Modelling and Optimisation of Batch Processes Using Computational Intelligence Techniques and Statistical Learning Approach

A Thesis Submitted By

Kazeem Opeyemi Alli

For the Award of Doctor of Philosophy

School of Engineering

Newcastle University

March 2022

ABSTRACT

Batch processes are commonly used for the manufacturing of high-value-added products such as specialty chemicals and pharmaceuticals. The efficient operation of batch processes is of great importance to produce high-quality products with minimal consumption of energy and resources. Batch process optimization and control are essential for achieving this.

One common approach to the modelling and optimization of batch process systems is to use the first principle concept in building the models of the process using mass balances, energy balances and reaction kinetics of the batch process operations. However, this type of mechanistic model is difficult to develop due to the complexity of the process operations, multiple variables, and batch-to-batch variations involved. The development of mechanistic models is time and effort-challenging, which may not be feasible for batch processes where frequent changes in product specifications occur and a type of product is usually manufactured for a limited time in response to the dynamic market demand.

Computational intelligence techniques address these shortcomings by making use of data-driven concepts in the modelling and optimization of batch processes. With the development and progress in research, data-driven modelling is becoming the more widely used method in modelling and analyses of batch/fed-batch process operations. Extreme learning machine (ELM) is a type of data-driven modelling technique with a fast-training process and can be used for modelling a different kind of process operation like the conventional neural network (NN). ELM has been established to be successful in modelling nonlinear (complex) batch operations as it provides good generalization performance at fast learning speed and gives accurate long-range or multi-step ahead prediction performance. However, it has its shortcomings as well, hence the need to combine other statistical learning techniques to improve its general prediction capabilities.

This work presents the modelling and optimisation of fed-batch processes using different data-driven modelling techniques such as the ELM, Bootstrap Aggregated ELM, and Iterative Learning Control. It also presents the strategy of merging extreme learning machine (ELM) and recursive least square (RLS) techniques in modelling and batch-to-batch optimization of fed-batch processes.

To cope with the batch-to-batch variations due to unknown disturbances such as unknown process condition drift, the RLS algorithm is integrated with the ELM to update the output layer weights recursively from batch to batch. This is because the number of hidden neurons selection together with the output layer weights computation are major criteria towards accurate model predictions in ELM. The recursive least square (RLS) adapts to the current process operation by recursively solving the least-squares problem in the considered model. RLS estimation algorithm nullifies the model plant mismatches caused by the occurrence of unknown disturbances. The offline trained output layer weights of the ELM are used as the initial parameter estimation in RLS. After updating the ELM model, optimisation is carried out to update the feeding policy for the next batch.

The proposed technique is thus applied to two fed-batch case studies including a simulated fed-batch reactor process and a simulated baker's yeast fermentation process. The results obtained from the use of the proposed technique show that the proposed technique can accurately cope with unknown disturbances and improve process operation from batch to batch.

ACKNOWLEDGEMENT

To God, the Almighty Allah, I would say Alhamdullilah for making this degree a reality and a successful one. I always appreciate your kind gestures even when I felt I wouldn't fit for such a role, you make it all possible, Masha Allah!

The PhD has been a tremendous 4 and a half years for me and my family. So much happened along this journey and I appreciate every moment of the programme.

Special appreciation goes to the funding organisation, The Petroleum Technology Development Fund Nigeria (PTDF-Nigeria), which made this research feasible financially. I would say, without the funding organisation, there wouldn't be any research findings. This is the only reliable and prestigious scholarship you can get in Nigeria and rest assured that all are being sorted for you throughout your programme (either master's or PhD).

The research would have been impossible without the constructive support given by my supervisor: Dr Jie Zhang, you are the best at what you do. I appreciate every drop of help you rendered in the cause of this research, be it in terms of valuable and constructive suggestions at the inception, and during the main research work, your patience guidance at all times, and your kind gestures. I wish we can have more of your type at the School of Engineering, Chemical Engineering Department, Merz Court, Newcastle University.

I would also like to extend many thanks to all members of staff (both academic and administrative) at the School of Engineering, Chemical Engineering Department, Merz Court, Newcastle University. I say a big thank you for being helpful in times of need.

My warm and humble regards to my ever-supportive Family members. Starting from my wife and son: Mrs Saidat Olajumoke Alli and Yassir Alli, I say a massive thanks to you all for your kind, moral and financial support always, you have been a blessing all days. To my unrelenting dad and siblings: Mr M. Alli, Dr Mrs M. M Adeyanju, Dr Mrs M. Taiwo, Mrs K. Oshoala, Mr L. Alli and Mrs Shakira Alli, you are all appreciated for your moral and academic support. Thank you for always been always there.

To all friends and families, I say a big thanks to you all for your moral support!

DEDICATION

This research work is fully dedicated to the loving memory of my mother: Mrs Fausat Alli who gave her all to me and my siblings before her soul departed the earth. I miss you mum and I pray to the Almighty Allah to clear all your sins and count you among the inmate of Aljanat Firdaus!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURES

a	coefficients of input variables of regression models
ai	the width of the Gaussian function
A	matrix of an appropriate size
b	bias of hidden nodes
Bi	the linguistic labels associated with input n,
ci	the centre of the Gaussian function.
c	a vector of size (2n2) ×1 containing the lower and upper bounds of K
$C_A(0)$	initial reactant A concentration mol/L
$C_B(0)$	initial reactant B concentration mol/L
$C_C(0)$	initial product C concentration mol/L
$C_D(0)$	initial by-product D concentration mol/L
d	is a disturbance matrix.
d_k	disturbance sequence in LTV perturbation
e_k	tracking error of measured product qualities at the under integrated control strategy
e_i	initial tracking error of measured product qualities at the under integrated control strategy
F	an indicator represents the extent of accuracy the model has improved
G	the process gain matrix
$\boldsymbol{G}_d$	the disturbance gain matrix
$\boldsymbol{G}_{diag}$	a diagonal matrix of G.
Gc(s)	PID controller transfer function

G_k	batch-wise LTV perturbation model
H	hidden layer output matrix
K_1	forward reaction rate
K	temperature
K	a steady state gain matrix
K_e	Saturation constant (eth)
K_i	Inhibition constant
K_o	Saturation constant (oxy)
Ks	Saturation constant (sub)
$\boldsymbol{K}_d$	disturbance gain matrix
K_p	proportional gain
K_i	integral gain
K_d	derivatives gain
$\acute{K}_{jk}$	the jkth element of K
m	mass fraction
oi	output of a standard SLFNs
O1, i	the degree to which the inputs m is related to Ai
O1, j	the degree to which the input n is related to Bj
Pu	the period of constant oscillation using ultimate cycle tuning rules
Q_s	weighting matrix for tracking error in ILC method
r	correlation coefficient
R	multiple correlation coefficient

S	the standard error
t	an indicator represents the extent of input contribution to predicted variable
T	a vector of the target values of network outputs.
T_{cond}	temperature of condenser
U_k	perturbation of control trajectory
U_s	nominal control trajectory
ΔU_k	control change between two adjacent batch runs
ΔU_{k+1}	control change between two adjacent batch runs for next batch
$\boldsymbol{v}_k$	measure disturbance variables
v	flow rate
w	weights between hidden nodes and input nodes
w_K	the sequence of model errors
$\boldsymbol{w}k$	unmeasured disturbance variables.
Xn	Input variable of regression model
Y	output variable of regression model
Y_d	specified reference trajectory
Y_k	measured product quality trajectory at the *k*th batch run
Y_{k+1}	measured product quality trajectory at the *k+1*th batch run
Y_i	initial product trajectory
Y_s	nominal product quality trajectory
$Y_{(x/s)^{\wedge}ox}$	yield component (ox)
$Y_{(x/s)^{\wedge}red}$	yield component (red)

y_{sp}	the desired output in the control loop
y	the controlled output in the control loop
$\overline{y}_k$	the desired output trajectory
$\hat{y}_k$	the predicted/measured output trajectory

Greek letters

β	weight linking the ith hidden node and the output node
βij	a function of K and $\boldsymbol{K}d$
εi	difference between predicted and actual output values.
λ	the values of elements in the RGA matrix
μ a	vector of manipulated variables
μ	control signal
τi	integral time constant
τD	derivative time constant
ɳ	production rate

Mathematical operators

Σ	summation
| |	Modulus
Exp	Exponential function
Ln	Natural logarithm function

log 10- base logarithm function

Abbreviations

ANN	Artificial neural networks
BP	Back propagation
CV	Controlled variable
ELM	Extreme learning machine
ILC	Integrated Learning Control
IMC	Internal Model Control
LM	Levenberg-Marquardt
MLR	Multiple linear regression
MSE	Mean squared error
PID	Proportional–integral–derivative
PCR	Principal component regression
RBM	Restricted Boltzmann machine
SLFNNs	Single layer feedforward neural networks
SISO	Single input single output
SVM	Support vector machine

CHAPTER 1.

INTRODUCTION

1.1 Research Background

Batch process operation is the production in time-sequential form, into distinct sets with characteristic features of flexibility, unsteady and nonlinear dynamical behaviour during the production process. Currently, many industrial companies make use of batch/semi-batch process techniques during production for their simplicity in the mode of operation and also support the production of high-value products, such as fine chemicals, and other suspension polymers with specific morphological properties, pharmaceutical products (bio and non-bio) and semiconductor products. Despite their limited usage in some big manufacturing industries, batch/semi-batch operation processes still form significant manufacturing processes that cannot be eliminated due to their simplified process operation and, consequently the sole interest of better monitoring of end-of-batch quality control parameters (Zhang, 2005a).

According to (Kulkarni *et al.*, 2004) there are three (3) main stages involved in running batch/semi-batch process operation, namely: (i) the feeding of reactants into the reactor, (ii) monitoring of the evolvement of operation and (iii) the yield delivery at the termination of process operation. The most decisive among these stages of operation is the monitoring of the process operation and to monitor such stage, the process control personnel need an accurate process model to achieve process optimisation. Therefore, the primary facet of process optimisation lies in the accurate process model which can either be in empirical (data-driven) or mechanistic (from the first principle) model form. In mechanistic modelling, detailed concepts of the process kinetics, reactions, energy, momentum, and mass balances phenomenon need to be well understood as these will provide precise insights into the mathematical representation of the process behaviour.

However, due to the complex nature of many chemical processes such as batch-to-batch variation and non-linearity of the processes, the detailed conceptualisation information proves more difficult and time-consuming to obtain. Developing mechanistic models usually require a significant amount of time and effort in understanding the process concepts, which may not be feasible for batch/fed-batch processes where frequent changes in product specifications occur and some product brand is usually manufactured for a limited period in response to the dynamic market demands. Data-driven modelling can be a very useful alternative in this case.

The new trending idea of machine learning and data science analysis in research developments has become widely acceptable insight used in modelling and analyses of any form of process operations. Thus, the data-driven modelling concept utilizes statistical theories in establishing the process performances, monitoring the progress of pre-set conditions of operation for optimization purposes and generalization capabilities in predicting unforeseen circumstances. In this concept, detailed knowledge of the process operation is not necessary but past historical data on the process operation is required. As long as the historical data is large enough, data-driven modelling and process optimization can be established with machine learning and statistical theories conceptualization (Liu *et al.*, 2019).

Machine learning (ML) concepts such as Neural Networks (NN) and Extreme Learning Machine (ELM) are like having as powerful means of approximating continuous nonlinear functions with an encouraging prospect in modelling and controlling any chemical processes through utilizing historical process operational data. These ML models can be used to forecast steady-state and dynamic process behaviour if properly trained and validated, hence resulting in better-quality control performance and process optimisation (Zhang, 1999; Tian;Zhang and Morris, 2001; Kulkarni et al., 2004; Xiong et al., 2004a; Zhang, 2004; Xiong and Zhang, 2005; Mukherjee and Zhang, 2008; Zhang, 2008). To model and control process plants effectively with a data-driven concept, large sets of data capturing the essential parts of the process settings will be required (Zhang, 2004). In most manufacturing industries, there is always insufficient batch-run data to use in training the model due to production cost minimisation. Thus, the neural network training with limited data sometimes leads to overfitting with an inaccurate model and poor generalisation.

To improve model prediction accuracy from insufficient data available while training during modelling, (Zhang, 1999) suggests the use of bootstrap aggregated or stacked neural networks. The bootstrap aggregated network combines several individual neural networks in building a model as different networks do perform well in different sections and combining those networks into a single network gave an improved model prediction.

Recently, various researchers have demonstrated the use of an extreme learning machine (ELM) instead of a conventional neural network for its fast training. An ELM is a single hidden layer feedforward neural network (SLFNN), but its training processes are different. In

ELM, the hidden layer weights are arbitrarily assigned and fixed without repeated adjustment, unlike the traditional neural network training approaches. The parameters to be learned in an ELM are the connections (weights) between the hidden layer and the output layer, which are determined with a one-step regression-type approach using Moore-Penrose (MP) generalized inverse matrix. Thus, "ELM is formulated as a linear-in-the-parameter model and then solved in form of the linear system of equations" (Huang *et al.*, 2015). ELM is impressively proficient, fast in training, with good generalization ability, and able to reach global optimum with the least human interference when compared to traditional feed-forward neural network learning methods. Previous studies have shown that ELM maintains its general approximation capability with arbitrarily generated hidden layer weights if "the activation function in the hidden layer is infinitely differentiable" (Huang;Zhu and Siew, 2006) and its learning algorithm could be used to train SLFNNs with either differentiable or non-differentiable activation functions (Cao and Yuan, 2011; Li et al., 2017) . Based on all these facts, ELM is a good choice for data-driven modelling in a situation where there's insufficient historical process data, unavoidable disturbances with transient characteristics process and for nonlinear batch/semi-batch process. To have an accurate model prediction, a large aggregate of data covering the diverse aspect of process operating conditions should be used in training of the model prediction (Jie, 2003; Zhang, 2004; Mukherjee and Zhang, 2008; Li *et al.*, 2017).

Regardless of all the facts mentioned above on ELM, its quick training and good generalization ability depend on the generation of random weights and selection of the number of hidden neurons, which is clearly by chances or probabilities and thus this sometimes led to model process mismatch. Furthermore, unknown disturbances commonly exist in production processes due to variations in raw materials composition, reactive impurities, process equipment degradation due to wearing and reactor fouling is common in many manufacturing plants (Zhang *et al.*, 1999). All these and insufficient historical data can lead to inaccurate model prediction or model plant mismatch. To overcome the model plant mismatches, the recursive least square technique (RLS) is proposed to be integrated with an ELM algorithm to correct the model plant mismatch prediction in ELM by updating the output layer weights of the ELM model from batch to batch. Based on the batch-wise updated ELM model, the optimal control policy can also be updated from batch to batch. This is termed as batch-to-batch optimal control strategy which utilises the repetitive feature of batch operation, by using immediate batch historical (previous) operational data to improve the operating trajectories of the next (current) batch runs intending to improve the process performance indicator of the overall batch operation. Due to the repetitive nature of batch processes, the error-tracking strategy from the previous batch is corrected onto the next batch using the error difference from the current and previous batch in achieving desired output trajectories close to actual product quality, although the final time product quality is the target of the control strategy (Jie, 2003).

Many conventional optimal control techniques have been invented for linear time-invariant systems, which are presumed to be well-known. In most real instances, the systems to be controlled are nonlinear and the detailed first principle knowledge is unknown. This unknown situation becomes difficult when the unknown parameters of the system model change endlessly during the process operation. These changes in the parameters bring about uncertainties in the model, which are sometimes difficult to control with the established adaptive control techniques. The adaptive control technique can adjust itself in unfamiliar uncertainties through either direct or indirect forms. It is said to be direct if the estimated parameters solution is used directly in an adaptive controller and indirect if parameter estimate details are used to adjust the controller. It has been shown to work well for many dynamic systems with unknown fixed parameters. However, their applications may not be easy to extend to unknown time-varying parameter systems (Ahn;Chen and Moore, 2007).

The RLS technique is a statistical tool in form of an adaptive filter algorithm for online parameter estimation, which estimates a plant model by repeatedly searching for the coefficients that minimise the weighted least squares solution of that model. RLS can solve least square estimation problems by attempting to minimise the sum of weighted squared errors between the measured and the predicted outputs. This concept is used repeatedly in obtaining the parameter estimation of the predicted output weight by using a model that is linear in predicted output weight from the ELM (Alli and Zhang, 2020).

Generally, parameter estimation is usually time-varying in many process systems which can be of two cases, namely: the parameter estimation can be constant for a long period and suddenly changes and sometimes changes with time slowly as the process operation progresses. In either case, a monitoring solution is sought and for the former case, covariance resetting is the solution for abrupt changes while for the latter case, the forgetting factor is added to correct the slow changes with time in the parameter estimation of that process (Wigren, 1993).

1.2 Motivation

Over a couple of decades, machine learning has been in vogue in modelling various industrial complex situations where traditional techniques have constraints in giving accurate model representation. Machine learning-based models may tend to be much easier to establish than mechanistic models but still encounter plant model mismatches and insufficient datasets in building reliable process models. Hence, the basic questions that always come to mind are how we can solve or minimize such problems to make machine learning much easier with negligible error for process modelling, monitoring and optimization purposes (Cao, 2018).

Many researchers have also worked on the improvements of machine learning for modelling complex process operations such as the advent of bootstrap aggregated NN and bootstrap aggregated ELM which majorly focused on maximal utilization of the historical dataset through bootstrapping of a dataset in building a reliable process model. However, this research work will propose another possible way of improving machine learning with a minimal historical dataset for modelling and optimization of chemical processes by merging the well-established machine learning (ELM) model with the statistical technique (RLS) concept.

1.3 Justification of the research

Admittedly, many strategies and concepts have been employed towards neural network-based models for batch-to-batch optimization in literature to compensate for process model discrepancy (modelling error) but as advancements and discoveries of new techniques of NN were being discovered, there is a need to apply the new concepts to new research to show how efficient and effective the new techniques is when applied to case study and compared to traditional concepts. Some of the earlier concepts suggested are: the use of bootstrap aggregated neural networks through principal component regression (PCR) concept for the development of robust nonlinear models was proposed (Zhang, 1999), comparative studies between single neural network and stacked neural networks in modelling and controlling batch polymerization reactor and the study admitted that stack neural networks outperform the single NN (Tian;Zhang and Morris, 2001), the use of recurrent neural networks for optimal control strategy in pilot-scale polymerisation reactor (Xiong and Zhang, 2005), and the use of support vector regression model for optimal batch to batch control (Liu *et al.*, 2005) etc.

Single hidden layer feedforward neural networks (SLFNNs) trained using the conventional backpropagation training have some limitations such as slow convergence speed, large numbers of iteration learning steps which lead to large training time, and the possibility of converging to local optima instead of global optima (Huang;Zhu and Siew, 2006). Moreover, NN with more than two layers is difficult to train using a traditional gradient descent scheme which suggested pre-processing data before data analysis and extraction for modelling (Jeong and Lee, 2018). This eliminates noises and outliers in the data and possibly reduces its searching dimensionality. With all these shortcomings and limitations of traditional NNs, it opens opportunities for more effective empirical forms such as extreme learning machines (ELM), deep learning machines, deep belief learning machines and restricted Boltzmann machine to be used for process modelling, monitoring and optimization purposes.

1.4 Aims and Objectives of the Research

This research work is aimed at developing improved model reliability and capability by using computational intelligence techniques for batch process modelling and control through the combination of an extreme learning machine and recursive least square technique.

The main objectives are to:

1. Develop models for batch/fed-batch chemical processes with ELM and Bootstrap aggregated ELM.
2. Investigate methods for improving ELM model reliability and generalization capability in (1) through a combination of ELM and RLS algorithms for the case studies.
3. Modelling and optimization of the batch/fed-batch chemical processes with the proposed ELM and RLS algorithm in (2).

1.5 Scope of the Research

Both extreme learning machine (ELM) and bootstrap aggregated ELM will be used to model some typical batch/fed-batch chemical processes. Furthermore, an improvement in the model reliability and generalization will then be sought through a combination of ELM and RLS algorithms. The RLS is expected to take care of the model plant mismatch that may arise due to outliers (unavoidable changes that do occur to the dataset during algorithm execution) in some of the datasets after data processing and analysis on the historical dataset.

This proposed technique will be applied to some typical batch/fed-batch chemical processes for process modelling, monitoring, control, and optimisation of the process operation.

1.6 Thesis organisation

The thesis is organised as follows:

In Chapter two, relevant general literature related to batch process modelling and computational intelligence is reviewed. This is to provide detailed insight into the progress in the improvement of machine learning techniques to process modelling. Some of the discussed modelling techniques are the Artificial Neural Network (NN), Bootstrap Aggregated Neural Network (BAGNET), Extreme learning machine (ELM), Bootstrap Aggregated ELM (BA-ELM), Iterative Learning Control (ILC) and lastly combining ELM with Recursive Least Squares technique (ELM-RLS), which is the focus of this research work.

Chapter three presents studies of the modelling of fed-batch reactor and baker's yeast fermentation process using different data-driven modelling techniques such as ELM, BA-ELM and ILC. This is done to show how effective, fast, reliable, and good generalisation performance each computational intelligence method will be achieved. Both static and dynamic models of these case studies are developed using either the full or partial simulated model-based datasets and they all offer different levels of accuracy in their model predictions.

Chapter four focuses on batch-to-batch adaptive modelling of the case studies (the fed-batch reactor system and baker's yeast fermentation process) using the proposed ELM-RLS modelling technique. These models developed are in the static form, with part of the simulated historical datasets used. Despite the use of insufficient historical process data in building the static model, the proposed batch-to-batch adaptive modelling offers an accurate and reliable model prediction for both case studies.

Chapter five focuses on the optimization control of the fed-batch process of the baker's yeast fermentation obtained with the proposed ELM-RLS technique. In this chapter, iterative batch-to-batch optimization control utilizes the repetitive nature of the process by finding the updated input variables using information from the previous and the current batch runs to enhance the future batch runs of the yeast fermentation process. The results obtained show an improved feeding control policy which yields an improved final biomass concentration when applied to the mechanistic simulation model.

Chapter six focuses on research conclusions and recommendations for some possible future works on the proposed hybrid technique of merging ELM with some other useful statistical techniques.

1.7 Conference Attended and Presentations

- Kazeem Alli and Jie Zhang (2018) ‘Batch-to-Batch Online Optimisation Control with Extreme Learning Machine and Integrated System Optimisation & Parameter Estimation’, Advances in Process Analytics and Control Technology 2018 Conference, 25th-27th April 2018, Gateshead, UK.
- Kazeem Alli and Jie Zhang (2020) ‘Adaptive Modelling of Fed-Batch Processes with Extreme Learning Machine and Recursive Least Square Technique’, 12th International Conference on Agents and Artificial Intelligence, 22nd-24th February 2020, Valletta, Malta.
- Kazeem Alli and Jie Zhang (2021) ‘Adaptive Optimal Control of Baker’s Yeast Fermentation with Extreme Learning Machine and Recursive Least Square Technique’, Advances in Process Analytics and Control Technology (APACT-21 On-line Conference), 19th-23rd April 2021, UK.
- Kazeem Alli and Jie Zhang (2021) ‘Adaptive Optimal Control of Baker’s Yeast Fermentation with Extreme Learning Machine and Recursive Least Square Technique’, 31st European Symposium on Computer Aided Process Engineering (ESCAPE-31), 6th-9th June 2021, Istanbul, Turkey.

Published Papers

- Alli, Kazeem, and Jie Zhang (2020). Adaptive Modelling of Fed-batch Processes with Extreme Learning Machine and Recursive Least Square Technique. *In Proceedings of the 12th International Conference on Agents and Artificial Intelligence-Volume (2): ICAART, ISBN 978-989-758-395-7 ISSN 2184-433X, pages 668-674”. DOI: 10.5220/0008980506680674. SciTePress - Publication Details*
- Alli, Kazeem, and Jie Zhang (2021). Adaptive Optimal Control of Baker’s Yeast Fermentation Process with Extreme Learning Machine and Recursive Least Square

Technique. Computer-Aided Chemical Engineering, Vol.50, The 31st European Symposium on Computer-Aided Process Engineering, Elsevier. *https://doi.org/10.1016/B978-0-323-88506-5.50191-1* .

- Alli, Kazeem, and Jie Zhang. “Optimal Control of Fed-Batch Processes with Computational Intelligence and Statistical Learning Technique”. (Drafted and ready for submission).

CHAPTER TWO

LITERATURE REVIEW

2.1 Batch Process Modelling and Control

Batch and semi-batch processes are well-known means of manufacturing high-quality and value-added products in the chemical, biochemical, pharmaceutical, and food industries. They are characterised by dynamic and non-stationary behaviour, high conversions of batch-end quality products, finite duration, and the cyclic repetition of all batch stages of the production process.

The main objective of the batch operation process is to achieve reliable and reproducible desired product qualities at all times of the production process but usually, this is not feasible due to batch-to-batch variations which affect the batch-end product quality. To reduce the batch-to-batch variations, efficient process monitoring, and control is developed to detect any deviation and take corrective measure in the process operation thereby increasing the output turnover of the constantly fluctuating market demands. Process monitoring of batch operation helps in fault diagnosis, maintaining process safety, reducing the cost of production through efficient use of materials in production and lastly for an improved understanding of the process operation. On the other part, quality control helps in maintaining uniform quality at all stages of production through the standardisation of the quality variables to the required settings (Qin, 2012) .

Some of the process variables in batch processes include temperature, pressure, feed rate and agitation rate etc. They are the key indicators of process performance which can either be offline or online measured. Information captured is thereby used to analyse the process behaviour for improved process operating conditions which leads to reliable product quality (Choi; Morris and Lee, 2008).

As both batch and semi-batch processes are highly nonlinear, time-varying, and highly intertwined with many uncertainties, building a process model from the governing physical and chemical process laws (i.e., first principle model or mechanistic model), is incredibly tough and a series of strategies have been developed to unravel these limitations in the mechanistic model for monitoring and controlling of batch processes.

Machine learning techniques are increasingly applied emerging technologies that can greatly boost the utilisation of plant data in the creation of data-based process models. This is known as a data-driven modelling approach, and it provides fast and accurate modelling of processes without requiring extensive process knowledge. To generate an improved model, machine learning is utilised in conjunction with process knowledge as well as engineering constraints. Data-driven models are typically referred to as empirical models, which are obtained either by linear regression or non-linear regression and provide easier alternatives to complicated mechanistic modelling of batch processes, bio-process reactors and fermenters, complex refining units and fluidized bed processes. It also allows model-building analysis in a broader set of data while leveraging advanced data science techniques for model predictions of these processes.

2.2 Modelling Techniques with Machine Learning

Machine learning is a broad term that describes how hardware or software enables machines to mimic human intelligence using algorithms programmed in computer codes in analysing large datasets in making intelligent decisions. Since the invention of machine learning, we have seen a significant increase in its applications in various fields of sciences, engineering, and medical technologies both at the academic and industrial levels. To health personnel, it is a great tool for diagnosing patients, developing new drugs through a combination of possible substances (like in the case of coronavirus vaccines) and making drug prescriptions more comfortable through an online application. To the automobile industry, it redefines the industry into the advent of autonomous (self-driving) cars, self-delivery drones and highly intelligent flight control, and to the manufacturing sectors, it boosts production quality and quantity through the application of intelligent components such as sensors and actuators which helps in effective production, thereby leading to saving billions of pounds through increased productivity.

Machine learning permits systems to acquire knowledge directly from historical data information using its computational algorithm to acquire the necessary information and to execute assignments logically and flawlessly. The computational algorithm is the brain behind the processing instruction procedure of any machine learning concept called artificial intelligence (AI). In other words, machine learning is a subset of AI that uses principles of computer science together with statistical techniques in creating models for predictions and inferences (Nwankpa *et al.*, 2018). Figure 2.1 gives a map of how machine learning and artificial intelligence are related.

