jiangpin Cai, and Yunxia Luo. 2018a. "On-line Calibration of Just in Time Learning and Gaussian Process Regression based Soft Sensor with Moving-Window Technology." *Chemical Engineering Transactions* 70: 1417-1422.

Mei, Congli, Yuhang Ding, Xu Chen, Yao Chen, and Jiangpin Cai. 2018b. Soft Sensor Modelling based on Just-in-Time Learning and Bagging-PLS for Fermentation Processes. *Chemical Engineering Transactions* 70: 1435-1440.

Miao, Xiaoye, Yunjun Gao, Su Guo, and Wanqi Liu. 2018. "Incomplete data management: a survey." *Frontiers of Computer Science* 12(1): 4-25.

Mongillo, Michael. 2011. "Choosing basis functions and shape parameters for radial basis function methods." *SIAM Undergraduate Research Online* 4: 190-209.

Naik, Ganesh R., and Dinesh K. Kumar. 2011. "An overview of independent component analysis and its applications." *Informatica* 35(1): 63 - 81.

Nakabayashi, Akio, Makoto Nakaya, Tetsuya, Ohtani, Dong Chen, Dong Wang, and Xinchun Li. 2010. "A process simulator based on hybrid model of physical model and Just-In-Time model." *Proceedings of SICE Annual Conference 2010*, Taipei, Taiwan, October 13, 2010.

Nakagawa, Hiroshi, Manabu Kano, Shinji Hasebe, Takuya Miyano, Tomoyuki Watanabe, and Naoki Wakiyama. 2014. "Verification of model development technique for NIR-based real-time monitoring of ingredient concentration during blending." *International journal of pharmaceutics* 471(1)**:** 264-275.

Nakagawa, Hiroshi, Takahiro Tajima, Manabu Kano, Sanghong Kim, Shinji Hasebe, Tatsuya Suzuki, and Hiroaki Nakagami. 2012. "Evaluation of infrared-reflection absorption spectroscopy measurement and locally weighted partial least-squares for rapid analysis of residual drug substances in cleaning processes." *Analytical chemistry* 84(8)**:** 3820-3826.

Nelson, Philip R. C., John F. MacGregor, and Paul A. Taylor. 2006. "The impact of missing measurements on PCA and PLS prediction and monitoring applications." *Chemometrics and intelligent laboratory systems* 80(1)**:** 1-12.

Nelson, Philip R. C., Paul A. Taylor, and John F. MacGregor. 1996. "Missing data methods in PCA and PLS: Score calculations with incomplete observations." *Chemometrics and intelligent laboratory systems* 35(1)**:** 45-65.

Novak, Bela, and John J. Tyson. 1993. "Modeling the cell division cycle: M-phase trigger, oscillations, and size control." *Journal of theoretical biology* 165(1)**:** 101-134.

Okada, Takeshi, Hiromasa Kaneko, and Kimito Funatsu. 2010. "Development of a model selection method based on the reliability of a soft sensor model." *Sonklanakarin Journal of Science and Technology* 34(2)**:** 217.

Opitz, David, and Richard Maclin. 1999. "Popular ensemble methods: An empirical study." *Journal of artificial intelligence research* 11: 169-198.

Orr, Mark J. L. 1996.*Introduction to radial basis function networks*.Scotland: University of Edinburgh.

Pangborn, Andrew, Gregor Von Laszewski, James Cavenaugh, Muhammad Shaaban, and Roy Melton. 2011. "Scalable Data Clustering using GPU Clusters."*Concurrency and computation: practice and experience* 00: 1 -24.

Pani, Ajaya Kumar and Hare Krishna Mohanta. 2011. "A survey of data treatment techniques for soft sensor design." *Chemical Product and Process Modeling* 6(1). doi**:** https://doi.org/10.2202/1934-2659.1536

Pantelides, Constantinos C., and Jeffrey G. Renfro. 2013. "The online use of first-principles models in process operations: Review, current status and future needs." *Computers & Chemical Engineering* 51**:** 136-148.

Parish, Eric J. and Karthik Duraisamy. 2016. "A paradigm for data-driven predictive modeling using field inversion and machine learning." *Journal of Computational Physics* 305**:** 758-774.

Peng, Kaixiang, Qianqian Li, Kai Zhang, and Jie Dong. 2016. "Quality-related process monitoring for dynamic non-Gaussian batch process with multi-phase using a new data-driven method." *Neurocomputing* 214**:** 317-328.

Peng, Xin, Yang Tang, Wenli Du, and Feng Qian. 2017. "Online Performance Monitoring and Modeling Paradigm based on Just-in-time Learning and Extreme Learning Machine for Non-Gaussian Chemical Process." *Industrial & Engineering Chemistry Research* 56(23): 6671-6684.

Peugh, James L., and Craig K. Enders. 2004. "Missing data in educational research: A review of reporting practices and suggestions for improvement." *Review of educational research* 74(4)**:** 525-556.

Qin, S. Joe. 2012. "Survey on data-driven industrial process monitoring and diagnosis." *Annual Reviews in Control* 36(2)**:** 220-234.

Reitermanova, Zuzana. 2010. "Data splitting." *Wisconsin Daylily Society* 10: 31-36.

Riggi, Simone, Daniele Riggi, and Francesco Riggi. 2015. "Handling missing data for the identification of charged particles in a multilayer detector: A comparison between different imputation methods." *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 780**:** 81-90.

Roffel, Brian, and Ben Betlem. 2007. *Process dynamics and control: modeling for control and prediction*, John Wiley & Sons.

Rosipal, Roman. 2010. "Nonlinear partial least squares: An overview." *Chemoinformatics and advanced machine learning perspectives: complex computational methods and collaborative techniques***,** 169-189.

Rosipal, Roman and Leonard J. Trejo. 2002. "Kernel partial least squares regression in reproducing kernel hilbert space." *The Journal of Machine Learning Research* 2**:** 97-123.

Saptoro, Agus. 2014. "State of the Art in the Development of Adaptive Soft Sensors based on Just-in-Time Models." *Procedia Chemistry* 9:226-234.

Saptoro, Agus, Hari B. Vuthalum and Moses O. Tade. 2006. "Design of feed-forward neural networks architecture for coal elemental composition prediction." *International conference on modeling and simulation.* K*uala Lumpur, Malaysia*, 3-5.

Saptoro, Agus, Hong Mei Yao, Moses O. Tadé, and Hari B. Vuthaluru. 2008. "Prediction of coal hydrogen content for combustion control in power utility using neural network approach." *Chemometrics and Intelligent Laboratory Systems* 94(2)**:** 149-159.

Schafer, Joseph L. and John W. Graham. 2002. "Missing data: our view of the state of the art." *Psychological methods* 7(2)**:** 147.

Schmitt, Peter, Jonas Mandel, and Mickael Guedj. "A Comparison of Six Methods for Missing Data Imputation. *Journal of Biometrics and Biostatistics* 6: 224. doi: 10.4172/2155-6180.100022.

Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller. 1998. "Nonlinear component analysis as a kernel eigenvalue problem." *Neural computation* 10(5)**:** 1299-1319.

Schön, Thomas. 2009. *An explanation of the expectation maximization algorithm.* Sweden: Linköpings universitet.

Sentas, Panagiotis, and Lefteris Angelis. 2006. "Categorical missing data imputation for software cost estimation by multinomial logistic regression." *Journal of Systems and Software* 79 (3): 404-414.

Severson, Kristen, Mark Molaro, and Richard Braatz. 2017. "Principal component analysis of process datasets with missing values." *Processes* 5 (3): 38.

Shan, Peng, Silong Peng, Yiming Bi, Liang Tang, Chixiao Yang, Qiong Xie, and Changwen Li. 2014. "Partial least squares–slice transform hybrid model for nonlinear calibration." *Chemometrics and Intelligent Laboratory Systems* 138**:** 72-83.

Shan, Peng, Silong Peng, Liang Tang, Chixiao Yang, Yuhui Zhao, Qiong Xie, and Changwen Li. 2015. "A nonlinear partial least squares with slice transform based piecewise linear inner relation." *Chemometrics and Intelligent Laboratory Systems* 143**:** 97-110.

Shao, Weiming, Xuemin Tian, and Ping Wang. 2014. "Local partial least squares based online soft sensing method for multi-output processes with adaptive process states division." *Chinese Journal of Chemical Engineering* 22(7)**:** 828-836.

Shao, Xueguang, Wei Wang, Zhenyu Hou, and Wensheng Cai. 2006. "A new regression method based on independent component analysis." *Talanta* 69(3)**:** 676-680.

Sharma, Suraj and Sanjeev S. Tambe. 2014. "Soft-sensor development for biochemical systems using genetic programming." *Biochemical Engineering Journal* 85**:** 89-100.

Shawe-Taylor, John and Nello Cristianini. 2004. *Kernel methods for pattern analysis*, Cambridge University press.

Shigemori, Hiroyasu. 2015. "Cooling temperature control for steel plates through locally weighted regression model." *24th International Conference on metallurgy and materials METAL 2015.* Hotel Voronez I, Brno, Czech Republic, EU.

Shigemori, Hiroyasu, Manabu Kano, and Shinji Hasebe. 2011. "Optimum quality design system for steel products through locally weighted regression model." *Journal of Process Control* 21(2)**:** 293-301.

Song, Kai, Fang Wu, Tuo-Peng Tong, and Xiao-Jing Wang. 2012. "A real-time Mooney-viscosity prediction model of the mixed rubber based on the Independent Component Regression-Gaussian Process algorithm." *Journal of Chemometrics* 26(11-12)**:** 557-564.

Souza, Francisco Alexandre Andrade de, Rui Araújo, and Jérôme Mendes. 2015. "Review of Soft Sensors Methods for Regression Applications." *Chemometrics and Intelligent Laboratory Systems* 152**:** 69-79.

Souza, Francisco Alexandre Andrade de, Rui Araújo, and Jérôme Mendes. 2016. "Review of soft sensor methods for regression applications." *Chemometrics and Intelligent Laboratory Systems* 152**:** 69-79.

Stulp, Freek and Olivier Sigaud. 2015. "Many regression algorithms, one unified model: A review." *Neural Networks* 69**:** 60-79.

Tong, Chudong, Ting Lan, and Xuhua Shi. 2016. "Soft sensing of non-Gaussian processes using ensemble modified independent component regression." *Chemometrics and Intelligent Laboratory Systems* 157**:** 120-126.

Toshiya, Hirai and Manabu Kano. 2015. "Adaptive Virtual Metrology Design for Semiconductor Dry Etching Process through Locally Weighted Partial Least Squares."*IEEE Transactions on Semiconductor Manufacturing,* 28(2)**:** 137 - 144.

Uchimaru, Taku and Manabu Kano. 2016. "Sparse Sample Regression Based Just-In-Time Modeling (SSR-JIT): Beyond Locally Weighted Approach." *IFAC-Papers on Line* 49(7)**:** 502-507.

Ukai, Shouta, Akio Nakabayashi, Hidehiko Wada, and Tetsuya Ohtani. 2011. "A plant simulator based on hybrid model of physical model and Just-In-Time model using statistical approach." *SICE Annual Conference (SICE), 2011 Proceedings of Tokyo IEEE.*

Walczak, Beata, and Désiré Luc Massart. 2001. "Dealing with missing data: Part I." *Chemometrics and Intelligent Laboratory Systems* 58 (1): 15-27.

Wang, Liang, Die Yang, Cheng Fang, Zuliang Chen, Peter Lesniewski, Megharaj Mallavarapu, and Ravi Naidu. 2015a. "Application of neural networks with novel independent component analysis methodologies to a Prussian blue modified glassy carbon electrode array." *Talanta* 131**:** 395-403.

Wang, Xichang, Pu Wang, Xuejin Gao, and Yongsheng Qi. 2016. "On-line quality prediction of batch processes using a new kernel multiway partial least squares method." *Chemometrics and Intelligent Laboratory Systems* 158**:** 138-145.

Wang, Yanxia, Hui Cao, Yan Zhou, and Yanbin Zhang. 2015b. "Nonlinear partial least squares regressions for spectral quantitative analysis." *Chemometrics and Intelligent Laboratory Systems* 148**:** 32-50.

Weaver, Kathleen F., Vanessa C. Morales, Sarah L. Dunn, Kanya Godde, and Pablo F. Weaver. 2018. *An Introduction to Statistical Analysis in Research: with Applications in the Biological and Life Sciences*. United state: John Wiley & Sons, Inc.

Wei, Chihang, Junghui Chen, Zhihuan Song, and Chun-I. Chen. 2019. "Adaptive virtual sensors using SNPER for the localized construction and elastic net regularization in nonlinear processes." *Control Engineering Practice* 83: 129-140.

Wentzell, Peter D. and Lorenzo Vega Montoto. 2003. "Comparison of principal components regression and partial least squares regression through generic simulations of complex mixtures." *Chemometrics and intelligent laboratory systems* 65(2)**:** 257-279.

Wold, Svante. 1992. "Nonlinear partial least squares modelling II. Spline inner relation." *Chemometrics and Intelligent Laboratory Systems* 14(1-3)**:** 71-84.

Xie, Lei, Jiusun Zeng, and Chuanhou Gao. 2014. "Novel just-in-time learning-based soft sensor utilizing non-Gaussian information." *IEEE Transactions on Control Systems Technology* 22(1)**:** 360-368.

Xiong, Weili, XianqiangYang, Liang Ke, and Baoguo Xu. 2015. "EM algorithm-based identification of a class of nonlinear Wiener systems with missing output data." *Nonlinear Dynamics* 80(1-2)**:** 329-339.

Xiong, Weili, Wei Zhang, Baoguo Xu, and Biao Huang. 2016. "JITL based MWGPR soft sensor for multi-mode process with dual-updating strategy." *Computers & Chemical Engineering* 90**:** 260-267.

Xu, Shu. 2016. "Data cleaning and knowledge discovery in process data." PhD diss. The University of Texas. https://repositories.lib.utexas.edu/bitstream/handle/2152/32920/XU-DISSERTATION-2015(3).pdf?seq

Yang, Kai, Huaiping Jin, Xiangguang Chen, Jiayu Dai, Li Wang, and Dongxiang Zhang. 2016. "Soft sensor development for online quality prediction of industrial batch rubber mixing process using ensemble just-in-time Gaussian process regression models." *Chemometrics and Intelligent Laboratory Systems* 155**:** 170-182.

Yao, Le, and ZhiqiangGe. 2017. "Locally weighted prediction methods for latent factor analysis with supervised and semisupervised process data." *IEEE Transactions on Automation Science and Engineering* 14(1)**:** 126-138.

Yeo, Wan Sieng, Agus Saptoro, and Perumal Kumar. 2017. "Development of adaptive soft sensor using locally weighted Kernel partial least square model." *Chemical Product and Process Modeling* 12(4).

Yin, Shen, Guang Wang, and Xu Yang. 2014. "Robust PLS approach for KPI-related prediction and diagnosis against outliers and missing data." *International Journal of Systems Science* 45(7)**:** 1375-1382.

Yu, Jie and S. Joe Qin. 2008. "Multimode process monitoring with Bayesian inference-based finite Gaussian mixture models." *AIChE Journal* 54(7): 1811-1829.

Yuan, Xiaofeng, Biao Huang, Zhiqiang Ge, and Zhihuan Song. 2016b. "Double locally weighted principal component regression for soft sensor with sample selection under supervised latent structure." *Chemometrics and Intelligent Laboratory Systems* 153**:** 116-125.

Yuan, Xiaofeng, Jiao Zhou, Yalin Wang, and Chunhua Yang. 2018. "Multi-similarity measurement driven ensemble just-in-time learning for soft sensing of industrial processes." *Journal of Chemometrics*.

Yuan, Xiaofeng, Lingjian Ye, Liang Bao, Zhiqiang Ge, and Zhihuan Song. 2015b. "Nonlinear feature extraction for soft sensor modeling based on weighted probabilistic PCA." *Chemometrics and Intelligent Laboratory Systems* 147**:** 167-175.

Yuan, Xiaofeng, Yalin Wang, Chunhua Yang, Zhiqiang Ge, Zhihuan Song, and Weihua Gui. 2018. "Weighted linear dynamic system for feature representation and soft sensor application in nonlinear dynamic industrial processes." *IEEE Transactions on Industrial Electronics* 65(2): 1508 - 1517.

Yuan, Xiaofeng, Zhiqiang Ge, Biao Huang, and Zhihuan Song. 2016a. "A Probabilistic Just-in-Time Learning Framework for Soft Sensor Development with Missing Data." *IEEE Transactions on Control Systems Technology* 25(3): 1124-1132.

Yuan, Xiaofeng, Zhiqiang Ge, and Zhihuan Song. 2014a. "Locally weighted kernel principal component regression model for soft sensing of nonlinear time-variant processes." *Industrial & Engineering Chemistry Research* 53(35)**:** 13736-13749.

Yuan, Xiaofeng, Zhiqiang Ge, and Zhihuan Song. 2014b. "Soft sensor model development in multiphase/multimode processes based on Gaussian mixture regression." *Chemometrics and Intelligent Laboratory Systems* 138**:** 97-109.

Yuan, Xiaofeng, Zhiqiang Ge, and Zhihuan Song. 2015a. "Spatio-temporal adaptive soft sensor for nonlinear time-varying and variable drifting processes based on moving window LWPLS and time difference model." *Asia-Pacific Journal of Chemical Engineering* 11(2): 209-219.

Zhang, Xinmin, Kano Manabu, and Yuan Li. 2017. "Locally weighted kernel partial least squares regression based on sparse nonlinear features for virtual sensing of nonlinear time-varying processes." *Computers & Chemical Engineering* 104**:** 164-171.

Zhang, Xinmin, Yuan Li, and Manabu Kano. 2015. "Quality Prediction in Complex Batch Processes with Just-in-Time Learning Model Based on Non-Gaussian Dissimilarity Measure." *Industrial & Engineering Chemistry Research* 54(31)**:** 7694-7705.

Zhang, Yingwei, and Zhiyong Hu. 2011. "On-line batch process monitoring using hierarchical kernel partial least squares." *Chemical Engineering Research and Design* 89(10)**:** 2078-2084.

Zhang, Yingwei, Shuai Li, Zhiyong Hu, and Chonghui Song. 2012. "Dynamical process monitoring using dynamical hierarchical kernel partial least squares." *Chemometrics and Intelligent Laboratory Systems* 118**:** 150-158.

Zhang, Yingwei, Yongdong Teng, and Yang Zhang. 2010. "Complex process quality prediction using modified kernel partial least squares." *Chemical Engineering Science* 65(6)**:** 2153-2158.

Zhang, Yingwei, and Yang Zhang. 2009. "Complex process monitoring using modified partial least squares method of independent component regression." *Chemometrics and Intelligent Laboratory Systems* 98(2)**:** 143-148.

Zheng, Wenjian, Yi Liu, Zengliang Gao, Jianguo Yang. 2018. "Just-in-time semi-supervised soft sensor for quality prediction in industrial rubber mixers." *Chemometrics and Intelligent Laboratory Systems* 180: 36-41.

Zhou, Xi-Yu, and Joon S. Lim. 2014. "Replace Missing Values with EM algorithm based on GMM and Naïve Bayesian." *International Journal of Software Engineering and Its Applications* 8(5)**:** 177-188.

Zhou, Yan-Ping, Jian-Hui Jiang, Wei-Qi Lin, Lu Xu, Hai-Long Wu, Guo-Li Shen and Ru-Qin Yu. 2007. "Artificial neural network-based transformation for nonlinear partial least-square regression with application to QSAR studies." *Talanta* 71(2)**:** 848-853.

Zou, Hui and Trevor Hastie. 2005. "Regularization and variable selection via the elastic net." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(2)**:** 301-320.