Machine learning can use supervised, unsupervised, semi-supervised, or reinforcement learning methods in obtaining detailed information about any process operation either through online or batch learning. In supervised learning, the algorithm learns information from historical data to forecast future outputs for given input data, while in unsupervised learning, the algorithm discovers hidden patterns or data structures of the historical data, and in semi-supervised learning, the algorithm is the combination of both the supervised and unsupervised algorithms. Figure 2.2 gives a general overview of the types of machine learning.

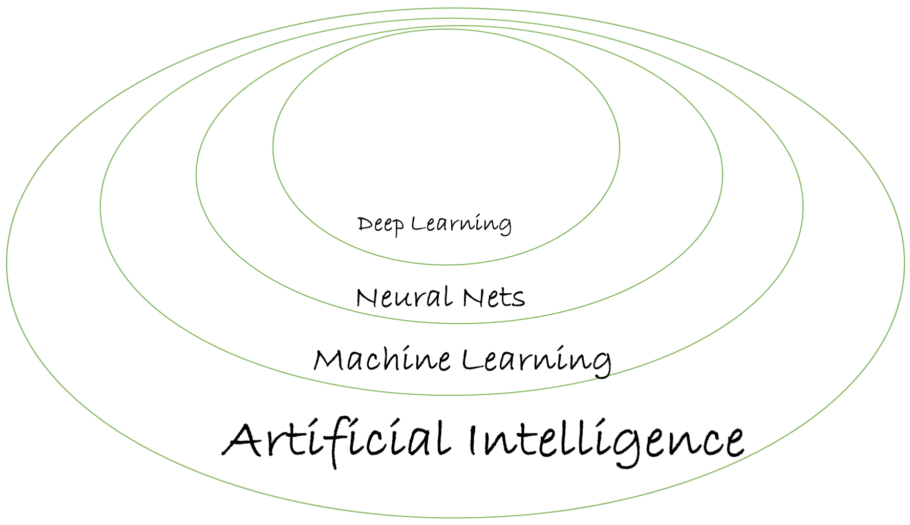


Figure 2.1 Relationship between Artificial Intelligence and Machine Learning

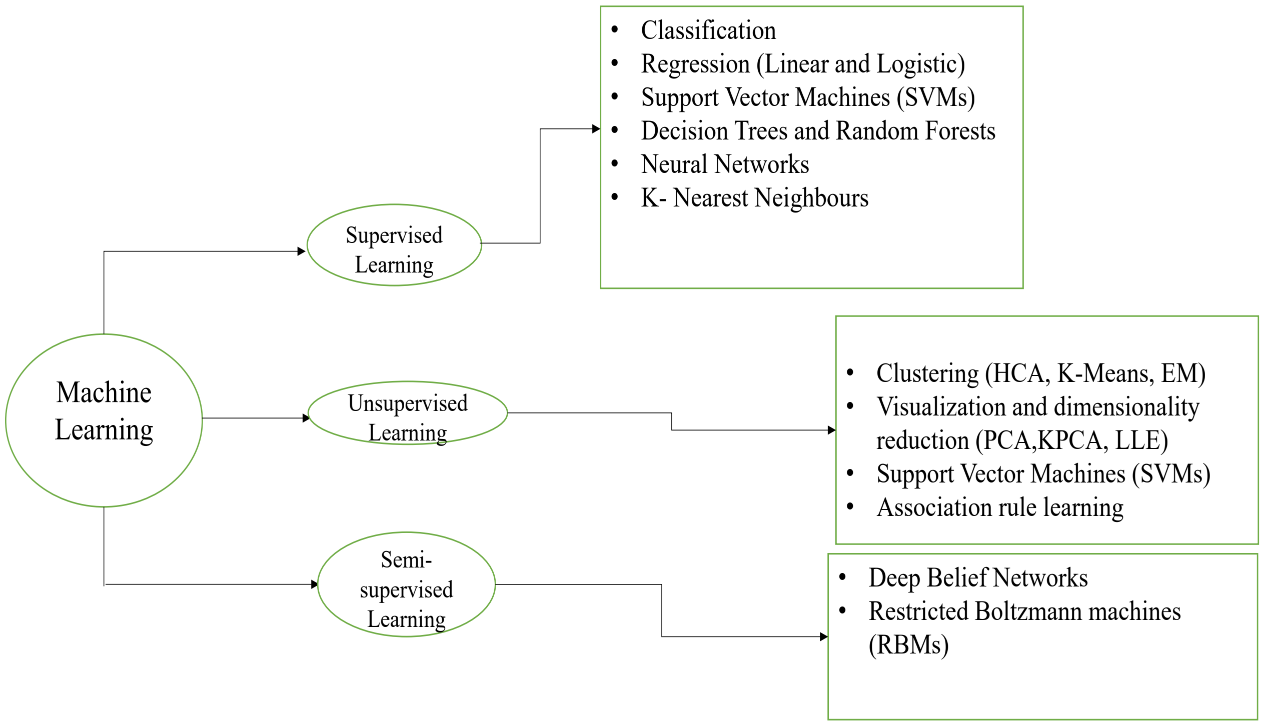

Machine Learning
Supervised Learning
Unsupervised Learning
Semi-supervised Learning
Classification
Regression (Linear and Logistic)
Support Vector Machines (SVMs)
Decision Trees and Random Forests
Neural Networks
K- Nearest Neighbours
Clustering (HCA, K-Means, EM)
Visualization and dimensionality reduction (PCA,KPCA, LLE)
Support Vector Machines (SVMs)
Association rule learning
Deep Belief Networks
Restricted Boltzmann machines (RBMs)

Figure 2.2 General overview of the types of machine learning

Advanced control and monitoring of industrial processes required accurate process models, and when choosing a modelling technique, the mechanistic modelling technique might not be the best option because of the complex nature and large-time execution of the process model involved. A data-driven or hybrid combination of data and mechanistic model provides a better alternative solution but still faces the effect of unexpected disturbance which leads to model misfit of the process operation.

Apart from the unexpected disturbances (such as impurities or catalyst activities) which affect both the data-driven and the mechanistic modelling, the primary cause of model misfit in data-driven modelling is the insufficient availability of data to train the model and the inability to obtain data of some quality variables of the process operation. A large amount of data is needed to build a correct data-driven model and without an adequate amount of data for training the model, it could easily lead to over-training and substantial errors when applied to unobserved data (Zhang, 1999; Zhang, 2008).

Thus, the mathematical representation of the general batch process modelling is given as follows:

Given the historical batch process operation data containing input and the product quality trajectories at all sampling times in the jth batch run as below,

$$U_j = [u_j(0), u_j(1), \ldots, u_j(N-1)]^T$$
$$Y_j = [y_j(1), y_j(2), \ldots, y_j(N)]^T \qquad (2.1)$$

where j represents the batch index, N is the number of sampling intervals within a batch, $u \in R^m$ is the manipulated input variable for the product quality variables $y \in R^n$ and the initial conditions for both input and output variables are given as (u_0, y_0) respectively. Thus, the non-linear relationship that exists between the U_j and Y_j of the batch process case study is modelled by using neural networks.

Each batch run of the manipulated input variable U_j is parameterised at a constant sampling interval to cover the whole batch run length of t_f of the process operation. This strategy of parameterisation at equal sampling intervals throughout the process operation time ensures that the product qualities obtained at each final batch get close to desired output values (Xiong and Zhang, 2003).

There are three (3) different ways of utilising the historical operational data in predicting the dynamic nonlinear relationship between the input control variables and output product quality stated in equation (2.1) above, namely:

The first type of dynamic model utilises the lagged values of the product quality variable and the input control variables up to time t-1 to predict in establishing the future time-series model prediction $y_f(t)$. This form of time-series model prediction is referred to as nonlinear autoregressive with exogenous (external) input (NARX), which is of the form:

$$y_f(t) = f[y(t-1), y(t-2), \dots, y(t-d_o), u(t-1), u(t-2), \dots u(t-d_i)] \quad (2.2)$$

The second form of time-series model prediction is close to the NARX type but only uses the manipulated inputs variable of the past historical process operational data $u(t)$ without past $y(t)$ operational data to predict the output/desired series of future $y(t)$. The form of the modelling representation equation is given as:

$$y_f(t) \ = f[u(t-1), u(t-2), \dots u(t-d_i)] \quad (2.3)$$

Lastly, the future time-series prediction $y_f(t)$ is predicted from only past time-series historical output data $y(t)$ and this is referred to as nonlinear autoregressive (NAR). The modelling equation is given as:

$$y_f(t) = f[y(t-1), y(t-2), \dots, y(t-d_o)] \quad (2.4)$$

where y_f represent the predicted future output, *y* is the process output, *u* is the process inputs, *t* represents the discrete time, $d_o \ and \ d_i$ represents the time delays in the model output and input respectively.

Among all the forms of time-series dynamic predictions mentioned above, the NARX model always gives superior model prediction based on its utilisation of both inputs and output historical data in capturing more information to make an informed decision in predicting future time-series output (Xiong and Zhang, 2003).Thus, the manipulated input variable and output product quality are given in equation (2.1) and are utilised in equation (2.2) in obtaining the model prediction for the different techniques used.

Moreover, suppose a fed-batch reaction process is represented as given in equation (2.1) below:

$$aA + bB \rightarrow cC + dD \tag{2.5}$$

The mass, mole balances and rate of reaction are given as:

For mass balance, Mass (m) = Density (ρ_0) x Volume (V),

$$\frac{dm}{dt} = \dot{m} \text{ and } \frac{dV}{dt} = \vartheta_0 \tag{2.6}$$

$$\frac{dm}{dt} = \rho_0 \frac{dV}{dt} + V \frac{d\rho_0}{dt} \tag{2.7}$$

$$\dot{m} = \rho_0 \vartheta_0 \tag{2.8}$$

For $t = 0 \; and \; V = \vartheta_0$

$$V = V_0 + \vartheta_0 t \tag{2.9}$$

The mole balance of reactant A is given as:

$$F_{A0} - F_A + G_A = \frac{dN_A}{dt} \text{ (i.e., IN – OUT + GEN = ACC)} \tag{2.10}$$

where F_{A0} is the input molar flow rate, F_A is the output molar flow rate, G_A is the generation and the differential term is the accumulation in moles per time. If the system variables are uniform throughout the system volume $(F_{A0} = F_A = constant)$, then equation (2.10) becomes:

$$G_A = \frac{dN_A}{dt} = r_A V \tag{2.11}$$

where $\frac{dN_A}{v.dt}$ denotes the amount in the mole of A disappearing per unit volume per unit time and

$$\frac{dN_A}{dt} = \frac{d[C_A V]}{dt} = V \frac{dC_A}{dt} + C_A \frac{dV}{dt} \tag{2.12}$$

Recall that from equation (2.6), equation (2.12) becomes:

$$\frac{dC_A}{dt} = r_A - \frac{\vartheta_0 C_A}{V} \tag{2.13}$$

Similarly, the other mole balances are given as:

$$\frac{dC_B}{dt} = r_B - \frac{\vartheta_0 (C_{B0} - C_B)}{V} \tag{2.14}$$

$$\frac{dC_C}{dt} = r_C - \frac{\vartheta_0 C_C}{V} \tag{2.15}$$

$$\frac{dC_D}{dt} = r_D - \frac{\vartheta_0 C_D}{V} \tag{2.16}$$

In addition, the rates of reaction of all materials from equation (2.14) to equation (2.16) are all related as follows:

$$-\frac{r_A}{a} = -\frac{r_B}{b} = \frac{r_C}{c} = \frac{r_d}{d} \tag{2.17}$$

Experience has shown that the rate of reaction is influenced by both the composition and the energy of the material (Levenspiel, 1999).

For this research work, two different case studies of fed-batch reaction processes will be discussed, namely: the isothermal fed-batch reactor and the baker's yeast fermentation process. In real terms, I used the fermentation model equations from (2.13) to (2.16) to explain the mass, mole balances and rate of reactions that occurred in the baker's yeast fermentation process.

2.3 Artificial Neural Network

A typical artificial neural network (ANN) is comprised of interconnected processing nodes called neurons and weights which are used for modelling complex non-linear patterns in data. A neural network learns the relationships and patterns in the data and uses the learnt knowledge in predicting the targeted output. These neurons are arranged into three (3) or more distinct layers, namely: the input layer, the hidden layers, and the output layer. The input layer accepts data and assigns weighted values by multiplying the scalar input with the corresponding scalar weights. The hidden layers establish relationships within the net input function through the summation of the weighted inputs to the scalar biases to form the net input signal for the activation function and lastly, the output layer produces the final network outputs.

The activation function controls the limits of the output to an acceptable range of finite values by manipulating the weighted sum of inputs and biases. These activation functions or transfer functions can either be linear or non-linear depending on the function it represents and there are many forms of the functions, namely: the sigmoid, step, hyperbolic, threshold, ramp, radial and many other functions but the sigmoid function is the most used. The sigmoid function is a non-linear activation function that is bounded by a differentiable real function. It is sometimes referred to as a 'logistic function' and it is represented as given in equation (2.18):

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.18}$$

where x is the input values and $f(x)$ is the output of a sigmoid function. In general, neural networks are trained on a data set containing system inputs and the corresponding system outputs and the network weights are adjusted based on the differences between the predicted outputs and the actual targets until the network outputs match the actual targets. For the network outputs to match with the actual targets, several weight-updating iterations are required to train such a network and a typical illustration of such is shown in Figure 2.3.

The behaviour of an ANN is determined by its architecture, weights, and transfer/activation function (Agatonovic-Kustrin and Beresford, 2000). Therefore, the architectures of ANN can be categorised into two groups, namely: the feedforward networks with a typical example as the single hidden layer feedforward neural network (SLFNN), and the Recurrent or feedback network. One of the simplest and most widely used feedforward NN is the SLFNN, in which signals move forward through all layers in only the forward direction (i.e., linked directly to an output layer). Typical architectures of a single neuron model, feedforward and recurrent network are shown in Figures 2.4, 2.5 and 2.6 respectively.

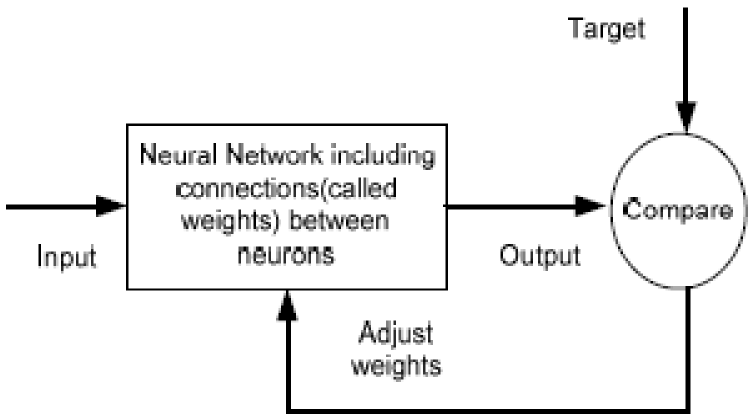

Target
Neural Network including connections(called weights) between neurons
Input
Output
Compare
Adjust weights

Figure 2.3 Typical structure of a supervised ANN (Adnan et al., 2012)

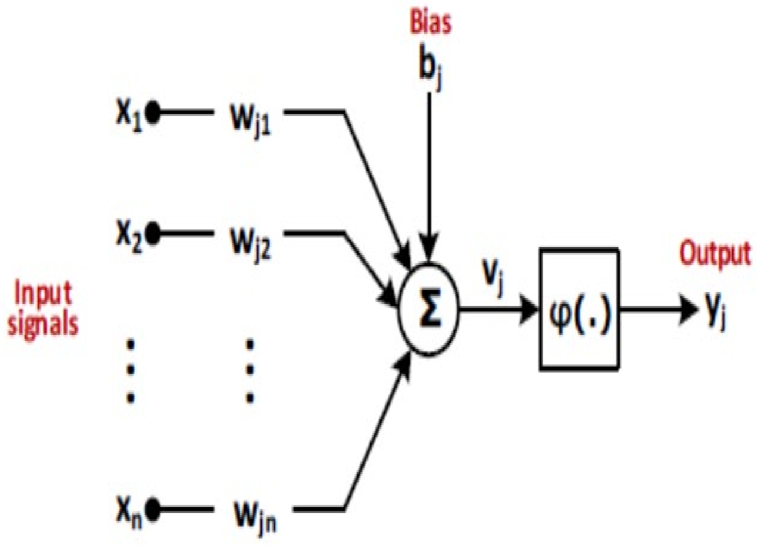


Fig 2.4 A typical single-neuron model (Haykin, 2009)

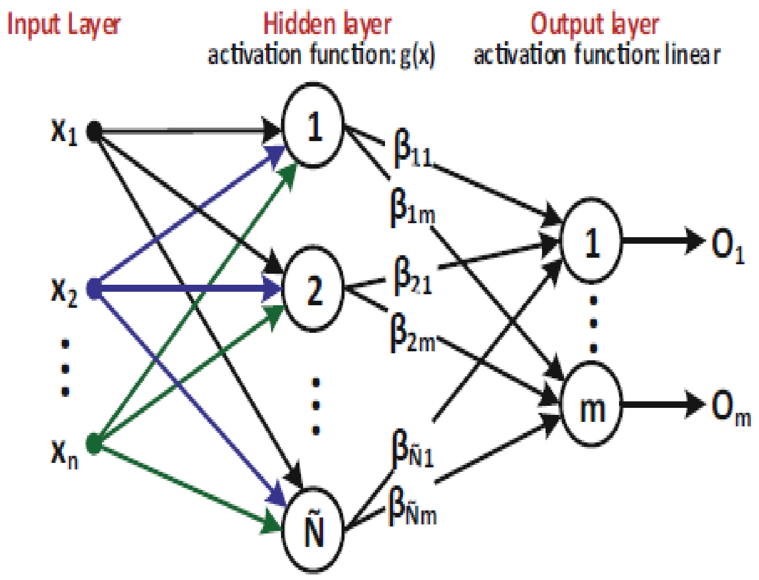


Figure 2.5 A single-layer feedforward network architecture (Baron and Zhang, 2017)

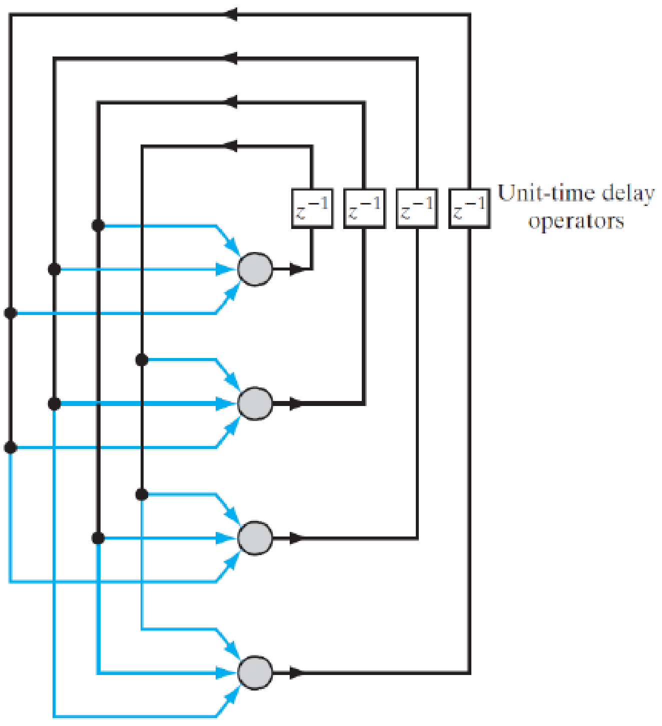


Fig 2.6 A recurrent network architecture with no self-feedback loops (Haykin, 2009)

Over the past few decades, ANN has been gaining a rapid push into research by diversifying into different aspects of science and technological innovations. One of its numerous benefits is seen in estimating continuous nonlinear functions for modelling and controlling chemical processes, especially when a detailed understanding of the process operation is restricted (Zhang, 2004). ANN model is useful in forecasting both steady and dynamic process behaviour accurately if trained and validated correctly (Xiong *et al.*, 2004a). Hence, it improves the control performance and later leads to enhanced process optimisation of the operation. Unlike the regression model, ANN can learn nonlinear relationships from observational data and use the captured knowledge in predicting targeted output through patterns structure in input data.

2.3.1 Single-hidden Layer Feedforward Network

SLFN consists of one input layer, one hidden layer and one output layer. The input layer accepts input signals while the output layer gives out the outcomes for the given input signals.

The hidden layer contains multiple hidden nodes where an activation function converts the sums of weighted inputs to neuron outputs. Figure 2.7 shows the structure of an SLFN.

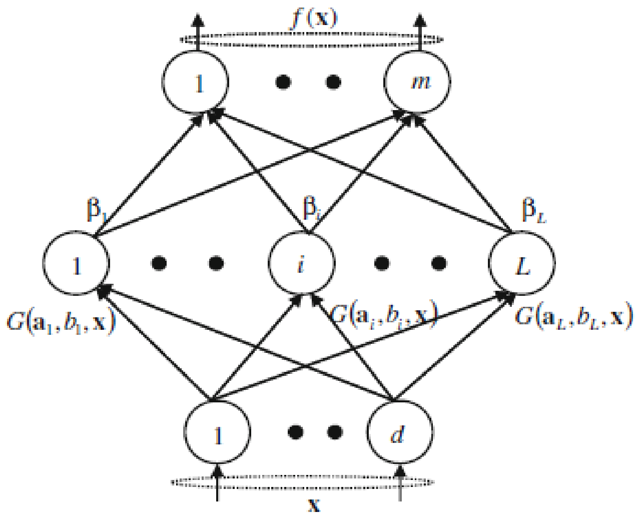

$f(\mathbf{x})$
1
m
β_1
β_i
β_L
1
i
L
$G(\mathbf{a}_1, b_1, \mathbf{x})$
$G(\mathbf{a}_i, b_i, \mathbf{x})$
$G(\mathbf{a}_L, b_L, \mathbf{x})$
1
d
$\mathbf{x}$

Figure 2.7 SLFN structure (Li et al., 2017)

For N arbitrary distinct samples (x_i, t_i), where $x_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in R^n$ is a vector of network inputs and $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in R^m$ is a vector of the target values of network outputs. The standard SLFNs with $\hat{L}$ hidden nodes and activation function $g(x)$ can be modelled mathematically as:

$$\sum_{i=1}^{\hat{L}} \beta_i g_i(x_j) = \sum_{i=1}^{\hat{L}} \beta_i G(a_i . x_j + b_i) = O_j \quad (2.19)$$

where j = 1, …, N, $a_i = [a_{i1}, a_{i2}, \dots, a_{in}]^T$ is a vector of weight connecting the *ith* hidden node with the inputs, b_i is the bias of the *ith* hidden nodes, x_j is the *jth* input sample, $\beta_i \in R^m$ is the weight linking the i*th* hidden node and the output node. The linear activation function is usually chosen for the output node. According to Xiong and Zhang (2004), "standard SLFNs can approximate any continuous nonlinear functions with zero error" which implies that $\sum_{j=1}^{\hat{L}} \|O_j - t_j\| = 0$, i.e., there exists (a_i, b_i) and β_i such that:

$$\sum_{i=1}^{\hat{L}} \beta_i G(a_i . x_j + b_i) = t_j, \ j = 1, \dots, N. \quad (2.20)$$

The above N equations can be written in compact form as:

$$H\beta = T \text{ , where:}$$

$$H(a_1, \dots, a_{\hat{L}}, b_1, \dots, b_{\hat{L}}, x_1, \dots, x_N) \quad (2.21)$$

$$= \begin{bmatrix} G(a_1.x_1 + b_1) & \cdots & G(a_{\hat{L}}.x_1 + b_{\hat{L}}) \\ \vdots & \ddots & \vdots \\ G(a_1.x_N + b_1) & \cdots & G(a_{\hat{L}}.x_N + b_{\hat{L}}) \end{bmatrix}_{N \times \hat{L}}$$

$$\beta = \begin{bmatrix} {\beta_1}^T \\ \vdots \\ {\beta^T}_{\hat{L}} \end{bmatrix}_{\hat{L}Xm} \quad and\ T = \begin{bmatrix} {t_1}^T \\ \vdots \\ {t^T}_N \end{bmatrix}_{NXm} \tag{2.22}$$

In the above equation (2.30), H is called the hidden layer output matrix of the neural network and the *i*th column of H is the *i*th hidden node output concerning inputs $x_1, x_2, \dots, x_N$;$h(x) = G(a_1.x_1 + b_1) \dots G(a_{\hat{L}}.x_N + b_{\hat{L}})$ is called the hidden node feature mapping; the *i*th row of H is the hidden layer feature mapping for the *i*th input x_j (Hsu; Chang and Hsu, 2017).

The backpropagation algorithm is usually used in training SLFN which is simply equivalent to finding the least square solution in $\hat{\beta}$ of the linear system $H\beta = T$ in equation (2.22). This can be described mathematically as:

$$\min_{\beta} ||H\beta - T|| \tag{2.23}$$

Equation (2.23) is equivalent to minimizing the error function given in equation (2.24). The MSE function plays a major role in the fast learning rate in learning algorithms and is expressed mathematically as follows:

$$E = MSE = \frac{1}{N \times m} \sum (||H_{N \times \hat{L}} \beta_{\hat{L} \times m} - T_{N \times m}||^2) \tag{2.24}$$

During the minimisation of MSE in equation (2.33), the parameters $\theta = (\beta, H)$ are adjusted iteratively as:

$$\alpha_{k+1} = \alpha_k - \eta \frac{\partial E(H)}{\partial (H)} \tag{2.25}$$

where η is the learning rate and the commonly used algorithm for computing the error update of the output layer to the input layer is the backpropagation algorithm.

Moreover, if the number of hidden nodes $\hat{L}$ is equal to the number of distinct training samples N ($\hat{L} = N$), matrix *H* becomes a squared matrix and could be invertible when the input weight vectors a_i and hidden biases b_i are randomly chosen. However, in most cases, the number of hidden nodes is always less than the number of distinct training samples and *H* resulted in a non-square matrix (Huang; Zhu and Siew, 2006).

Thus, this conventional gradient descent algorithm can also be substituted with different forms of optimisation techniques such as the Levenberg-Marquardt algorithm, bayesian regularisation and scaled conjugate gradient method to achieve the minimum error and parameters update of the network (Ahmad, 2005).

2.3.2 Bootstrap Aggregated Neural Network

The bootstrap aggregated neural network (BAGNET) was created to address the shortcomings of neural networks in terms of poor generalisation of unknown input data. This concept focuses on the generalisation accuracy of future predictions (i.e., predictions based on unknown data). Training individual neural networks tend to overfit leading to poor generalisation capabilities of the model. As a result, it ends up in local optima instead of global optima during process optimisation. Figure 2.8 shows the relationship between generalisation error and the training of networks. The training error can be minimised by either increasing the number of hidden neurons or running more training iterations; however, in both cases, overfitting does always occur. As the accuracy of training data estimations improves, the training error decreases and the generalisation error tends to improve (Engineers Handbook, 2006).

Moreover, the trained network may not be able to minimise the error on the training data set due to the uncontrolled excess nonlinearity in the training data. According to Noor et al. (2010), the size (number of neurons) of a neural network is a measure of its ability to represent detailed information about the data set. If the networks are large enough, a wide range of solutions can be obtained to fit the training data provided there’s minimal noise in the data set and if otherwise, there will be a lack of robustness on the individual neural networks which leads to the second form of overfitting or under-fitting.