Appendix A Diferrential equations for simulation

This appendix shows the differential equations used in the case studies to produce the simulated data using MATLAB SIMULINK. Appendices A.1, A.2, A.3, and A.4 are for case studies 2, 3, 5 and 6, respectively.

Appendix A.1 Case study 2: A single chemical reactor

The following differential equations are used to generate simulated data for a single chemical reactor in case study 2.

$$\frac{dT}{dt} = \frac{F}{V}\left(T_{A0} - T\right) + \sum_{i=1}^{3} \frac{\left(-\Delta H_i\right)}{\rho C_p} R_i\left(C_A, T\right) + \frac{Q}{\rho C_p V} \tag{A.1.1}$$

$$R_i\left(C_A, T\right) = k_{i0} \exp\left(\frac{-E_i}{RT}\right) C_A \tag{A.1.2}$$

$$\frac{dC_A}{dt} = \frac{F}{V}\left(C_{A0} - C_A\right) - \sum_{i=1}^{3} R_i\left(C_A, T\right) \tag{A.1.3}$$

$$\frac{dC_B}{dt} = -\frac{F}{V} C_B + R_1\left(C_A, T\right) \tag{A.1.4}$$

where T, T_{A0}, F, Q, V, and ρ are the reactor temperature, the temperature of the pure reactant species A, the flow rate, the rate of heat input, the volume of the reactor, and the density of the fluid in the reactor, respectively. Then, ΔH_i, k_i, E_i, $i = 1, 2, 3$, and c_p are the enthalpies, the pre-exponential constants, the activation energies of the three reactions and the heat capacity, respectively. C_{A0}, C_A, and C_B are the molar concentration of pure A, the concentrations of the species A and B, respectively. The values of process parameters and the respective steady-state values are summarised in Table A.1.1.

Table A.1 The values for the process parameters and steady-state for the chemical reactor in case study 2

Parameters	**Values**
F	4.998 m^3/hr
V	1.0 m^3
R	8.314 $KJ/kmol.K$
T_{A0}	300.0 K
C_{A0}	4.0 $kmol/m^3$
C_{B0}	0.0 $kmol/m^3$
ΔH_1	-5.0×10^4 $KJ/kmol$
ΔH_2	-5.2×10^4 $KJ/kmol$
ΔH_3	-5.4×10^4 $KJ/kmol$
k_{10}	3.0×10^6 hr^{-1}
k_{20}	3.0×10^5 hr^{-1}
k_{30}	3.0×10^5 hr^{-1}
E_1	5.0×10^4 $KJ/kmol$
E_2	7.53×10^4 $KJ/kmol$
E_3	7.53×10^4 $KJ/kmol$
ρ	1000 kg/m^3
c_p	0.231 $KJ/kg.K$
T^s	388.57 K
C_A^s	3.59 $kmol/m^3$
C_B^s	0.41 $kmol/m^3$

Appendix A.2 Case study 3: Wastewater treatment

The differential equations for the wastewater treatment process in case study 3 are shown below:

$$\frac{dX_B}{dt} = \mu(t)X_B(t) - D(t)(1+r)X_B(t) + rD(t)X_r(t) \tag{A.2.1}$$

$$\frac{dS_s}{dt} = -\frac{\mu(t)}{Y}X_B(t) - D(t)(1+r)S_s(t) + D(t)S_{in} \tag{A.2.2}$$

$$\frac{dDO}{dt} = -K_0\frac{\mu(t)}{Y}X_B(t) - D(t)(1+r)DO(t) + \alpha W(DO_{\max} - DO(t)) + D(t)DO_{in} \tag{A.2.3}$$

$$\frac{dX_r}{dt} = D(t)(1+r)X_B(t) - D(t)(\beta + r)X_r(t) \tag{A.2.4}$$

$$\mu(t) = \mu_{\max}\frac{S_s(t)}{k_S + S_s(t)}\frac{DO(t)}{K_{DO} + DO(t)} \tag{A.2.5}$$

where $X_B(t)$, $S_s(t)$, $DO(t)$, DO_{max}, $X_r(t)$, $D(t)$, S_{in}, and DO_{in} are the biomass, the substrate, the dissolved oxygen, the maximum dissolved oxygen, the recycled biomass, the dilute rate, the substrate and dissolved oxygen concentrations in the influent, respectively. Besides, Y, μ , $\mu_{\max}$, k_s, K_{DO}, α , W, K_0 , r, and β are the biomass yield factor, the biomass growth rate, the maximum specific growth rate, the saturation constants for the substrate and the dissolved oxygen, the oxygen transfer rate, the aeration rate, the model constant, the ratios of recycled and waste flow to the influent, respectively.

Table A.2 The values for the process parameters and steady-state for the wastewater treatment process in case study 3

Parameters	**Values**
Y	0.65
β	0.2
α	0.018
K_{DO}	2 *mg/l*
K_0	0.5
μ_{max}	0.15 hr^{-1}
K_S	100 *mg/l*
DO_{max}	10 *mg/l*
r	0.6
$X_B(0)$	200 *mg/l*
$S_s(0)$	88 *mg/l*
$DO(0)$	5 *mg/l*
$X_r(0)$	320 *mg/l*
DO_{in}	0.5 *mg*
S_{in}	200 *mg/l*
D	0.02 to 0.05 hr^{-1}
W	20 to 80 hr^{-1}

Appendix A.3 Case study 5: Eukaryotic cell cycle regulation

In case study 5, the following differential equations for eukaryotic cell cycle regulation are used to capture the basic stages of frog egg development.

$$\frac{du}{dt} = \frac{k_1^{'}}{G} - (k_2^{'} + k_2^{''} u^2 + k_{wee})u + (k_{25}^{'} + k_{25}^{''} u^2)(\frac{v}{G} - u) \tag{A.3.1}$$

$$G = 1 + \frac{k_{INH}}{k_{CAK}} \tag{A.3.2}$$

$$\frac{dv}{dt} = k_1^{'} - (k_2^{'} + k_2^{''} u^2)v \tag{A.3.3}$$

where u, v, k_{INH}, and k_{CAK} are the dimensionless concentration of active M-phase promoting factor (MPF), the dimensionless concentration of total cyclin, the rate constant for inhibition of INH (a protein that negatively regulates MPF), and the rate constant for activation of CAK (a cdc2-acting kinase), respectively. $k_1^{'}$, k_{wee}, $k_2^{'}$, $k_2^{''}$, $k_{25}^{'}$, and $k_{25}^{''}$ are a rate constant for cyclin synthesis, the rate constant for inhibition of Weel (an enzyme responsible for the tyrosine phosphorylation of MPF which inhibits MPF activity), the rate constant for the low-activity form of cyclin degradation, the rate constant for high activity form of cyclin degradation, the rate constant for the low-activity form of tyrosine dephosphorylation of MPF, and the rate constant for the high activity form of tyrosine dephosphorylation of MPF, respectively. The parameters used in this cell cycle model are illustrated in Table A.3.1.

Table A.3 The values for the process parameters and steady-state for the wastewater treatment process in case study 5

Parameters	**Values**
$k_1^{'}$	0.01
$k_{25}^{'}$	0.04
$k_{25}^{''}$	100
k_{INH}	0.1
k_{CAk}	1
$k_2^{'}$ (Oscillatory mode)	0.01
$k_2^{''}$ (Oscillatory mode)	10
k_{wee} (Oscillatory mode)	2.0

Appendix A.4 Case study 6: A highly nonlinear CSTR

The reactor dynamics of a highly nonlinear continuously stirred tank reactor (CSTR) in case study 6 can be described by using the below differential equations.

$$\frac{dC_A}{dt} = \frac{\dot{V}}{V_R}(C_{A0} - C_A) - k_1 C_A - k_3 C_A^2 \qquad \text{(A.4.1)}$$

$$\frac{dC_B}{dt} = -\frac{\dot{V}}{V_R} C_B + k_1 C_A - k_2 C_B \qquad \text{(A.4.2)}$$

$$\frac{dT}{dt} = \frac{\dot{V}}{V_R}(T_0 - T) + \frac{k_w A_R}{\rho C_p V_R}\left(T_c - T\right) - \frac{1}{\rho C_p}(k_1 C_A \Delta H_{RAB} + k_2 C_B \Delta H_{RBC} + k_3 C_A^2 \Delta H_{RAD})$$

(A.4.3)

$$\frac{dT_C}{dt} = \frac{1}{m_c C_{pc}} \left(\dot{Q} + k_w A_R (T - Tc) \right) \tag{A.4.4}$$

The reaction rate constants for this highly nonlinear CSTR, k_i can be described by

$$k_i = k_{i0} e^{\left(\frac{E_i}{T}\right)}, i = 1,2,3 \tag{A.4.5}$$

where $\dot{Q}$, $\dot{V}$, V_R, T_0, C_{A0}, C_A, C_B, and T_c are the heat flow, the inlet flow, the reactor volume, the temperature contains reactant A, the concentration of pure reactant A, the concentration of reactant A, the concentration of B, and the cooling medium temperature, respectively. Moreover, E_i, C_p, m_c, C_{pc}, ρ, k_w, and A_R are the activation energy for the reaction, the heat capacity, the coolant mass, the heat capacity of coolant, the density, the heat transfer coefficient for the cooling jacket, and the surface of the cooling jacket, respectively. Besides, $\angle H_{RAB}$, $\angle H_{RBC}$, and $\angle H_{RAD}$ are the enthalpies of reaction k_1, the enthalpies of reaction k_2, and the enthalpies of reaction k_3, respectively. Table A.4.1 provides the parameters values required for the mathematical model described the highly nonlinear CSTR in the case study 6.

Table A.4 The values for the process parameters and steady-state for the highly nonlinear continuously stirred tank reactor in case study 6

Parameters	Values
C_{A0}	5.10 *mol/l*
C_A	2.14 *mol/l*
C_B	1.09 *mol/l*
T	114.2 ^oC

T_0	104.9 ^oC
T_c	112.9 ^oC
$\frac{\dot{V}}{V_R}$	3 hr^{-1} to 35 hr^{-1}
$\dot{Q}$	-9000 kJ/hr to 0 kJ/hr
k_1	$(1.287 \pm 0.04) \times 10^{12}$ hr^{-1}
k_2	$(1.287 \pm 0.04) \times 10^{12}$ hr^{-1}
k_3	$(9.043 \pm 0.27) \times 10^{9}$ $molA^{-1}hr^{-1}$
E_1	-9758.3 K
E_2	-9758.3 K
E_3	-8560 K
$\angle H_{RAB}$	(4.2 ± 2.36) $kJ/molA^{-1}$
$\angle H_{RBC}$	$-(11.0 \pm 1.92)$ $kJ/molB^{-1}$
$\angle H_{RAD}$	$-(41.85 \pm 1.41)$ $kJ/molA^{-1}$
C_p	(3.01 ± 0.04) $kJkg^{-1}K^{-1}$
ρ	$(0.9342 \pm 4.0 \times 10^{-4})$ kg/l
k_w	(4032 ± 120) $kJhr^{-1}m^{-2}K^{-1}$
A_R	0.215 m^2
V_R	0.01 m^3
m_k	5.0 kg
C_{pc}	(2.0 ± 0.05) $kJkg^{-1}K^{-1}$

Appendix B MATLAB code for an ensemble locally weighted Kernel partial least square algorithm

Appendix B.1 MATLAB code for an ensemble locally weighted Kernel partial least square (E-LW-KPLS) algorithm

This section displays a sample of the MATLAB code for the E-LW-KPLS algorithm. The main program code for the E-LW-KPLS (main2.m) with two subsets of locally weighted Kernel partial least square (LW-KPLS) is presented. Additionally, the other MATLAB codes for the LW-KPLS function (lw_kpls.m), centering function (centering.m), Kernel matrix function (kernelmatrix.m), Kernel centering function (kernelcentering.m), dualpls function (dualpls.m) and estimate Sigma function (estimateSigma.m) are given in website. The web site's link is https://drive.google.com/open?id=1NXUPXG23hSj59sCVpKs2T_7hOSwQFRWC.

```
clear, clc

%% --- Description ---
% This is a sample program of ensemble locally weighted
Kernel partial least squares (E-LW-KPLS)

%% --- Nomeclature ---
% N1            : number of training samples
% N2            : number of testing samples
% M1            : number of input variables
% M2            : number of output variables
%
% --- Input ---
% X  (N1 * M1): input observed data matrix for training
data
% Y  (N1 * M2): output observed data matrix for training
data
% Xq (N2 * M1): query/new input matrix
% Yq (N2 * M2): true or query/new output matrix
% LV (1 * 1): number of latent variable to be calculated
% phi (1 * 1): localization parameter for distance
% b (1 * 1): Kernel parameter
%
% --- Output ---
% RMSE2 (1 * 1): root mean square for testing data
% MAE2 (1 * 1): mean absolute errorfor testing data
% MSE2 (1 * 1): mean square errorfor testing data
% t2 (1 * 1): central processing unit time for testing
data
% RMSE1 (1 * 1): root mean square for training data
% MAE1 (1 * 1): mean absolute errorfor training data
% MSE1 (1 * 1): mean square errorfor training data
% t1 (1 * 1): central processing unit time for training
data
% Avr_Y (1 * 1): average value of real output variable
matrix for training data
% Avr_YY (1 * 1): average value of predicted output
variable matrix for training data
% Avr_Yq (1 * 1): average value of real output variable
matrix for testing data
% Avr_Yest (1 * 1): average value of predicted output
variable matrix for testing data
% Rsq_test (1 * 1): R-squared for testing data
% Rsq_train(1 * 1): R-squared for training data
% E (1 * 1)       : error of approximation (Ea) value

%% --- Data loading for testing data ---
load single.mat
tic
%% --- Parameter setting for testing data ---
XA  = [CA(1:2:7500,1),T(1:2:7500,1)];
XB  = [CA(2:2:7500,1),T(2:2:7500,1)];
YA  = CB(1:2:7500,1);
YB  = CB(2:2:7500,1);

XqA = [CA(7501:2:10000,1),T(7501:2:10000,1)];
```

```
XqB = [CA(7502:2:10000,1),T(7502:2:10000,1)];
YqA = CB(7501:2:10000,1);
YqB = CB(7502:2:10000,1);

%% --- Parameter setting for training and testing data -
LV  = 1;    % number of latent variables
phi = .1;   % localization parameter
b   = 23;   % Kernel parameter

%% --- Output estimation for testing data ---
[YestA,    W,    T,    P,    q,    omega,    tq]    =
lw_kpls(XA,YA,XqA,LV,phi,b);
[YestB,    W,    T,    P,    q,    omega,    tq]    =
lw_kpls(XB,YB,XqB,LV,phi,b);

Yest = [YestA;YestB];
Yq   = [YqA;YqB];

%% --- Results for testing data ---
formatlong
RMSE2 = sqrt(sum((Yq(:)-Yest(:)).^2)/numel(Yq(:)))

MAE2 = mean(abs(Yq(:)-Yest(:)))

MSE2 = mean((Yq(:)-Yest(:)).^2)

toc
t2 = toc

%% --- Data loading for training data ---
load single.mat
tic
%% --- Parameter setting for training data ---
XA = [CA(1:2:7500,1),T(1:2:7500,1)];
XB = [CA(2:2:7500,1),T(2:2:7500,1)];

YA = CB(1:2:7500,1);
```

```
YB = CB(2:2:7500,1);


%% --- Output estimation for training data ---
[YYA, W, T, P, q, omega, tq] = lw_kpls(XA,YA,XA,LV,phi,b);
[YYB, W, T, P, q, omega, tq] = lw_kpls(XB,YB,XB,LV,phi,b);


YY = [YYA;YYB];
Y  = [YA;YB];


%% --- Results for training data ---
RMSE1 = sqrt(sum((Y(:)-YY(:)).^2)/numel(Y(:)))

MAE1 = mean(abs(Y(:)-YY(:)))

MSE1 = mean((Y(:)-YY(:)).^2)


toc
t1 = toc


%% --- Results for testing and training data ---
Avr_Y      = sum(Y(:))/numel(Y(:));
Avr_YY     = sum(YY(:))/numel(YY(:))


Avr_Yq     = sum(Yq(:))/numel(Yq(:));
Avr_Yest   = sum(Yest(:))/numel(Yest(:))


Rsq_test   =  1-sum((Yq(:)  -  Yest(:)).^2)/sum((Yest(:)  -
mean(Yq(:))).^2)
Rsq_train   =   1-sum((Y(:)   -   YY(:)).^2)/sum((YY(:)   -
mean(Y(:))).^2)


N2 = size(Yq,1);
N1 = size(Y,1);
E   =  (N1/(N2+N1))*RMSET  +  (N2/(N2+N1))*RMSE  +  abs(RMSET-
RMSE)


%   coded   by   Christine   yeo   @   Curtin   Uni.
Malaysia(christineyeo@curtin.edu.my)
```

```
% created date: 7th Jan 2016.
% last update : 7th Dec 2017.
```

Listing B.1.main2.m

Appendix C Results for the ensemble locally weighted Kernel partial least square algorithm

Appendix C.1 Results for the ensemble locally weighted Kernel partial least square (E-LW-KPLS) algorithm

For case studies 1, 2 and 3, the obtained current results from the newly developed E-LW-KPLS, the locally weighted Kernel partial least square (LW-KPLS) and the locally weighted partial least square (LW-PLS) algorithms are illustrated in this section. Moreover, the E-LW-KPLS consists of numerous sets of LW-KPLS models running simultaneously. Hence, the obtained results from 1 set of LW-KPLS to 30 subsets of LW-KPLS models are also shown in the following tables.

C.1.1 Case study 1: Numerical example 1

Tables C.1.1.1 to C.1.1.6 show the results from the E-LW-KPLS, LW-KPLS and LW-PLS models for case study 1. The Kernel function used in the E-LW-KPLS and LW-KPLS algorithms is the inverse multi-quadric Kernel, and the Kernel parameter *b* is fixed at 0.3.

Table C.1.1.1 Results for E-LW-KPLS, LW-KPLS and LW-PLS algorithms for case study 1

No. of set	**1 set**	**1 set**	**2 sets**	**3 sets**	**4 sets**	**5 sets**
Models	LW-PLS	LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.4822	0.2244	0.2266	0.2229	0.2244	0.2245
MAE_1	0.4264	0.1721	0.1740	0.1717	0.1741	0.1738
MSE_1	0.2325	0.0504	0.0513	0.0497	0.0504	0.0504
t_1 (sec)	14	1003	196	98	61	42
$RMSE_2$	0.4859	0.2227	0.2266	0.2259	0.2313	0.2329
MAE_2	0.4287	0.1706	0.1724	0.1725	0.1776	0.1779
MSE_2	0.2325	0.0496	0.0513	0.0510	0.0535	0.0542
t_2 (sec)	4	218	54	26	16	11
E_a	0.4868	0.2257	0.2266	0.2267	0.2330	0.2350

Table C.1.1.2 Results for E-LW-KPLS algorithm for case study 1

No. of set	**6 sets**	**7 sets**	**8 sets**	**9 sets**	**10 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.2239	0.2270	0.2227	0.2235	0.2250
MAE_1	0.1732	0.1758	0.1734	0.1730	0.1750
MSE_1	0.0501	0.0515	0.0496	0.0499	0.0506
t_1 (sec)	27	20	19	15	12
$RMSE_2$	0.2345	0.2369	0.2364	0.2443	0.2460
MAE_2	0.1796	0.1819	0.1801	0.1858	0.1880
MSE_2	0.0550	0.0561	0.0559	0.0597	0.0605
t_2 (sec)	8	6	5	4	4
E_a	0.2372	0.2393	0.2398	0.2495	0.2513

Table C.1.1.3 Results for E-LW-KPLS algorithm for case study 1

No. of set	**11 sets**	**12 sets**	**13 sets**	**14 sets**	**15 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.2202	0.2225	0.2222	0.2228	0.2225
MAE_1	0.1706	0.1738	0.1735	0.1729	0.1728
MSE_1	0.0485	0.0495	0.0494	0.0496	0.0495
t_1 (sec)	10	10	9	8	7
$RMSE_2$	0.2458	0.2449	0.2537	0.2448	0.2546
MAE_2	0.1862	0.1887	0.1943	0.1864	0.1944
MSE_2	0.0604	0.0600	0.0644	0.0599	0.0648
t_2 (sec)	4	4	3	3	2
E_a	0.2521	0.2487	0.2616	0.2503	0.2626

Table C.1.1.4 Results for E-LW-KPLS algorithm for case study 1

No. of set	**16 sets**	**17 sets**	**18 sets**	**19 sets**	**20 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.2239	0.2228	0.2229	0.2230	0.2240
MAE_1	0.1750	0.1741	0.1737	0.1745	0.1762
MSE_1	0.0502	0.0496	0.0497	0.0497	0.0502
t_1 (sec)	6	6	6	6	6
$RMSE_2$	0.2564	0.2612	0.2650	0.2686	0.2699
MAE_2	0.1954	0.2013	0.2029	0.2045	0.2061
MSE_2	0.0657	0.0682	0.0702	0.0721	0.0729
t_2 (sec)	2	2	2	2	2
E_a	0.2645	0.2709	0.2755	0.2800	0.2814

Table C.1.1.5 Results for E-LW-KPLS algorithm for case study 1

No. of set	**21 sets**	**22 sets**	**23 sets**	**24 sets**	**25 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.2227	0.2191	0.2221	0.2198	0.2214
MAE_1	0.1743	0.1713	0.1745	0.1728	0.1748
MSE_1	0.0496	0.0480	0.0493	0.0483	0.0490
t_1 (sec)	6	6	6	5	5
$RMSE_2$	0.2690	0.2737	0.2768	0.2755	0.2800
MAE_2	0.2076	0.2079	0.2126	0.2121	0.2145
MSE_2	0.0724	0.0749	0.0766	0.0759	0.0784
t_2 (sec)	2	2	2	2	2
E_a	0.2806	0.2873	0.2904	0.2894	0.2946

Table C.1.1.6 Results for E-LW-KPLS algorithm for case study 1

No. of set	**26 sets**	**27 sets**	**28 sets**	**29 sets**	**30 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.2201	0.2205	0.2202	0.2210	0.2202
MAE_1	0.1731	0.1733	0.1727	0.1737	0.1724
MSE_1	0.0484	0.0486	0.0485	0.0488	0.0485
t_1 (sec)	5	5	5	5	4
$RMSE_2$	0.2827	0.2886	0.2792	0.2877	0.2917
MAE_2	0.2156	0.2191	0.2140	0.2238	0.2215
MSE_2	0.0799	0.0833	0.0780	0.0828	0.0851
t_2 (sec)	2	2	2	2	2
E_a	0.2983	0.3056	0.2940	0.3044	0.3096

C.1.2 Case study 2: A single chemical reactor

The results from the E-LW-KPLS, LW-KPLS and LW-PLS models for case study 2 are shown in Tables C.1.2.1 to C.1.2.6. The polynomial Kernel is used as the Kernel function in the E-LW-KPLS, and LW-KPLS algorithms and then the Kernel parameter *b* is set at 0.01.

Table C.1.2.1 Results for E-LW-KPLS, LW-KPLS and LW-PLS algorithms for case study 2

No. of set	**1 set**	**1 set**	**2 sets**	**3 sets**	**4 sets**	**5 sets**
Models	LW-PLS	LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.0075	0.0065	0.0064	0.0064	0.0058	0.0052
MAE_1	0.0028	0.0028	0.0028	0.0028	0.0028	0.0028
MSE_1	0.0001	$4.2\text{x}10^{-5}$	$4.1\text{x}10^{-5}$	$4.1\text{x}10^{-5}$	$3.4\text{x}10^{-5}$	$2.7\text{x}10^{-5}$
t_1 (sec)	10	907	256	98	61	38
$RMSE_2$	0.0032	0.0033	0.0033	0.0033	0.0033	0.0033
MAE_2	0.0026	0.0027	0.0027	0.0027	0.0027	0.0027
MSE_2	$1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$
t_2 (sec)	3	205	64	27	18	13
E_a	0.0108	0.0089	0.0087	0.0087	0.0077	0.0067

Table C.1.2.2 Results for E-LW-KPLS algorithm for case study 2

No. of set	**6 sets**	**7 sets**	**8 sets**	**9 sets**	**10 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.0061	0.0050	0.0061	0.0060	0.0051
MAE_1	0.0028	0.0027	0.0028	0.0028	0.0027
MSE_1	$3.8\text{x}10^{-5}$	$2.5\text{x}10^{-5}$	$3.8\text{x}10^{-5}$	$3.6\text{x}10^{-5}$	$2.6\text{x}10^{-5}$
t_1 (sec)	27	21	19	15	13
$RMSE_2$	0.0033	0.0033	0.0033	0.0034	0.0033
MAE_2	0.0027	0.0027	0.0027	0.0027	0.0027
MSE_2	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$	$1.1\text{x}10^{-5}$
t_2 (sec)	9	7	6	5	5
E_a	0.0082	0.0063	0.0083	0.0080	0.0065

Table C.1.2.3 Results for E-LW-KPLS algorithm for case study 2

No. of set	**11 sets**	**12 sets**	**13 sets**	**14 sets**	**15 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.0047	0.0042	0.0040	0.0041	0.0049
MAE_1	0.0027	0.0027	0.0026	0.0027	0.0027
MSE_1	$2.2x10^{-5}$	$1.8x10^{-5}$	$1.6x10^{-5}$	$1.7x10^{-5}$	$2.4x10^{-5}$
t_1 (sec)	13	10	9	8	8
$RMSE_2$	0.0033	0.0033	0.0033	0.0034	0.0034
MAE_2	0.0027	0.0027	0.0027	0.0027	0.0027
MSE_2	$1.1x10^{-5}$	$1.1x10^{-5}$	$1.1x10^{-5}$	$1.1x10^{-5}$	$1.1x10^{-5}$
t_2 (sec)	4	4	4	3	3
E_a	0.0057	0.0049	0.0045	0.0047	0.0060

Table C.1.2.4 Results for E-LW-KPLS algorithm for case study 2

No. of set	**16 sets**	**17 sets**	**18 sets**	**19 sets**	**20 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.0078	0.0087	0.0078	0.0079	0.0080
MAE_1	0.0029	0.0029	0.0029	0.0029	0.0029
MSE_1	$6.1x10^{-5}$	$7.6x10^{-5}$	$6.1x10^{-5}$	$6.2x10^{-5}$	$6.4x10^{-5}$
t_1 (sec)	7	7	7	7	6
$RMSE_2$	0.0034	0.0035	0.0034	0.0036	0.0036
MAE_2	0.0027	0.0028	0.0027	0.0028	0.0028
MSE_2	$1.1x10^{-5}$	$1.2x10^{-5}$	$1.2x10^{-5}$	$1.3x10^{-5}$	$1.3x10^{-5}$
t_2 (sec)	3	3	3	3	3
E_a	0.0111	0.0126	0.0111	0.0111	0.0113

Table C.1.2.5 Results for E-LW-KPLS algorithm for case study 2

No. of set	**21 sets**	**22 sets**	**23 sets**	**24 sets**	**25 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.0074	0.0071	0.0071	0.0069	0.0070
MAE_1	0.0029	0.0029	0.0028	0.0028	0.0028
MSE_1	$5.7x10^{-5}$	$5.1x10^{-5}$	$5x10^{-5}$	$4.8x10^{-5}$	$4.9x10^{-5}$
t_1 (sec)	6	6	5	5	5
$RMSE_2$	0.0036	0.0036	0.0035	0.0036	0.0037
MAE_2	0.0028	0.0028	0.0028	0.0028	0.0028
MSE_2	$1.3x10^{-5}$	$1.3x10^{-5}$	$1.2x10^{-5}$	$1.3x10^{-5}$	$1.4x10^{-5}$
t_2 (sec)	3	3	2	2	2
E_a	0.0102	0.0097	0.0098	0.0094	0.0094

Table C.1.2.6 Results for E-LW-KPLS algorithm for case study 2

No. of set	**26 sets**	**27 sets**	**28 sets**	**29 sets**	**30 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.0068	0.0068	0.0068	0.0068	0.0067
MAE_1	0.0028	0.0028	0.0027	0.0028	0.0028
MSE_1	$4.6x10^{-5}$	$4.6x10^{-5}$	$4.7x10^{-5}$	$4.6x10^{-5}$	$4.5x10^{-5}$
t_1 (sec)	5	5	5	5	5
$RMSE_2$	0.0036	0.0035	0.0036	0.0037	0.0037
MAE_2	0.0028	0.0028	0.0027	0.0028	0.0029
MSE_2	$1.3x10^{-5}$	$1.2x10^{-5}$	$1.3x10^{-5}$	$1.4x10^{-5}$	$1.4x10^{-5}$
t_2 (sec)	2	2	2	2	2
E_a	0.0091	0.0093	0.0093	0.0091	0.0090

C.1.3 Case study 3: Wastewater treatment

In this section, Tables C.1.3.1 to C.1.3.6 illustrate the obtained current results from E-LW-KPLS, LW-KPLS and LW-PLS algorithms for case study 3. An inverse multi-quadric Kernel function is adopted in E-LW-KPLS and LW-KPLS algorithms and then the selected Kernel parameter, *b* is 8.

Table C.1.3.1 Results for E-LW-KPLS, LW-KPLS and LW-PLS algorithms for case study 3

No. of sets	**1 set**	**1 set**	**2 sets**	**3 sets**	**4 sets**	**5 sets**
Models	LW-PLS	LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.7103	0.7330	0.7322	0.7309	0.7294	0.7283
MAE_1	0.5811	0.5859	0.5851	0.5841	0.5828	0.5820
MSE_1	0.5045	0.5373	0.5361	0.5342	0.5320	0.5304
t_1 (sec)	11	1067	225	101	61	42
$RMSE_2$	0.7744	0.7413	0.7412	0.7415	0.7425	0.7429
MAE_2	0.6284	0.6053	0.6052	0.6054	0.6060	0.6062
MSE_2	0.5997	0.5496	0.5494	0.5498	0.5513	0.5519
t_2 (sec)	4	238	64	30	19	13
E_a	0.7905	0.7434	0.7435	0.7441	0.7458	0.7466

Table C.1.3.2 Results for E-LW-KPLS algorithm for case study 3

No. of sets	**6 sets**	**7 sets**	**8 sets**	**9 sets**	**10 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.7259	0.7243	0.7223	0.7212	0.7176
MAE_1	0.5800	0.5791	0.5784	0.5761	0.5738
MSE_1	0.5269	0.5246	0.5217	0.5201	0.5149
t_1 (sec)	31	25	20	16	14
$RMSE_2$	0.7418	0.7443	0.7401	0.7436	0.7475
MAE_2	0.6047	0.6068	0.6025	0.6061	0.6095
MSE_2	0.5502	0.5539	0.5477	0.5530	0.5588
t_2 (sec)	10	8	7	5	5
E_a	0.7458	0.7493	0.7445	0.7493	0.7550

Table C.1.3.3 Results for E-LW-KPLS algorithm for case study 3

No. of sets	**11 sets**	**12 sets**	**13 sets**	**14 sets**	**15 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.7157	0.7139	0.7137	0.7066	0.7040
MAE_1	0.5712	0.5693	0.5692	0.5641	0.5605
MSE_1	0.5123	0.5097	0.5094	0.4993	0.4956
t_1 (sec)	12	11	10	9	8
$RMSE_2$	0.7461	0.7314	0.7462	0.7432	0.7508
MAE_2	0.6087	0.5966	0.6090	0.6018	0.6124
MSE_2	0.5567	0.5350	0.5568	0.5524	0.5637
t_2 (sec)	4	4	3	3	3
E_a	0.7537	0.7358	0.7543	0.7523	0.7625

Table C.1.3.4 Results for E-LW-KPLS algorithm for case study 3

No. of sets	**16 sets**	**17 sets**	**18 sets**	**19 sets**	**20 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.7046	0.6978	0.6982	0.6887	0.6881
MAE_1	0.5621	0.5564	0.5562	0.5492	0.5479
MSE_1	0.4965	0.4869	0.4874	0.4743	0.4735
t_1 (sec)	8	7	7	7	6
$RMSE_2$	0.7510	0.7581	0.7545	0.7501	0.7608
MAE_2	0.6125	0.6130	0.6108	0.6059	0.6182
MSE_2	0.5641	0.5748	0.5693	0.5626	0.5789
t_2 (sec)	3	3	3	3	3
E_a	0.7626	0.7732	0.7686	0.7654	0.7790

Table C.1.3.5 Results for E-LW-KPLS algorithm for case study 3

No. of sets	**21 sets**	**22 sets**	**23 sets**	**24 sets**	**25 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.6876	0.6879	0.6809	0.6752	0.6693
MAE_1	0.5471	0.5464	0.5409	0.5369	0.5327
MSE_1	0.4727	0.4732	0.4637	0.4559	0.4479
t_1 (sec)	6	6	5	5	5
$RMSE_2$	0.7486	0.7572	0.7640	0.7676	0.7685
MAE_2	0.6112	0.6174	0.6166	0.6219	0.6230
MSE_2	0.5605	0.5734	0.5838	0.5892	0.5906
t_2 (sec)	3	3	2	2	2
E_a	0.7639	0.7746	0.7848	0.7907	0.7933

Table C.1.3.6 Results for E-LW-KPLS algorithm for case study 3

No. of sets	**26 sets**	**27 sets**	**28 sets**	**29 sets**	**30 sets**
Models	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS	E-LW-KPLS
$RMSE_1$	0.6628	0.6632	0.6694	0.6599	0.6628
MAE_1	0.5241	0.5240	0.5300	0.5238	0.5241
MSE_1	0.4394	0.4398	0.4481	0.4355	0.4394
t_1 (sec)	5	5	5	5	5
$RMSE_2$	0.7605	0.7870	0.7990	0.7760	0.7605
MAE_2	0.6143	0.6375	0.6416	0.6324	0.6143
MSE_2	0.5784	0.6193	0.6385	0.6022	0.5784
t_2 (sec)	2	2	2	2	2
E_a	0.7849	0.8179	0.8315	0.8050	0.7849

Appendix D MATLAB code for an ensemble locally weighted independent component Kernel partial least square algorithm

Appendix D.1 MATLAB code for an ensemble locally weighted independent component Kernel partial least square (E-LW-IC-KPLS) algorithm

A sample of the MATLAB code for the E-LW-IC-KPLS algorithm is illustrated in this section. The main MATLAB code for the E-LW-IC-KPLS (main2.m) with two sets of a locally weighted independent component Kernel partial least square (LW-IC-KPLS) model is presented first. Moreover, the other MATLAB codes for the LW-IC-KPLS function (lw_ic_kpls.m), centering function (centering.m), Kernel matrix function (kernelmatrix.m), Kernel centering function (kernelcentering.m), dualpls function (dualpls.m), estimate Sigma function (estimateSigma.m), independent component analysis function (ica.m) and principal component analysis with singular value decomposition function (pca_svd.m) can be found in https://drive.google.com/open?id=1rS83rQqiyPLkS7lZMq-RllqZRZo1flv2.

```
clear, clc
%% --- Description ---
% This is a sample program of ensemble locally weighted
independent component Kernel partial least squares (E-LW-
IC-KPLS)
%
%% --- Nomeclature ---
% N1             : number of training samples
% N2             : number of testing samples
% M1             : number of input variables
% M2             : number of output variables
%
% --- Input ---
% X  (N1 * M1): input observed data matrix for training
data
% Y (N1 * M2): output observed data matrix for training
data
```

```
% Xq (N2 * M1): query/new input matrix
% Yq (N2 * M2): true or query/new output matrix
% LV (1 * 1): number of latent variable to be calculated
% phi (1 * 1): localization parameter for distance
% b (1 * 1): Kernel parameter
% numPC (1 * 1): number of principal components to be
calculated
% epsilon (1 * 1): maximum error for convergence (default:
0.1)
% max_itr (1 * 1): maximum number of iteration (default:
1000)
%
% --- Output ---
% RMSE2 (1 * 1): root mean square for testing data
% MAE2 (1 * 1): mean absolute errorfor testing data
% MSE2 (1 * 1): mean square errorfor testing data
% t2 (1 * 1): central processing unit time for testing
data
% RMSE1 (1 * 1): root mean square for training data
% MAE1 (1 * 1): mean absolute errorfor training data
% MSE1 (1 * 1): mean square errorfor training data
% t1 (1 * 1): central processing unit time for training
data
% Avr_Y (1 * 1): average value of real output variable
matrix for training data
% Avr_YY (1 * 1): average value of predicted output
variable matrix for training data
% Avr_Yq (1 * 1): average value of real output variable
matrix for testing data
% Avr_Yest (1 * 1): average value of predicted output
variable matrix for testing data
% Rsq_test (1 * 1): R-squared for testing data
% Rsq_train(1 * 1): R-squared for training data
% E (1 * 1)       : error of approximation (Ea) value


%% --- Data loading for testing data ---
load sine.mat
```

```
tic

%% --- Parameter setting for training data ---
XA = x(1:2:7500,1);
XB = x(2:2:7500,1);

YA = y(1:2:7500,1);
YB = y(2:2:7500,1);

XqA = x(7501:2:10000,1);
XqB = x(7502:2:10000,1);
YqA = y(7501:2:10000,1);
YqB = y(7502:2:10000,1);

%% --- Parameter setting for training and testing data -
LV  = 1;    % number of latent variables
phi = .1;    % localization parameter
numIC = 1; % number of ICs
epsilon = 0.00001;
max_itr = 1000;
b = 2; % Kernel parameter

%% --- Output estimation for testing data ---
[YestA,     W,     T,     P,     q,     omega,     tq]     =
lw_ic_kpls(XA,YA,XqA,LV,phi,numIC,epsilon,max_itr,b);
[YestB,     W,     T,     P,     q,     omega,     tq]     =
lw_ic_kpls(XB,YB,XqB,LV,phi,numIC,epsilon,max_itr,b);


Yest = [YestA;YestB];
Yq = [YqA;YqB];

%% --- Result for testing data ---
formatlong
RMSE2 = sqrt(sum((Yq(:)-Yest(:)).^2)/numel(Yq(:)))

MAE2 = mean(abs(Yq(:)-Yest(:)))
```

```
MSE2 = mean((Yq(:)-Yest(:)).^2)
toc
t2 = toc

%% --- Data loading for training data ---
load sine.mat
tic

%% --- Parameter setting for training data ---
XA = x(1:2:7500,1);
XB = x(2:2:7500,1);
YA = y(1:2:7500,1);
YB = y(2:2:7500,1);

%% --- Output estimation for training data ---
[YYA,     W,     T,     P,     q,     omega,     tq]     =
lw_ic_kpls(XA,YA,XA,LV,phi,numIC,epsilon,max_itr,b);
[YYB,     W,     T,     P,     q,     omega,     tq]     =
lw_ic_kpls(XB,YB,XB,LV,phi,numIC,epsilon,max_itr,b);

YY = [YYA;YYB];
Y = [YA;YB];

%% --- Result for training data ---
formatlong
RMSE1 = sqrt(sum((Y(:)-YY(:)).^2)/numel(Y(:)))

MAE1 = mean(abs(Y(:)-YY(:)))

MSE1 = mean((Y(:)-YY(:)).^2)

toc
t1 = toc

%% --- Results for testing and training data ---
Avr_Y = sum(Y(:))/numel(Y(:))
Avr_YY = sum(YY(:))/numel(YY(:))
```

```
Avr_Yq = sum(Yq(:))/numel(Yq(:))
Avr_Yest = sum(Yest(:))/numel(Yest(:))


Rsq_test = 1-sum((Yq(:) - Yest(:)).^2)/sum((Yest(:) -
mean(Yq(:))).^2)
Rsq_train = 1-sum((Y(:) - YY(:)).^2)/sum((YY(:) -
mean(Y(:))).^2)


N = size(Yq,1);
NN = size(Y,1);
E = (NN/(N+NN))*RMSET + (N/(N+NN))*RMSE + abs(RMSET -
RMSE)


% coded by Christine yeo @ Curtin Uni.
Malaysia(christineyeo@curtin.edu.my)
% created date: 7th Jan 2016.
% last update : 7th Dec 2017.
```

Listing D.1.main2.m

Appendix E Results for the ensemble locally weighted independent component Kernel partial least square algorithm

Appendix E.1 Results for an ensemble locally weighted independent component Kernel partial least square (E-LW-IC-KPLS) algorithm

In this section, the results of case studies 4, 5 and 6 from the newly developed E-LW-IC-KPLS, locally weighted independent component Kernel partial least square (LW-IC-KPLS), locally weighted Kernel partial least square (LW-KPLS) and locally weighted partial least square (LW-PLS) algorithms are presented. For the E-LW-IC-KPLS model, several LW-IC-KPLS models are run spontaneously. The tables below report the results for 1 set of the LW-IC-KPLS to 30 subsets of LW-IC-KPLS models.