There are quite a several techniques have been established for improving the robustness of neural networks. These include the Bayesian learning approach (MacKay, 1992), the early stopping technique (Bishop, 1995), the regularisation technique (Bishop, 1991), training with static and dynamic process data (Zhang, 2001) and a combination of several neural networks (Zhang *et al.*, 1997; Zhang, 1999; Jie, 2002; Ahmad and Zhang, 2005; Mukherjee and Zhang, 2008). Among all these techniques, the bootstrap aggregated neural network has been established to be highly effective in improving model robustness and generalisation accuracy for predictions on unseen data. According to Noor et al. (2010), there are three (3) different algorithm techniques to resample training data in a stacked neural network, namely: adaptive boosting, bootstrap resampling, and the ensemble technique. Individual neural networks may perform better in different segments of the input space

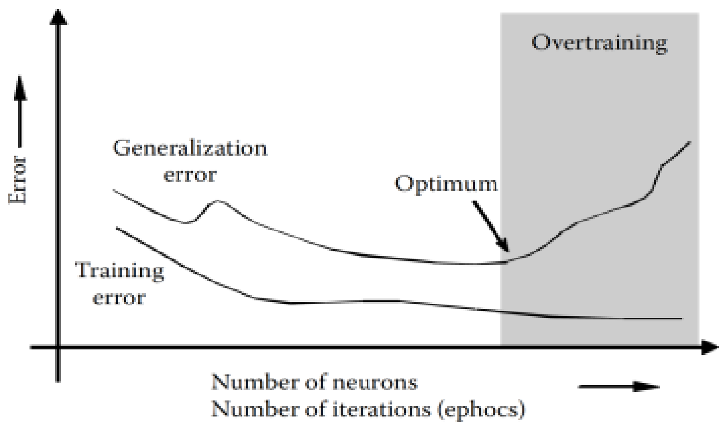


Figure 2.8 Training error and generalisation error (Baron and Zhang, 2017)

while developing neural network models. Prediction accuracy and robustness throughout the entire input space could be enhanced by combining several individual neural networks using different training data sets or different initial weights for the training of the individual networks. Moreover, instead of choosing the single best from the trained individual networks, aggregated neural networks are being used (Zhang *et al.*, 1997; Zhang *et al.*, 1998; Tian; Zhang and Morris, 2001; Mukherjee and Zhang, 2008; Zhang, 2008). Another technique of combining individual neural network is known as bootstrap resampling, where multiple neural networks data are resampled from the same original data to create different randomised data set for training and testing (Zhang *et al.*, 1998; Zhang, 1999; Baron and Zhang, 2017) and the last method, is to create a good ensemble through adjustment of the initial random weight assigned to the individual networks. Figure 2.9 shows a representation of bootstrap aggregated neural networks, where multiple network models are combined to model the same process data.

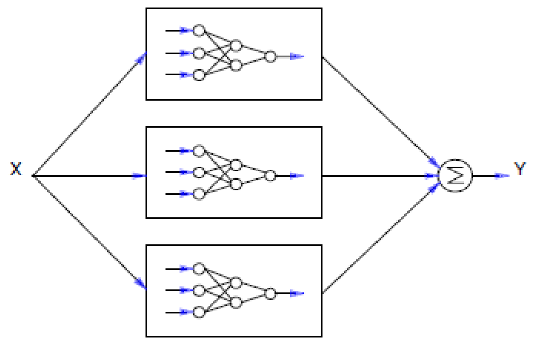


Figure 2.9 A typical representation of a bootstrap aggregated neural network (Mukherjee and Zhang, 2008)

The weighted sum of the individual/multiple neural network outputs is the overall aggregated neural network and can be represented in form of the following equation:

$$f(X) = \sum_{i=1}^{k} w_i f_i(X) \tag{2.26}$$

where $f(X)$ is the aggregated neural network predictor, $f_i(X)$ is the *ith* neural network, w_i is the aggregated weight for combining the *ith* neural network, *k* is the number of networks and X is a vector of neural network inputs. For an effective modelling performance, the stacking weights need to be accurately determined but due to the high correlations of the individual/multiple neural networks, this could be obtained using either the principal component regression (PCR) or partial least square (PLS) regression as the weighted averaging method of networks combination (Noor *et al.*, 2010).

However, the main motive behind these techniques of neural network combination is to reduce the number of shared failures or lack of robustness in the individual networks. When combining several neural networks, some may not necessarily enhance the model generalisation capability due to the repetition of the same captured information from other individual networks or networks that are sternly over-fit. Therefore, a selective combination of neural networks may be sought such as choosing the lowest sum of squared errors in a combination of individual networks suggested by Perrone and Cooper (1992). Bootstrap aggregated neural network also has the added advantage of finding the model prediction confidence bounds of the individual neural networks in BAGNET. This is achieved by finding the smallest standard errors of the predicted values among the number of neural networks and can be represented as (Mukherjee and Zhang, 2008):

$$\sigma_e = \left\{\frac{1}{k-1}\sum_{i=1}^{k}[f_i(X) - y(X)]^2\right\}^{1/2} \tag{2.27}$$

where $y(X) = \sum_{i=1}^{k} f_i(X)/k$ and k is the number of neural networks. Provided that prediction errors are normally distributed, the 95% confidence bounds can be calculated as $y(x_i;.) \pm 1.96\sigma_e$.

2.4 Extreme learning machine

Despite the successful applications of feedforward artificial neural networks in numerous fields, it has limitations of slower training convergence speed, over-fitting of training data, large numbers of learning/training steps, convergence to local optima instead of global optima and time consumption when using the gradient descent method (Huang; Zhu and Siew, 2006). These limitations of ANN become more pronounced when there are multiple hidden layers. Many researchers have been working on these shortcomings and one of the findings by Jeong and Lee (2018) suggested a way of reducing these limitations of ANN by pre-detecting the outliers from the data through unsupervised learning before training of the data set begins.

As stated earlier while describing the SLFNNs above, an extreme learning machine (ELM) was primarily proposed for SLFNNs and later used for generalising SLFNNs to reduce training time and enhance generalisation capability. In ELM, the hidden layer weights and biases are assigned randomly to overcome the slow training problem of the backpropagation training method. The gradient descent method is generally slow and required many iterative learning steps and according to Bartlett (1998), the smaller the norm of weights in a network, the better the generalisation performance of such a network. Thus, in ELM, one may not necessarily need to tune the hidden layer weights and biases which resulted in the learning technique being thousands of times faster than the traditional feedforward neural network learning algorithm. The ELM was later extended to the radial basis function case (RBF), in which the kernels were arbitrarily assigned instead of adjusted like in the case of the backpropagation method in ANN (Guang-Bin and Chee-Kheong, 2004).

Recently, ELM has been improved by various researchers for fast training of single hidden layer feedforward neural networks (SLFNNs). ELM is one of the modern learning algorithm concepts used for SLFNs in learning from data. Its structure is like traditional SLFNNs but their ways of parameter updating are different and its learning strategy is extremely fast. In ELM, the hidden layer weights are arbitrarily assigned and fixed without any iterative adjustment unlike the traditional training for SLFNs. Parameters to be learned are the connecting output weights between the hidden layer and the output layer which can be determined with a one-step regression-type approach using Moore-Penrose (MP) generalised inverse. Thus, ELM training is expressed as a linear-in-parameter model which could be solved as a linear system of equations (Huang et al., 2015).

ELM is impressively proficient, fast, has good generalisation ability and reaches global optimum with the least human interference when compared to traditional feedforward neural network learning methods. Reports on its studies have shown that ELM maintains its general approximation capability with arbitrarily generated hidden layer weights if "the activation function in the hidden layer is infinitely differentiable" and its learning algorithm could be used to train SLFNs with both non-differentiable and differentiable activation functions (Wang; Cao and Yuan, 2011).

However, as the input weights and hidden biases are randomly assigned, there may exist additional and dissatisfied selections. This makes ELM require more hidden neurons than the traditional SLFNNs and gave rise to slow response in testing data and later resulted in variant forms of ELM inventions by researchers. There is a lot of growing interest in ways of improving ELM generalisation and later extending its usage for classification and regression problems. Noteworthy theoretical validations and sensational ELM algorithms have been developed and established in a wide range of fields that ELM and its new variants such as Incremental-ELM, Bidirectional-ELM, Pruning-ELM, Adaptive-ELM, sparse-ELM, Online Sequential-ELM, and Generalized-ELM are capable if not better than ordinary ELM, precise and stress-free to implement (Huang et al., 2011).

In the use of ELM for modelling chemical processes in this research work, the PCR algorithm replaces the least square algorithm in basic ELM for obtaining the output layer weights of the ELM. This concept can solve the problem of correlation/collinearity issues among hidden neuron outputs which can be used for long-range or multi-step-ahead model prediction on a fed-batch reactor (Li *et al.*, 2017).

2.4.1 Basic ELM

As stated earlier, ELM has been proposed by various researchers for fast training of single hidden layer feedforward neural networks (SLFNs) in which the hidden layer is not neuron-like, with an output function given as:

$$f_l(x) = \sum_{i=1}^{L} \beta_i h_i(x) = h(x)\beta \tag{2.28}$$

where $\beta = [\beta_1, \dots, \beta_L]^T$ is the output weight vector between the hidden layer L nodes to the output nodes, and $h(x) = [h_1(x), \dots, h_L(x)]$ is ELM nonlinear feature mapping.

$$h_i(x) = G\big(a_i . x_j + b_i\big), \tag{2.29}$$

Equation (2.29) is equivalent to equations (2.28) and (2.19), where $G(a, b, x)$ with hidden node parameters (a, b) is a nonlinear piecewise continuous function satisfying ELM universal approximation capability theorems.

Essentially, the ELM training process involves two (2) key stages, namely: the random weight allocation, where the hidden layer is randomly initialized to allocate the input data into feature space with some nonlinear activation functions such as SVM, RBM etc. and the minimization of approximation error in the least square error form, where the connecting weights in the hidden layer and the output layer, denoted by β, are solved in the form of:

$$\min\|H\beta - T\|^2 \tag{2.30}$$

where H is the hidden layer output matrix (randomized matrix):

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_N) & \cdots & h_L(x_N) \end{bmatrix} \tag{2.31}$$

and T is the training data-target matrix:

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix} = \begin{bmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{N1} & \cdots & t_{Nm} \end{bmatrix} \tag{2.32}$$

where $\|.\|$ represents the Frobenius norm.

The optimal solution (smallest norm least-squares solution) of equation (2.22) is given by:

$$\hat{\beta} = H^{\dagger}T \tag{2.33}$$

where $H^{\dagger}$ denotes the Moore-Penrose generalised inverse of matrix H and this can also be solved with many other competing methods such as singular value decomposition (SVD), Gaussian elimination, iterative method etc.

Summarily, the basic ELM provides the following features: the ability to learn without iterative tuning of the hidden nodes, the fast learning speed, a unified learning algorithm for a generalization of networks, better generalisation performance through reaching the smallest output weights and training error and lastly, it satisfies the universal approximation capability (Huang *et al.*, 2015).

ELM has its major limitation of random allocation of weights to the hidden layers, and this makes it require a considerably large number of hidden neurons. This resulted in poor modelling capability due to the collinearity that exists within the hidden layer outputs. According to Li et al. (2017), to improve the modelling capability of the ELM, the default least square method in the ELM can be substituted with the PCR technique to solve multicollinearity challenges. However, the first two (2) terms of the principal components decomposition of the H matrix are used as a regressor in place of regressing H and T. The H matrix principal component is decomposed into the sum of rank-one matrices and represented as:

$$H = p_1 t_1^T + p_2 t_2^T + \cdots + p_m t_m^T \tag{2.34}$$

In the above equation (2.34), p_i is called the ith score vector and t_i is called the ith loading vector. The score vectors are mutually orthogonal and so are the loading vectors which are of unit length. The loading vector t_1 describes the utmost variability and the score vector p_1, known as the first principal component, represent the projection of each column in H onto t_1. In the principal component regression technique, the model output is obtained as a linear combination of the first *m* principal components of H as:

$$\hat{T} = P_m w = H t_m w \tag{2.35}$$

where *w* denotes the vector of the model parameters in terms of principal components and its least square estimate is given as:

$$w = ({P_m}^T P_m)^{-1} {P_m}^T \hat{T} = (H^T {t_m}^T H t_m)^{-1} H^T {t_m}^T \hat{T} \tag{2.36}$$

The model parameters in equation (2.33) can be calculated through PCR as given below:

$$\beta = P_m w = P_m[(H^T {t_m}^T H t_m)^{-1} H^T {t_m}^T \hat{T}] \tag{2.37}$$

In other forms, $\beta = (H^T H\,)^{-1} H^T T^T$

Thus, the $H^\dagger$ in equation (2.33) can also be solved through several techniques such as the singular value decomposition (SVD), orthogonal projection, orthogonalization and iterative method. For the orthogonal projection method to perform well, $H^T H$ need to be non-singular and $H^\dagger = (H^T H\,)^{-1} H^T$ but not in all cases will $H^T H$ be singular and in all these cases, the SVD method performs well. The data set for building a model is partitioned into training and testing data sets. PCR models are developed on training data and validated on testing data with the least testing errors as the appropriate number of principal components. Appendices A and B give details explanations of PCA and linear regression respectively.

2.4.2 Bootstrap Aggregated ELM

The bootstrap aggregated ELM has the same concept as the BAGNET described above. It involves combining multiple ELM networks to improve the prediction accuracy and generalization capability of the network on unseen data. Original training data were replicated/resampled to generate individual ELM networks and the overall stacked network output is a weighted combination of the individual ELM networks.

Summarily, for a given original training data as $(x_i, t_i) \dots (x_i, t_i)_m | x_i \in R^m,\ t_i \in R^n, i = 1, 2, \dots N$ with activation function of $g(x)$ and $\widehat{N}$ number of hidden neurons, the bootstrap aggregated ELM can be described in three (3) steps, namely:

- Apply bootstrap re-sampling to the original training data to produce n replications depending on the number of ELMs required.
- Build ELM models with the individual bootstrap replications obtained in step one above through:
  - Assign random values to the hidden layer weights w_i and bias b_i, where $i = 1, 2, \dots\ \widehat{N}$.
  - Calculate the hidden layer output matrix H and
  - Calculate the output layer weights $\beta = (H^T H\ )^{-1} H^T T^T$
- Combine the *n* numbers of the individual ELM models through the weighted average of their predictions.

According to Mukherjee and Zhang (2008), it is possible to calculate the model prediction confidence bounds from the individual predictions of the bootstrap aggregated ELM. The standard error of the *ith* predicted value is estimated as given in equation (2.27). The smallest confidence bound shows the most accurate model prediction among all.

2.5 Iterative Learning Control

Many conventional optimal control techniques have been proposed for linear time-invariant systems, which are presumed to be well-known. In most real instances, the systems to be controlled are nonlinear and the basic physical processes are unknown. This unknown situation becomes difficult when the unknown parameters of the system model change endlessly or several times during the process operation. These changes in the parameters bring about uncertainties in the model, which are difficult to control with the state-of-the-art established adaptive control techniques. Adaptive control techniques have been shown to work well for many dynamic systems with unknown fixed parameters. However, their applications may not be easy to extend to unknown time-varying parameter systems.

As an approach to solving the time-varying parameter systems highlighted above, several iterative learning control (ILC) techniques have been established. The concept of an ILC strategy was first introduced by Arimoto *et al.* (1984) for robotic manipulations and from then onward, the strategy has been in use for industrial application processes such as pharmaceutical industries or general chemical industries that readily use batch processes in the manufacturing of their products (Xu *et al.*, 1999; Ahn *et al.*, 2014; Oh and Lee, 2016). Batch chemical processes such as batch distillation operation, heat treatment processes for metallic products and batch reactors can successfully gain from this ILC strategy.

Apart from applications of ILC strategy to industrial batch process operations, it has also been applied to various processes such as computer-numerical machine control (Krishnamoorthy and Tsao, 2004), robotics (Tayebi, 2004; Wallén; Norrlöf and Gunnarsson, 2011), in injection moulding machine (Havlicek and Alleyne, 1999), induction motors, even in accelerators, free-electron laser (Rezaeizadeh and Smith, 2018), automated manufacturing plants, injection moulding and food processing industries.

The ILC control strategy, its main target is to improve the transitory tracking performance of systems that operate repeatedly over some specific time interval. Its basic principle is to update the current trajectory inputs of a system with the use of the previous/last batch calculated error of the process operation. This is done by using the tracking error of the current batch to update the manipulated input variables or trajectory sequence for the current/next batch run with the sole aim of obtaining improved tracking performance from batch to batch until the predicted output meets up or is close to the desired output. One major feature of ILC is its learning strategy, which is capable of tracking the entire reference trajectory while many other control methods only achieve asymptotic tracking in the time domain (Kuc; Lee and Nam, 1992). Instead of this, it differs from adaptive control (AC) strategies and repetitive control (RC) in that the ILC modify/updates the input trajectories while AC updates a whole system controller. As for the RC and ILC, they are similar in operation but differ as RC is applied to continuous operation while ILC is applicable to batch/discontinuous operation. Similarly, the ELM learns from historical data through training to predict future value on unseen input data. This occurs because of modification in controller parameters (output weights) rather than the control input trajectories modification by ILC. Other features that distinguish ILC from the rest of the controller are: (i) the desired/reference input trajectory is specified over a time interval and (ii) initialization of the reference input trajectories is updated and reset at each batch step of the iteration.

Over the past few years, large numbers of variant forms of ILC have been developed which makes ILC virtually applicable in all facets of control field operations. All variant forms of an ILC have their peculiar features, which makes them distinct from one another, but they all try to achieve a common task (improve tracking performance) although ways of tracking the target trajectory differ in terms of convergence, learning speed, robustness, or suitability for specific plants processes. Detail overview of many of these variant forms of ILC can be seen in the work of (Bristow; Tharayil and Alleyne, 2006; Ahn *et al.*, 2014).

2.5.1 Iterative Learning Algorithms

Generally, the main task of an ILC is for error tracking convergence and stability of the system during repetitive tracking of operation. This has attracted a lot of research interest, as there are many possible ways to track the desired trajectory of any system, which leads to various forms of ILC algorithms. However, two divisions of an ILC algorithm, namely: (i) algorithms operate without historical knowledge of the plant model (i.e., simple computation of the control updating vector by the control error). (ii) Algorithms that use the mathematical model-based representation of the plant.

According to Arimoto *et al.* (1985), there is a derivative ILC algorithm (D-type), which uses the first order derivative of the tracking error as the satisfactory learning filter and proportional gain (P-type) algorithm ILC that is simple to implement without the use of tracking error derivatives. Moreover, there were many other types of ILC algorithms mentioned earlier, such as the combined PID-type ILC proposed by (Dong-Il and Sungkwun, 1996) which can turn to P-type or D-type depending on the setting value of the gains (i.e. if the integral and the proportional or the integral and the derivative are set equals to zero). To eradicate time delay in the implementation of the ILC controller, (Barton; Lewin and Brown, 2000) reported that step-ahead prediction error should be used instead of the instantaneous error. Both (Longman, 2000; Freeman *; Lewin and Rogers, 2005; Freeman *et al.*, 2009), expand on the technique of (Barton; Lewin and Brown, 2000) of the phase-lead algorithm on successful control of non-minimum phase plant by appropriate selection of learning filter.

2.5.2 Model-based ILC Algorithms

These ILC algorithms utilise the mathematical model-based representation of the plant for transient response of tracking the performance of the repetitive system. Thus, a perfect or close-to-perfect model is required for ILC convergence and stability of the system. The ELM model prediction has been established to provide good approximation in long-range or multi-steps ahead predictions on nonlinear batch systems but there are always unavoidable disturbances in the real process such as raw material variations in composition, reactive impurities and at times fouling in reactors, which do cause model mismatches.

Although the model-based concept is useful in providing key and concealed information the non-model-based technique cannot reveal the desired process system. Phan and Frueh (1999) reported new ways by which controllers can learn (i.e., training) from plant models at each batch of process operation of the system". Lee et al., (2000) revisited the update on the model-based ILC algorithm concept (i.e., the quadratic criterion for time-varying linear systems) by generalizing quadratic criterion performance for time-varying linear constrained systems. They reported that the generalization principle could only apply to "chemical process control, where excess input trajectories are unavoidable and many process variables are subject to difficult constraints".

The general mathematical representation for a discrete LTI SISO system is given below:

$$x(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) \tag{2.38}$$

where $x(t)\, and\, y(t)$ are input and output signals respectively. The plant dynamics represented by (G) can be written in matrix form over a finite time interval $0 \leq t \leq K$ as tends to $K < \infty$.

$$G = \begin{bmatrix} CB & \cdots & 0 \\ \vdots & \ddots & \vdots \\ CA^{K-1} & \cdots & CB \end{bmatrix} \tag{2.39}$$

where A, B, and C represent the Markov parameters of the input and output systems represented above. Thus, the classical ILC update algorithm is given as:

$$U_{K+1} = U_K + Le_K \tag{2.40}$$

where U_{K+1} is the next batch input trajectory update, U_K is the current input trajectory, L is called the learning filter, which penalises the error signal to the input update. Equation (2.40) above can be re-written as:

$$\Delta_{U_{k+1}} = U_{k+1} - U_k$$

$$\Delta_{U_{k+1}} = L_k e_k \tag{2.41}$$

where $e_k(t) = \overline{y}_k(t) - \hat{y}_k(t)$ and $\overline{y}_k$ is the desired output trajectory with $\hat{y}_k$ as the predicted/measured output trajectory. According to (Barton; Lewin and Brown, 2000), equation (2.40) can also be re-written in the time domain as:

$$U_{K+1}(t) = U_K(t) + L_K(t)\sum_{i=1}^{K+1} e_i(t) \qquad (2.42)$$

The learning filter L_k in equation (2.42) is the diagonal learning gain matrix that ensures convergence of the next batch error, $e_{k+1} = 0$ (Ahn; Chen and Moore, 2007).

Many researchers have shown that the learning filter can be obtained as the ILC model inverse (i.e., $L_k = G^{-1}$), where G represents the matrix form of the relationship between the output trajectory and the input trajectories for the entire batch run of the process.

Sogo *et al.* (2000) reported a simple way of achieving stable model based ILC inversion to avoid the conventional casual ILC model inversion. They explained further, by saying "inversion of non-minimum phase plant will lead to an unstable system". This implies, at least a single pole will be found on the right half of the complex plane. Moreover, (Lee; Lee and Kim, 2000) stated that computation of the ILC model inverse on the real plant will always contain differentiators (if at all will exist) in continuous time case which becomes very sensitive to components of modelling errors of the system. In short, ILC model inversion cannot be used directly without the inclusion of a general objective function for non-square MIMO processes and other notable constraints peculiar to physical or safety considerations.

2.5.3 Optimal Model-based ILC Algorithm

The optimisation is the process of searching for the best solution among the feasible solutions. Optimal model based ILC was proposed by many researchers but differs in format and strategies of implementation. Its concept is to accommodate the shortcomings of ordinary model-based ILC such as the ability to accept non-square MIMO systems, relaxed zero-tracking error constraints through minimum likely error with the help of the least-square principle (Togai and Yamano, 1985; Moore, 1999; Lee; Lee and Kim, 2000; Norrlöf and Gunnarsson, 2002).

There are two categories of optimisation according to the variables involved being in either continuous or discrete form. (Tao;Kosut and Aral, 1994) and many other researchers have proposed discrete-time learning control strategies, which take into consideration the simplicity of implementation, error convergence, noise sensitivity and improper learning constraints with input penalisation terms. Its mathematical representation is given as:

$$\min J_K = \|e_K\|_Q^2 + \|u_K\|_R^2 \text{ as } k \to \infty \tag{2.43}$$

A similar optimisation concept has also been considered by (Amann *et al.*, 1996) and (Lee; Lee and Kim, 2000) but as a continuous time domain and change in input, a penalisation term was proposed. Therefore, equation (2.43) can be re-written as:

$$\min J_K = \|e_K\|_Q^2 + \|u_K - u_{K-1}\|_R^2 \tag{2.44}$$

thus, the next batch (k+1) of equation (2.44) will be of the form:

$$\min J_{K+1}(u_{K+1}) = \|e_K\|_Q^2 + \|u_{K+1} - u_K\|_R^2 \tag{2.45}$$

where Q and R are the weight matrices, selected as $Q = \lambda_q . I_M$ and $R = \lambda_r . I_N$ that should be symmetric and positive at all time (t). The minimisation of either equation (2.43) or (2.44) ensures minimum possible error and prevents excessive control to be applied to the input trajectories. The error at batch (k+1) is given as

$$e_{k+1}(t) = \overline{y}_k\,(t) - \hat{y}_{k+1}(t) \tag{2.46}$$

where $\hat{y}_{k+1}(t)$ is the model prediction from an ELM at the (k+1)th batch. This is approximated with the use of first-order Taylor's series expansion approximation given as:

$$\hat{y}_{k+1}(t) = G.u_{K+1} \tag{2.47}$$

where G is the linear operator of the system dynamics relating the output trajectory to the input trajectories. (Amann; Owens and Rogers, 1996) and Lee *et al.* (2000) respectively reported the derivation for the input update as:

$$u_{K+1} = u_K + R^{-1}G^TQe_{k+1} \tag{2.48}$$

and

$$u_{K+1} = u_K + (G^TQG + R)^{-1}G^TQe_{k+1} \tag{2.49}$$

Equation (2.41) was derived from $\frac{\delta J_k}{\delta u_k} = 0$, then equation (2.48) and equation (2.49) are quite similar concerning the learning filter L_k to the transpose of the linear operator G^T and the weighing matrix *R* and *Q*, given as:

$$L_k = R^{-1}G^TQ \tag{2.50}$$

Amann *et al.* (1998) later extend the work of Amann and Owens (1994) and Amann *et al.* (1996) to improve on the shortcomings of their discoveries. One of his findings stated that the plant model's non-minimum phase zeros are linked to a specific direction in the input space where error convergence is delayed. Their form of optimal model-based ILC incorporates future predicted errors and is of the form:

$$\min J_{K+1,N}(u_{K+1}) = \sum_{i=1}^{N} \lambda^{i-1}(\|e_K\|_Q^2 + \|u_{K+1} - u_K\|_R^2) \tag{2.51}$$

Equation (2.42) and equation (2.49) are quite similar; the only difference is the weight parameter that captured the unseen predicted errors and updates/changes on inputs at each batch of iterations.

2.5.4 Linearized perturbation model in ILC

Most (if not virtually all) batch processes are nonlinear, but in ILC, the linear model is required for the control of the systems. Therefore, it is necessary to seek the departure of variables from their standardized trajectories. Given the input and output variables ($X\ and\ Y$), as functions of time, then their perturbation variables, $[\overline{X}, \overline{Y}]$, will also be a function of time.

Xiong and Zhang (2003) reported that the linearization of the non-linear batch process around the nominal trajectory is done by removing the time-varying nominal trajectories away from the desired trajectories operation. This idea will ensure the removal of the bias effect from each batch iteration through an incremental update of the variables of the system. The linearized perturbation model can be represented as:

$$\Delta\overline{Y}_{K+1} = G_{K+1}\Delta\overline{X}_{K+1} \tag{2.52}$$

where,

$\Delta\overline{Y}_{K+1} = \overline{Y}_{K+1}$- $\overline{Y}_K$ and $\Delta\overline{X}_{K+1} = \overline{X}_{K+1}$- $\overline{X}_K$, then

G_{K+1} is the linear operator that can be obtained through the linearized perturbation concept and varies from batch to batch as the input trajectories vary at each step of iteration of the process operation. Provided there are sufficient historical batch runs, G_{K+1} can be obtained through any regression method such as partial least square regression (PLS), multiple linear regression (MLR), or principal component regression (PCR). Recall that a batch process operation is always modelled in the form of a dynamic model just because the process of operation is in a transient state and therefore no feasible steady state at any point in time, but steady state operation is quite easy to work with. However, if the whole batch input and output trajectories are considered, it is possible to establish a static model linking the whole batch input and output trajectories. The steady-state input control policy relating to the output quality over the whole batch duration is of the form:

$$Y_K = F(X_K) + V_K \tag{2.53}$$

where F(.) represents the nonlinear dynamic state function between $X_K(t)$ and $Y_K(t)$ at different batch times and V_K is the vector of noises. Therefore, to obtain the linearized model concerning X_K around nominal trajectories X_S and Y_S, the following equation can be obtained.

$$Y_K = Y_S + \frac{\partial F(X_K)}{\partial (U_K)} \mathrm{I}_{X_S}(X_K - X_S) + v_K + w_K \tag{2.54}$$

where w_K is the sequence of model errors resulting from the neglect of other higher terms, v_K is the noise with unexpected disturbances that may arise and the linearized model G_K is

$$G_K = \frac{\partial F(X_K)}{\partial (U_K)} \mathrm{I}_{X_S} \tag{2.55}$$

The G_{K+1} in equation (2.52) and $G_K$in equation (2.55) is the same, only that the former is representing the updated/next batch while the latter is representing the current batch. Its structure is restricted to a lower block-triangular form to prevent causality (due to correlated data) given as:

$$G_K = \begin{bmatrix} g_{10}^k & 0 & \cdots & 0 \\ g_{20}^k & g_{21}^k & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{N0}^k & g_{N1}^k & \cdots & g_{NN-1}^k \end{bmatrix} \tag{2.56}$$

For every batch process operation, optimisation of product quality at the end of the operation is the major goal whereby the control policies are the determinant of optimisation. Moreover, the control actions at each batch iteration could be correlated, in such a situation, a regression method capable of removing the causalities will be used in obtaining the linearized models (Jie, 2003; Xiong and Zhang, 2004; Xiong *et al.*, 2004b; Xiong *et al.*, 2007).