E.1.1 Case study 4: Numerical example 2

Tables E.1.1.1 to E.1.1.6 show the results from the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 4. The Kernel function used in the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS models is the Log Kernel, and the Kernel parameter *b* is fixed at 2.

Table E.1.1.1 Results for E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 4

No. of set	**1 set**	**1 set**	**1 sets**	**2 sets**	**3 sets**	**4 sets**
Models	LW-PLS	LW-KPLS	LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
RMSE$_1$	0.2574	0.2593	0.2269	0.2270	0.2272	0.2271
MAE$_1$	0.2140	0.2149	0.1911	0.1911	0.1912	0.1911
MSE$_1$	0.0663	0.0672	0.0515	0.0515	0.0516	0.0516
t$_1$ (sec)	9	449	7146	1726	728	437
RMSE$_2$	0.2591	0.2643	0.2322	0.2324	0.2329	0.2334
MAE$_2$	0.2170	0.2206	0.1971	0.1972	0.1975	0.1977
MSE$_2$	0.0671	0.0699	0.0539	0.0540	0.0542	0.0545
t$_2$ (sec)	3	112	1289	309	147	89
E$_a$	0.2595	0.2656	0.2335	0.2338	0.2343	0.2350

Table E.1.1.2 Results for E-LW-IC-KPLS algorithm for case study 4

No. of set	**5 sets**	**6 sets**	**7 sets**	**8 sets**	**9 sets**	**10 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
RMSE$_1$	0.2270	0.2271	0.2252	0.2272	0.2270	0.2271
MAE$_1$	0.1912	0.1911	0.1892	0.1911	0.1909	0.1911
MSE$_1$	0.0515	0.0516	0.0507	0.0516	0.0515	0.0516
t$_1$ (sec)	291	210	169	133	105	81
RMSE$_2$	0.2334	0.2334	0.2364	0.2347	0.2339	0.2361
MAE$_2$	0.1979	0.1979	0.1997	0.1985	0.1981	0.1997
MSE$_2$	0.0545	0.0545	0.0559	0.0551	0.0547	0.0557
t$_2$ (sec)	60	43	33	28	24	18
E$_a$	0.2350	0.2350	0.2392	0.2366	0.2356	0.2383

Table E.1.1.3 Results for E-LW-IC-KPLS algorithm for case study 4

No. of set	**11 sets**	**12 sets**	**13 sets**	**14 sets**	**15 sets**	**16 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.2212	0.2282	0.2187	0.2277	0.2274	0.2281
MAE_1	0.1861	0.1916	0.1843	0.1907	0.1912	0.1914
MSE_1	0.0489	0.0521	0.0478	0.0518	0.0517	0.0520
t_1 (sec)	69	61	45	43	41	41
$RMSE_2$	0.2346	0.2368	0.2313	0.2407	0.2365	0.2378
MAE_2	0.1980	0.1993	0.1954	0.2019	0.1995	0.2004
MSE_2	0.0551	0.0561	0.0535	0.0579	0.0559	0.0566
t_2 (sec)	15	15	11	11	11	11
E_a	0.2380	0.2389	0.2345	0.2439	0.2387	0.2402

Table E.1.1.4 Results for E-LW-IC-KPLS algorithm for case study 4

No. of set	**17 sets**	**18 sets**	**19 sets**	**20 sets**	**21 sets**	**22 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.2302	0.2288	0.2313	0.2202	0.2277	0.2263
MAE_1	0.1930	0.1915	0.1931	0.1846	0.1908	0.1885
MSE_1	0.0530	0.0524	0.0535	0.0485	0.0518	0.0512
t_1 (sec)	39	36	34	33	21	20
$RMSE_2$	0.2403	0.2404	0.2403	0.2417	0.2420	0.2423
MAE_2	0.2020	0.2021	0.2017	0.2030	0.2027	0.2025
MSE_2	0.0578	0.0578	0.0577	0.0584	0.0586	0.0587
t_2 (sec)	10	10	9	8	7	6
E_a	0.2429	0.2433	0.2426	0.2471	0.2456	0.2463

Table E.1.1.5 Results for E-LW-IC-KPLS algorithm for case study 4

No. of set	**23 sets**	**24 sets**	**25 sets**	**26 sets**	**27 sets**	**28 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.2257	0.2322	0.2182	0.2247	0.2259	0.2321
MAE_1	0.1891	0.1932	0.1798	0.1872	0.1864	0.1924
MSE_1	0.0510	0.0539	0.0476	0.0505	0.0510	0.0539
t_1 (sec)	18	18	18	18	18	17
$RMSE_2$	0.2371	0.2440	0.2452	0.2407	0.2620	0.2492
MAE_2	0.1997	0.2034	0.2035	0.2008	0.2185	0.2071
MSE_2	0.0562	0.0595	0.0601	0.0579	0.0686	0.0621
t_2 (sec)	6	6	6	6	6	5
E_a	0.2399	0.2469	0.2519	0.2446	0.2710	0.2535

Table E.1.1.6 Results for E-LW-IC-KPLS algorithm for case study 4

No. of set	**29 sets**	**30 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.2310	0.2297
MAE_1	0.1907	0.1920
MSE_1	0.0533	0.0528
t_1 (sec)	17	17
$RMSE_2$	0.2442	0.2443
MAE_2	0.2029	0.2047
MSE_2	0.0596	0.0597
t_2 (sec)	5	5
E_a	0.2475	0.2480

E.1.2 Case study 5: Eukaryotic cell cycle regulation

The results from the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 5 are shown in Tables E.1.2.1 to E.1.2.7. The Log Kernel is used as the Kernel function in the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS models and then the selected Kernel parameter *b* is 0.8.

Table E.1.2.1 Results for E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 5

No. of set	**1 set**	**1 set**	**1 sets**	**2 sets**	**3 sets**	**4 sets**
Models	LW-PLS	LW-KPLS	LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0755	0.0702	0.0702	0.0700	0.0697	0.0695
MAE_1	0.0641	0.0556	0.0556	0.0554	0.0552	0.0550
MSE_1	0.0057	0.0049	0.0049	0.0049	0.0049	0.0048
t_1 (sec)	9	521	6850	1740	765	458
$RMSE_2$	0.0746	0.0697	0.0697	0.0697	0.0701	0.0698
MAE_2	0.0639	0.0554	0.0553	0.0554	0.0555	0.0554
MSE_2	0.0056	0.0049	0.0049	0.0049	0.0049	0.0049
t_2 (sec)	3	140	1310	335	141	84
E_a	0.0761	0.0706	0.0706	0.0702	0.0702	0.0699

Table E.1.2.2 Results for E-LW-IC-KPLS algorithm for case study 5

No. of set	**5 sets**	**6 sets**	**7 sets**	**8 sets**	**9 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0690	0.0691	0.0686	0.0696	0.0683
MAE_1	0.0546	0.0548	0.0542	0.0546	0.0540
MSE_1	0.0048	0.0048	0.0047	0.0048	0.0047
t_1 (sec)	296	204	157	123	98
$RMSE_2$	0.0702	0.0707	0.0709	0.0718	0.0705
MAE_2	0.0554	0.0562	0.0559	0.0576	0.0553
MSE_2	0.0049	0.0050	0.0050	0.0052	0.0050
t_2 (sec)	58	41	31	25	22
E_a	0.0705	0.0712	0.0715	0.0724	0.0710

Table E.1.2.3 Results for E-LW-IC-KPLS algorithm for case study 5

No. of set	**10 sets**	**11 sets**	**12 sets**	**13 sets**	**14 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0676	0.0690	0.0682	0.0675	0.0674
MAE_1	0.0533	0.0546	0.0542	0.0533	0.0533
MSE_1	0.0046	0.0048	0.0046	0.0046	0.0045
t_1 (sec)	83	71	62	54	48
$RMSE_2$	0.0711	0.0716	0.0719	0.0699	0.0717
MAE_2	0.0557	0.0567	0.0570	0.0549	0.0560
MSE_2	0.0051	0.0051	0.0052	0.0049	0.0051
t_2 (sec)	19	16	15	13	12
E_a	0.0720	0.0723	0.0728	0.0704	0.0728

Table E.1.2.4 Results for E-LW-IC-KPLS algorithm for case study 5

No. of set	**15 sets**	**16 sets**	**17 sets**	**18 sets**	**19 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0677	0.0678	0.0669	0.0667	0.0669
MAE_1	0.0536	0.0535	0.0529	0.0530	0.0530
MSE_1	0.0046	0.0046	0.0045	0.0045	0.0045
t_1 (sec)	41	40	38	33	32
$RMSE_2$	0.0719	0.0737	0.0737	0.0718	0.0722
MAE_2	0.0566	0.0591	0.0575	0.0565	0.0564
MSE_2	0.0052	0.0054	0.0054	0.0052	0.0052
t_2 (sec)	11	10	10	9	9
E_a	0.0730	0.0752	0.0753	0.0730	0.0735

Table E.1.2.5 Results for E-LW-IC-KPLS algorithm for case study 5

No. of set	**20 sets**	**21 sets**	**22 sets**	**23 sets**	**24 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0657	0.0672	0.0666	0.0670	0.0665
MAE_1	0.0518	0.0530	0.0527	0.0528	0.0524
MSE_1	0.0043	0.0045	0.0044	0.0045	0.0044
t_1 (sec)	30	21	21	21	21
$RMSE_2$	0.0720	0.0746	0.0727	0.0765	0.0752
MAE_2	0.0559	0.0596	0.0571	0.0597	0.0595
MSE_2	0.0052	0.0056	0.0053	0.0058	0.0057
t_2 (sec)	8	7	7	7	6
E_a	0.0735	0.0765	0.0742	0.0789	0.0773

Table E.1.2.6 Results for E-LW-IC-KPLS algorithm for case study 5

No. of set	**25 sets**	**26 sets**	**27 sets**	**28 sets**	**29 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0661	0.0652	0.0647	0.0645	0.0650
MAE_1	0.0521	0.0511	0.0509	0.0508	0.0516
MSE_1	0.0044	0.0043	0.0042	0.0042	0.0042
t_1 (sec)	20	19	18	18	18
$RMSE_2$	0.0732	0.0711	0.0731	0.0729	0.0733
MAE_2	0.0576	0.0551	0.0563	0.0564	0.0573
MSE_2	0.0054	0.0051	0.0053	0.0053	0.0054
t_2 (sec)	6	6	6	6	6
E_a	0.0750	0.0726	0.0752	0.0750	0.0753

Table E.1.2.7 Results for E-LW-IC-KPLS algorithm for case study 5

No. of set	**30 sets**
Models	E-LW-IC-KPLS
$RMSE_1$	0.0650
MAE_1	0.0512
MSE_1	0.0042
t_1 (sec)	18
$RMSE_2$	0.0785
MAE_2	0.0617
MSE_2	0.0062
t_2 (sec)	6
E_a	0.0819

E.1.3 Case study 6: A highly nonlinear CSTR

The results from the E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 6 are shown in Tables E.1.3.1 to E.1.3.6. The Log Kernel is used as the Kernel function in the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS algorithms and the chosen Kernel parameter *b* is 9.

Table E.1.3.1 Results for E-LW-IC-KPLS, LW-IC-KPLS, LW-KPLS and LW-PLS algorithms for case study 6

Output variable : Product concentration						
No. of set	**1 set**	**1 set**	**1 set**	**2 sets**	**3 sets**	**4 sets**
Models	LW-PLS	LW-KPLS	LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0155	0.0153	0.0153	0.0151	0.0149	0.0151
MAE_1	0.0111	0.0110	0.0110	0.0108	0.0107	0.0108
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0074	0.0076	0.0079	0.0082	0.0078	0.0106
MAE_2	0.0061	0.0063	0.0065	0.0066	0.0063	0.0077
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0216	0.0211	0.0209	0.0202	0.0202	0.0185
Output variable : Reactor temperature						
No. of set	**1 set**	**1 set**	**1 set**	**2 sets**	**3 sets**	**4 sets**
Models	LW-PLS	LW-KPLS	LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0152	0.0086	0.0086	0.0151	0.0029	0.0152
MAE_1	0.0006	0.0004	0.0004	0.0006	0.0003	0.0005
MSE_1	0.0002	$7.4x10^{-5}$	$7.4x10^{-5}$	0.0002	$8.2x10^{-5}$	$2.3x10^{-5}$
t_1 (sec)	17	1978	18307	9328	7024	4416
$RMSE_2$	0.0005	0.0003	0.0022	0.0013	0.0014	0.0003
MAE_2	0.0003	0.0002	0.0004	0.0005	0.0002	0.0002
MSE_2	$2.9x10^{-5}$	$7.2x10^{-5}$	$4.8x10^{-5}$	$1.7x10^{-5}$	$1.9x10^{-5}$	$6.6x10^{-5}$
t_2 (sec)	6	446	3243	1826	985	851
Ea_2 (2nd output)	0.0262	0.0148	0.0134	0.0255	0.0040	0.0264
Ea_3 (average)	0.0239	0.0180	0.0172	0.0229	0.0121	0.0225

Table E.1.3.2 Results for E-LW-IC-KPLS algorithm for case study 6

Output variable : Product concentration						
No. of set	**5 sets**	**6 sets**	**7 sets**	**8 sets**	**9 sets**	**10 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0147	0.0143	0.0147	0.0152	0.0143	0.0141
MAE_1	0.0106	0.0103	0.0105	0.0109	0.0103	0.0102
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0086	0.0099	0.0086	0.0104	0.0092	0.0090
MAE_2	0.0069	0.0076	0.0069	0.0079	0.0072	0.0072
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0194	0.0177	0.0193	0.0187	0.0181	0.0179

Output variable : Reactor temperature						
No. of set	**5 sets**	**6 sets**	**7 sets**	**8 sets**	**9 sets**	**10 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0152	0.0029	0.0152	0.0024	0.0029	0.0018
MAE_1	0.0005	0.0003	0.0005	0.0003	0.0003	0.0003
MSE_1	0.0002	$8.4\text{x}10^{-6}$	0.0002	$5.7\text{x}10^{-6}$	$8.3\text{x}10^{-6}$	$3.3\text{x}10^{-6}$
t_1 (sec)	3285	2024	1934	1230	1146	968
$RMSE_2$	0.0003	0.0004	0.0004	0.0007	0.0005	0.0004
MAE_2	0.0002	0.0002	0.0002	0.0003	0.0002	0.0002
MSE_2	$1.1\text{x}10^{-7}$	$1.7\text{x}10^{-7}$	$1.4\text{x}10^{-7}$	$4.8\text{x}10^{-7}$	$2.2\text{x}10^{-7}$	$1.4\text{x}10^{-7}$
t_2 (sec)	475	377	305	291	273	253
Ea_2 (2nd output)	0.0263	0.0048	0.0263	0.0037	0.0047	0.0029
Ea_3 (average)	0.0229	0.0113	0.0228	0.0112	0.0114	0.0104

Table E.1.3.3 Results for E-LW-IC-KPLS algorithm for case study 6

Output variable : Product concentration						
No. of set	**11 sets**	**12 sets**	**13 sets**	**14 sets**	**15 sets**	**16 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0138	0.0140	0.0136	0.0134	0.0138	0.0136
MAE_1	0.0101	0.0102	0.0099	0.0097	0.0100	0.0098
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0098	0.0094	0.0099	0.0104	1.0588	1.0586
MAE_2	0.0076	0.0075	0.0077	0.0078	0.0100	0.0108
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0078	0.0082
Ea_1 (1st output)	0.0168	0.0175	0.0164	0.0156	0.0166	0.0156
Output variable : Reactor temperature						
No. of set	**11 sets**	**12 sets**	**13 sets**	**14 sets**	**15 sets**	**16 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0152	0.0028	0.0028	0.0013	0.0013	0.0015
MAE_1	0.0005	0.0003	0.0003	0.0003	0.0003	0.0003
MSE_1	2.3×10^{-4}	8.1×10^{-6}	8×10^{-6}	1.7×10^{-6}	1.8×10^{-6}	2.2×10^{-6}
t_1 (sec)	881	764	683	505	404	397
$RMSE_2$	0.0005	0.0004	0.0005	0.0005	0.0004	0.0004
MAE_2	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
MSE_2	2.2×10^{-7}	2.0×10^{-7}	2.9×10^{-7}	2.1×10^{-7}	1.3×10^{-7}	1.2×10^{-7}
t_2 (sec)	205	194	188	171	105	91
Ea_2 (2nd output)	0.0262	0.0046	0.0045	0.0019	0.0021	0.0023
Ea_3 (average)	0.0215	0.0111	0.0105	0.0088	0.0094	0.0090

Table E.1.3.4 Results for E-LW-IC-KPLS algorithm for case study 6

Output variable : Product concentration						
No. of set	**17 sets**	**18 sets**	**19 sets**	**20 sets**	**21 sets**	**22 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0137	0.0145	0.0130	0.0130	0.0129	0.0131
MAE_1	0.0099	0.0104	0.0094	0.0095	0.0095	0.0095
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0106	1.0591	1.0588	0.0106	0.0102	0.0106
MAE_2	0.0081	0.0090	0.0103	0.0082	0.0079	0.0081
MSE_2	0.0001	0.0071	0.0079	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0160	0.0187	0.0149	0.0148	0.0150	0.0149
Output variable : Reactor temperature						
No. of set	**17 sets**	**18 sets**	**19 sets**	**20 sets**	**21 sets**	**22 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0028	0.0013	0.0029	0.0029	0.0026	0.0027
MAE_1	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003
MSE_1	$7.9x10^{-6}$	$1.8x10^{-6}$	$8.1x10^{-6}$	$8.5x10^{-6}$	$7x10^{-6}$	$7.1x10^{-6}$
t_1 (sec)	305	285	272	263	251	248
$RMSE_2$	0.0006	0.0003	0.0003	0.0004	0.0006	0.0005
MAE_2	0.0003	0.0002	0.0002	0.0002	0.0002	0.0002
MSE_2	$3.5x10^{-7}$	$7.2x10^{-8}$	$1.2x10^{-7}$	$1.5x10^{-7}$	$3.9x10^{-7}$	$2.8x10^{-7}$
t_2 (sec)	85	78	66	57	46	35
Ea_2 (2nd output)	0.0045	0.0022	0.0047	0.0048	0.0041	0.0043
Ea_3 (average)	0.0103	0.0105	0.0098	0.0098	0.0096	0.0096

Table E.1.3.5 Results for E-LW-IC-KPLS algorithm for case study 6

Output variable : Product concentration						
No. of set	**23 sets**	**24 sets**	**25 sets**	**26 sets**	**27 sets**	**28 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0128	0.0131	0.0128	0.0124	0.0127	0.0129
MAE_1	0.0093	0.0094	0.0093	0.0091	0.0092	0.0093
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0102	0.0105	0.0102	0.0106	0.0106	0.0111
MAE_2	0.0080	0.0081	0.0080	0.0083	0.0083	0.0086
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0147	0.0150	0.0148	0.0138	0.0143	0.0142
Output variable : Reactor temperature						
No. of set	**23 sets**	**24 sets**	**25 sets**	**26 sets**	**27 sets**	**28 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0023	0.0012	0.0029	0.0021	0.0011	0.0012
MAE_1	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003
MSE_1	$5.2x10^{-6}$	$1.5x10^{-6}$	$8.4x10^{-6}$	$4.4x10^{-6}$	$1.2x10^{-6}$	$1.5x10^{-6}$
t_1 (sec)	224	206	187	158	106	103
$RMSE_2$	0.0004	0.0003	0.0003	0.0004	0.0005	0.0006
MAE_2	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
MSE_2	$1.3x10^{-7}$	$9.6x10^{-8}$	$8.6x10^{-8}$	$1.3x10^{-7}$	$2.3x10^{-7}$	$3.9x10^{-7}$
t_2 (sec)	31	28	27	26	25	23
Ea_2 (2nd output)	0.0037	0.0019	0.0049	0.0034	0.0016	0.0017
Ea_3 (average)	0.0092	0.0085	0.0099	0.0089	0.0080	0.0080

Table E.1.3.6 Results for E-LW-IC-KPLS algorithm for case study 6

Output variable : Product concentration		
No. of set	**29 sets**	**30 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0132	0.0128
MAE_1	0.0094	0.0092
MSE_1	0.0002	0.0002
$RMSE_2$	0.0113	0.0106
MAE_2	0.0086	0.0082
MSE_2	0.0001	0.0001
Ea_1 (1st output)	0.0146	0.0144
Output variable : Reactor temperature		
No. of set	**29 sets**	**30 sets**
Models	E-LW-IC-KPLS	E-LW-IC-KPLS
$RMSE_1$	0.0048	0.0029
MAE_1	0.0004	0.0003
MSE_1	$2.3\text{x}10^{-5}$	$8.3\text{x}10^{-6}$
t_1 (sec)	95	87
$RMSE_2$	0.0006	0.0003
MAE_2	0.0002	0.0002
MSE_2	$3.7\text{x}10^{-7}$	$1.2\text{x}10^{-7}$
t_2 (sec)	22	21
Ea_2 (2nd output)	0.0079	0.00477
Ea_3 (average)	0.0113	0.0096

Appendix F MATLAB code for an expectation maximization ensemble locally weighted independent component Kernel partial least square algorithm

Appendix F.1 MATLAB code for the expectation maximization ensemble locally weighted independent component Kernel partial least square (EM-E-LW-IC-KPLS) algorithm

In this section, a sample of the MATLAB code for the EM-E-LW-IC-KPLS algorithm is illustrated. Firstly, the main MATLAB code for the EM-E-LW-IC-KPLS (main2.m) is presented. The other MATLAB codes for the LW-IC-KPLS are the LW-IC-KPLS function (lw_ic_kpls.m), centering function (centering.m), Kernel matrix function (kernelmatrix.m), Kernel centering function (kernelcentering.m), dualpls function (dualpls.m), estimate Sigma function (estimateSigma.m), independent component analysis function (ica.m), principal component analysis with singular value decomposition function (pca_svd.m), expectation maximization function (mixGaussEm.m) and log-sum-exp function (logsumexp.m) are provided in https://drive.google.com/open?id=1o1GTRjh_e3vpucGEyejQPripWOcxDoLf.

```
clear, clc

%% --- Description ---
% This is a sample program of expectation maximization
ensemble locally weighted independent component Kernel
partial least squares(EM-E-LW-IC-KPLS)

%% --- Nomeclature ---
% N1            : number of training samples
% N2            : number of testing samples
% M1            : number of input variables
% M2            : number of output variables
%
% --- Input ---
% X (N1 * M1): input observed data matrix for training
data
% Y (N1 * M2): output observed data matrix for training
data
% Xq (N2 * M1): query/new input matrix
% Yq (N2 * M2): true or query/new output matrix
% LV (1 * 1): number of latent variable to be calculated
% phi (1 * 1): localization parameter for distance
% b (1 * 1): Kernel parameter
% numPC (1 * 1): number of principal components to be
calculated
% epsilon (1 * 1): maximum error for convergence
(default: 0.1)
% max_itr (1 * 1): maximum number of iteration (default:
1000)
% kt (1 * 1): number of Gaussian component for
expectation maximization algorithm
%
% --- Output ---
% RMSE2 (1 * 1): root mean square for testing data
% MAE2 (1 * 1): mean absolute error for testing data
% MSE2 (1 * 1): mean square error for testing data
% t2 (1 * 1): central processing unit time for testing
data
% RMSE1 (1 * 1): root mean square for training data
% MAE1 (1 * 1): mean absolute error for training data
% MSE1 (1 * 1): mean square error for training data
% t1 (1 * 1): central processing unit time for training
data
% Avr_Y (1 * 1): average value of real output variable
matrix for training data
% Avr_YY (1 * 1): average value of predicted output
variable matrix for training data
% Avr_Yq (1 * 1): average value of real output variable
matrix for testing data
% Avr_Yest (1 * 1): average value of predicted output
variable matrix for testing data
% Rsq_test (1 * 1): R-squared for testing data
% Rsq_train(1 * 1): R-squared for training data
% E (1 * 1)        : error of approximation (Ea) value

%% --- Data loading for testing data ---
load Data_ECCR.mat

tic

% Generating missing data
u(1:20:10000,1)= NaN; % Generating 5% of missing data

%% --- Parameters setting for expectation maximization
algorithm ---
X = u(1:7500,1); % set input variables as X
kt = 1; % Set one gaussian component
```

```
%% --- Initial guess for expectation maximization
algorithm ---
%--- Fill in the missing values by their estimated
values ---
% Calculate mean and then input data has missing values
that filles with its mean
[n,d] = size(X);
mis = zeros(n,d);
cmis = zeros(n,d);
m = zeros(1,d);
mt = zeros(n,d);

while sum(isnan(X))>0,
mis    = isnan(X);
X(mis) = 0; % let missing data equal to zeros
cmis   = ones(n,d) - mis; % complement of mis
m  = (sum(X)./(n - sum(mis))); % calculating mean value
from observed data
mt = repmat(m,[n,1]); % creating a matrix with repeated
means
X = cmis.*X + mt.*mis; % replacing missing data with
mean value
end

X = X';

%% --- Impute new value for missing data using
expectation maximization algorithm ---
% training dataset
[z1,model,llh] = mixGaussEm(X,kt);

mu = model.mu';
u1 = cmis.*X' + repmat(mu,n,1).*mis; % replacing missing
data with new value

u2 = u(7501:10000,1); % test data

% substituting imputed value into test data
[n,d] = size(u2);
mis = zeros(n,d);
cmis = zeros(n,d);
%m = zeros(1,d);
mt = zeros(n,d);

while sum(isnan(u2))>0,
mis    = isnan(u2);
u2(mis) = 0; % let missing data equal to zeros
cmis   = ones(n,d) - mis; % complement of mis
%m  = (sum(X)./(n - sum(mis))); % calculating mean value
from observed data
```

```
mt = repmat(mu,[n,1]); % creating a matrix with repeated
means
u2 = cmis.*X + mt.*mis; % replacing missing data with
mean value
end

%% --- Parameter setting for training and testing data -
XA = u1(1:2:7500,1);
XB = u1(2:2:7500,1);

YA = v(1:2:7500,1);
YB = v(2:2:7500,1);

XqA = u2(1:2:2500,1);
XqB = u2(2:2:2500,1);

YqA = v(7501:2:10000,1);
YqB = v(7502:2:10000,1);

LV  = 1;   % number of latent variables
phi = .1;   % localization parameter
numIC = 1; % number of indepedence component
epsilon = 0.1;
max_itr = 1000;
b = 0.8; % Kernel parameter

%% --- output estimation ---
[YestA, W, T, P, q, omega, tq] =
ic_lw_kpls(XA,YA,XqA,LV,phi,numIC,epsilon,max_itr,b);
[YestB, W, T, P, q, omega, tq] =
ic_lw_kpls(XB,YB,XqB,LV,phi,numIC,epsilon,max_itr,b);

Yest = [YestA;YestB];
Yq = [YqA; YqB];

%% --- Results for testing data ---
format long
RMSE = sqrt(sum((Yq(:)-Yest(:)).^2)/numel(Yq(:)))

MAE = mean(abs(Yq(:)-Yest(:)))

MSE = mean((Yq(:)-Yest(:)).^2)

toc
t2 = toc

%% --- Data loading for testing data ---
load Data_ECCR.mat

tic
```

```
% Generating missing data
u(1:20:10000,1)= NaN; % Generating 5% of missing data

%% --- Parameters setting for expectation maximization
algorithm ---
X = u(1:7500,1); % set input variables as X
kt = 1; % Set one gaussian component

%% --- Initial guess for expectation maximization
algorithm ---
%--- Fill in the missing values by their mean value ---
% Calculate mean and then input data has missing values
that filles with its mean
[n,d] = size(X);
mis = zeros(n,d);
cmis = zeros(n,d);
m = zeros(1,d);
mt = zeros(n,d);
while sum(isnan(X))>0,
mis    = isnan(X);
X(mis) = 0; % let missing data equal to zeros
cmis   = ones(n,d) - mis; % complement of mis
m  = (sum(X)./(n - sum(mis))); % calculating mean value
from observed data
mt = repmat(m,[n,1]); % creating a matrix with repeated
means
X = cmis.*X + mt.*mis; % replacing missing data with
mean value
end

X = X';

%% --- Impute new value for missing data using
expectation maximization algorithm ---
% training dataset
[z1,model,llh] = mixGaussEm(X,kt);

mu = model.mu';
u1 = cmis.*X' + repmat(mu,n,1).*mis; % replacing missing
data with new value

u1 = u(:,1); % new input variables

%% --- parameter setting ---
XA = u1(1:2:7500,1);
XB = u1(2:2:7500,1);

YA = v(1:2:7500,1);
YB = v(2:2:7500,1);
%% --- output estimation ---
```

```
[YYA, W, T, P, q, omega, ~] =
ic_lw_kpls(XA,YA,XA,LV,phi,numIC,epsilon,max_itr,b);
[YYB, W, T, P, q, omega, tq] =
ic_lw_kpls(XB,YB,XB,LV,phi,numIC,epsilon,max_itr,b);

YY = [YYA;YYB];
Y = [YA; YB];

%% --- Results for training data ---
format long
RMSET = sqrt(sum((Y(:)-YY(:)).^2)/numel(Y(:)))

MAET = mean(abs(Y(:)-YY(:)))

MSET = mean((Y(:)-YY(:)).^2)

toc
t1 = toc

%% --- Results for testing and training data ---
Avr_Y = sum(Y(:))/numel(Y(:));
Avr_YY = sum(YY(:))/numel(YY(:))

Avr_Yq = sum(Yq(:))/numel(Yq(:));
Avr_Yest = sum(Yest(:))/numel(Yest(:))

Rsq_test = 1-sum((Yq(:) - Yest(:)).^2)/sum((Yest(:) -
mean(Yq(:))).^2)
Rsq_train = 1-sum((Y(:) - YY(:)).^2)/sum((YY(:) -
mean(Y(:))).^2)

N = size(Yq,1);
NN = size(Y,1);
E = (NN/(N+NN))*RMSET + (N/(N+NN))*RMSE + abs(RMSET-
RMSE)
```

Listing F.1.main2.m

Appendix G Results for the expectation maximization ensemble locally weighted independent component Kernel partial least square algorithm

Appendix G.1 Results for the expectation maximization ensemble locally weighted independent component Kernel partial least square (EM-E-LW-IC-KPLS) algorithm

Case studies 4, 5 and 6 with 5% to 60% of missing data were used to evaluate the predictive performance of the EM-E-LW-IC-KPLS algorithm. Likewise, integration of the singular value decomposition (SVD) with ensemble locally weighted independent component Kernel partial least square, the E-LW-IC-KPLS (SVD-E-LW-IC-KPLS) and locally weighted partial least square, the LW-PLS (SVD-LW-PLS) while the trimmed score regression (TSR) with the E-LW-IC-KPLS (TSR-E-LW-IC-KPLS) and LW-PLS (TSR-LW-PLS) algorithms are also used to solve the missing data problem. Besides, expectation maximization (EM) model was also integrated with LW-PLS (EM-LW-PLS). The results from the abovementioned algorithms are demonstrated in this section.

G.1.1 Case study 4: Numerical example 2

Tables G.1.1.1 to G.1.1.12 show the results from the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS models for case study 4. These algorithms were tested on process data with 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55% and 60% of missing data. In case study 4, the Kernel function used in the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS algorithms is also the Log Kernel, and the Kernel parameter *b* is also fixed at 2. The std(x) and std(xq) are the standard deviation for the training data and test data, respectively.

Table G.1.1.1 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

5%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.2442	0.2442	0.2553	0.2717	0.2717	0.2856
MAE_1	0.1969	0.1969	0.2050	0.2205	0.2205	0.2284
MSE_1	0.0596	0.0596	0.0652	0.0738	0.0738	0.0815
t_1 (sec)	54	54	54	9	9	9
std(x)	2.8129	2.8129	2.8617	2.8129	2.8129	2.8617
$RMSE_2$	0.2560	0.2560	0.2677	0.2734	0.2734	0.2845
MAE_2	0.2089	0.2089	0.2173	0.2243	0.2243	0.2301
MSE_2	0.0655	0.0655	0.0716	0.0747	0.0747	0.0810
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.7668	2.7668	2.8428	2.7668	2.7668	2.8428
E_a	0.2589	0.2589	0.2707	0.2738	0.2738	0.2863

Table G.1.1.2 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS and SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS, SVD-LW-PLS algorithms for case study 4

10%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.2621	0.2621	0.2778	0.2877	0.2877	0.3053
MAE_1	0.2061	0.2061	0.2174	0.2297	0.2297	0.2393
MSE_1	0.0687	0.0687	0.0772	0.0828	0.0828	0.0932
t_1 (sec)	54	54	54	9	9	9
std(x)	2.7380	2.7380	2.8323	2.7380	2.7380	2.8323
$RMSE_2$	0.2727	0.2727	0.2926	0.2886	0.2886	0.3025
MAE_2	0.2168	0.2168	0.2332	0.2324	0.2324	0.2396
MSE_2	0.0743	0.0743	0.0856	0.0833	0.0833	0.0915
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.6960	2.6960	2.8421	2.6960	2.6960	2.8421
E_a	0.2753	0.2753	0.2963	0.2888	0.2888	0.3075

Table G.1.1.3 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

15%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.2798	0.2798	0.2958	0.3026	0.3026	0.3213
MAE_1	0.2154	0.2154	0.2276	0.2381	0.2381	0.2486
MSE_1	0.0783	0.0783	0.0875	0.0916	0.0916	0.1032
t_1 (sec)	54	54	54	9	9	9
std(x)	2.6675	2.6675	2.8034	2.6675	2.6675	2.8034
$RMSE_2$	0.2893	0.2893	0.3201	0.3022	0.3022	0.3159
MAE_2	0.2272	0.2272	0.2538	0.2403	0.2403	0.2479
MSE_2	0.0837	0.0837	0.1024	0.0913	0.0913	0.0998
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.6327	2.6327	2.8490	2.6327	2.6327	2.8490
E_a	0.2916	0.2916	0.3261	0.3029	0.3029	0.3254

Table G.1.1.4 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

20%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.2986	0.2986	0.3131	0.3163	0.3163	0.3352
MAE_1	0.2266	0.2266	0.2380	0.2461	0.2461	0.2570
MSE_1	0.0892	0.0892	0.0980	0.1001	0.1001	0.1123
t_1 (sec)	54	54	54	9	9	9
std(x)	2.5814	2.5814	2.7564	2.5814	2.5814	2.7564
$RMSE_2$	0.3078	0.3078	0.3370	0.3168	0.3168	0.3313
MAE_2	0.2370	0.2370	0.2644	0.2483	0.2483	0.2562
MSE_2	0.0947	0.0947	0.1136	0.1004	0.1004	0.1098
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.5487	2.5487	2.8166	2.5487	2.5487	2.8166
E_a	0.3101	0.3101	0.3430	0.3169	0.3169	0.3380

Table G.1.1.5 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

25%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.3172	0.3172	0.3286	0.3301	0.3301	0.3483
MAE_1	0.2374	0.2374	0.2471	0.2539	0.2539	0.2650
MSE_1	0.1006	0.1006	0.1079	0.1089	0.1089	0.1213
t_1 (sec)	54	54	54	9	9	9
std(x)	2.4995	2.4995	2.7097	2.4995	2.4995	2.7097
$RMSE_2$	0.3297	0.3297	0.3569	0.3323	0.3323	0.3467
MAE_2	0.2514	0.2514	0.2768	0.2570	0.2570	0.2654
MSE_2	0.1087	0.1087	0.1274	0.1104	0.1104	0.1202
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.4710	2.4710	2.7910	2.4710	2.4710	2.7910
E_a	0.3328	0.3328	0.3640	0.3329	0.3329	0.3494

Table G.1.1.6 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

30%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.3287	0.3287	0.3429	0.3429	0.3429	0.3607
MAE_1	0.2442	0.2442	0.2564	0.2615	0.2615	0.2727
MSE_1	0.1080	0.1080	0.1176	0.1176	0.1176	0.1301
t_1 (sec)	54	54	54	9	9	9
std(x)	2.4290	2.4290	2.6686	2.4290	2.4290	2.6686
$RMSE_2$	0.3395	0.3395	0.3992	0.3429	0.3429	0.3558
MAE_2	0.2590	0.2590	0.3166	0.2640	0.2640	0.2724
MSE_2	0.1153	0.1153	0.1594	0.1176	0.1176	0.1266
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.4142	2.4142	2.8056	2.4142	2.4142	2.8056
E_a	0.3422	0.3422	0.4133	0.3429	0.3429	0.3643

Table G.1.1.7 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

35%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.3522	0.3522	0.3625	0.3569	0.3569	0.3732
MAE_1	0.2600	0.2600	0.2690	0.2702	0.2702	0.2809
MSE_1	0.1240	0.1240	0.1314	0.1274	0.1274	0.1393
t_1 (sec)	54	54	54	9	9	9
std(x)	2.3051	2.3051	2.5778	2.3051	2.3051	2.5778
$RMSE_2$	0.4653	0.4653	0.3849	0.3589	0.3589	0.3690
MAE_2	0.3487	0.3487	0.2963	0.2724	0.2724	0.2787
MSE_2	0.2165	0.2165	0.1482	0.1288	0.1288	0.1361
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.2169	2.2169	2.6699	2.2169	2.2169	2.6699
E_a	0.4935	0.4935	0.3905	0.3594	0.3594	0.3764