2.6 Recursive Least Square Technique

The recursive least square (RLS) technique is a form of adaptive filter algorithm concept of online parameter estimation, which estimates a plant model by repeatedly updating the model parameters that minimise the weighted linear least square cost function of that model. This is obtained by recursively updating the model parameters based on the error difference between desired model output and the model prediction until the desired model is realized.

Model parameters are usually time-varying in many practical process systems where there can be two cases, namely: the parameters can suddenly change or sometimes change with time slowly as the process operation progresses. In either case, monitoring solutions are sought. For the former case, covariance resetting is the solution for abrupt changes while for the latter case; the forgetting factor needs to be included to correct the slow changes with time in the parameter estimation of that process (Wigren, 1993).

In the recursive algorithm technique, the parameter estimation of the next batch is obtained by using the current and the previous parameter estimates with correction terms in the computation of the predicted model. The correction term is proportional to the deviation of the prediction model from the desired model.

There are two categories of a recursive algorithm for online parameter estimation of any process operation (i.e., the finite-history and infinite-history algorithms) and the offline or batch method of parameter estimation. For the finite-history algorithms, the algorithms target to minimise the error between the desired and the predicted outputs over a specific number of past time steps while the infinite-history algorithms also object to minimise the error between the desired and the predicted outputs over the whole process operation from the start until the end of the operation (The Math Works, Inc., 2017).

2.6.1 Recursive Least Square Estimation

Given the following system of equations as the input samples and desired output samples respectively as: $\{ X(1), X(2), \dots X(t)\}$ and $\{ d(1), d(2), \dots d(t)\}$

$$y(t) = a_1 X(t-1) + a_2 X(t-2) + \cdots + a_k X(t-M)$$

$$= [X(t-1)\ X(t-2)\ \ldots\ \ X(t-M)]\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix}$$

$$y(t) = \sum_{k=0}^{M} a_k X(t-k) = a_k^T X(t) \text{ where k= 1, 2, 3,..., M} \qquad (2.57)$$

From equation (2.57), a_k^T is termed as the parameter vector to be estimated and $X(t)$ as the regressor. Therefore, to find the parameters estimates at time t, the sum of the square error between the predicted and the desired models needs to be minimized, with the error signal at each iteration of time (t) given as:

$$e(t) = d(t) - y(t) = d(t) - a_k^T X(t) \qquad (2.58)$$

The error signal can be represented as the negative feedback loop in the diagram below:

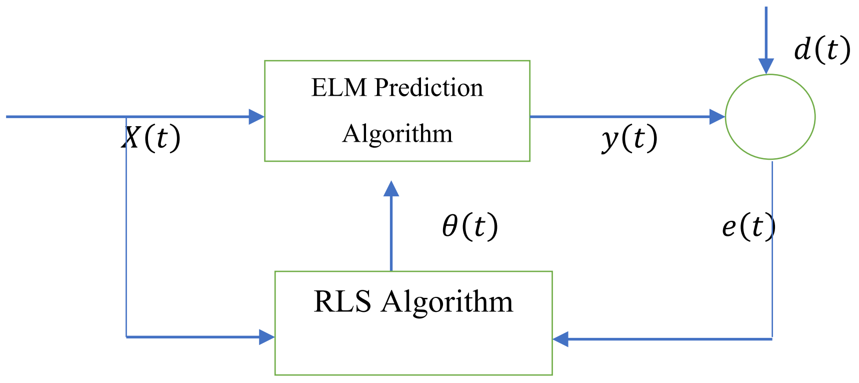


Figure 2.10 Schematic representation of ELM with RLS

The basic goal in the RLS technique is to minimize a cost function F through a selection of appropriate parameter estimates of the model parameters and thereby updating the parameters at each iteration, i.e., at each sampling time (t) throughout the operation of the batch process. The parameter estimation in equation (2.57) can be solved with the method of ordinary least squares (Yuxin *et al.*, 2004). This results in finding the optimal parameter estimates of $a^T(t)$ in equation (2.58) which resulted to minimising the following:

$$J(e_t) = \sum_{i=0}^{t} \lambda^{t-i}(d(i) - y(i))^2 \qquad (2.59)$$

The equation above is the weighted least square error function. It is a sum of the weighted error terms and therefore depends on the forgetting factor λ, which penalises the previous (historical) measurement exponentially in general. The forgetting factor typically has a positive value between the range of 0.98 and 0.995 and it is significant for parameter estimation such that if systems remain constant over time (T_0), λ will be chosen as:

$$T_0 = \frac{1}{1-\lambda} \qquad (2.60)$$

Setting $\lambda = 1$, signifies that there's no forgetting factor and a constant (time-invariant) coefficient will be estimated, but if $\lambda < 1$, this signifies that less weight is given to historical measurements and can as well be forgotten during parameter estimation (Bai *et al.*, 2005). Moreover, the lesser the value of λ, the faster the detail information captured from historical data will be forgotten and hence the λ is termed as the forgotten factor.

2.6.2 Recursive Least Square Derivation

The minimisation of equation (2.59) leads to the normal equations given by (Gaensler and Benesty, 2004):

$$\mathrm{R}_t . \mathrm{y}_t = \mathrm{P}_t \qquad (2.61)$$

Where R_t and P_t is an estimate of the input signal covariance matrix and an estimate of the cross-correlation vector between input and output respectively and are represented as:

$$\mathrm{R}_t = \sum_{i=0}^{M} \lambda^{M-i} X(i).X(i)^T \text{ and}$$

$$\mathrm{P}_t = \sum_{i=0}^{t} \lambda^{M-i} X(i).d(i)$$

Therefore, equation (2.61) can be re-written as

$$\mathrm{y}_t = \mathrm{R}_t^{-1}.\mathrm{P}_t \qquad (2.62)$$

The RLS algorithm parameters estimation y_t is solved through iterative steps of the process by incrementally updating the equation (2.62) and obtaining the inverse of R_t the matrix in solving for the parameter estimation of the filter weight in equation (2.62), the matrix inversion formula is employed to avoid the rigorous matrix inversion computation (Chi Sing *et al.*, 1996).

The matrix inversion formula stated that if A and B are $M \times M$ positive definite matrices, D is a $N \times N$ matrix, and C is $M \times N$ matrix, then they are related as follows:

$$A = B^{-1} + CD^{-1}C^{T}$$

$$\text{Then,} \quad A^{-1} = B - BC(D + C^{T}BC)^{-1}C^{T}B$$

with matrix inversion formula on equation (2.62), the $\mathrm{R}_t{}^{-1}$ give rise to:

$$\mathrm{R}_t{}^{-1} = \lambda^{-1}[\mathrm{R}_{t-1}{}^{-1} - \mathrm{K}_t.X_t^T.\mathrm{R}_{t-1}{}^{-1}] \quad (2.63)$$

Therefore, the parameter estimate update in equation (2.62) is of the form:

$$y_{t+1} = y_t + K_t.e_t \quad (2.64)$$

where, e_t is termed as the prediction error. Given as:

$$e_t = d_{t+1} - X_{t+1}^T.y_t \quad (2.65)$$

K_t is termed as the filter gain, given as:

$$K_t = P_{t-1}.X_t.Z_t \quad (2.66)$$

Where Z_t is the error dispersion and it is given as:

$$Z_t = \left(\lambda + X_t.X_t{}^{-1}.P_{t-1}\right)^{-1} \quad (2.67)$$

$$P_t = P_{t-1}.\,\lambda^{-1}[I - K_t.X_t] \quad (2.68)$$

In the RLS algorithm stated above from equation (2.64) to equation (2.68), equations (2.68) is termed as the dispersion estimate and equation (2.64) is termed the weighted update, which were the major equations for parameter estimation of the initial output weight predicted from the ELM. The output weight prediction from the ELM will be used to initialize the weight update for the parameter estimation and $P(t)$ will be initialized as:

$$P_t = P(0) = \delta I \quad (2.69)$$

Thus P_t is proportional to the covariance matrix of the parameters $y(t)$ but the initial values of $y(0)$ is uncertain and then a very high covariance matrix of the parameters is estimated as:

$$\delta > 100\sigma^2 \quad (2.70)$$

According to (Chen; Billings and Grant, 1990), the recommended value for δ is given in above equation and for large data, the initialization step does not really matter since the exponential forgetting factor will take care of it.

In summary, at the end of this chapter, I have been able to give some basic background knowledge on different forms of modelling and process controlling techniques. All these techniques will be utilized to model and optimize case studies in the rest of the chapters.

CHAPTER THREE

MODELLING AND OPTIMISATION OF FED-BATCH PROCESSES USING DIFFERENT MODELLING TECHNIQUES

3.1 Introduction

Fed-batch processes are commonly used for the manufacturing of high value-added products such as specialty chemicals and pharmaceuticals (Ruppen;Benthack and Bonvin, 1995; Bonvin, 1998). Both fed batch and batch processes operate in similar pattern only that the intermittent addition of the reactants is not required in the case of batch operation.

These process operations possess great benefit in many manufacturing industries as they help to ascertain controlled situation during the progress of a reaction whereby operating variables, such as the feed-rate, temperature, pressure, and agitation rate are varied according to a specified dynamic or steady trajectory. According to (Xiong and Zhang, 2005), the significant requirement of batch process optimization is the accurate mathematical representation of the process capable of providing accurate and reliable long range predictions. Process models can be generally classified into three forms, namely: mechanistic models, data-driven models, and hybrid models. Mechanistic models are developed based on the first principles governing the processes such as mass balance, energy balance, and reaction kinetics. Due to the processing of multiple materials, batch-to-batch variation and complexity involved (e.g., large numbers of reactions), first principle mechanistic models for batch processes are usually difficult to obtain. Developing mechanistic models usually requires significant amount of time and effort, which may not be feasible for batch processes where frequent changes in product specifications occur and a type of product is usually manufactured for a limited time in response to the dynamic market demand. Data-driven modelling can be a very useful alternative in this case. With the development and progress in research, data-driven modelling is becoming the most widely used method in modelling and analyses of batch/fed-batch process operations.

Moreover, to obtain good control policy for a batch or fed-batch process, a reliable model capable of providing accurate predictions is required and some examples of batch processes includes polymerisation reaction, in which the mechanistic models are usually complicated and difficult to implement for both offline and on-line control system. However, low-quality, or inadequate operational data, as well as variability in process conditions frequently cause a mismatch between the predicted model and the actual plant model. As a result, the optimal control profile obtained from the model may not be optimal when applied to the actual plant but due to the repetitive nature of batch operation, this can be improved using the previous and current batch knowledge to improve the next batch operation (Liu *et al.*, 2005). This is termed as batch-to-batch optimisation and many researchers have worked on this by trying different modelling techniques.

To employ the concept of data-driven in modelling the nonlinear characteristics relationship between the manipulated input variable and the output product quality, substantial amount of historical (previous) process operational data is necessary for an accurate model prediction of final product quality. Different modelling techniques such as ELM, bootstrap aggregated ELM (BAGNET) and iterative learning control (ILC) will be used to model and control an isothermal fed-batch and baker's yeast fermentation process.

The remaining part of this chapter is organised as follows. Section 3.2 gives the general background knowledge on both the isothermal fed-batch reactor and the baker's yeast fermentation process while the rest of the sections talks about modelling of the case studies with different computational techniques such as ELM, BA-ELM, and an ILC.

3.2 Case studies

Two case studies are used in this thesis. One is a fed-batch reactor and the other one is fed-batch fermentation process. Simulations of the two case studies based on detailed mechanistic models are used here to represent the actual industrial processes. These two case studies have been widely used in the batch process optimisation and control area.

3.2.1 An Isothermal Fed-Batch Reactor Process

The isothermal fed-batch reactor is taken from (Tian;Zhang and Morris, 2001) with the following reaction system kinetics:

$$A + B \overset{K_1}{\rightarrow} C \qquad 3.18$$

$$B + B \overset{K_2}{\rightarrow} D \qquad 3.19$$

The above reactions are conducted in an isothermal fed-batch reactor. In this process, reactant A and B are raw materials, C is the desired product, and D is the undesired by-product. The objective of the process operation is to produce maximum amount of desired product *C* while minimizing the amount of the undesired by-product *D* at the end of a batch with a specified final time t_f by feeding reactant *B* in an optimal way. Adding all the reactant B at the beginning of the batch will lead to more side reactions (3.19) which will eventually lead to yield of the undesired by-product *D*. To keep this undesired by-product *D* as low as possible, the reactor is operated in fed-batch mode, where *B* is added in a feed stream at a constant concentration. The following mechanistic model equations were derived based on material balances and reaction kinetics.

$$\frac{dC_A}{dt} = K_1 C_A C_B - \frac{C_A}{V} u \quad (3.19)$$

$$\frac{dC_B}{dt} = K_1 C_A C_B - 2K_2 {C_B}^2 + \frac{b_{feed} - C_B}{V} u \quad (3.20)$$

$$\frac{dC_C}{dt} = K_1 C_A C_B - \frac{C_C}{V} u \quad (3.21)$$

$$\frac{dC_D}{dt} = 2K_2 {C_B}^2 - \frac{C_D}{V} u \quad (3.22)$$

$$\frac{dV}{dt} = u \quad (3.23)$$

where C_A, C_B, C_C, C_D denote the concentrations of *A*, *B*, *C*, and *D* respectively*, V* is the current reaction volume, *u* is the reactant feed rate, and K_1 and K_2 are the reaction rate constants. The operation conditions and other physical properties of the fed-batch reactor is given in Table 3.1:

Table 3.1 Operation conditions and physical properties of the fed-batch reactor.

Parameters	Value
b_{feed}	$0.2\ m^3.min^{-1}$
K_1 and K_2	$0.5\ L.moles^{-1}.min^{-1}$
$C_A(0)$	$0.2\ mol.L^{-1}$
$C_B(0) = C_D(0) = C_C(0)$	$0\ mol.L^{-1}$
$V(0)$	$0.5\ m^3$
Batch time (t_f)	$120\ min$

3.2.2 Process Model for Baker's Yeast Fermentation Process

Fermentation is a biochemical process that converts glucose extract to an alcohols and organic acids in the presence (aerobic) or an absence (anaerobic) of oxygen. Mostly in biotechnology and food industries, fermentation process plays vital role in the production of baker's yeast also known as *Saccharomyces cerevisiae*, alcoholic beverages, organic solvents and antibiotics or biopolymers. The importance of these products in food industries cannot be over-emphasised which leads to their highly competitive market demands and hence, the industrial production of baker's yeast and ethanol.

Fed-batch processes operation are commonly seen in many fermentation industries. The mode of operation allows the substrate concentration and all other operating variables (such as the feed-rate, temperature, pressure, and agitation rate) to be varied and monitored during the progress of production to obtain maximum and desired biomass yield at the end of production. According to (Xiong and Zhang, 2005), the significant requirement of batch process modelling lies in the accurate mathematical representation of the process capable of providing accurate and reliable model predictions but due to the complexity nature of the fermentation processes such as multiple process variables, batch-to-batch variation and non-linear and dynamic mathematical equations of the process, accurate kinetic model representation is difficult and challenging to obtain.

The new trending concept of machine learning coupled with the data science analysis in research developments has become widely accepted concept used in modelling and analyses of any form of process operations. Thus, the data-driven concept utilizes statistical theories in establishing the process performances, monitoring the progress of preset conditions of operation for optimization purpose and generalization capabilities in predicting the unforeseen circumstances. In this concept, detailed knowledge of the process operation is not necessary but past historical data of the process operation is required. As much as the historical data is large enough, data-driven model of the process with optimization can be establish with machine learning and statistical theories conceptualization (Liu *et al.*, 2019).

3.2.3 Kinetic Model

According to (Yüzgeç;Türker and Hocalar, 2009), the yeast cell kinetic model of the baker's yeast fermentation metabolism process is based on the hypothesis developed by (Sonnleitner and Käppeli, 1986), which was later updated with glucose uptake and oxidative capacity terms by (Karakuzu, 2003). The cell kinetic model equations are given as follows:

The glucose uptake: $Q_s = Q_{s,max}\frac{C_s}{K_{s+}C_s}\left(1 - e^{-t/t_d}\right)$ (3.24)

Oxidative capacity: $Q_{o,lim} = Q_{o,max}\frac{C_o}{K_{o+}C_o}\frac{K_i}{K_{i+}C_e}$ (3.25)

Specific growth rate limit: $Q_{s,lim} = \frac{\mu_{cr}}{Y_{x/s}^{ox}}$ (3.26)

Oxidative glucose metabolism: $Q_{s,ox} = min\begin{pmatrix} Q_s \\ Q_{s,lim} \\ Q_{o,lim}/Y_{o/s} \end{pmatrix}$ (3.27)

Reductive glucose metabolism: $Q_{s,red} = Q_s - Q_{s,ox}$ (3.28)

Ethanol uptake rate: $Q_{e,up} = Q_{e,max}\frac{C_e}{K_{e+}C_e}\frac{K_i}{K_{i+}C_e}$ (3.29)

Oxidative ethanol metabolism: $Q_{e,ox} = min\begin{pmatrix} Q_{e,up} \\ (Q_{o,lim} - Q_{s,ox}Y_{\frac{o}{s}})Y_{e/o} \end{pmatrix}$ (3.30)

Ethanol production rate: $Q_{e,pr} = Q_{s,red}Y_{e/s}$ (3.31)

Total specific growth rate: $\mu = Q_{s,ox}Y_{x/s}^{ox} + Q_{s,red}Y_{x/s}^{red} + Q_{e,ox}Y_{x/e}$ (3.32)

Oxygen consumption rate: $Q_o = Q_{s,ox}Y_{o/s} + Q_{e,ox}Y_{o/e}$ (3.33)

Carbon (IV) oxide production rate: $Q_c = Q_{s,ox}Y_{c/s}^{ox} + Q_{s,red}Y_{c/s}^{red} + Q_{e,ox}Y_{c/e}$

(3.34)

Respiratory quotient: $RQ = Q_c/Q_o$ (3.35)

3.2.4 Reactor Dynamic Model

This is based on mass balance equations describing the glucose, biomass, oxygen and ethanol concentrations in the fermenter as follows (Yüzgeç;Türker and Hocalar, 2009):

For substrate concentration rate:

$$\frac{dC_s}{dt} = \frac{F}{V}(S_o - C_s) - \left(\frac{\mu}{Y_{x/s}^{ox}} + \frac{Q_{e,pr}}{Y_{e/s}} + Q_m\right) C_x \tag{3.36}$$

For oxygen concentration rate:

$$\frac{dC_o}{dt} = -Q_o C_x + K_L a_o (C_0^* - C_0) - \frac{F}{V} C_0 \tag{3.37}$$

For ethanol concentration rate:

$$\frac{dC_e}{dt} = \left(Q_{e,pr} - Q_{e,ox}\right) C_x - \frac{F}{V} C_e \tag{3.38}$$

For biomass: $\frac{dC_x}{dt} = \mu C_x \frac{F}{V} C_x$ (3.39)

Volume: $\frac{dV}{dt} = F$ (3.40)

Volumetric mass transfer: $K_L a_o = 113\left(\frac{F_a}{A_R}\right)^{0.25}$ (3.41)

where F_a is the air feedrate and A_R is the cross-sectional area of the fermentor? Table 1 shows the values of the other model parameters from equation (3.24) to equation (3.41) as stated in (Yüzgeç;Türker and Hocalar, 2009). Other model parameter values are given in Table 3.2 and a typical example of a computer control fermentor is given in Figure 3.1

3.3 Modelling of an Isothermal Fed-Batch Process Using Different Computational Intelligence Techniques

3.3.1 Dynamic Modelling of Fed-Batch Reactor Process using ELM

Based on the reaction kinetics, material balances in equation (3.19) to (3.23) and the process operational conditions in Table 3.1, one hundred (100) batches of process operations under different feeding profiles were simulated as inputs historical process data sets. These 100 feeding rate profiles were generated by adding random perturbations with normal distribution and zero mean to some base profiles. Each process batch time is 120 minutes, with 12 minutes sampling time at 10 equal intervals and the simulation of the fed-batch reactor process solved using the ODE 45 solver in MATLAB R2020a. The feed rate is constant within each interval. Thus, a feeding profile is a vector of 10 elements. The feeding rate constraint is given as: $0 < u_i < 0.01\ (i = 1, 2, \ldots, N)$ and the nonlinear dynamic modelling is of the form:

$$y(t) = f[\,u(t-1), u(t-2), \ldots u(t-d_i), y(t-1), y(t-2), \ldots, y(t-d_o)] \tag{3.42}$$

where y is the process output variables (C_C and C_D), u is the process inputs variables (the feeding rate profile generated), $f[\,]$ is the nonlinear function represented by an ELM network, t represents the discrete time, d_o and d_i represents the time lags in the model output and input respectively, and depend on the chosen number of batches considered for process model. Equation (3.42) is called a nonlinear autoregressive with exogenous (external) input model and it was implemented by an ELM, where 20 hidden neurons was used for building the process model. This is used in predicting future values of a time-series $y(t)$ from past values of its time-series and past values of a second time-series $u(t)$. Output $y(t)$ is regressed on its previous output signal value and previous independent input signal value $u(t)$.

Table 3.2 Model parameters value for the fermentation process.

Parameters	Values	Parameters	Values
K_e-Saturation constant (eth)	0.1 g/L	$Y_{c/e}$	0.6450 g/g
K_i-Inhibition constant	3.5 g/L	$Y_{c/s}^{red}$	0.5744 g/g
K_o-Saturation constant (oxy)	9.6E-5 g/L	$Y_{c/s}^{ox}$	0.4620 g/g
K_s-Saturation constant (sub)	0.612 g/L	$Q_{e,max}$	0.2380 g/gh
$Y_{x/s}^{ox}$- yield component (ox)	0.5850 g/g	$Q_{s,max}$	2.9430 g/gh
$Y_{x/s}^{red}$- yield component (red)	0.0500 g/g	$Q_{o,max}$	0.2550 g/gh
$Y_{e/s}$	0.4859 g/g	Q_m	0.0300 g/gh
$Y_{o/s}$	0.3857 g/g	μ_{cr}	0.2100 /h
$Y_{o/e}$	0.8904 g/g	C_0^*	0.0060 g/L
$Y_{e/o}$	1.1236 g/g	S_o	325 g/L
$Y_{x/e}$	0.7187 g/g	A_R	12.56 m^2

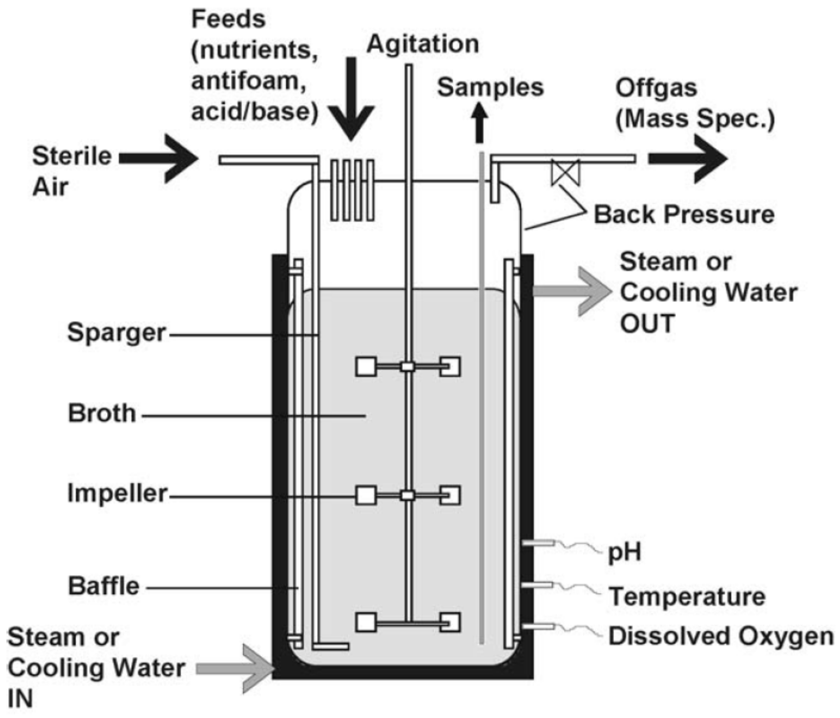


Figure 3.1 A typical computer controlled fermentor (Alford, 2006)

The batch simulation of each feeding rate profile used end of the batch process output variables in building the historical target process data for the ELM model network. The historical input and output process data are given as:

$$U = \begin{pmatrix} Uk_1^T \\ \vdots \\ Uk_{100}^T \end{pmatrix}, \; Y = \begin{pmatrix} {Y_i}_1^T \\ \vdots \\ {Y_i}_{100}^T \end{pmatrix} \; and \; Y_i = \begin{pmatrix} y_i^C(t_f) \\ y_i^D(t_f) \end{pmatrix} \tag{3.43}$$

While building the ELM model, the data set was pre-processed by scaling to zero mean and unit variance to work easily with the data magnitudes and units. The data samples were divided into 70% of training and testing and 30% for validation data set and the ELM model was developed using the training and testing data sets as in form of equation (3.42).

The developed ELM model is used for obtaining both one step ahead and multi-step ahead predictions. For the calculation of multi-step ahead predictions, the network output is fed back repeatedly to the network input via one or more-time delay units. To evaluate the performance of this ELM dynamic process modelling, performance on one of the unseen validation batches is shown in detail in the next subsection.

In this study of an isothermal fed-batch reactor process, the performance of ELM modelling predictions in terms of one-step-ahead and multi-step-ahead predictions for both C_C and C_D are compared with their actual values. Many researchers have reported and established ELM models which are fast in learning and reaches global optimum with least human intervention when compared to SLFN network.

Hence, there is a need to see the feasibility of the ELM model predictions on both short- and long-term model prediction on an isothermal fed-batch case study.

The root mean square error (RMSE) is used to check the prediction performance on both C_C and C_D. The RMSE is the root of the mean squared errors between the predicted and the actual/observed values. The larger the values of the RMSE, the worse the ELM model fits the isothermal fed-batch reactor data. Conversely, the smaller the RMSE, the better the ELM model prediction of the data. Figures 3.1, 3.2, 3.3 and 3.4 show the one-step and multi-step ahead ELM model prediction on first batch of the simulated validation data for both C_C and C_D respectively. The RMSE performance values of the ELM model predictions on any of the validation data batches is given in Table 3.3.