Table G.1.1.8 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

40%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.3687	0.3687	0.3799	0.3689	0.3689	0.3855
MAE_1	0.2717	0.2717	0.2812	0.2782	0.2782	0.2897
MSE_1	0.1360	0.1360	0.1443	0.1361	0.1361	0.1486
t_1 (sec)	54	54	54	9	9	9
std(x)	2.2316	2.2316	2.5139	2.2316	2.2316	2.5139
$RMSE_2$	0.4422	0.4422	0.5266	0.3690	0.3690	0.3795
MAE_2	0.3357	0.3357	0.4181	0.2794	0.2794	0.2868
MSE_2	0.1955	0.1955	0.2773	0.1362	0.1362	0.1440
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.1935	2.1935	2.7153	2.1935	2.1935	2.7153
E_a	0.4605	0.4605	0.5633	0.3691	0.3691	0.3901

Table G.1.1.9 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

45%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.3754	0.3754	0.3854	0.3800	0.3800	0.3950
MAE_1	0.2772	0.2772	0.2868	0.2864	0.2864	0.2973
MSE_1	0.1409	0.1409	0.1486	0.1444	0.1444	0.1560
t_1 (sec)	54	54	54	9	9	9
std(x)	2.1645	2.1645	2.4735	2.1645	2.1645	2.4735
$RMSE_2$	0.3909	0.3909	0.6367	0.3788	0.3788	0.3894
MAE_2	0.2983	0.2983	0.4889	0.2866	0.2866	0.2946
MSE_2	0.1528	0.1528	0.4053	0.1435	0.1435	0.1516
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.1738	2.1738	2.7017	2.1738	2.1738	2.7017
E_a	0.3948	0.3948	0.6995	0.3809	0.3809	0.3992

Table G.1.1.10 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

50%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.3890	0.3890	0.3975	0.3915	0.3915	0.4051
MAE_1	0.2868	0.2868	0.2955	0.2944	0.2944	0.3050
MSE_1	0.1513	0.1513	0.1580	0.1533	0.1533	0.1641
t_1 (sec)	54	54	54	9	9	9
std(x)	2.0408	2.0408	2.3717	2.0408	2.0408	2.3717
$RMSE_2$	0.4022	0.4022	0.4388	0.3929	0.3929	0.4036
MAE_2	0.3050	0.3050	0.3458	0.2958	0.2958	0.3042
MSE_2	0.1617	0.1617	0.1925	0.1544	0.1544	0.1629
t_2 (sec)	14	14	14	4	4	4
std(xq)	2.0306	2.0306	2.5236	2.0306	2.0306	2.5236
E_a	0.4055	0.4055	0.4491	0.3933	0.3933	0.4062

Table G.1.1.11 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

55%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.4063	0.4063	0.4127	0.4056	0.4056	0.4187
MAE_1	0.3013	0.3013	0.3080	0.3051	0.3051	0.3157
MSE_1	0.1650	0.1650	0.1703	0.1646	0.1646	0.1753
t_1 (sec)	54	54	54	9	9	9
std(x)	1.9617	1.9617	2.2785	1.9617	1.9617	2.2785
$RMSE_2$	0.4640	0.4640	0.4204	0.4072	0.4072	0.4155
MAE_2	0.3711	0.3711	0.3197	0.3069	0.3069	0.3122
MSE_2	0.2153	0.2153	0.1767	0.1658	0.1658	0.1727
t_2 (sec)	14	14	14	4	4	4
std(xq)	1.8446	1.8446	2.2817	1.8446	1.8446	2.2817
E_a	0.4784	0.4784	0.4223	0.4076	0.4076	0.4211

Table G.1.1.12 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 4

60%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.4154	0.4154	0.4188	0.4147	0.4147	0.4238
MAE_1	0.3078	0.3078	0.3134	0.3114	0.3114	0.3200
MSE_1	0.1725	0.1725	0.1754	0.1720	0.1720	0.1796
t_1 (sec)	54	54	54	9	9	9
std(x)	1.8259	1.8259	2.1980	1.8259	1.8259	2.1980
$RMSE_2$	0.4197	0.4197	0.4232	0.4168	0.4168	0.4261
MAE_2	0.3137	0.3137	0.3196	0.3138	0.3138	0.3228
MSE_2	0.1762	0.1762	0.1791	0.1738	0.1738	0.1816
t_2 (sec)	14	14	14	4	4	4
std(xq)	1.8288	1.8288	2.2018	1.8288	1.8288	2.2018
E_a	0.4208	0.4208	0.4243	0.4174	0.4174	0.4267

G.1.2 Case study 5: Eukaryotic cell cycle regulation

The results from the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS models for case study 5 are illustrated in Tables G.1.2.1 to G.1.2.12. The process data with 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55% and 60% of missing data were used to examine the performance of these algorithms.The Log Kernel was used as the Kernel function in the E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS algorithms and the selected Kernel parameter, *b* was 0.8. The std(x) and std(xq) were the standard deviations for training data and test data respectively.

Table G.1.2.1 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

5%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0692	0.0692	0.0693	0.0761	0.0761	0.0762
MAE_1	0.0554	0.0554	0.0552	0.0652	0.0652	0.0650
MSE_1	0.0048	0.0048	0.0048	0.0058	0.0058	0.0058
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0665	0.0665	0.0667	0.0665	0.0665	0.0667
$RMSE_2$	0.0716	0.0716	0.0717	0.0755	0.0755	0.0755
MAE_2	0.0571	0.0571	0.0569	0.0650	0.0650	0.0649
MSE_2	0.0051	0.0051	0.0051	0.0057	0.0057	0.0057
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0666	0.0666	0.0669	0.0666	0.0666	0.0669
E_a	0.0722	0.0722	0.0723	0.0766	0.0766	0.0767

Table G.1.2.2 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

10%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0698	0.0698	0.0700	0.0768	0.0768	0.0769
MAE_1	0.0561	0.0561	0.0563	0.0660	0.0660	0.0659
MSE_1	0.0049	0.0049	0.0049	0.0059	0.0059	0.0059
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0647	0.0647	0.0653	0.0647	0.0647	0.0653
$RMSE_2$	0.0721	0.0721	0.0752	0.0760	0.0760	0.0761
MAE_2	0.0578	0.0578	0.0600	0.0657	0.0657	0.0656
MSE_2	0.0052	0.0052	0.0057	0.0058	0.0058	0.0058
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0648	0.0648	0.0654	0.0648	0.0648	0.0654
E_a	0.0727	0.0727	0.0765	0.0774	0.0774	0.0774

Table G.1.2.3 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

15%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0708	0.0708	0.0712	0.0773	0.0773	0.0776
MAE_1	0.0571	0.0571	0.0573	0.0666	0.0666	0.0666
MSE_1	0.0050	0.0050	0.0051	0.0060	0.0060	0.0060
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0627	0.0627	0.0636	0.0627	0.0627	0.0636
$RMSE_2$	0.0739	0.0739	0.0743	0.0765	0.0765	0.0766
MAE_2	0.0598	0.0598	0.0602	0.0662	0.0662	0.0664
MSE_2	0.0055	0.0055	0.0055	0.0058	0.0058	0.0059
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0631	0.0631	0.0640	0.0631	0.0631	0.0640
E_a	0.0747	0.0747	0.0750	0.0780	0.0780	0.0782

Table G.1.2.4 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

20%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0714	0.0714	0.0716	0.0779	0.0779	0.0781
MAE_1	0.0581	0.0581	0.0580	0.0672	0.0672	0.0672
MSE_1	0.0051	0.0051	0.0051	0.0061	0.0061	0.0061
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0609	0.0609	0.0620	0.0609	0.0609	0.0620
$RMSE_2$	0.0735	0.0735	0.0788	0.0770	0.0770	0.0773
MAE_2	0.0597	0.0597	0.0640	0.0669	0.0669	0.0671
MSE_2	0.0054	0.0054	0.0062	0.0059	0.0059	0.0060
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0612	0.0612	0.0623	0.0612	0.0612	0.0623
E_a	0.0741	0.0741	0.0806	0.0786	0.0786	0.0786

Table G.1.2.5 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

25%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0743	0.0743	0.0744	0.0784	0.0784	0.0786
MAE_1	0.0619	0.0619	0.0616	0.0678	0.0678	0.0678
MSE_1	0.0055	0.0055	0.0055	0.0062	0.0062	0.0062
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0590	0.0590	0.0603	0.0590	0.0590	0.0603
$RMSE_2$	0.0777	0.0777	0.0781	0.0778	0.0778	0.0780
MAE_2	0.0650	0.0650	0.0654	0.0676	0.0676	0.0679
MSE_2	0.0060	0.0060	0.0061	0.0061	0.0061	0.0061
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0592	0.0592	0.0606	0.0592	0.0592	0.0606
E_a	0.0786	0.0786	0.0790	0.0789	0.0789	0.0791

Table G.1.2.6 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

30%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0733	0.0733	0.0736	0.0788	0.0788	0.0792
MAE_1	0.0603	0.0603	0.0604	0.0681	0.0681	0.0685
MSE_1	0.0054	0.0054	0.0054	0.0062	0.0062	0.0063
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0571	0.0571	0.0586	0.0571	0.0571	0.0586
$RMSE_2$	0.0757	0.0757	0.0762	0.0781	0.0781	0.0786
MAE_2	0.0623	0.0623	0.0626	0.0680	0.0680	0.0685
MSE_2	0.0057	0.0057	0.0058	0.0061	0.0061	0.0062
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0572	0.0572	0.0588	0.0572	0.0572	0.0588
E_a	0.0763	0.0763	0.0768	0.0792	0.0792	0.0796

Table G.1.2.7 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

35%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0739	0.0739	0.0744	0.0796	0.0796	0.0798
MAE_1	0.0608	0.0608	0.0614	0.0689	0.0689	0.0691
MSE_1	0.0055	0.0055	0.0055	0.0063	0.0063	0.0064
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0549	0.0549	0.0566	0.0549	0.0549	0.0566
$RMSE_2$	0.0760	0.0760	0.0768	0.0786	0.0786	0.0790
MAE_2	0.0626	0.0626	0.0635	0.0686	0.0686	0.0691
MSE_2	0.0058	0.0058	0.0059	0.0062	0.0062	0.0062
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0550	0.0550	0.0568	0.0550	0.0550	0.0568
E_a	0.0765	0.0765	0.0774	0.0803	0.0803	0.0805

Table G.1.2.8 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

40%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0747	0.0747	0.0752	0.0799	0.0799	0.0802
MAE_1	0.0618	0.0618	0.0622	0.0693	0.0693	0.0696
MSE_1	0.0056	0.0056	0.0057	0.0064	0.0064	0.0064
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0528	0.0528	0.0547	0.0528	0.0528	0.0547
$RMSE_2$	0.0768	0.0768	0.0773	0.0790	0.0790	0.0795
MAE_2	0.0636	0.0636	0.0641	0.0690	0.0690	0.0696
MSE_2	0.0059	0.0059	0.0060	0.0062	0.0062	0.0063
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0529	0.0529	0.0548	0.0529	0.0529	0.0548
E_a	0.0774	0.0774	0.0778	0.0806	0.0806	0.0807

Table G.1.2.9 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

45%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0757	0.0757	0.0758	0.0803	0.0803	0.0806
MAE_1	0.0630	0.0630	0.0630	0.0698	0.0698	0.0701
MSE_1	0.0057	0.0057	0.0057	0.0065	0.0065	0.0065
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0506	0.0506	0.0526	0.0506	0.0506	0.0526
$RMSE_2$	0.0780	0.0780	0.0783	0.0796	0.0796	0.0800
MAE_2	0.0650	0.0650	0.0654	0.0696	0.0696	0.0702
MSE_2	0.0061	0.0061	0.0061	0.0063	0.0063	0.0064
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0508	0.0508	0.0528	0.0508	0.0508	0.0528
E_a	0.0786	0.0786	0.0789	0.0809	0.0809	0.0811

Table G.1.2.10 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

50%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0764	0.0764	0.0764	0.0807	0.0807	0.0810
MAE_1	0.0639	0.0639	0.0638	0.0702	0.0702	0.0706
MSE_1	0.0058	0.0058	0.0058	0.0065	0.0065	0.0066
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0482	0.0482	0.0503	0.0482	0.0482	0.0503
$RMSE_2$	0.0783	0.0783	0.0787	0.0801	0.0801	0.0805
MAE_2	0.0656	0.0656	0.0662	0.0701	0.0701	0.0707
MSE_2	0.0061	0.0061	0.0062	0.0064	0.0064	0.0065
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0484	0.0484	0.0506	0.0484	0.0484	0.0506
E_a	0.0788	0.0788	0.0793	0.0812	0.0812	0.0814

Table G.1.2.11 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

55%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0770	0.0770	0.0770	0.0812	0.0812	0.0814
MAE_1	0.0648	0.0648	0.0647	0.0708	0.0708	0.0710
MSE_1	0.0059	0.0059	0.0059	0.0066	0.0066	0.0066
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0458	0.0458	0.0480	0.0458	0.0458	0.0480
$RMSE_2$	0.0792	0.0792	0.0796	0.0805	0.0805	0.0810
MAE_2	0.0667	0.0667	0.0672	0.0706	0.0706	0.0713
MSE_2	0.0063	0.0063	0.0063	0.0065	0.0065	0.0066
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0460	0.0460	0.0483	0.0460	0.0460	0.0483
E_a	0.0798	0.0798	0.0803	0.0816	0.0816	0.0817

Table G.1.2.12 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 5

60%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**13 sets**	**13 sets**	**13 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0776	0.0776	0.0777	0.0814	0.0814	0.0818
MAE_1	0.0656	0.0656	0.0656	0.0711	0.0711	0.0715
MSE_1	0.0060	0.0060	0.0060	0.0066	0.0066	0.0067
t_1 (sec)	54	54	54	11	11	11
std(x)	0.0432	0.0432	0.0454	0.0432	0.0432	0.0454
$RMSE_2$	0.0804	0.0804	0.0806	0.0809	0.0809	0.0814
MAE_2	0.0683	0.0683	0.0683	0.0711	0.0711	0.0717
MSE_2	0.0065	0.0065	0.0065	0.0066	0.0066	0.0066
t_2 (sec)	14	14	14	4	4	4
std(xq)	0.0435	0.0435	0.0459	0.0435	0.0435	0.0459
E_a	0.0811	0.0811	0.0813	0.0818	0.0818	0.0820

G.1.3 Case study 6: A highly nonlinear CSTR

Tables G.1.3.1 to G.1.3.12 present the results from the EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS and SVD-E-LW-IC-KPLS algorithms for case study 6. 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55% and 60% of missing data were included in the process data utilised to investigate the performance of the EM-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS, and SVD-LW-PLS algorithms. The Log Kernel was used as the Kernel function in E-LW-IC-KPLS, LW-IC-KPLS and LW-KPLS models and the chosen Kernel parameter, *b* was 9. The std(x) and std(xq) were the standard deviations for training data and test data, respectively.

Table G.1.3.1 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
5%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0124	0.0126	0.0128	0.0155	0.0155	0.0155
MAE_1	0.0090	0.0091	0.0092	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0102	0.0104	0.0115	0.0074	0.0074	0.0074
MAE_2	0.0079	0.0081	0.0087	0.0061	0.0061	0.0061
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0141	0.0143	0.0137	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
5%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0145	0.0145	0.0145	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006
MSE_1	0.0002	$2.1x10^{-4}$	$2.1x10^{-4}$	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	9.0486	9.0486	9.2941	9.0486	9.0486	9.2941
std(x2)	2552.7293	2552.7293	2601.8756	2552.7293	2552.7293	2601.8756
$RMSE_2$	0.0039	0.0039	0.0039	0.0006	0.0006	0.0005
MAE_2	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
MSE_2	$1.5x10^{-5}$	$1.5x10^{-5}$	$1.5x10^{-5}$	$3.8x10^{-7}$	$3.8x10^{-7}$	$2.8x10^{-7}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	8.9891	8.9891	9.2369	8.9891	8.9891	9.2369
std(xq2)	2559.3288	2559.3288	2604.8644	2559.3288	2559.3288	2604.8644
Ea_2 (2nd output)	0.0224	0.0224	0.0224	0.0261	0.0261	0.0262

Table G.1.3.2 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
10%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0129	0.0131	0.0129	0.0155	0.0155	0.0155
MAE_1	0.0093	0.0094	0.0093	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0105	0.0105	0.0101	0.0075	0.0075	0.0075
MAE_2	0.0081	0.0081	0.0080	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0147	0.0149	0.0150	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
10%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0147	0.0147	0.0150	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0004	0.0006	0.0006	0.0006
MSE_1	0.0002	$2.2x10^{-4}$	$2.2x10^{-4}$	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	8.8159	8.8159	9.2587	8.8159	8.8159	9.2587
std(x2)	2490.8909	2490.8909	2579.4034	2490.8909	2490.8909	2579.4034
$RMSE_2$	0.0029	0.0029	0.0003	0.0005	0.0005	0.0005
MAE_2	0.0004	0.0004	0.0002	0.0003	0.0003	0.0003
MSE_2	$8.6x10^{-6}$	$8.6x10^{-6}$	$8.2x10^{-8}$	$2.7x10^{-7}$	$2.7x10^{-7}$	$2.2x10^{-7}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	8.7586	8.7586	9.2246	8.7586	8.7586	9.2246
std(xq2)	2491.1706	2491.1706	2579.8916	2491.1706	2491.1706	2579.8916
Ea_2 (2nd output)	0.0235	0.0235	0.0260	0.0262	0.0262	0.0263

Table G.1.3.3 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
15%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0130	0.0130	0.0129	0.0155	0.0155	0.0155
MAE_1	0.0093	0.0094	0.0093	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0105	0.0107	0.0099	0.0075	0.0075	0.0075
MAE_2	0.0081	0.0082	0.0078	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0149	0.0148	0.0152	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
15%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0149	0.0149	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006
MSE_1	0.0002	$2.2\text{x}10^{-4}$	0.0002	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	8.6012	8.6012	9.2463	8.6012	8.6012	9.2463
std(x2)	2421.9051	2421.9051	2551.4447	2421.9051	2421.9051	2551.4447
$RMSE_2$	0.0014	0.0014	0.0014	0.0004	0.0004	0.0004
MAE_2	0.0004	0.0004	0.0004	0.0003	0.0003	0.0003
MSE_2	$1.9x10^{-6}$	$1.9x10^{-6}$	$1.9x10^{-6}$	$2x10^{-7}$	$2x10^{-7}$	$1.7x10^{-7}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	8.5210	8.5210	9.1900	8.5210	8.5210	9.1900
std(xq2)	2415.4177	2415.4177	2543.8760	2415.4177	2415.4177	2543.8760
Ea_2 (2nd output)	0.0251	0.0251	0.0251	0.0263	0.0263	0.0263

Table G.1.3.4 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
20%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0135	0.0134	0.0132	0.0155	0.0155	0.0155
MAE_1	0.0096	0.0095	0.0094	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0106	0.0106	0.0094	0.0075	0.0075	0.0075
MAE_2	0.0081	0.0081	0.0074	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0157	0.0154	0.0161	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
20%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0151	0.0151	0.0150	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	8.3152	8.3152	9.1472	8.3152	8.3152	9.1472
std(x2)	2343.8782	2343.8782	2511.3571	2343.8782	2343.8782	2511.3571
$RMSE_2$	0.0003	0.0003	0.0003	0.0004	0.0004	0.0004
MAE_2	0.0002	0.0002	0.0002	0.0003	0.0003	0.0003
MSE_2	$1.2x10^{-7}$	$1.2x10^{-7}$	$7x10^{-8}$	$1.6x10^{-7}$	$1.6x10^{-7}$	$1.4x10^{-7}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	8.2505	8.2505	9.1085	8.2505	8.2505	9.1085
std(xq2)	2361.0369	2361.0369	2525.4588	2361.0369	2361.0369	2525.4588
Ea_2 (2nd output)	0.0261	0.0261	0.0260	0.0263	0.0263	0.0263