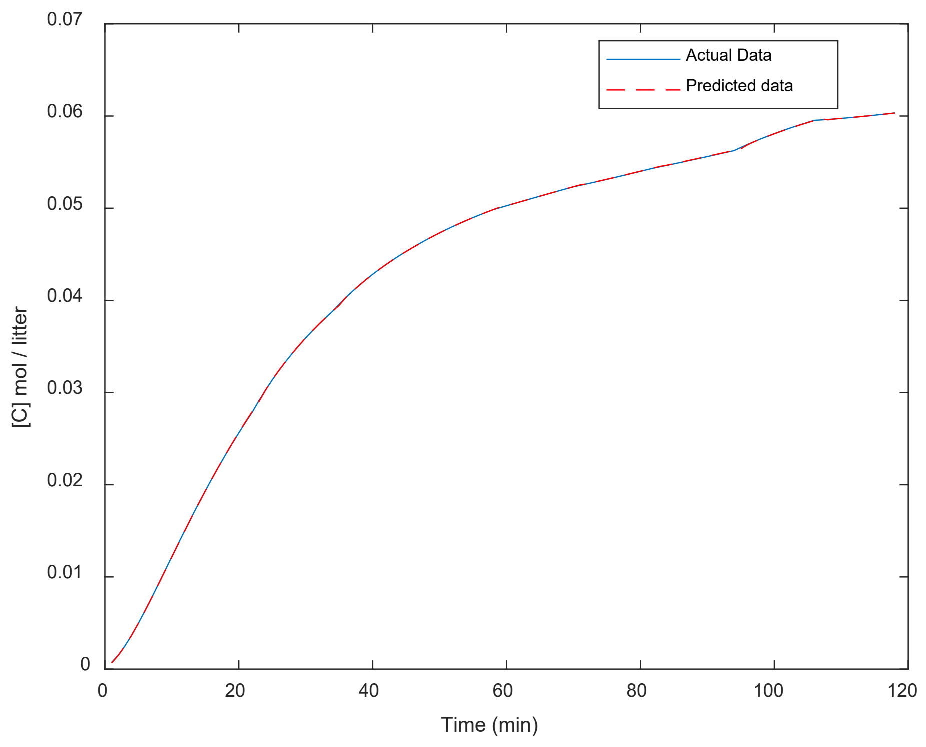


Figure 3.1 One-step ahead ELM model prediction on C_C

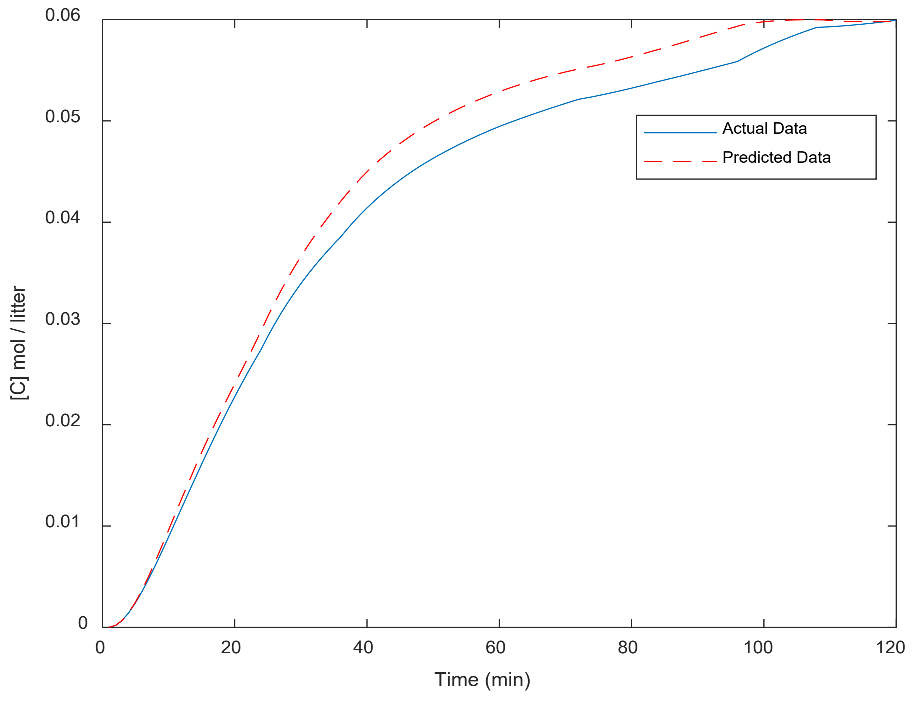


Figure 3.2 Multi-step ahead ELM model prediction on C_C

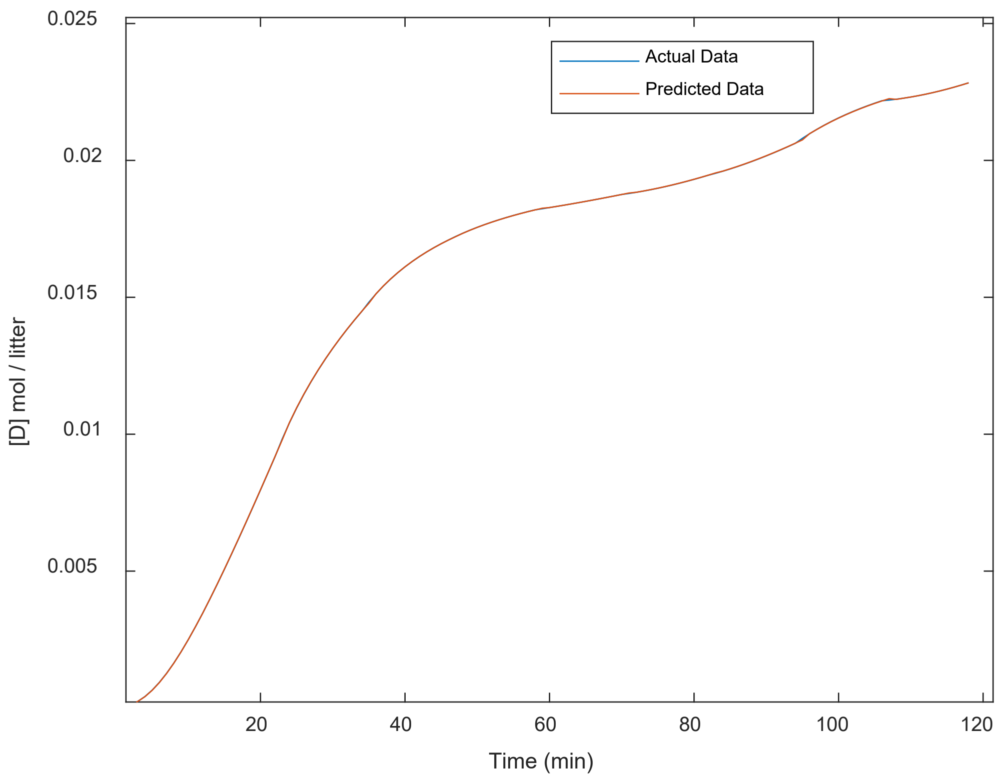


Figure 3.3 One-step ahead ELM model prediction on C_D

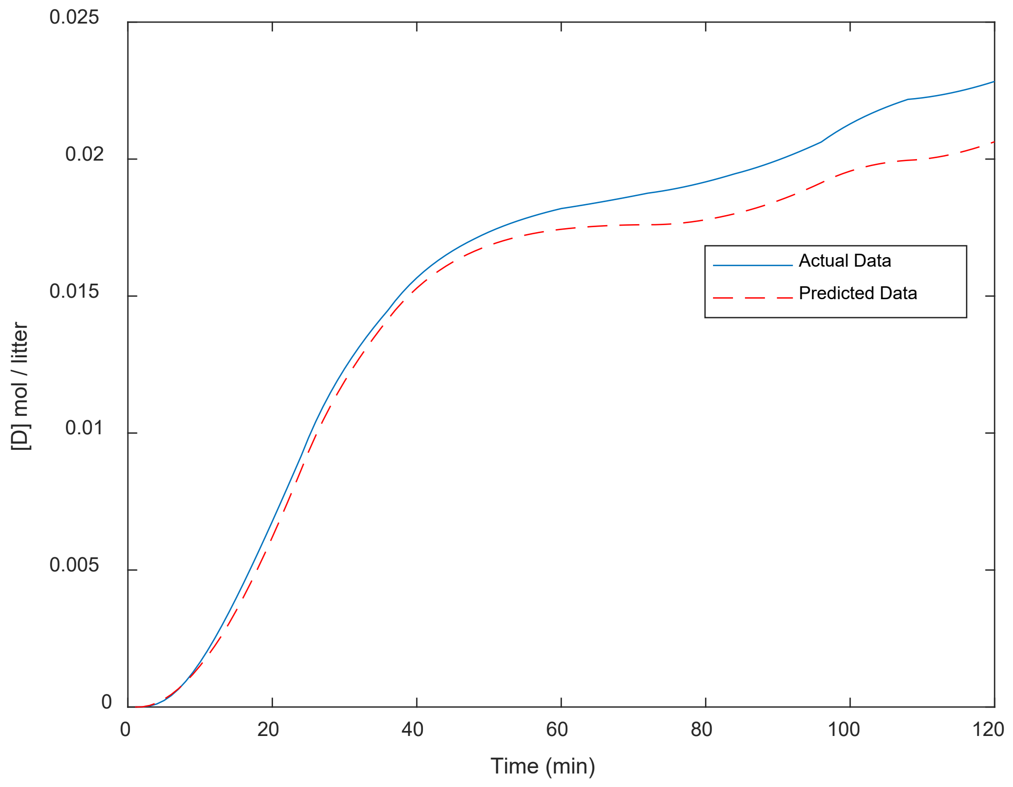


Figure 3.4 Multi-step ahead ELM model prediction on C_D

Table 3.3: Comparison of modelling performance on one-step and multi-step predictions

Predictions	RMSE for C_C $(mol.L^{-1})$	RMSE for C_D $(mol.L^{-1})$
One-step ahead	0.00063	0.00019
Multi-step ahead	0.00270	0.00120

Obviously from the ELM model predictions figures shown for both the C_C *and* C_D $(mol.L^{-1})$ above, ELM can predict very accurately for one-step ahead predictions and reasonably well for multi-step model predictions. In the case of one-step ahead predictions for C_C *and* C_D $(mol.L^{-1})$, the RMSE values are 0.00063 $mol.L^{-1}$ and 0.00019 $mol.L^{-1}$ respectively. For the multi-step ahead predictions, the RMSE values obtained for

$C_C \, and \, C_D \, (mol. L^{-1})$ are 0.0027 $mol. L^{-1}$ and 0.00120 $mol. L^{-1}$ respectively. The RMSE performance evaluation results in Table 3.3 and the Figure 3.3 and 3.4 plots shows that ELM can accurately predict for both short- and long-term prediction horizon but short-term predictions are more accurate than long term model predictions. This is due to the accumulations of errors in multi-step ahead model predictions.

The long-range model prediction is quite important in industries for model predictive control and real-time optimisation. The more accuracy of modelling technique for multi-step ahead predictions, the better the model predictive control performance. Therefore, there is a need to try some other modelling techniques or improve on ELM modelling predictions by merging some other statistical learning techniques.

3.3.2 Static Modelling of Fed-Batch Reactor Process using ELM

In the static models, the product quality variables at the end of each batch are to be predicted by the ELM models using the feed profile as model inputs. The historical input and output process data, batch duration and batch interval remain the same as described in dynamic modelling above. The essence of having many sampling intervals within a batch is to improve the performance of the network model by increasing the degree of freedom in control policy. The more the sampling interval, the more the tendency to improve the performance and vice-versa.

The main objective of this process modelling is to maximise the amount of the final product $C_C \, (t_f)V(t_f)$ and at the same time, keeping the amount of undesirable product $C_D(t_f)V(t_f)$ to the barest minimum. Adding all the reactant B at the start of the reaction will lead to the side reaction II in equation (3.19) and this leads to the yield of the undesired product D. To keep the undesired product D to the barest minimum, the reactor is operated in semi-batch (fed-batch) mode, where B is added in a feed stream intermittently. The static ELM network models for the prediction of $C_C \, (t_f)V(t_f)$ and $C_D(t_f)V(t_f)$ is of the form:

$$y_C = f_1(X) \tag{3.44}$$

$$y_D = f_2(X) \tag{3.45}$$

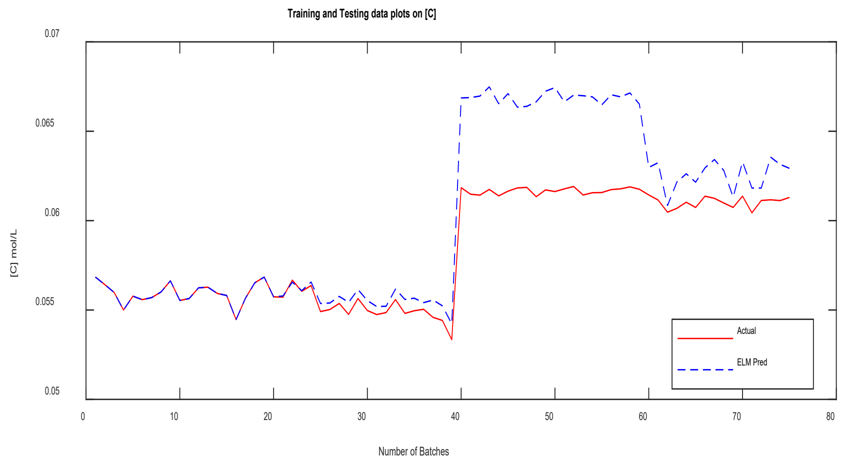
where $y_C = C_C \, (t_f)V(t_f)$, $y_D = C_D \, (t_f)V(t_f)$, $X = [x_{1,} \, x_{2,} \dots \, x_{10,}]^T$ is a vector of the reactant feed rates over a batch, $f_1 \, and \, f_2$ are the nonlinear functions representing the ELM networks.

As stated previously in dynamic modelling, 100 batches of historical simulated process data were generated with normal random noise distribution in the range of $0 < x_i < 0.01$. Out of these 100 batches, 75% of the historical process data were used for training and testing in building the ELM models while the remaining 25% were used as unseen validation data.

Results and Discussions

In the static models, the performance of ELM model predictions on training, testing and validation data for $C_C \ and \ C_D$ are shown in Figures 3.5 and 3.6 respectively. 26% out of the 75% training and testing historical data were used as training data to build the ELM models. The 26% out of the 75% training data was used in building the ELM models to show the effect of having insufficient (few) datasets in the process of learning from the training data. Although, it is expected that inaccurate model predictions may occur due to insufficient data to learn. Many other techniques can be added to the ELM concept to give a good generalisation capability of the model predictions despite the use of insufficient training data.

Moreover, the mean square errors (MSE) between the actual historical data and the predicted data for both the product C_C and by-product C_D are also given in Table 3.4.



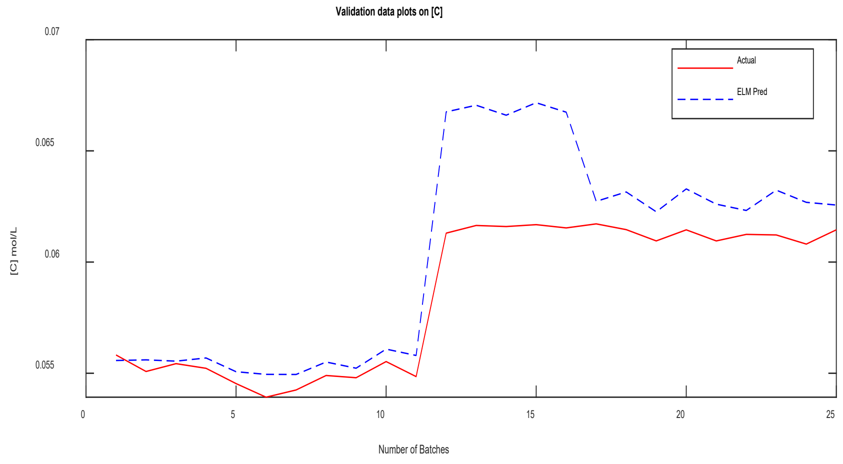
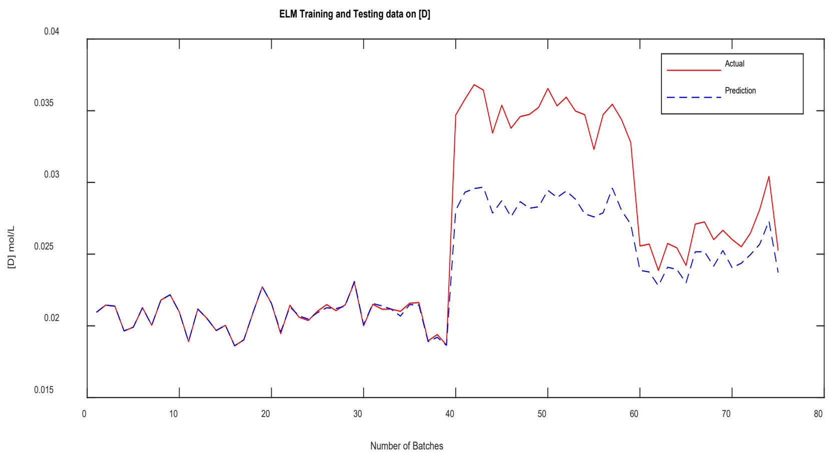


Figure 3.5 ELM Model predictions of [C] $mol. L^{-1}$ at batch end.



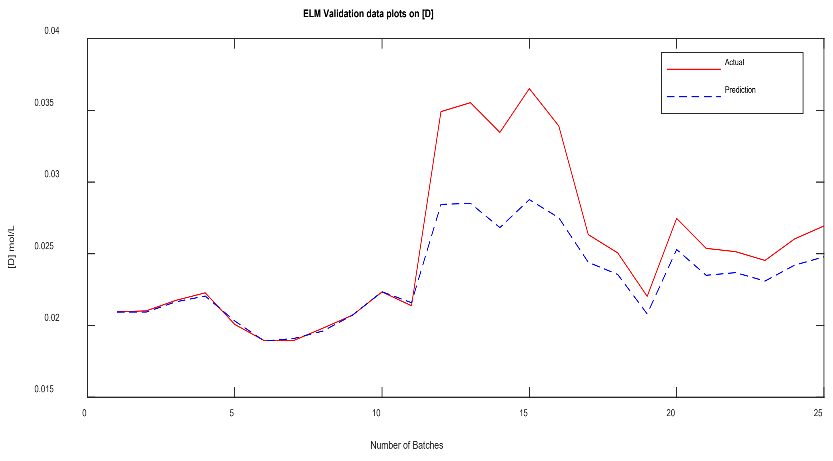


Figure 3.6 ELM Model predictions of [D] $mol. L^{-1}$ at batch end.

Table 3.4 Predicted MSE Values on C_C and C_D Using ELM

Data	MSE for C_C $(mol. L^{-1})$	MSE for C_D $(mol. L^{-1})$
Training &Testing	0.8506	0.3287
Validation	0.7161	0.2982

The ELM model predictions on [C] given in Figure 3.5 shows accurate model predictions on only the first 20 batches of the training and testing datasets. This is due to the 26% of datasets that were used in modelling the training and testing data. For the rest of the batches, the model prediction errors tend to become larger, from the 21st batch till the 65th batch when the prediction error becomes smaller on the training and testing dataset. This is caused by an insufficient dataset the model can learn from before being able to predict accurately on an unseen dataset. In general, the model prediction on the validation dataset is said to be inaccurate due to wide model mismatches between the actual and the predicted data shown in Figure 3.5. The same explanation goes for the model predictions seen on [D] in Figure 3.6.

However, based on the low values of the MSE obtained in Table 3.4 above, some might believe that the ELM model predictions seem to be accurate and predicted well enough for both the by-product C_D and the main product C_C. The MSE values are $0.8506\ and\ 0.7161$ for training, testing and validation data of C_C were close because it's an overall batches error where some batches of less error are added up with batches of large errors. The same explanation goes for the MSE values of C_D given as 0.3287 for the training and testing data and 0.2982 for the validation data.

To re-validate and be certain of the mean square error performance of the ELM static modelling, some techniques can be integrated with the ELM (which will be discussed in Chapter 4) or re-sample the historical data in form of bootstrap aggregated ELM (BA-ELM) instead of the ordinary ELM in the steady-state modelling of the process.

3.3.3 Static Modelling of Fed-batch Reactor Process using BA-ELM

As stated earlier in building steady-state ELM modelling, the pre-processing steps, and the data allocations: 70% for training and testing and 30% for validation remain the same for building BA-ELM static model. The bootstrap resampling technique was applied to the training data set to generate $nm = 20$ different bootstrap replication data sets for training individual ELM networks. 56% of the original training data were selected for training, 24% for testing and 20% for unseen validation data to build the BA-ELM model. The individual ELM model networks are linearly stacked as a weighted combination of the individual ELM networks to improve the model generalisation capability in predictions. Figures 3.7, 3.8, and 3.9 show the BA-ELM model predictions and the mean square error (MSE) plots of the individual BA-ELM output on training, testing and validation data sets for product C_C , while

Figures 3.10, 3.11 and 3.12 show BA-ELM model predictions and the MSE for the by-product C_D. Moreover, Figures 3.13 and 3.14 show the 95% confidence bound of BA-ELM model prediction plots for both C_C and C_D.

Results and Discussions

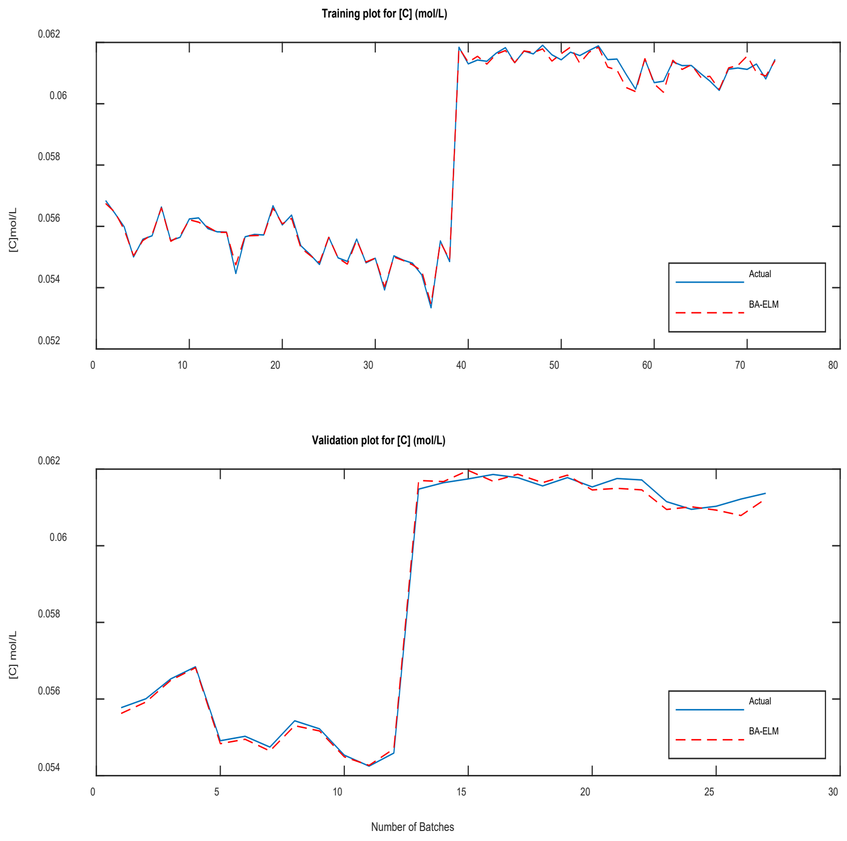


Figure 3.7 Training and Validation data Plots using BA-ELM for $C_C(mol.L^{-1})$

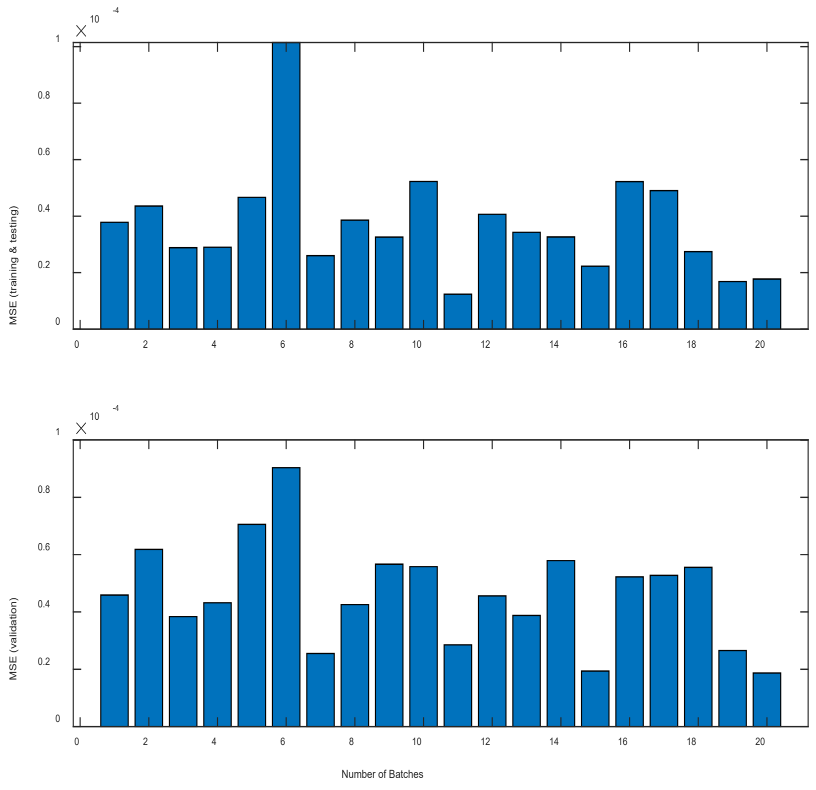


Figure 3.8 Plot of MSE for Individual Single ELM Network on $C_C(mol.L^{-1})$

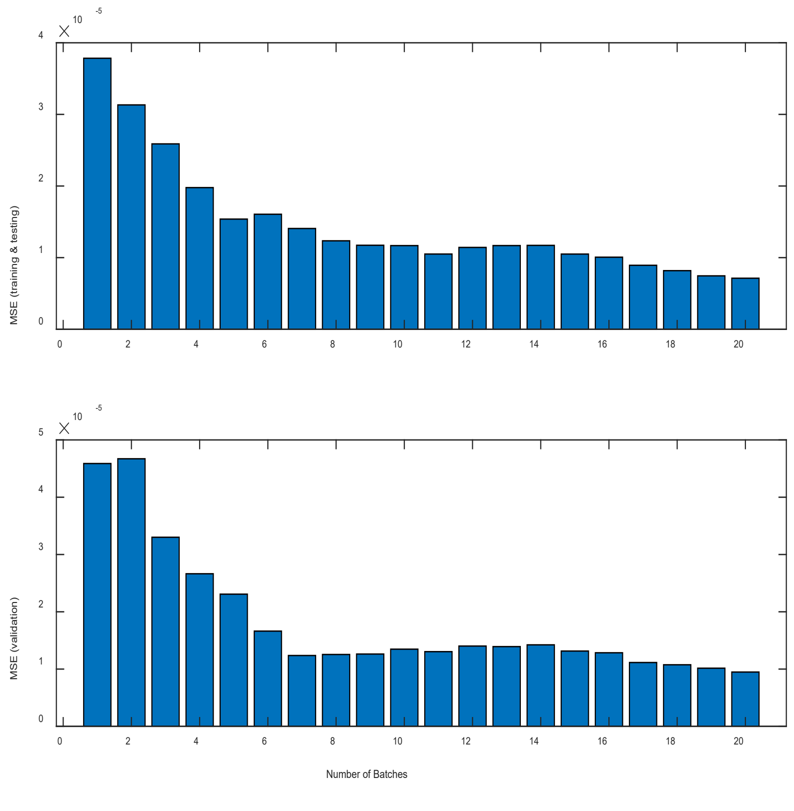


Figure 3.9 Plot of MSE for Individual Aggregated ELM Network on $C_C(mol.L^{-1})$

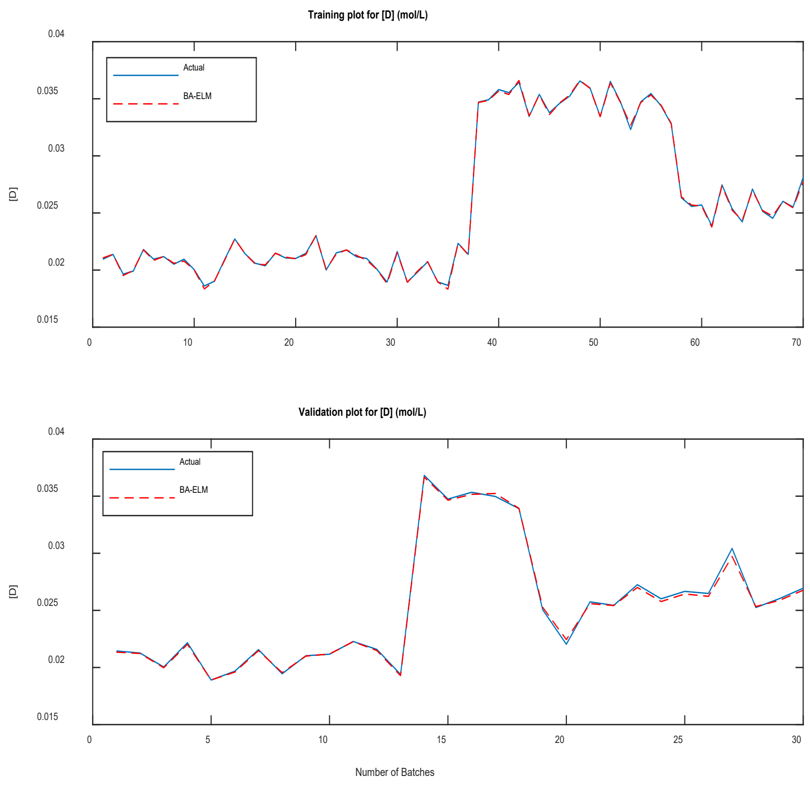


Figure 3.10 Training and Validation data Plots using BA-ELM for $C_D(mol.L^{-1})$

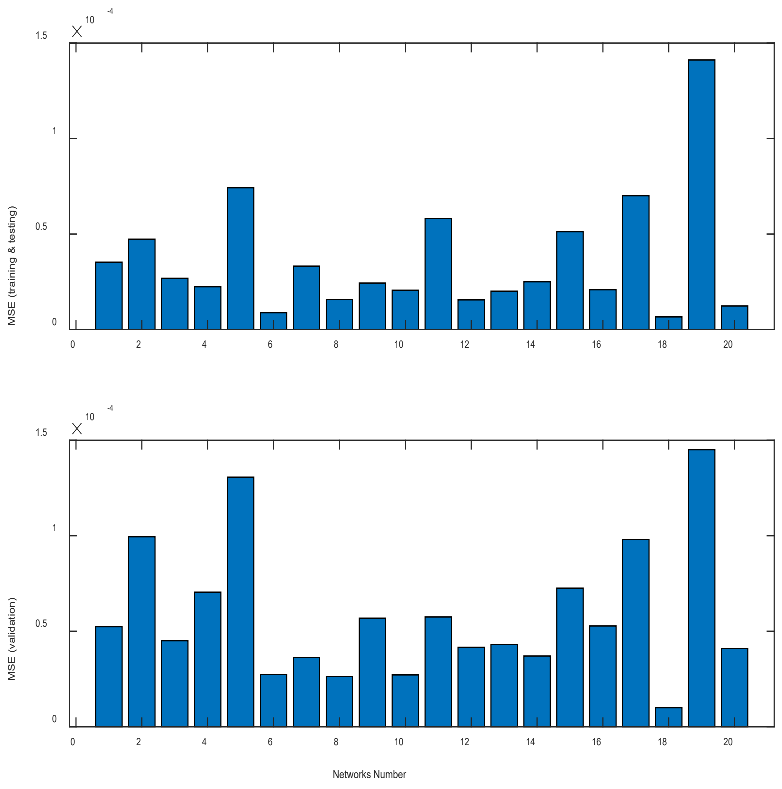


Figure 3.11 Plot of MSE for Individual Single ELM Network on $C_D(mol.L^{-1})$

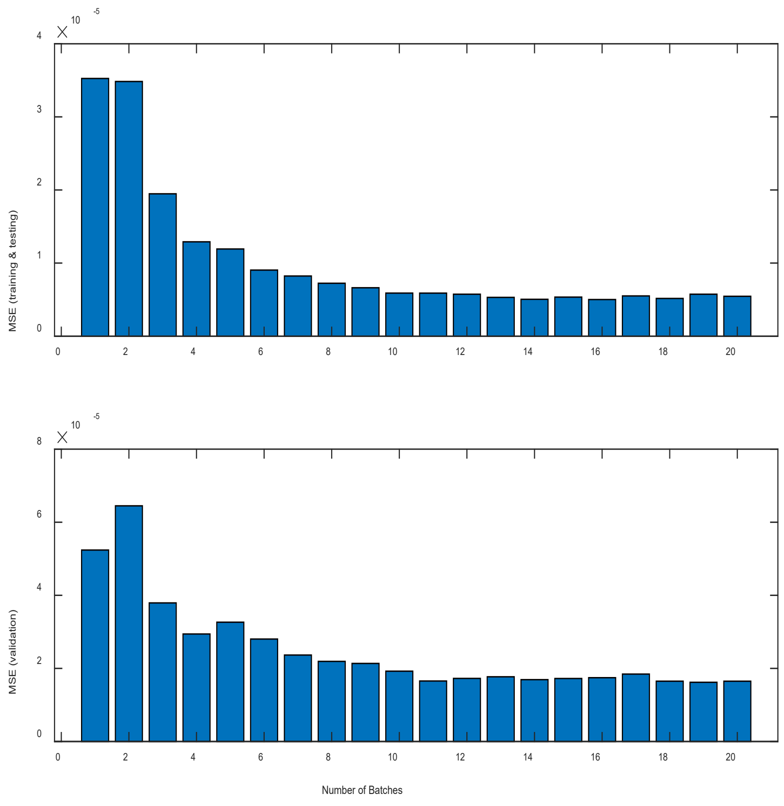


Figure 3.12 Plot of MSE for Individual Aggregated ELM Network on $C_D(mol.L^{-1})$

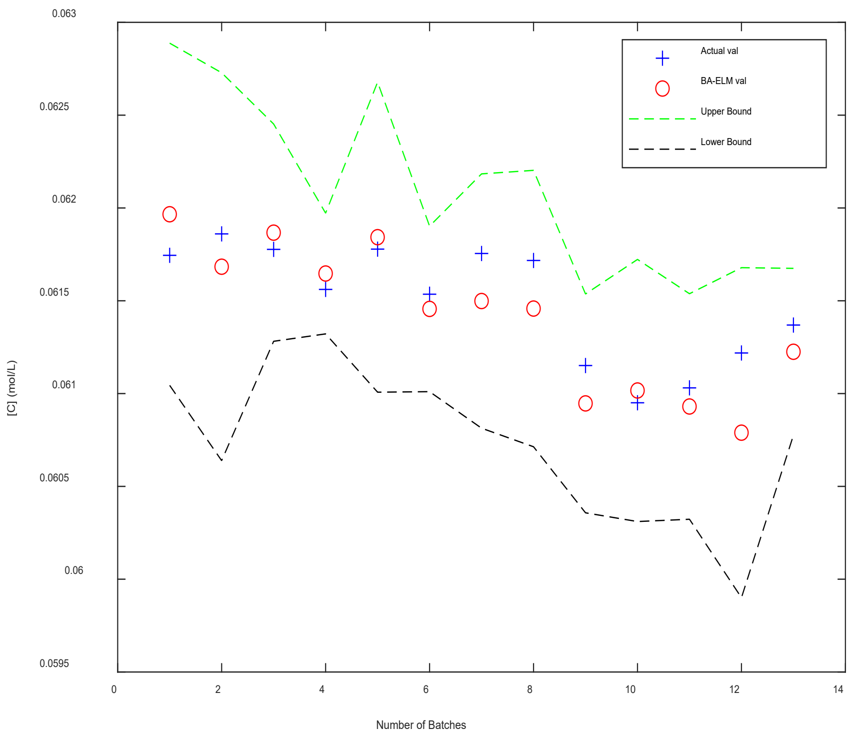


Figure 3.13 Confidence Bound Plot of BA-ELM model prediction on $C_C(mol.L^{-1})$

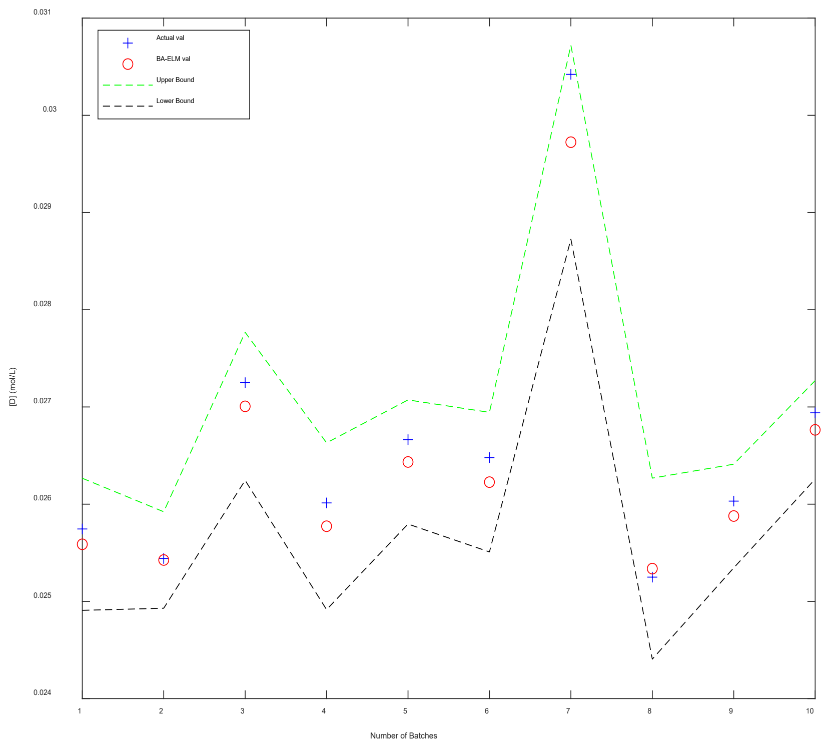


Figure 3.14 Confidence Bound Plot of BA-ELM model prediction on $C_D(mol.L^{-1})$

Based all the plots of the BA-ELM model predictions on the fed-batch reactor process shown above, it shows that BA-ELM models predict the process operation more accurately than single ELM models on the same process. In comparing the plots of the model predictions of static ELM and BA-ELM, Figure 3.7 shows a more accurate prediction when compared to Figure 3.5. Likewise, Figure 3.10 shows better predictions than Figure 3.6.

Figure 3.8 shows the MSE in training and validation data sets for 20 different individual ELM models generated with bootstrap technique while Figure 3.9 shows the MSE in training and validate data sets for the same 20 different data set using BA-ELM models for product C_C. Figure 3.8 also shows the individual network MSE's in both training and validation data in which their values differ widely and are unstable. The patterns of MSE values of the individual network on training data differ from those on the validation data. This is signifying that different model do perform differently (either good or bad based on the value of the MSE) in different networks because each network was trained from different random selections on original data sets and likewise, Figure 3.9 shows how the effect of aggregating the networks can bring about improvement in model accuracy and robustness to a process. Its

MSE can be seen to be stable (from network 10 till the last network), in the same range and becoming lower from one trained network data to the next network. Same explanation for comparing Figures 3.11 and 3.12 but to the by-product C_D.

Both Figure 3.13 and Figure 3.14 show the 95% confidence limit for the predictions of C_C *and* C_D respectively on the validation data. In these plots, the upper green dashed curves indicate the upper confidence limit, the lower black dashed curves denote the lower confidence limit of the historical process data, the blue plus marks represent the actual values and the red circles denote the BA-ELM predictions. The essence of the confidence limit plot is to show the prediction reliability of the BA-ELM model in that the true values lie within the confidence interval (i.e., between the lower and upper limit) with 95% probability. If the confidence intervals are narrow, the predictive model will be reliable and accurate. It can be seen from both Figures 3.13 and 3.14 that the actual and the predicted values

3.4 Product Quality Control of Fed-Batch Reactor using Iterative Learning Control

The iterative learning control (ILC) strategy can improve the final product quality of an isothermal fed-batch reactor process by learning from batch-to-batch historical control performance data of that process. The general idea of batch-to-batch control is to use the repetitive feature of the batch process to reformulate the control profile to meet the desired product quality. This is achieved using past process knowledge to update the current batch operation so that the product trajectory will converge asymptotically to the desired reference trajectory.

In this study, only the 20 batches out of the 100 simulated process model of the fed-batch reactor system are used and all other data cleaning (the sampling interval, time, input constraint and batch end) remains exactly same as given in equation (3.47) below. Remember that the main aim is to achieve the highest possible value of the desired product C_C and the lowest corresponding value for undesired by-product C_D. We want to make predictions on both the product and the by-product C_C *and* C_D at the same batch end time (120 mins). For batch processes, the bulk of the non-linearity in the process operation can be removed by subtracting time-varying nominal trajectories from process operation trajectories which will eventually allow the linear modelling technique to perform well on the perturbation variables (Xiong and Zhang, 2003).

In a linear time-varying (LTV) model given by equation (3.46) below:

$$Y = GX \tag{3.46}$$

where Y is the output trajectory, X is the input trajectory and G is the linear model between the input and the output trajectories for the batch process. Let the input and output nominal trajectory matrix be represented as:

$$X = \begin{pmatrix} Xk_1^T \\ \vdots \\ Xk_{20}^T \end{pmatrix},\ Y = \begin{pmatrix} Y_{i_1}^T \\ \vdots \\ Y_{i_{20}}^T \end{pmatrix}\ and\ Y_i = \begin{pmatrix} y_i^C(t_f) \\ y_i^D(t_f) \end{pmatrix}, Xs = \begin{pmatrix} X_{i_1}^T \\ \vdots \\ X_{i_{20}}^T \end{pmatrix},\ Ys = \begin{pmatrix} Y_{i_1}^T \\ \vdots \\ Y_{i_{20}}^T \end{pmatrix} \tag{3.47}$$

The perturbation model matrix, Gs can be considered in two cases, namely: the fix and the time-varying matrix Gs. For the fix matrix Gs, its perturbation model depends on the nominal trajectories and when it is fixed then the perturbation model is also fixed. Gs can be calculate using PLS, PCR, or MLR techniques and for MLR calculation, it is represented as:

$$Gs = ((\mathrm{X} - \mathrm{Xs})^T(\mathrm{X} - \mathrm{Xs}))^{-1}(\mathrm{X} - \mathrm{Xs})^T(Y - Ys)) \tag{3.48}$$

and the control gain is calculated as:

$$D_i = (G^TQG + R)^{-1}Gs^TQ \tag{3.49}$$

Where Q, as a (2×2) matrix, and R, as a (10×10) matrix, are the positive-definitive weighting matrices, D_i is a (10×2) matrix, and Gs is a (2×10) matrix. The R matrix is a determinant of the convergence level, the larger the weight on the input, the slower the convergence. The Q and R matrices are given as:

$$R = \lambda \times I \quad and \quad Q = \begin{pmatrix} \beta & 0 \\ 0 & \theta \end{pmatrix} \tag{3.50}$$

The $\beta\ and\ \theta$ values in Q were also determinants for convergences to reference trajectories. A higher value of either the $\beta\ or\ \theta$ will leads to a better convergence to the desired reference trajectories of both product and by-product.

To improve the feeding policy from batch to batch, the calculation for the tracking error e_i for the ith batch is given as:

$$e_i = Y_d - Y_i \tag{3.51}$$

where $Y_d = \begin{pmatrix} C_C(t_f) \\ C_D(t_f) \end{pmatrix}_d$ and the calculation for the next batch control is given as:

$$X_{i+1} = X_i + D_i e_i \tag{3.52}$$

The process runs until all the 20th batches has been completed where we can see all the progress of the quality of both product and the by-product batch by batch.

For the changing matrix Gs, the tracking of the model prediction can be improved through the update of Gs at each batch of operation (Xiong *et al.*, 2007). At each batch completion, the matrices of the output and the input trajectories increases by a row. The new row defines the update for the nominal trajectory matrices $Xs\ and\ Ys$ which lead to new update for Gs and the control action D_i. The looping process can be listed as:

- Calculation of the e_i
- Calculation of the next input trajectory X_{i+1}
- Simulation of the next output trajectory Y_{i+1}
- Definition of new nominal trajectory matrices $Xs\ and\ Ys$, and lastly
- Calculation of the new $Gs\ and\ D_i$

Results and Discussions

The accuracy of fixed Gs depends on the initialisation of its historical input matrix. Bad initialisation leads to a bad perturbation model Gs and control actions of the batches. Likewise, good initialisation allows easy convergence of the product quality to the desired trajectories.

The last vector of the input matrix X is used as the nominal trajectory Xs and the desired parameters are given as:

$$Yd = \begin{pmatrix} 0.0618 \\ 0.0189 \end{pmatrix}, R = 0.1 \times 1\ and\ Q = \begin{pmatrix} 50 & 0 \\ 0 & 1 \end{pmatrix}$$

Figure 3.15 shows the tracking evolution of the iterative learning control (ILC) on a fed-batch reactor without perturbation. The figure shows how both the desired product C_C and undesired by-product C_D reaches the desired values of 0.0618g/L and 0.0189g/L from starting point value of 0.0560g/L and 0.0220g/L respectively. It also shows how the tracking error e_k evolved through all the batches for both $C_C\ and\ C_D$.

For the progress of product C_C in fixed Gs without perturbation, the desired product quality value has been reached from batch 6 while the undesired by-product value was reached at batch 15 but seem to get close to the desired value from batch 8. This is signifying that there is quite an improvement in product quality without increasing the by-product value.

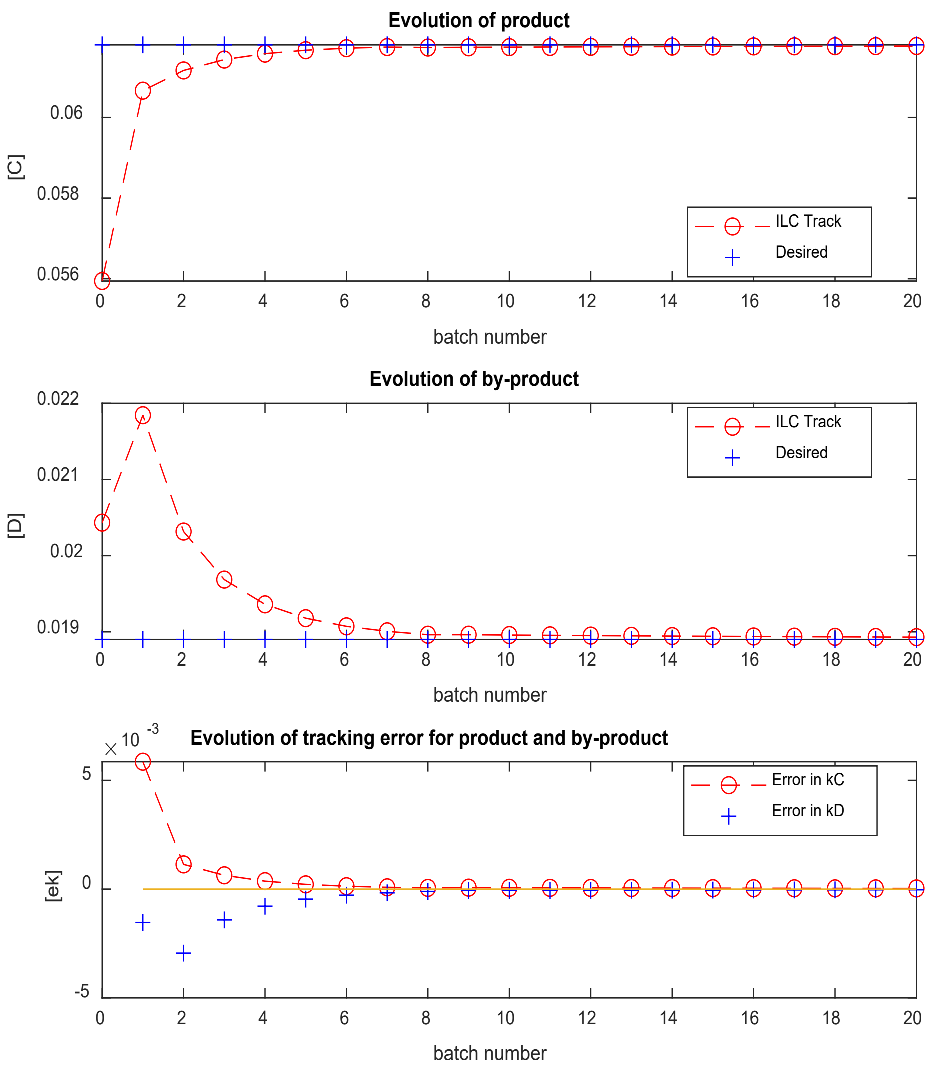


Figure 3.15 ILC tracking of the fed-batch reactor without Perturbation

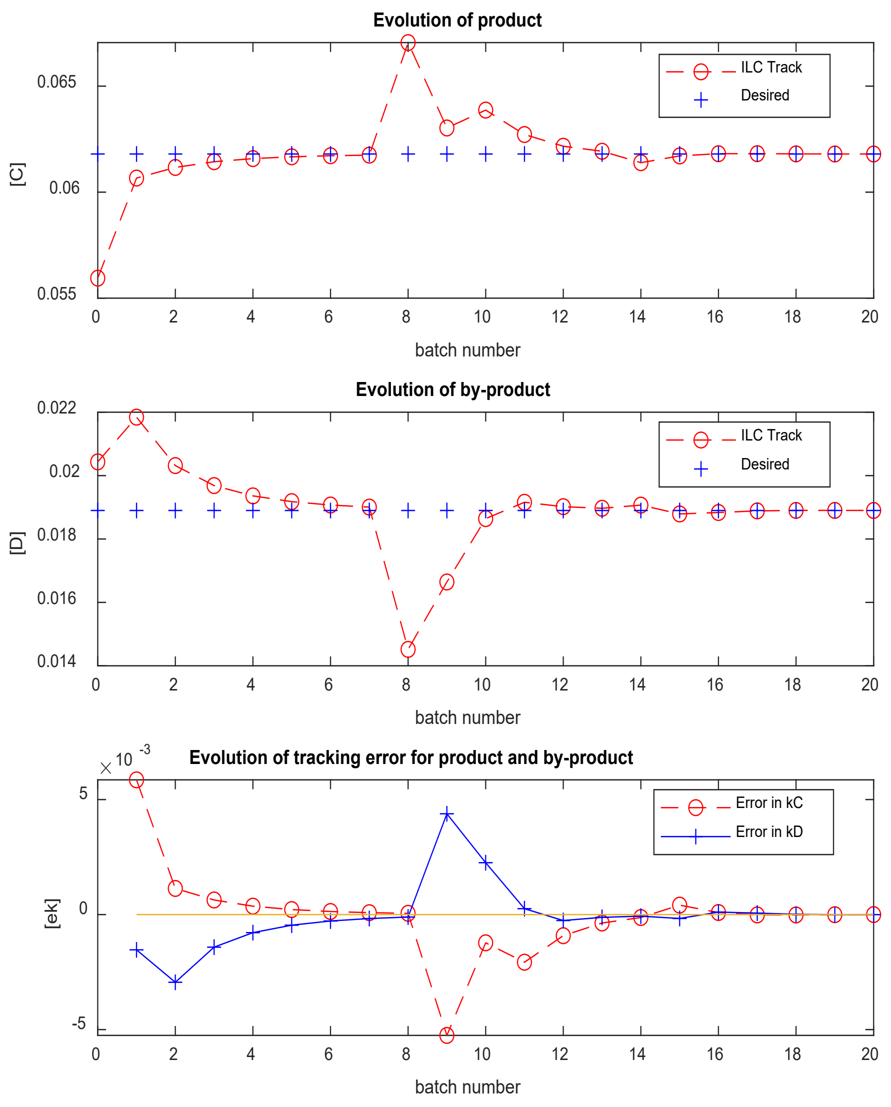


Figure 3.16 ILC tracking of the fed-batch reactor with Perturbation

Figure 3.16 shows the plots of the tracking progress for the product, undesired by-product, and batch-to-batch progress of their errors. The constant of proportionalities $K_1 and\ K_2\ is$ 0.5 from the beginning of the process tracking and at batch 8, K_1 value is increased to 0.65, favouring the fed-batch reaction to proceed to the right (i.e., an increase in the C_C product). The perturbation was applied at batch 8 because the ILC tracked value will have reached the desired product quality value and stabilised for some time as seen in Figure 3.15 before introducing perturbation at batch number 8.

Despite the introduction of the perturbation, it can be seen from Figure 3.16 that the product quality value can still be maintained and tracked to reach the desired product value. From batch 8 to batch 14, it shows the effect of the perturbation on product C_C and from batch 15, the ILC has been able to track the product quality value to the desired product value. Both the product and the undesired by-product have been able to be tracked accurately to the desired values in the presence of the perturbation.

3.5 Modelling of Baker's Yeast Fermentation Process

3.5.1 Static Modelling of Baker's Yeast Fermentation process using ELM

The cell kinetic and the reactor dynamic model equations given in equations (3.24) to (3.41) are termed as the mechanistic model for the yeast fermentation process. Simulated process operation data generated from the mechanistic model of the fermentation process is used to develop data-driven models. These simulated data serve as the process historical data of the baker's yeast fermentation in the absence of real-life operational data (i.e., if there were historical process data of the fermentation process from industry, there would be no need to simulate the process operation data).

The model inputs are the substrate feed flow rate, randomly generated in the range of 0 to 2500 L/h for 100 batches with some uniform distribution random noise of $\pm 0.5L/h$ added to each batch to show that the fermentor is operated in the real situation. The total batch time of 16.5h is divided into 10 equal intervals with constant substrate feed rate at each interval. Thus, the ELM model is of the static model form, relating the final biomass to the substrate input vector as given below:

$$Y_i = f(x_i) \tag{3.53}$$

where Y_i is the final biomass concentration of the ith batch and $x_i = [x_1, x_2 \ldots x_{10}]$ represent the substrate feed rates at the *ith* batches.

Simulated process data were obtained using equation (3.24) to (3.41). Model parameter values and initial condition set values given in Table 3.2. The simulated data were scaled to zero mean and unit variance before dividing the data set into training and testing (70% of the data) while the remaining data were used as validation data (30% of the data). Only the first 28.6% of the training and testing data were used for ELM modelling of the fermentation process and the essence of modelling with few data sets is to show how inaccurate the ELM predictions can be with few historical process data availabilities.