Table G.1.3.5 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
25%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0136	0.0138	0.0133	0.0155	0.0155	0.0155
MAE_1	0.0097	0.0097	0.0094	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0109	0.0109	0.0104	0.0074	0.0074	0.0074
MAE_2	0.0082	0.0082	0.0078	0.0062	0.0062	0.0061
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0156	0.0160	0.0155	0.0216	0.0216	0.0216

Output variables: Reactor temperature						
25%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0151	0.0149	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	8.0308	8.0308	9.0227	8.0308	8.0308	9.0227
std(x2)	2276.5987	2276.5987	2474.7320	2276.5987	2276.5987	2474.7320
$RMSE_2$	0.0020	0.0020	0.0019	0.0004	0.0004	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0003	0.0003	0.0003
MSE_2	$3.8x10^{-6}$	$3.8x10^{-6}$	$3.5x10^{-6}$	$1.4x10^{-7}$	$1.4x10^{-7}$	$1.2x10^{-7}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	8.0114	8.0114	9.0715	8.0114	8.0114	9.0715
std(xq2)	2286.7744	2286.7744	2479.5615	2286.7744	2286.7744	2479.5615
Ea_2 (2nd output)	0.0249	0.0245	0.0246	0.0263	0.0263	0.0263

Table G.1.3.6 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
30%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0135	0.0134	0.0131	0.0155	0.0155	0.0155
MAE_1	0.0096	0.0095	0.0094	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0097	0.0098	0.0098	0.0075	0.0075	0.0075
MAE_2	0.0076	0.0078	0.0078	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0164	0.0160	0.0156	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
30%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0149	0.0149	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006

MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
t_1 (sec)	110	110	110	20	20	20
std(x1)	7.8007	7.8007	8.9329	7.8007	7.8007	8.9329
std(x2)	2193.2745	2193.2745	2423.8835	2193.2745	2193.2745	2423.8835
$RMSE_2$	0.0010	0.0010	0.0010	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	$3.8\text{x}10^{-4}$	0.0003	0.0003	0.0003
MSE_2	$9.9\text{x}10^{-7}$	$9.9\text{x}10^{-7}$	$9.9\text{x}10^{-7}$	$1.2\text{x}10^{-7}$	$1.2\text{x}10^{-7}$	$1.1\text{x}10^{-7}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	7.7040	7.7040	8.8910	7.7040	7.7040	8.8910
std(xq2)	2195.6077	2195.6077	2422.5456	2195.6077	2195.6077	2422.5456
Ea_2 (2nd output)	0.0254	0.0254	0.0254	0.0263	0.0263	0.0264

Table G.1.3.7 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
35%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0136	0.0136	0.0137	0.0155	0.0155	0.0155
MAE_1	0.0098	0.0098	0.0098	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0090	0.0090	0.0093	0.0075	0.0075	0.0075
MAE_2	0.0072	0.0072	0.0072	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0171	0.0171	0.0171	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
35%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0149	0.0149	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006

MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
t_1 (sec)	110	110	110	20	20	20
std(x1)	7.5428	7.5428	8.8019	7.5428	7.5428	8.8019
std(x2)	2115.7468	2115.7468	2376.4203	2115.7468	2115.7468	2376.4203
$RMSE_2$	0.0015	0.0015	0.0015	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0003	0.0003	0.0002
MSE_2	$2.3x10^{-6}$	$2.3x10^{-6}$	$2.3x10^{-6}$	$1x10^{-7}$	$1x10^{-7}$	$9.5x10^{-8}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	7.4590	7.4590	8.7843	7.4590	7.4590	8.7843
std(xq2)	2109.0245	2109.0245	2360.3395	2109.0245	2109.0245	2360.3395
Ea_2 (2nd output)	0.0249	0.0249	0.0250	0.0264	0.0264	0.0264

Table G.1.3.8 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
40%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0137	0.0138	0.0139	0.0155	0.0155	0.0155
MAE_1	0.0099	0.0099	0.0100	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0090	0.0090	0.0088	0.0074	0.0074	0.0074
MAE_2	0.0070	0.0070	0.0069	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0173	0.0173	0.0177	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
40%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0149	0.0149	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	7.2561	7.2561	8.6186	7.2561	7.2561	8.6186
std(x2)	2030.4564	2030.4564	2314.1938	2030.4564	2030.4564	2314.1938
$RMSE_2$	0.0014	0.0014	0.0014	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0003	0.0003	0.0002
MSE_2	$2.1x10^{-6}$	$2.1x10^{-6}$	$2x10^{-6}$	$9.9x10^{-8}$	$9.9x10^{-8}$	$9.1x10^{-8}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	7.1622	7.1622	8.6005	7.1622	7.1622	8.6005
std(xq2)	2040.3829	2040.3829	2315.6844	2040.3829	2040.3829	2315.6844
Ea_2 (2nd output)	0.0250	0.0250	0.0250	0.0264	0.0264	0.0264

Table G.1.3.9 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
45%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0137	0.0139	0.0137	0.0155	0.0155	0.0155
MAE_1	0.0098	0.0099	0.0097	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0107	0.0107	0.0095	0.0075	0.0075	0.0075
MAE_2	0.0079	0.0079	0.0073	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0160	0.0162	0.0168	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
45%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0149	0.0149	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006

MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
t_1 (sec)	110	110	110	20	20	20
std(x1)	6.9304	6.9304	8.3860	6.9304	6.9304	8.3860
std(x2)	1942.6921	1942.6921	2246.7901	1942.6921	1942.6921	2246.7901
$RMSE_2$	0.0014	0.0014	0.0014	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0003	0.0003	0.0002
MSE_2	$1.8x10^{-6}$	$1.8x10^{-6}$	$1.8x10^{-6}$	$1x10^{-7}$	$1x10^{-7}$	$9.3x10^{-8}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	6.8925	6.8925	8.4512	6.8925	6.8925	8.4512
std(xq2)	1949.6352	1949.6352	2241.5826	1949.6352	1949.6352	2241.5826
Ea_2 (2nd output)	0.0251	0.0251	0.0251	0.0264	0.0264	0.0264

Table G.1.3.10 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
50%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0138	0.0146	0.0150	0.0155	0.0155	0.0155
MAE_1	0.0097	0.0103	0.0105	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0096	0.0088	0.0098	0.0075	0.0075	0.0075
MAE_2	0.0073	0.0069	0.0072	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0169	0.0189	0.0190	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
50%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0152	0.0152	0.0149	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002

t_1 (sec)	110	110	110	20	20	20
std(x1)	6.6211	6.6211	8.1466	6.6211	6.6211	8.1466
std(x2)	1859.7109	1859.7109	2177.7359	1859.7109	1859.7109	2177.7359
$RMSE_2$	0.0013	0.0013	0.0013	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0002	0.0002	0.0002
MSE_2	$1.6x10^{-6}$	$1.6x10^{-6}$	$1.6x10^{-6}$	$9.8x10^{-8}$	$9.8x10^{-8}$	$9x10^{-8}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	6.6659	6.6659	8.2903	6.6659	6.6659	8.2903
std(xq2)	1889.8303	1889.8303	2202.9966	1889.8303	1889.8303	2202.9966
Ea_2 (2nd output)	0.0256	0.0256	0.0252	0.0264	0.0264	0.0264

Table G.1.3.11 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
55%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0138	0.0138	0.0139	0.0155	0.0155	0.0155
MAE_1	0.0098	0.0098	0.0099	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0083	0.0083	0.0084	0.0075	0.0075	0.0075
MAE_2	0.0066	0.0066	0.0067	0.0062	0.0062	0.0062
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0179	0.0179	0.0181	0.0215	0.0215	0.0215
Output variables: Reactor temperature						
55%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0151	0.0151	0.0151	0.0152	0.0152	0.0152
MAE_1	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006

MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
t_1 (sec)	110	110	110	20	20	20
std(x1)	6.2426	6.2426	7.8309	6.2426	6.2426	7.8309
std(x2)	1764.6585	1764.6585	2091.8953	1764.6585	1764.6585	2091.8953
$RMSE_2$	0.0013	0.0013	0.0012	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0002	0.0002	0.0002
MSE_2	$1.7x10^{-6}$	$1.7x10^{-6}$	$1.6x10^{-6}$	$9.5x10^{-8}$	$9.5x10^{-8}$	$8.7x10^{-8}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	6.1962	6.1962	7.8258	6.1962	6.1962	7.8258
std(xq2)	1758.9717	1758.9717	2085.8460	1758.9717	1758.9717	2085.8460
Ea_2 (2nd output)	0.0255	0.0255	0.0254	0.0264	0.0264	0.0264

Table G.1.3.12 Results for EM-E-LW-IC-KPLS, TSR-E-LW-IC-KPLS, SVD-E-LW-IC-KPLS, EM-LW-PLS, TSR-LW-PLS and SVD-LW-PLS algorithms for case study 6

Output variable: Production concentration						
60%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0137	0.0137	0.0138	0.0155	0.0155	0.0155
MAE_1	0.0098	0.0098	0.0099	0.0111	0.0111	0.0111
MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
$RMSE_2$	0.0084	0.0084	0.0084	0.0074	0.0074	0.0074
MAE_2	0.0067	0.0067	0.0068	0.0061	0.0061	0.0061
MSE_2	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Ea_1 (1st output)	0.0177	0.0177	0.0179	0.0216	0.0216	0.0216
Output variables: Reactor temperature						
60%	**EM**	**TSR**	**SVD**	**EM**	**TSR**	**SVD**
No. of set	**27 sets**	**27 sets**	**27 sets**	**1 set**	**1 set**	**1 set**
Models	EM-E-LW-IC-KPLS	TSR-E-LW-IC-KPLS	SVD-E-LW-IC-KPLS	EM-LW-PLS	TSR-LW-PLS	SVD-LW-PLS
$RMSE_1$	0.0151	0.0151	0.0150	0.0152	0.0152	0.0152
MAE_1	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006

MSE_1	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
t_1 (sec)	110	110	110	20	20	20
std(x1)	5.8799	5.8799	7.4917	5.8799	5.8799	7.4917
std(x2)	1666.7046	1666.7046	1998.7383	1666.7046	1666.7046	1998.7383
$RMSE_2$	0.0013	0.0013	0.0012	0.0003	0.0003	0.0003
MAE_2	0.0004	0.0004	0.0004	0.0002	0.0002	0.0002
MSE_2	$1.8\text{x}10^{-6}$	$1.8\text{x}10^{-6}$	$1.4\text{x}10^{-6}$	$1\text{x}10^{-7}$	$1\text{x}10^{-7}$	$9.3\text{x}10^{-8}$
t_2 (sec)	27	27	27	7	7	7
std(xq1)	5.8541	5.8541	7.5579	5.8541	5.8541	7.5579
std(xq2)	1654.2430	1654.2430	1996.0462	1654.2430	1654.2430	1996.0462
Ea_2 (2nd output)	0.0255	0.0255	0.0253	0.0264	0.0264	0.0264

Appendix H Figures

This appendix provides figures used for case studies 1 to 6 in Chapter 6. Some of these figures are either adopted from journal papers or generated from MATLAB software or drawn from Microsoft Excel. All these figures are labeled from Figures H.1 to H.28.

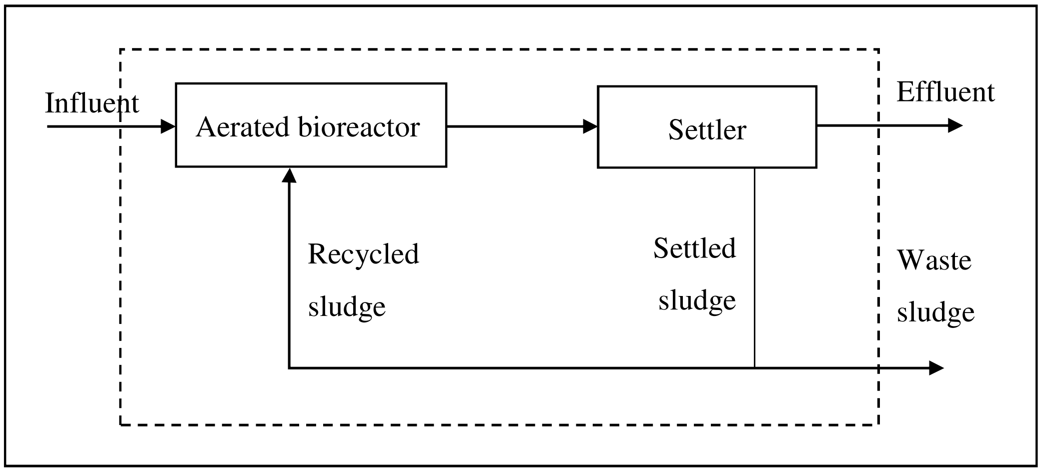

Figure H.1 Schematic of wastewater treatment process (Caraman *et al.* 2007)

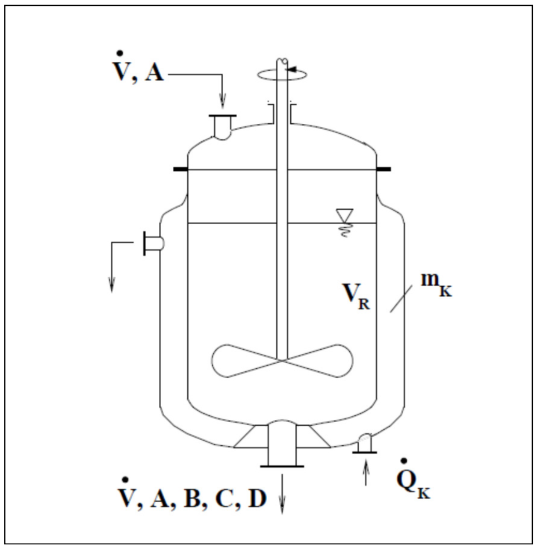

Figure H.2 Schematic diagram of the CSTR (Chen *et al.* 1995)

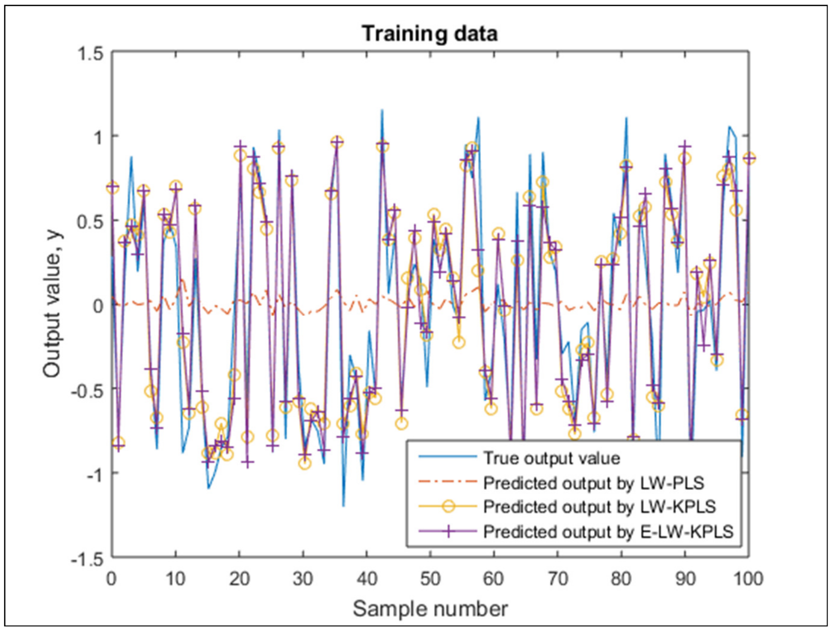

Figure H.3 Prediction results for training data in case study 1 using LW-PLS, LW-KPLS and E-LW-KPLS algorithms

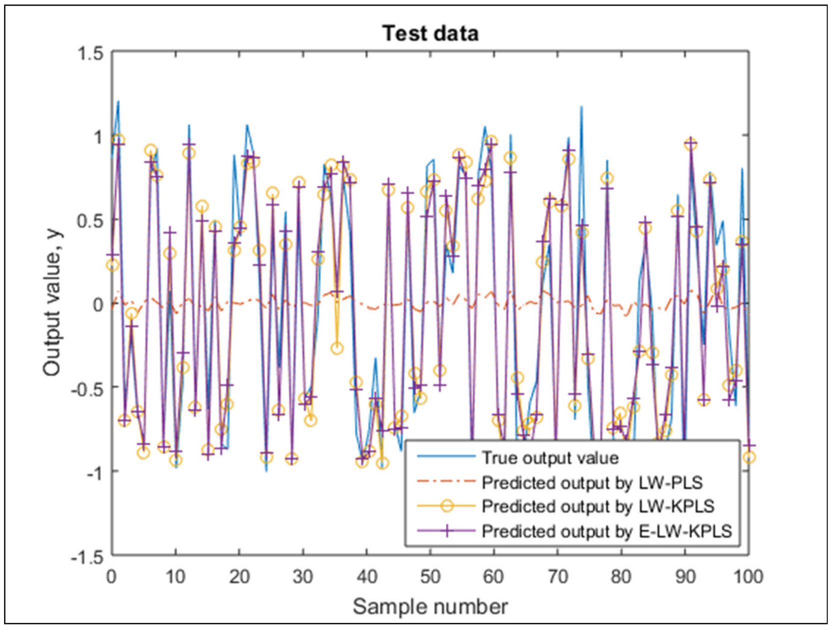

Figure H.4 Prediction results for test data in case study 1 using the LW-PLS, LW-KPLS and E-LW-KPLS algorithms

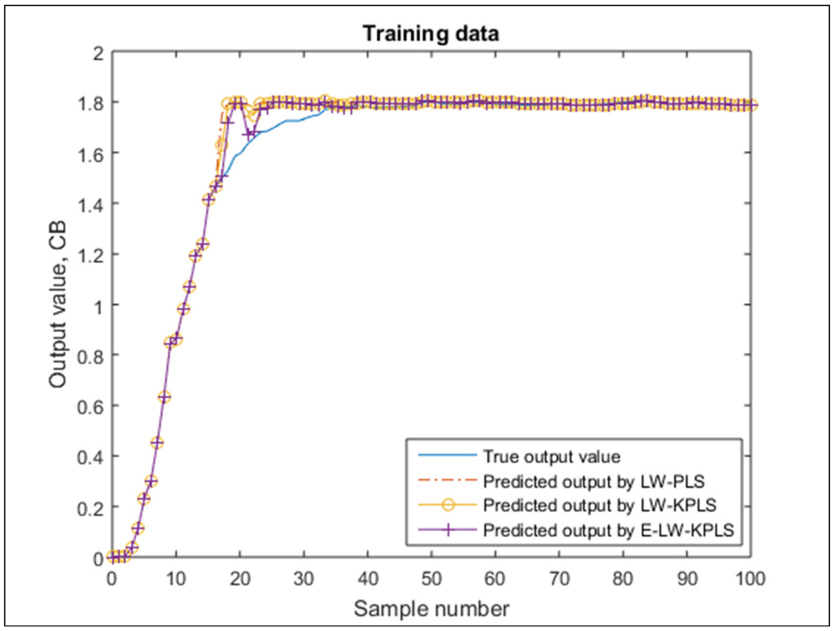

Figure H.5 Prediction results for training data of the case study 2 using LW-PLS, LW-KPLS and E-LW-KPLS models