3.5.2 Results and Discussions

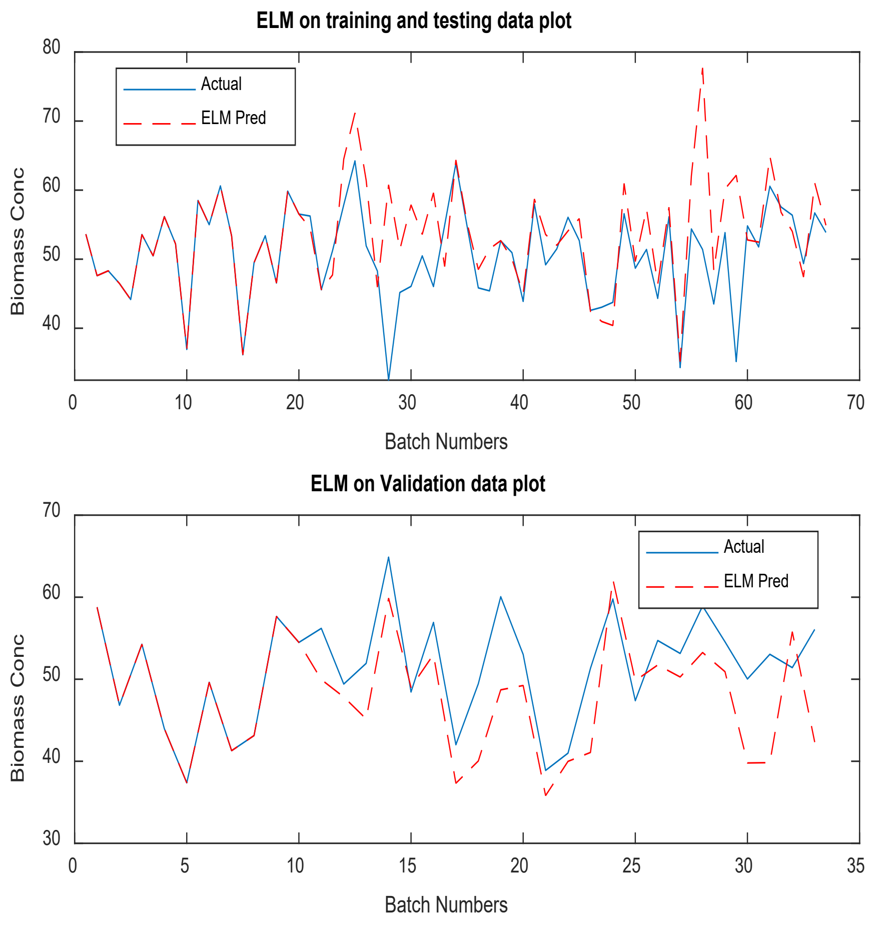


Figure 3.17 ELM modelling on Baker's Yeast Fermentation

The simulation result of the final biomass concentration of the baker's yeast fermentation is shown in Figure 3.17. The first plot in Figure 3.17 shows the ELM model prediction on the actual training and testing data set and the first 20 batch samples were tracked accurately with ELM model. This is because only 28.6% of the training data were used for the ELM modelling to show the effect of having few or insufficient data for modelling with an ELM method. Although, the whole training data set can be used in modelling the fermentation process to have better and accurate model representation of the process but what will happen in a situation where we have insufficient data to model the process system?

Both plots on Figure 3.17 shows the effects of using insufficient historical process data for ELM model prediction on an unseen validation and the actual validation dataset. The ELM model predictions are not accurate due to the limited historical data used in modelling the fermentation process.

3.6 Static Modelling of Baker's Yeast Fermentation process using BA-ELM

In modelling of the baker's yeast fermentation with BA-ELM, the simulated process data for the ELM modelling stated in Section 3.5.1 and all other cases like data pre-processing all remain the same. At the beginning of the learning process, simulated process biomass data is divided into training, testing and validation data. 75% of the simulated data were allocated for training and testing while the remaining 25% of the data were used as validation. Thus, the training and testing data were re-sampled to generate 30 different bootstrap replications for training the individual ELM networks for the fermentation process.

For the development of each of the 30 different individual networks, the numbers of hidden neurons from 2 to 30 were tried and the ELM model with the smallest sum of squared errors on the testing data was selected. At the end of all the learning process, the main objective is to model the relationship between the input feeding rates (substrate) to the output biomass concentration (g/L) as given in equation (3.53) which can be described fully with sets of weights and biases of the hidden and output layers represented by $(w_1, b_1) and\ (w_2, b_2)$ respectively.

Therefore, the individual ELM model networks are linearly stacked as a weighted combination of the individual ELM networks to improve the model generalisation capability in predictions. Figure 3.18 shows the performance of the BA-ELM model network and the accuracy of the model where the predicted final biomass concentration values are plotted against the actual biomass concentration to show the predicted error distributions over the diagonal. Thus, it is referred to as the measure of dispersion, to describe the variability of the predicted error between the actual and the predicted biomass concentration (g/L).

Figure 3.19 shows the BA-ELM model predictions on unseen validation data of the actual biomass concentration and shows the 95% confidence bounds while Figures 3.20 and 3.21 shows MSE plots of the training, testing and validation data sets on both single and aggregated networks.

3.6.1 Result and Discussions

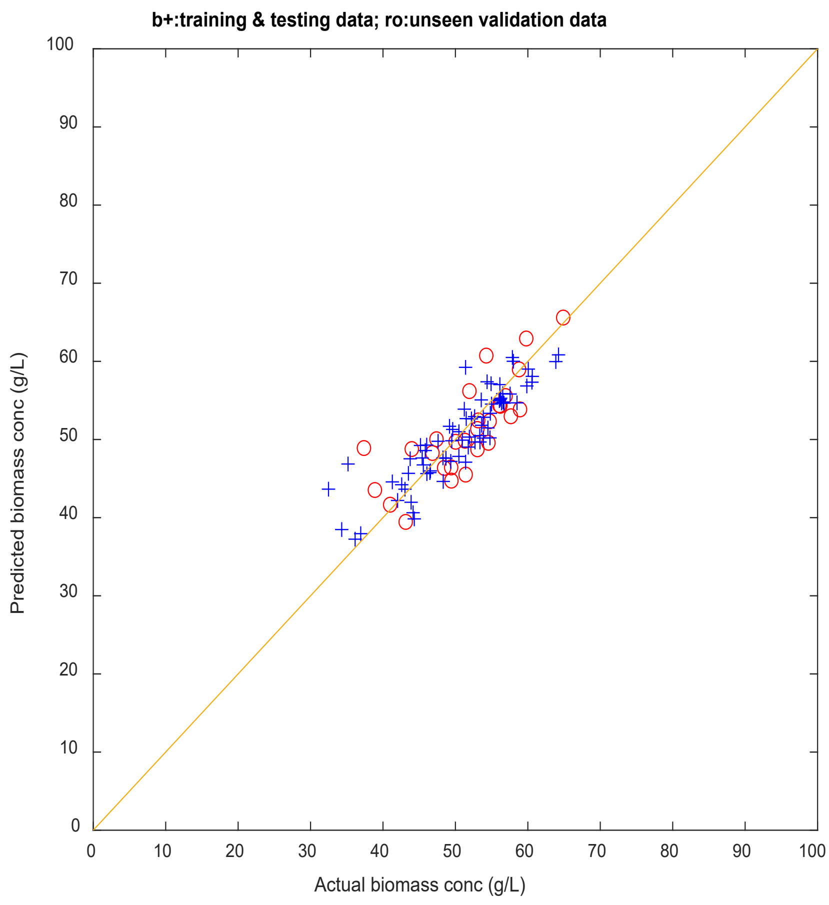


Figure 3.18 Measure of Dispersion of BA-ELM predictions

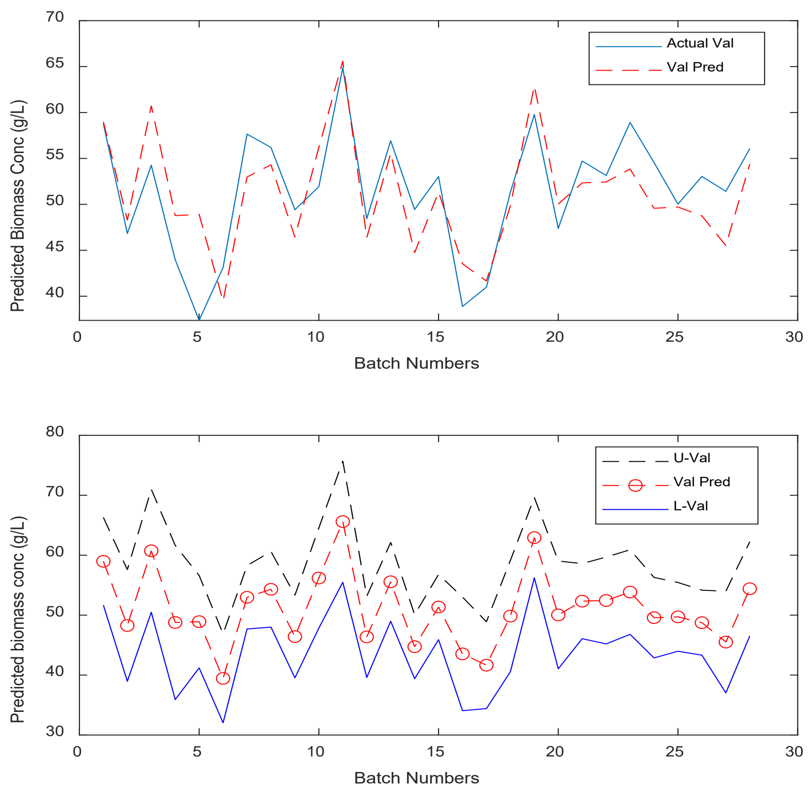


Figure 3.19 BA-ELM model predictions on unseen validation data

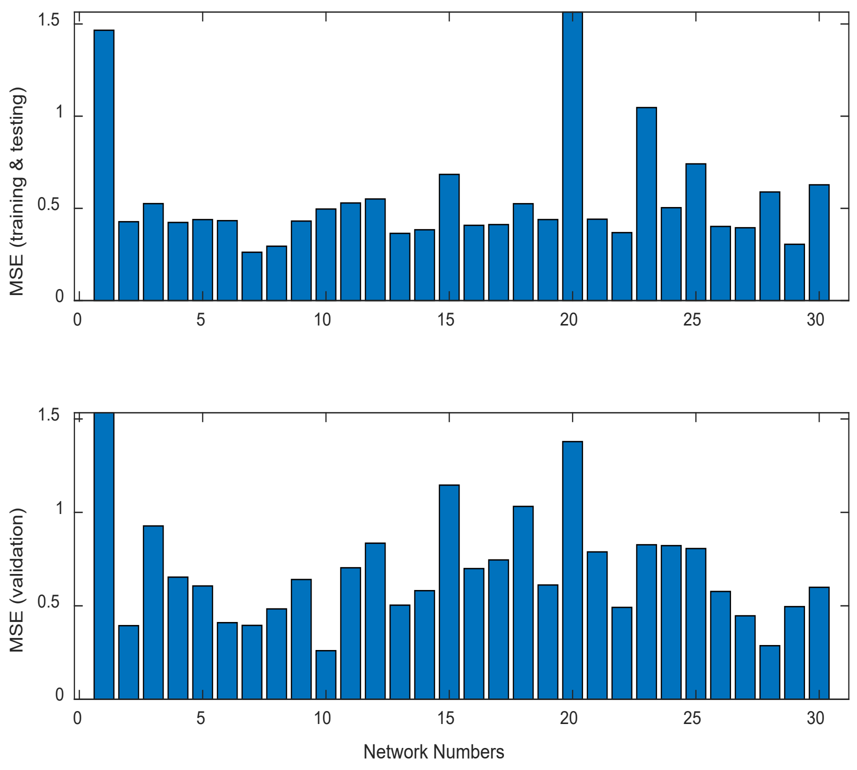


Figure 3.20 MSE values of individual networks

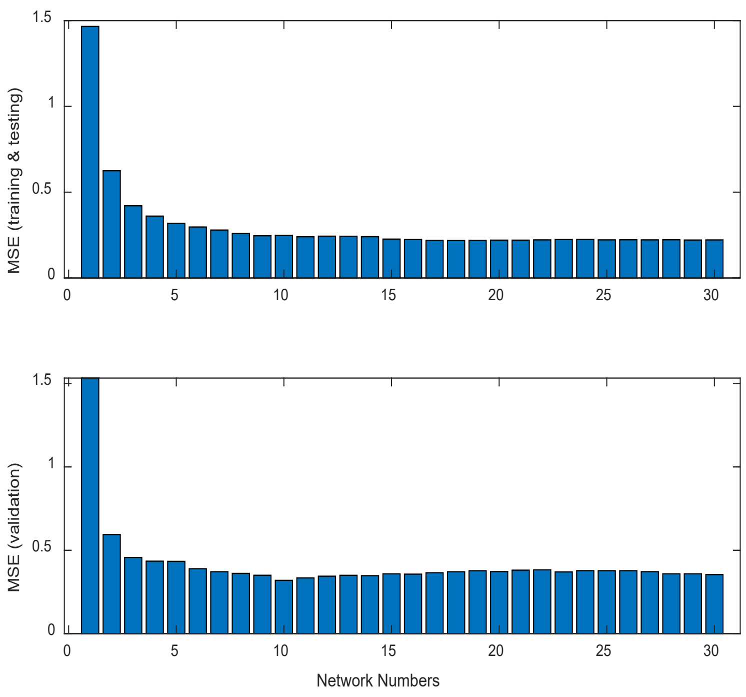


Figure 3.21 MSE values of individual BA-ELM with different numbers of aggregated networks

Figure 3.18 shows the measures of dispersion of the predicted and actual biomass concentration on the validation data. It can be seen from the plot that the spread of the biomass concentration for both the actual and predicted values were very close to the perfect model line. The more the closeness of the biomass yield spread along the perfect model line, the more accurate the predicted biomass yields. Thus, the BA-ELM seems to predict well enough with some sense of high accuracy as the actual biomass yield plot tend to be superimposed by the BA-ELM biomass yield concentration.

Figure 3.19 shows the BA-ELM predictions of biomass yield on the unseen validation data together with the 95% confidence bounds. The BA-ELM gives reliable and accurate predictions as can be seen from the relative narrow confidence bounds and the small errors.

Moreover, Figure 3.20 shows the mean squared error (MSE) plot of the individual BA-ELM networks on training, testing and validation data. The individual ELM models were unreliable as they give quite different performances on the training and testing data and the unseen validation data. For instance, the minimum MSE on training data in Figure 3.20 is obtained by network 7 but network 10 performs the best on the unseen validation data instead of network 7. Thus, it can be concluded that the individual networks are not robust enough to give accurate predictions on the unseen data. However, Figure 3.21 shows consistency in the model performance on both training and testing data and unseen validation data. This is due to applying the bootstrap aggregation on the individual models leading to robust and accurate model predictions on the unseen validation data. For instance, there are some batches (1, 3, 15, 20, 23 and 25) in Figure 3.20 with high MSE values on both training and validation data and these have been successfully reduced to the barest minimum values in both training and validation data on Figure 3.21. Thus, lower values of MSE on the unseen validation data indicate that the BA-ELM model provides more robust and accurate predictions than the single ELM models.

3.6.2 Conclusion

Summarily, at the end of this chapter, I have been able to demonstrate ways of developing models and controlling fed-batch case studies through either a static or dynamic ELM model, bootstrap aggregated ELM (BA-ELM) and an iterative learning control (ILC) technique. Moreover, different prediction performances such as MSE, RMSE, and validation plots were all used in confirming the reliability and robustness of any of the model prediction methods.

CHAPTER FOUR

BATCH-TO-BATCH ADAPTIVE MODELLING USING ELM AND RLS

4.1 Introduction

Both fed batch and batch processes are vital mode of process operation commonly used in many chemical and pharmaceutical manufacturing industries. In these process operations, the interest lies in the final-batch product quality. To obtain an optimal control policy for the process operation, model capable of accurate predicting the final product quality variables (i.e., at batch-end) is required. The optimal control policy obtained from the ELM model may not be optimal when applied to the actual process due to model-plant mismatches and unforeseen disturbances that may arises due to changes in operating conditions from batch to batch. To overcome this, model updating is required. Previous ELM batch model prediction errors can be used in the recursive least square (RLS) algorithm to improve the ELM model predictions for the current batch. The improved predictions are proven by Xiong et al., (2004) to decreased from batch to batch. Therefore, batch-to-batch adaptive modelling with ELM and RLS is sought to overcome the problem of plant model mismatches.

This chapter presents a technique of merging ELM and RLS in obtaining a batch-to-batch improved model for the fed-batch case studies (the baker's yeast and the reactor system) mentioned in Chapter 3. ELM has some characteristic features of fast training together with good generalisation capability. To cope with batch-to-batch variations due to unknown disturbances such as unknown process condition drift, the RLS algorithm is integrated with the ELM to update its output layer weights recursively from batch to batch. The offline trained output layer weights of the ELM are used as the initial parameter estimation in RLS.

The rest of this chapter is organised as follows. Section 4.2 presents batch to batch modelling with an updated ELM. Applications of the proposed method to the fed-batch reactor system and the baker's yeast fermentation process are discussed in detail in Sections 4.3 and 4.4 respectively. Section 4.5 conclude this chapter.

4.2 Batch-to-Batch Modelling with an Updated ELM Model

4.2.1 Batch Process Modelling Using ELM

An ELM is a single-hidden layer feedforward networks (SLFNs) where the hidden layer weights and bias are assigned randomly. The output layer usually uses the linear activation function, and the output is calculated as:

$$y = \sum_{i=1}^{L} \beta_i h_i(x) = h(x)\beta \tag{4.1}$$

where $\beta = [\beta_1, \ldots, \beta_L]^T$ is a vector of the output layer weights, L is the number of hidden neurons, and $h(x) = [h_1(x), \ldots, h_L(x)]$ is a vector of the hidden layer outputs calculated as:

$$h_i(x) = G(a_i.x + b_i\ ) \tag{4.2}$$

where $j = 1, \ldots, N$, $a_i = [a_{i1}, a_{i2}, \ldots, a_{iN}]^T$ is a vector of weights connecting the ith hidden node to the inputs, bi is the bias of the ith hidden nodes, x_j is the jth input sample, and G is the hidden layer neuron activation function typically taken as the sigmoid function. The output layer weights, denoted by β, are obtained by solving a regression type problem as follows:

$$\min \| H\beta - T \|^2 \tag{4.3}$$

where H is the hidden layer output matrix and T is the training target matrix.

The optimal solution of Eq. (4.3) is given by:

$$\beta = H^{\dagger}T \tag{4.4}$$

where $H^{\dagger}$ denotes the Moore-Penrose generalised inverse of matrix H.

Both the input and the output historical process data are represented like the equation (3.43) given Chapter 3 and the ELM static model is the form given in equation (4.6).

$$U = \begin{pmatrix} Uk_1^T \\ \vdots \\ Uk_{100}^T \end{pmatrix},\ Y = \begin{pmatrix} Y_{i_1}^T \\ \vdots \\ Y_{i_{100}}^T \end{pmatrix} \tag{4.5}$$

$$Y_i(t_f) = f(X_0,\ U_i) \tag{4.6}$$

where Y is the final biomass concentration at each batch, X_0 is the initial condition of process operation and $U_i = [u_1, u_2 \dots u_{10}]$ represent the substrate, feed rates at *ith* time intervals.

4.2.2 Batch-wise Updated ELM Model with RLS

Batch to batch variations exist due to the presence of unknown disturbances and due to changes in operating conditions from batch to batch. Any variations in the process operation could make the ELM model inaccurate in predictions on unseen data. To cope with operating condition changes, the ELM can be updated from batch to batch by using the RLS algorithm.

In RLS, the parameter estimation at the current batch is obtained by using the previous parameter estimation and data from the current batch. The correctional term applied to the output layer weights is proportional to the prediction error at the current batch. The updating output layer weights by the RLS algorithm is given as:

$$\beta(k+1) = \beta(k) + F(k).E(k+1) \tag{4.5}$$

$$F(k) = P(k).\mathrm{H}_{k+1}{}^{T}[\lambda + \mathrm{H}_{k+1}.P(k).\mathrm{H}_{k+1}{}^{T}]^{-1} \tag{4.6}$$

$$\mathrm{P}(k+1) = [\mathrm{I} - F(\mathrm{k}).\mathrm{H}_{k+1}]\,\mathrm{P(k)}/\,\lambda \tag{4.7}$$

$$P(0) = I\,.R \tag{4.8}$$

where $E(k+1) = y_{k+1} - H_{k+1}\beta(k)$ is the ELM model prediction error at the current batch, $\beta(k)$ is the current parameter estimate of the ELM output layer weights, $F(k)$ is the Kalman gain, H_{k+1} is a vector of the hidden neuron outputs at the current batch, λ is a forgetting factor = 0.99, I = identity matrix specified at number of hidden neurons magnitude and R being a large positive number, e.g. 10000 .

Equations (4.5) to (4.7) are the key RLS equations that need to be updated at each batch of the recursive process. In integrating the RLS with ELM in this work, equation (4.5) is termed as the weight update, which is the major equation for parameter estimation of the initial output weight predicted from the ELM. The output weight prediction from the ELM is used to initialise the weight update for the parameter estimation and $P(k)$ will be initialise as equation (4.8). The RLS algorithm adjusts the ELM to fit the most recent process data by recursively solving the least squares problem in estimating the output layer weights of the ELM model. The RLS algorithm tries to eliminate the model plant mismatches caused by the occurrence of unknown disturbances through correction of the differences between the actual model output and the predicted process output.

The offline trained output layer weights of an ELM are used as the initial values for the model parameter estimation in RLS and the ELM output layer weights can thus be update iteratively with the two major recursion terms in RLS algorithm technique mentioned earlier in Chapter 2.

The main reason of integrating extreme learning machine (ELM) with recursive least square technique (RLS) is to provide a better model performance under unknown disturbances that may affect the accuracy of an ELM model prediction. However, the number of hidden neurons selection together with the output weights in ELM modelling computation are major criteria towards accurate model prediction for an ELM but combining RLS with the ELM, the number of hidden neurons selection seem to be insignificant criteria in model predictions. The schematic representation of the proposed method, ELM-RLS, is illustrated in Figure 4.12 below:

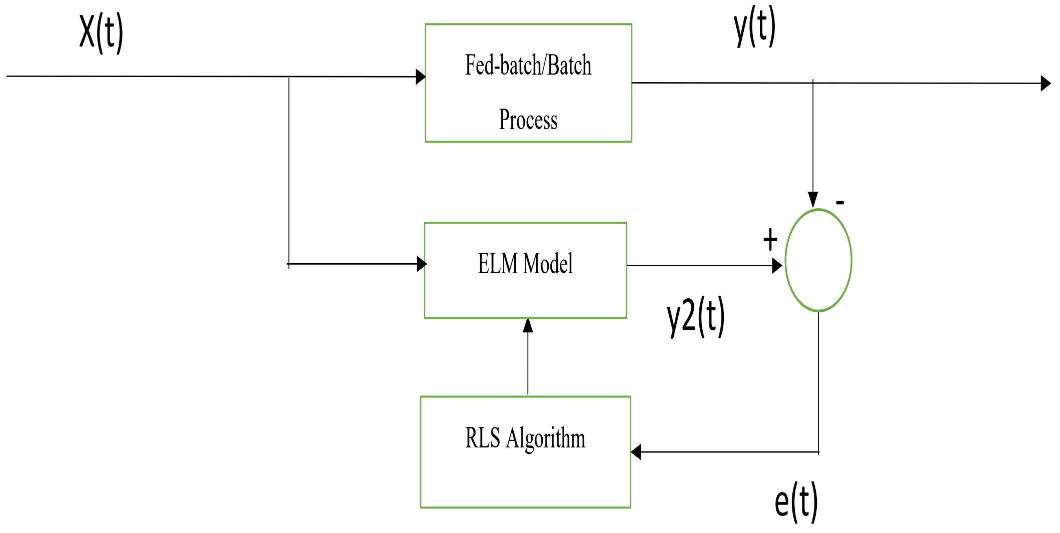


Fig 4.1 Schematic Integration of ELM and RLS.

4.3 Modelling of an Isothermal Fed-batch Reactor using the proposed ELM-RLS

All data simulations and pre-processing of the fed-batch reactor systems remain same as used in ELM modelling stated in section 3.3.1. Recall that 26% of the 75% of the historical process data were used for training and testing while 25% of the entire process data were used for the validation dataset. The model equations given in equation (3.44) and (3.45) were both re-modelled using the proposed algorithms of ELM-RLS.

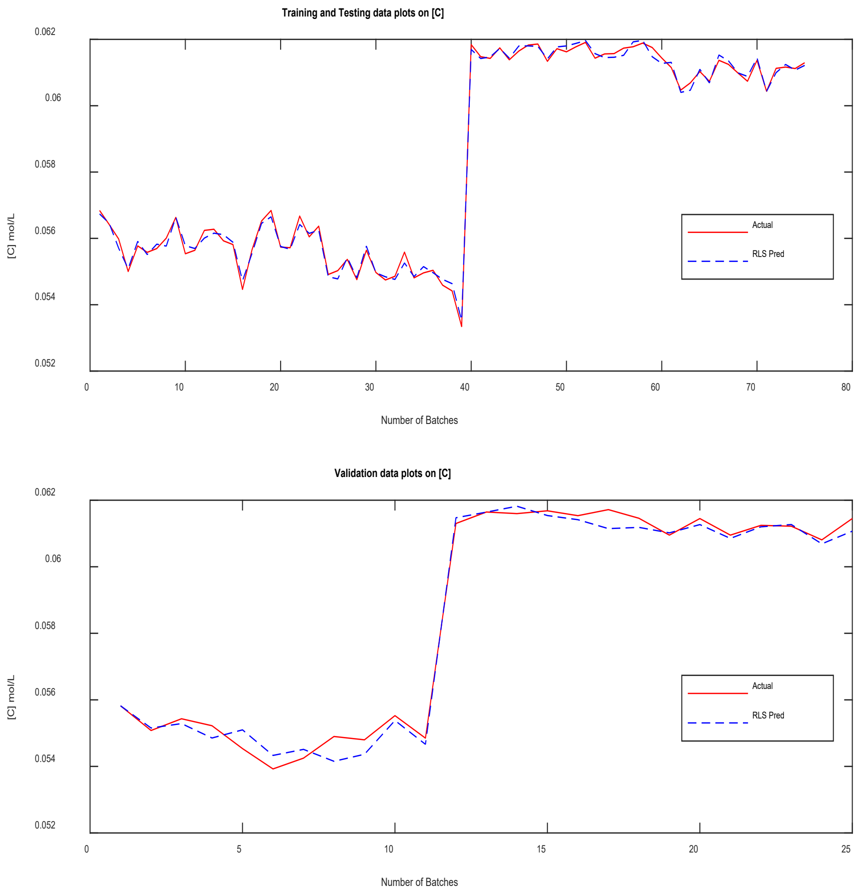
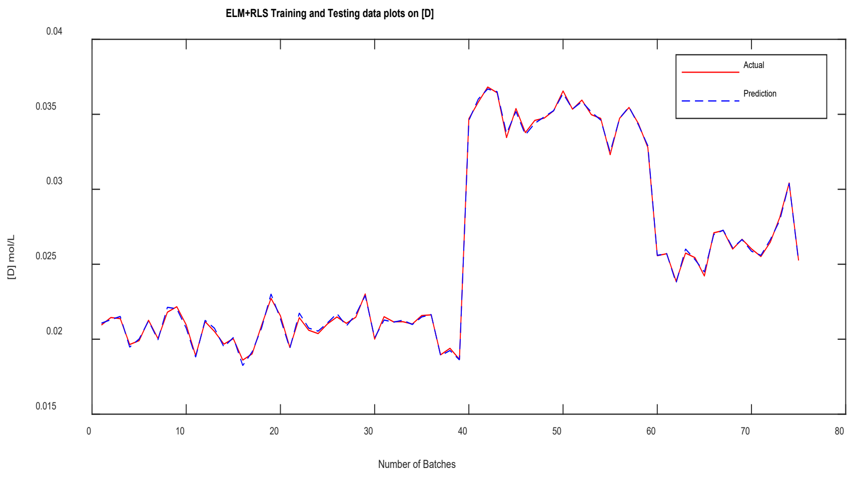


Figure 4.2 ELM-RLS model predictions of $C_C(mol.L^{-1})$ at batch end



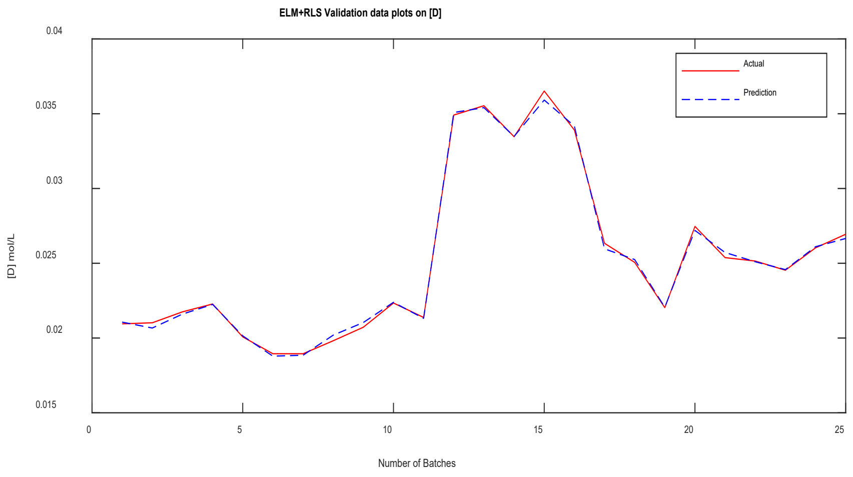


Figure 4.3 ELM-RLS model predictions of $C_D(mol.L^{-1})$ at batch end

Table 4.1 MSE Values on C_C *and* C_D Using ELM and ELM-RLS

Models	MSE (Training and Testing)		MSE (Validation)	
	C	D	C	D
ELM	0.8506	0.3287	0.7161	0.2982
ELM-RLS	**0.0021**	**0.0007**	**0.0096**	**0.0016**

Figures 4.2 and 4.3 show the performance of the proposed ELM-RLS model predictions on the product C_C and the by-product C_D respectively. Moreover, their MSE values are also shown in Table 4.1.