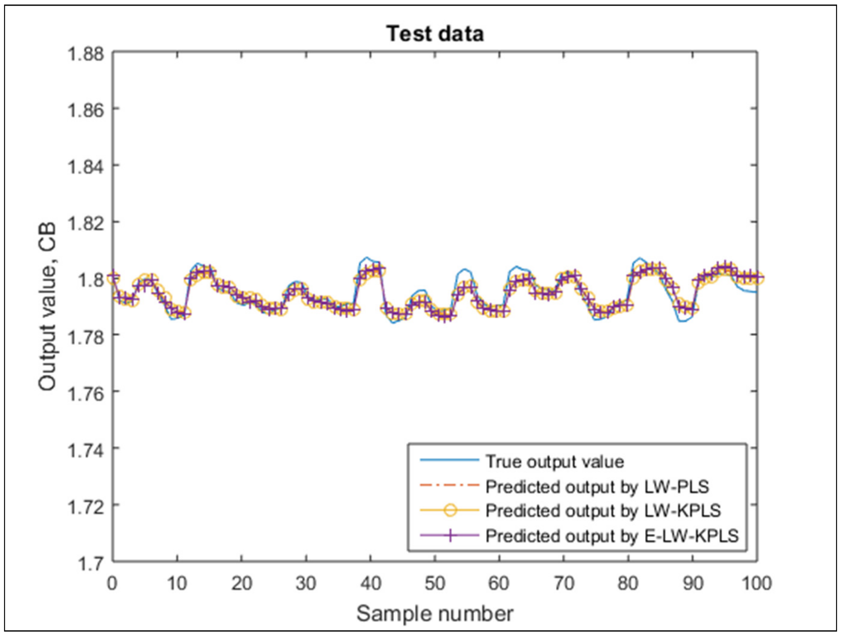

Figure H.6 Prediction results for test data of the case study 2 using LW-PLS, LW-KPLS and E-LW-KPLS models

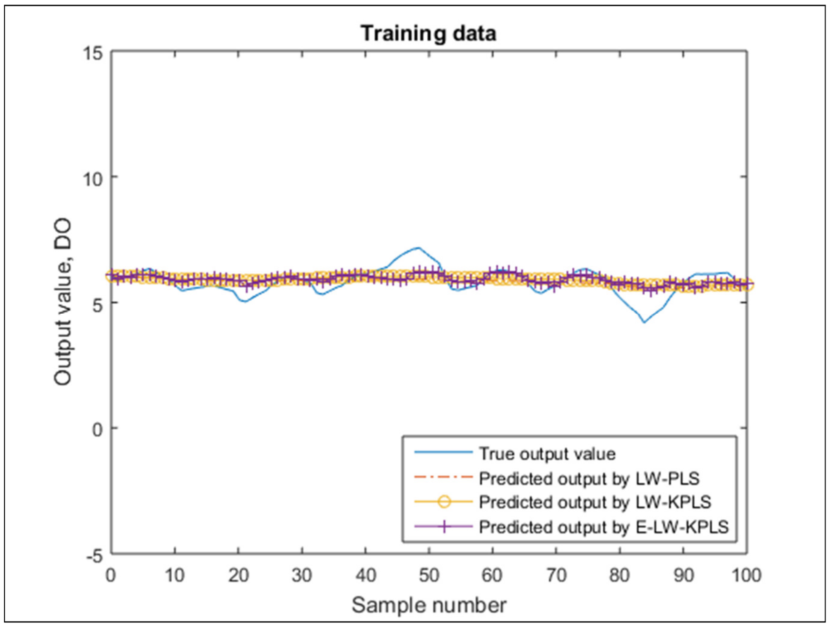

Figure H.7 Prediction results for training data of the case study 3 using the LW-PLS, LW-KPLS and E-LW-KPLS algorithms

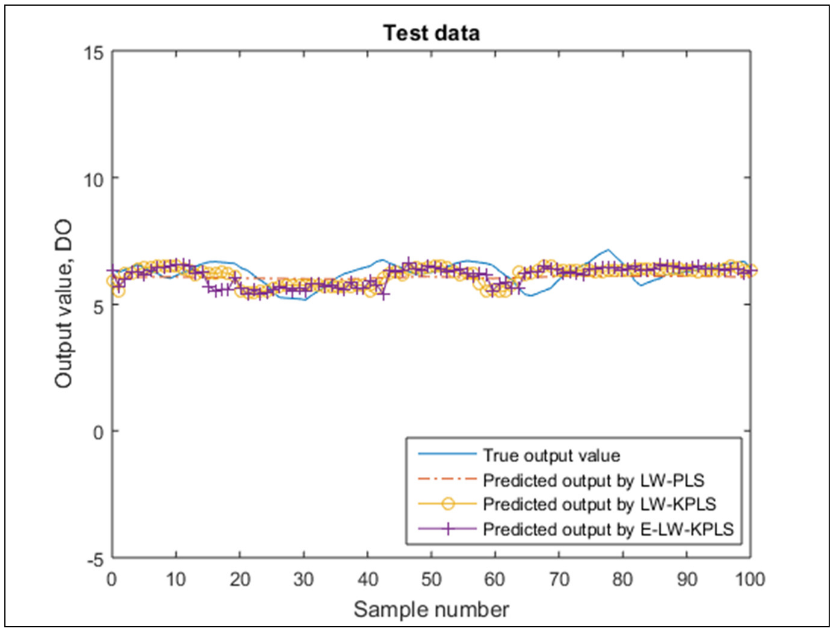

Figure H.8 Prediction results for test data of case study 3 using the LW-PLS, LW-KPLS and E-LW-KPLS models

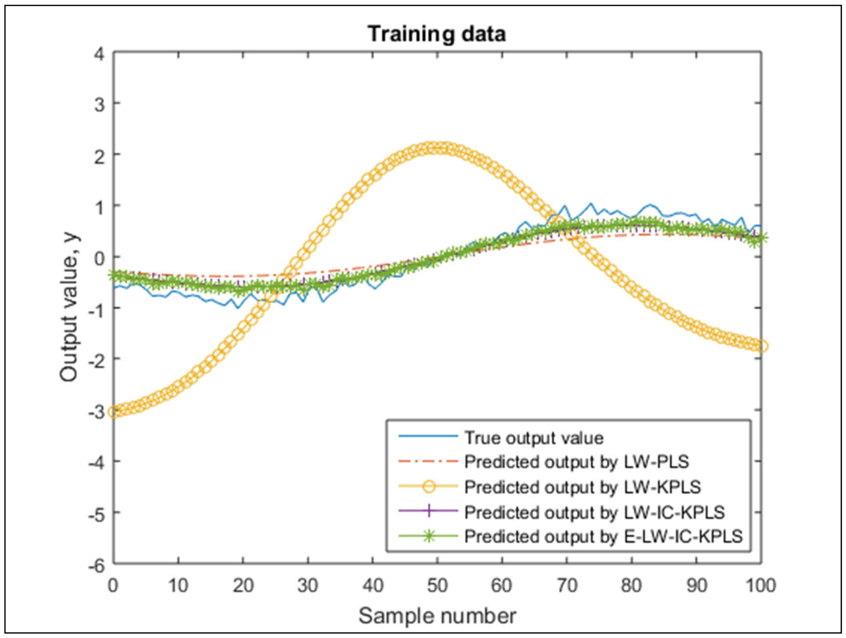

Figure H.9 Prediction results for training data from case study 4 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS models

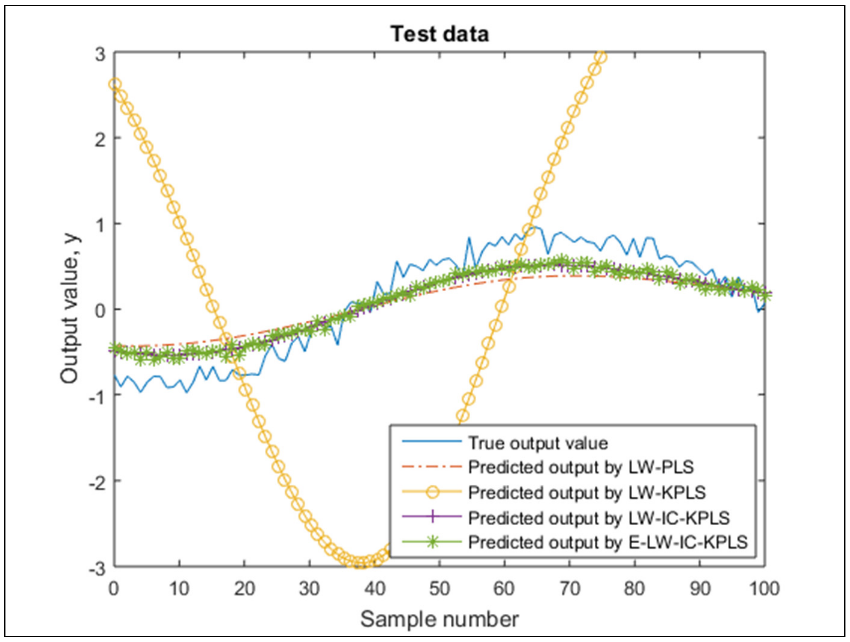

Figure H.10 Prediction results for the test data of case study 4 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms

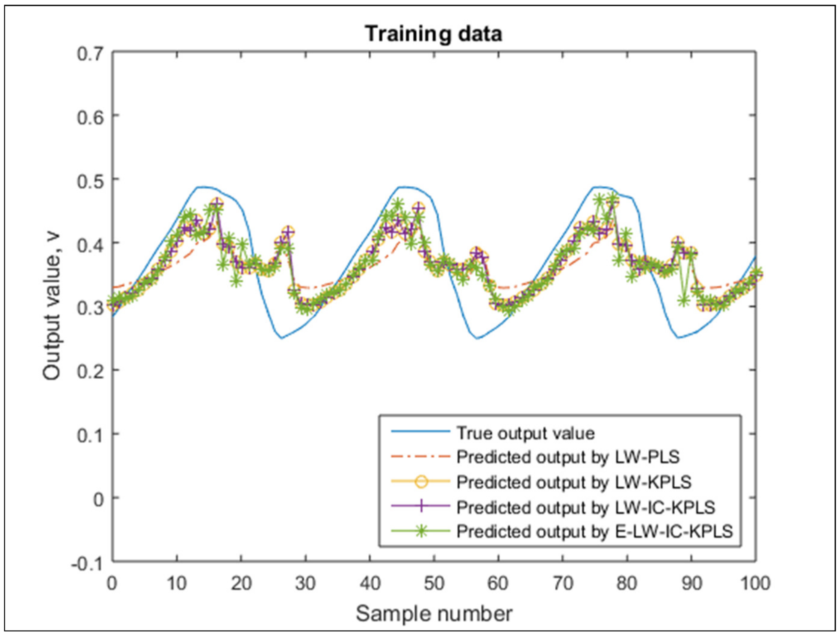

Figure H.11 Prediction results for training data of case study 5 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS models

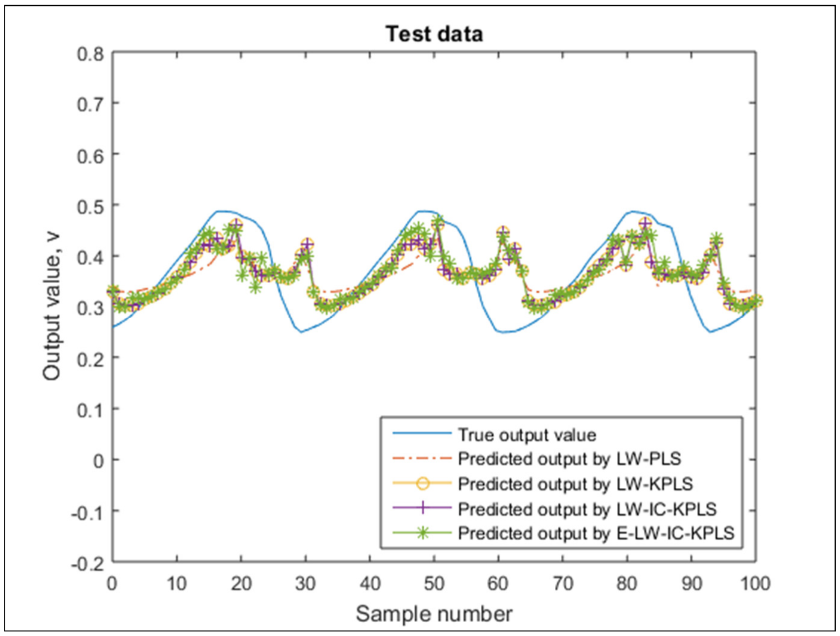

Figure H.12 Prediction results for test data of case study 5 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms

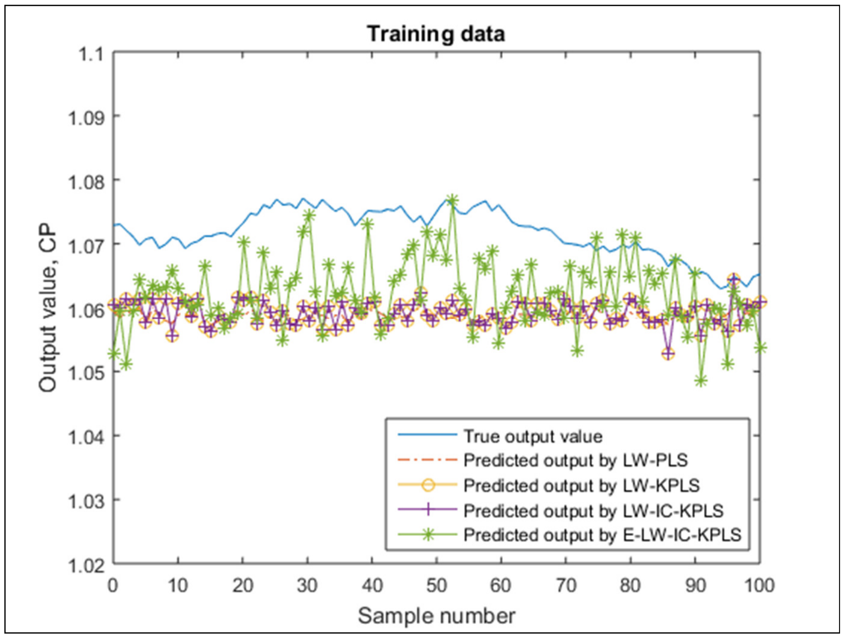

Figure H.13 Prediction results of product concentration for training data in case study 6 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms

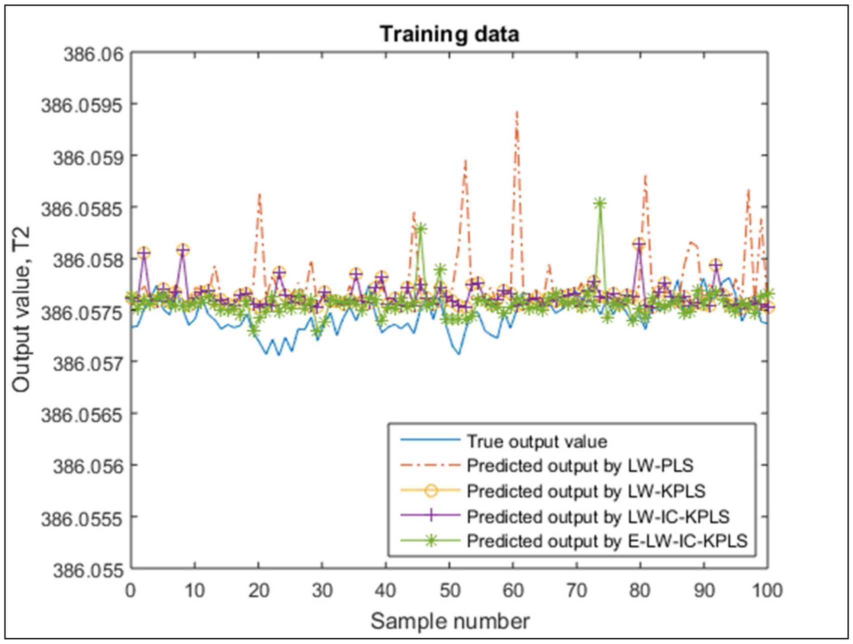

Figure H.14 Prediction results of reactor temperature for training data in case study 6 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms

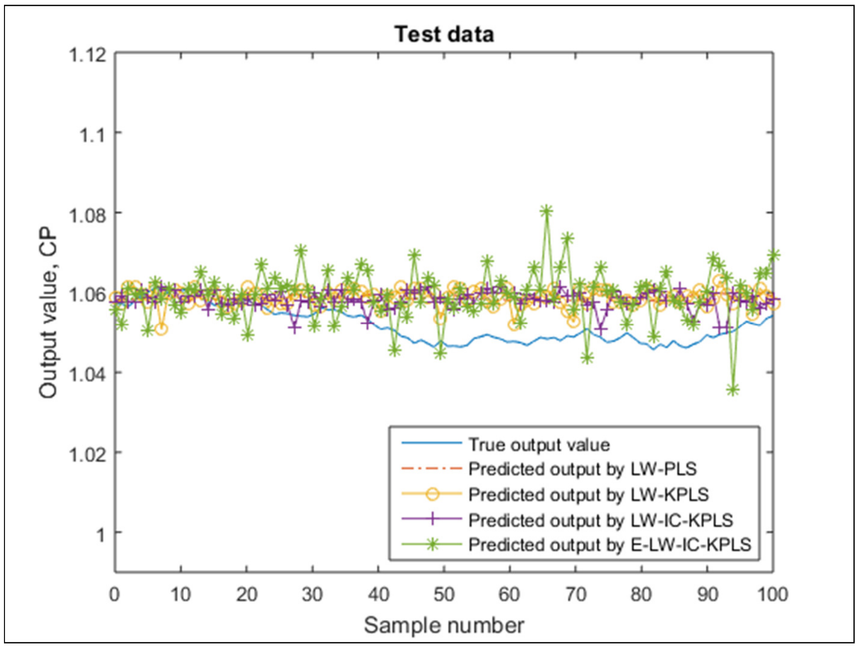

Figure H.15 Prediction results of product concentration for test data in case study 6 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS algorithms

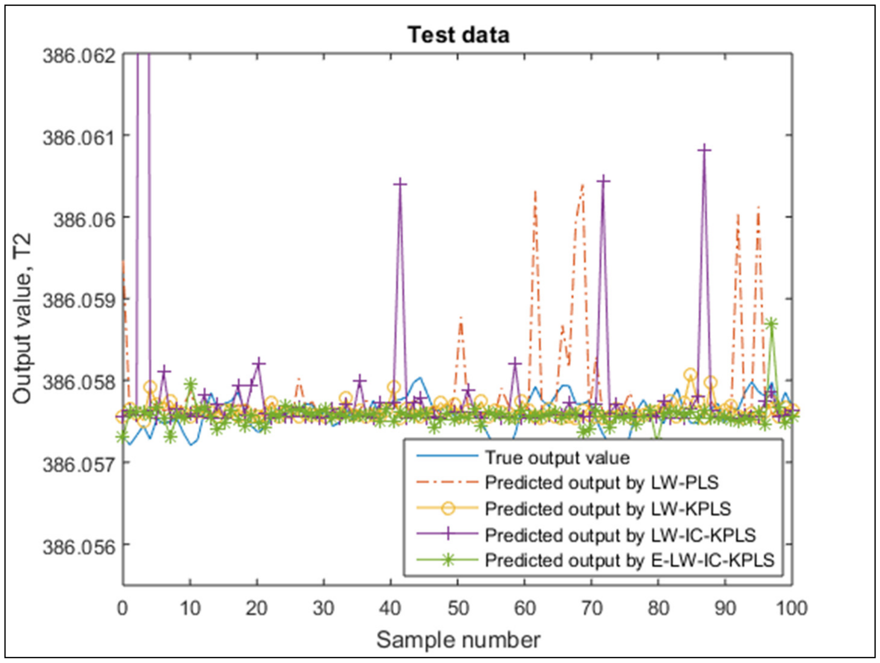

Figure H.16 Prediction results of reactor temperature for test data in case study 6 using the LW-PLS, LW-KPLS, LW-IC-KPLS and E-LW-IC-KPLS models