Figure 3.5 and Figure 4.2 show the model predictions on C_C for conventional ELM and the proposed integrated ELM-RLS respectively. Likewise, the model predictions on C_D are shown in Figure 3.6 and Figure 4.3 for conventional ELM and the proposed ELM-RLS respectively. In comparing Figures 3.6 and 4.3, the proposed ELM-RLS model predictions are more robust and accurate than that of the conventional ELM model. Provided that there are no unknown disturbances (which may not be feasible in reality) on the fed-batch reactor system and there are large historical process simulation data, the ELM model can also predict the system accurately without the need to merge RLS algorithms with ELM.

During the process simulations and building of the ELM models for both $C_C\ and\ C_D$, it is assumed that there is insufficient historical process data by using 26% out of the 75 % allocated training and testing data which makes the ELM model inaccurate in its prediction and generalisation capability on the fed-batch reactor systems. However, with the proposed ELM-RLS technique, both predictions on training, testing and validation data are shown to be accurate and with good generalisation capabilities.

Moreover, the MSE values shown in Table 4.1confirm the prediction accuracy of the proposed technique. It shows how the ELM MSE values of the C_C in training and testing data changes from 0.8506 $mol.L^{-1}$ to 0.0021 $mol.L^{-1}$ of the ELM-RLS proposed technique and likewise its validation MSE values changes from 0.7161 $mol.L^{-1}$ to 0.0096 $mol.L^{-1}$. The MSE value for the C_D also reduces close to zero for both the training, testing and validation data using the proposed technique. For training and testing data, it changes from 0.3287 $mol.L^{-1}$ to 0.0006 $mol.L^{-1}$ and for its validation data, it changes from 0.2982 $mol.L^{-1}$ to 0.0016 $mol.L^{-1}$.

In general, the MSE measures the average squared errors between the predicted data and the actual data. An MSE of zero or values very close to zero like the case of the values in Table 4.1, means that the proposed technique predicts with perfect accuracy and good generalisation capabilities of the model prediction.

4.4 Modelling of Baker's Yeast Fermentation Process using ELM-RLS

Modelling of the baker's yeast fermentation with the proposed ELM-RLS technique uses the same simulated process data obtained from the ELM modelling in Section 3.5.1 and all other cases of data cleaning such as pre-processing of the simulated process data remained same. At the beginning of the learning process with the proposed technique (ELM-RLS), the simulated process biomass data is divided into training, testing and validation by 75% to 25 % respectively.

The steady-state model equation given in equation (3.53) is re-modelled using the proposed algorithms of ELM-RLS and the performance evaluation plot of the ELM-RLS model predictions on the actual model for both the training and validation data sets are given in Figure 4.4. The MSE values were also shown in Table 4.2 under the results and discussions.

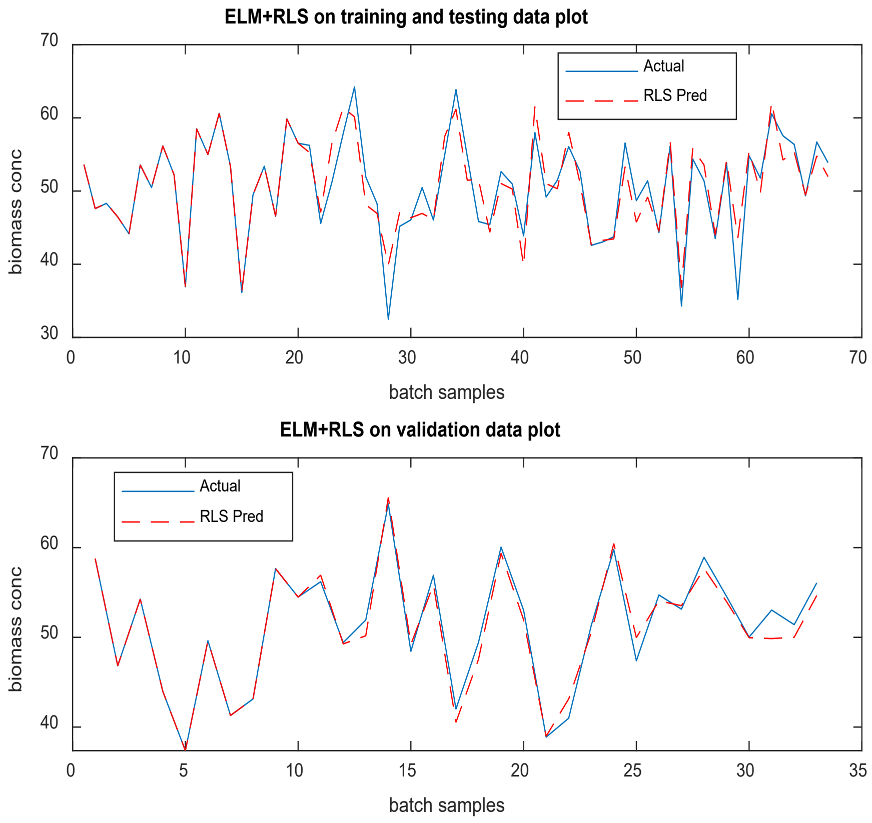


Figure 4.4 ELM-RLS model performance for predicting fermentor final biomass yield

Table 4.2 MSE Values on Baker's Yeast Fermentation Using ELM and ELM-RLS

Models	MSE (Training and Testing)	MSE (Validation)
ELM	1.0369	0.7044
ELM-RLS	0.1262	0.0281

The model predictions of the final biomass concentration of the baker's yeast fermentation using the proposed technique are shown in Figure 4.4. The first plot in Figure 4.4 represents the ELM-RLS model prediction on the training and testing data set while the second plot show shows the ELM-RLS model prediction on the validation data set.

Figure 3.17 and Figure 4.4 show the model predictions on using the conventional ELM and the proposed integrated ELM-RLS technique respectively. In comparing these figures, the model predictions of the proposed ELM-RLS technique show more accuracy in model prediction on the final biomass concentration when compared to the model prediction of the conventional ELM model predictions. This is because the RLS technique has been merged with the conventional ELM to cater for the unexpected disturbances that may arise as an error in the fermentation process system.

During the process simulations and building of the conventional ELM models for the baker's yeast fermentation, it is assumed that there is insufficient historical process data by using 29% out of the 70 % allocated training and testing simulated data which makes the ELM model to be inaccurate in its prediction and generalisation capability on the yeast fermentation process. Despite the use of insufficient data in modelling the fermentation process, which caused large model errors between the actual and the predicted data, the line plots of the model prediction in Figure 4.4 shows that the proposed ELM-RLS technique can adapt to the process changes and provide accurate model predictions with good generalisation capability.

Moreover, the MSE values shown in Table 4.2 confirms the prediction accuracy of the proposed technique. It shows that the MSE value for training and testing data decrease from 1.0369 (of the ELM model predictions) to 0.1262 (of the proposed ELM-RLS technique), and likewise the validation MSE values changes from 0.7044 (of the ELM model predictions) to 0.0281 (of the proposed ELM-RLS technique).

In general, the MSE measures the average squared errors between the predicted data and the actual data. An MSE of zero or values very close to zero like the case of the values in Table 4.2 for the proposed ELM-RLS technique, it means that the proposed technique predicts with good accuracy and good generalisation capabilities.

Provided that there are no unknown disturbances on the process operation system and there is large amount of historical process data, the ELM model would model and predict the process operation accurately without the need to merge RLS algorithms with ELM. However, the presence of unknown disturbance during real industrial process operation is inevitable.

4.5 Conclusions

An adaptive batch-to-batch modelling approach integrating ELM with RLS for modelling fed-batch processes is proposed in this Chapter. Through batch-to-batch adaptation of ELM output layer weights, the ELM model can track unknown disturbances and process drift. Such a model is very useful in batch-to-batch optimal control which will be focused on next chapter. Based on the results obtained and comparison between the conventional ELM and the proposed ELM-RLS method, the proposed method showed better modelling performance of both the isothermal fed-batch reactor process and baker's yeast fermentation process, compared to the ordinary ELM.

CHAPTER FIVE

BATCH-TO-BATCH OPTIMIZATION CONTROL

5.1 Introduction

Machine learning has recently taken the world by storm, engaging with a broad audience of practitioners, research institutes and academics. It has grown in acceptance and is used in a variety of manufacturing industries, including both sales and production stages, machine translation, speech recognition, picture recognition, recommendation systems, and many other ones. Machine learning includes optimization as an important factor and its algorithms work by creating an optimization model through learning the parameters of the objective function in the datasets. Thus, machine learning optimization is the process of modifying hyperparameters to identify the extremum (either minimize or maximize) of the cost function and satisfy the constraints. In doing this, we tend to find the discrepancy between the actual data set and the model predictions.

The popularity and implementation of machine learning models are heavily influenced by the effectiveness and efficiency of their numerical optimization techniques in the presence of large datasets. Several effective optimization strategies have been proposed to accelerate the development of machine learning models, which have enhanced the performance and efficiency of machine learning models. Generally, optimization methods are divided into three categories, namely: the first-order optimization method, which involves the commonly used stochastic gradient methods, the higher-order optimization method, and the heuristic derivative-free optimization methods (Sun et al., 2020).

The progress of optimization makes a significant contribution to the advancement of machine learning. However, there are still many shortcomings and unresolved problems in machine learning for optimization problems, such as the case of ways to improve optimization efficiency in neural networks modelling with insufficient data. This causes significant variances and overfitting in the modelling based on machine learning. Furthermore, one other major issue with neural networks is that the training usually involves non-convex optimization, which causes the optimization to produce a locally optimal solution rather than a global optimal one.

Optimization techniques have a great impact on several aspects of machine learning. This makes the machine learning problem to be conceived as an optimization problem during which the goal is to identify the extremum of the training objective function. The initial stage in machine learning approaches is to build models and develop realistic objective functions. To address the optimization problem, relevant numerical or analytical optimization methods are usually utilised with the determined objective function. Some of these well-known optimization techniques in machine learning are

(i) Exhaustive search, which involves searching for all possible solution options whether it is a good match of the extremum. This type of search is inefficient and slow when the dataset involves are hundreds and thousands.

(ii) Gradient descent is the most used numerical optimization technique. This involves finding the local minimum of a differentiable function and it is referred to as finding the partial derivatives in the cost function with respect to the decision variables. Its main goal is to minimize the cost function by searching to the position where these partial derivatives are close to zero. The major shortcoming in this method is its possibility of getting stuck to a local minimum rather than the global minimum.

(iii) Genetic algorithms use the theory of evolution in form of generic population-based metaheuristic optimization to machine learning. This is done by finding the best models among the system of multiple models by calculating the accuracy of each model. The second generation of models is derived from the initial best models which are randomly mutated to obtain other sets of best models. One possible advantage of the Genetic algorithm technique is: it has great possibility of reaching for global optimum, but it is hard to come up with its heuristic's formulation. Figures 5.1 and 5.2 give an overview description of both the Genetic algorithm and Gradient descent methods.

Thus, the key responsibilities in machine learning are to construct a model hypothesis through building a reliable model, set the objective function, and find the objective function's maximum or minimum (extremum) to decide the model's parameters. The first two steps of these 3 main processes are machine learning modelling issues, and the third step is to solve the desired model using optimization approaches.

The batch-to-batch optimization control approach for batch operations is based on the proposed ELM-RLS modelling method. The proposed ELM-RLS models are built using process operation data to address the difficulties of developing detailed mechanistic models. The optimal control policy calculated using this model may not be optimal when applied to the actual process due to model process mismatches and unknown/disturbances. The ELM-

RLS model-based iterative learning control is used to enhance the process performance from batch to batch by harnessing the repeated nature of batch processes. Thus, making the batch-to-batch control to be able to enhance the future batch's performance but cannot improve the performance of the current batch. The proposed approach of batch-to-batch optimization control is successfully applied to a simulated baker’s yeast fermentation and fed-batch reactor processes.

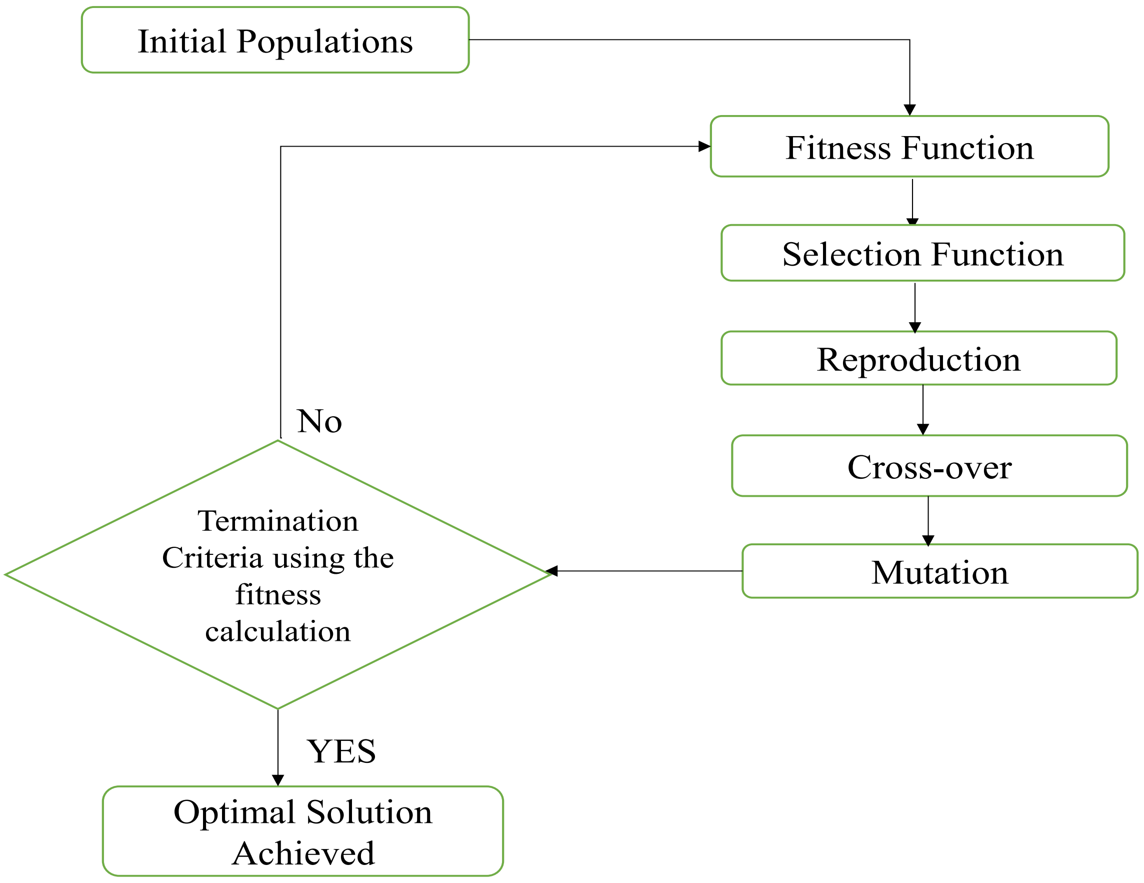


Figure 5.1 An Overview of Genetic Algorithm Technique

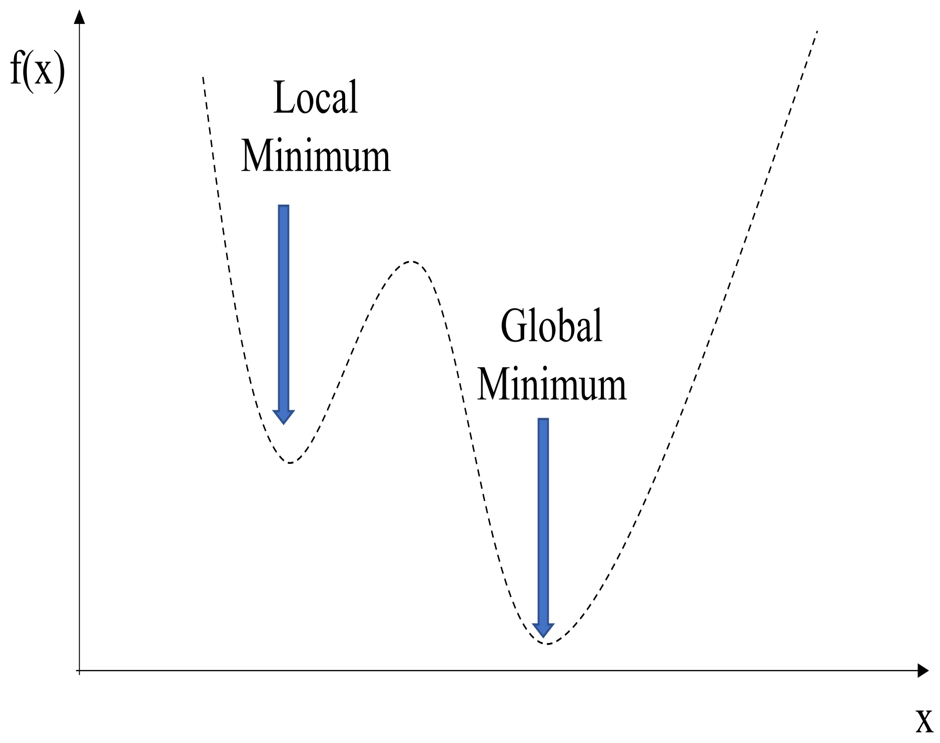


Figure 5. 2 An overview of Local and Global minimum

The rest of this chapter is organised as follows. Section 5.2 discusses the iterative batch-to-batch optimization control in relation to the baker yeast fermentation process, followed by Section 5.3 which gives the simulation results and discussion on the fermentation process and lastly, Section 5.3 states the general conclusion of the batch-to-batch optimization of the fermentation case study.

5.2 Iterative Batch-to-Batch Optimization Control

During the baker's yeast fermentation process, the substrate is constantly delivered to the fermentor in a fed-batch mode, where no cells or products are removed during the batch run. This action provides a great chance to change the substrate feeding profile, which would allow for more precise control of bioactivities such as cell biomass growth, nutrient uptake, and metabolite production. As a result, the feeding profile could have a significant impact on the intended product's productivity and output at the end of the production process. Here, the main objective will be to improve the final biomass concentration by modifying the control policy which is composed of the substrate feed rate subject to the process constraints.

Moreover, many fermentation industries are currently keen to establish an inexpensive and effective control system to reduce manufacturing costs and enhance yields while retaining the quality of the desired products. Hence, the need to work on this case study and apply the proposed technique in modelling and optimization to help in achieving the main goal of finding an updated profile of input variables that will maximise the objective function of the baker's yeast fermentation process. However, according to Jin et al. (2014), the modelling and optimization of fermentation processes are still faced with many challenges like the unpredictability of varying model parameters which causes unavoidable plant model mismatches, the nonlinear complexity and dynamic of the process operation, and the lack of robust online sensors for significant variables such as biomass or product concentration is a major hindrance to reliable optimal control of bioprocesses.

The proposed iterative optimal control of the baker's yeast fermentation is based on the static ELM-RLS model prediction given by equation (3.53) and the optimization problem can be solved with the following equation, represented as:

$$\min_{F} J = -f_{ELM\pm RLS}\big(Y(t_f)\big) \tag{5.1}$$

$$subject\ to\colon \begin{cases} 0 \le Y_i \le 3000,\ i = 1,2,\dots,10 \\ \qquad V_f \le 50000 \end{cases}$$

where $f_{ELM\pm RLS}\big(Y(t_f)\big)$ is the ELM-RLS predicted final biomass concentration and F is the vector of substrate feeding rates. The constrained optimisation problem in Eq. (5.1) was solved using the nonlinear programming (interior-point) method of "*fmincon*" function in the MATLAB 2019a Optimization Toolbox.

In general, the batch-to-batch optimization control of the fermentation process explores the repetitive nature of the fed-batch/batch processes to improve the feeding control policy by using information from the previous and the current batch runs to enhance the future batch runs of the process. This strategy has been reported by (Xiong & Zhang, 2005; Zhang, 2004a, 2004b, 2004d, 2005a) to help in controlling the declining control performance exhibited in many model-based control techniques due to model-plant mismatch and unavoidable disturbances. These model plant mismatches and disturbances always exist due to an insufficient amount of historical process operational data and sometimes from raw material variations, reactor fouling and reactive impurities. Thus, the optimal control policy is only optimal on the model and may not be optimal when applied to the real process (Zhang, 2004b) . To address this problem, (Zhang, 2004a) established a technique of ensuring a reliable optimum control policy through the incorporation of model prediction confidence bounds within the optimization objective function. This ensures penalization of any wide model prediction confidence bounds that may arise during the optimization.

The governing equation that explains the basis of the batch-to-batch optimization control has been given in equation (2.58) through equation (2.66) when discussing about the optimal model-based ILC algorithm. According to (Zhang, 2004b; Xiong et al., 2004), there may be failing of control performance which is caused by model-plant mismatches and some unavoidable disturbances. To solve this, batch-to-batch optimal control technique was developed using ILC concept with the neural network model.

In this chapter, instead of using the previously reported neural network model-based batch-to-batch optimal control technique (Zhang, 2004b; Xiong et al., 2004), the proposed ELM-RLS model will be used in batch-to-batch optimal control. Based on the yeast fermentation process established in Chapter 4 and remember for static batch modelling, the objective function is set to the function of the batch state variables at the batch end time. This is represented as given in equation (4.6) above. This equation can be expanded around a nominal control profile with Taylor series expansion as:

$$\hat{Y}(t_f) = f_0 + \frac{\partial f}{\partial u_1}\Delta u_1 + \frac{\partial f}{\partial u_2}\Delta u_2 + \ldots + \frac{\partial f}{\partial u_N}\Delta u_N \quad (5.2)$$

For the ith batch, the actual biomass concentration can be written as the model prediction and an error term, given as:

$$Y_i(t_f) = \hat{Y}(t_f) + e_i \quad (5.3)$$

where $\hat{Y}(t_f)$ and $Y_i(t_f)$ represents the predicted biomass and actual (mechanistic model simulated) biomass concentration values respectively. The e_i is the model predicted model error representing model-plant mismatches.

For model prediction at the end of the next $(k+1)th$ batch, equation (5.3) becomes:

$$\ddot{Y}_{i+1}(t_f) = \hat{Y}_{i+1}(t_f) + e_{i+1} \quad (5.4)$$

where the optimal control solution is similar to equation (2.58). Thus, the following summarizes the stepwise approach of the batch-to-batch optimal optimization control:

- An ELM model is built from historical data to predict the yielding output $[y_i(t_f)]$ using the substrate input trajectory as model inputs.
- Optimal control profile, i.e., input trajectory (U_i), is calculated using the ELM model-based optimisation.
- The substrate input trajectory (U_i) is fed into the current batch run i under the initial conditions (X_0) of the fermentation process, which results in yielding output $[y_i(t_f)]$ at the batch end time.
- Prediction errors from the previous batch runs are used in the RLS algorithm to update the output layer weights of the ELM model.
- The optimization problem is solved again using the improved model predictions from the proposed technique, and a new input control policy U_{i+1} for the next batch is formed.
- The optimal input control policy is applied to the mechanistic model simulation to obtain the desired product quality at batch end time.
- This procedure is repeated from batch to batch until the magnitude of the desired product quality value is not significantly increasing. This will indicate that the batch-to-batch optimal control technique to have converged. The schematic representation of the stepwise approach can be illustrated in Figure 5.3 below:

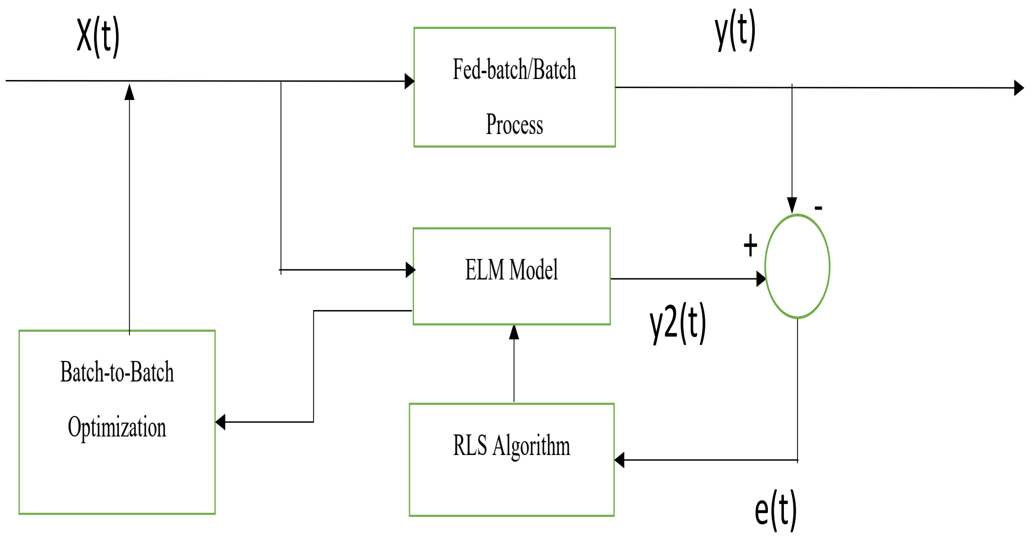


Figure 5. 3 Schematic representation of Batch-to-Batch Optimization Control

Moreover, on the other hand, the batch-to-batch control can only increase the performance of future batch runs and not the current batch run. Furthermore, batch-to-batch control is ineffective when the disturbance occurs only in a single batch rather than a series of batches. Some mid-batch process data can be used to detect the consequences of an unknown disruption in the current batch run. As a result, the impacts of unknown disturbances on the starting batch state might be inferred from these mid-batch process measures. Corrective control steps can thus be done throughout the remainder of the batch process operation, reducing any serious impacts of the unknown disturbance on the end-product quality.

5.3 Simulation results and Discussions

The optimal simulation results obtained in Table 5.1 shows a significant improvement in model accuracy and reliability of the proposed optimization method when applied to unseen data. The initial final biomass concentration increases from 40.79g/L of the ELM model prediction to 55.32g/L of the ELM-RLS model prediction values when the optimal control policy obtained is applied to mechanistic model simulation to obtain the desired biomass concentration yield at batch end time after 1st batch optimization.

Table 5.1 Optimal Control Results on Final Biomass Concentration

Models	Optimal biomass value (g/L)	Final biomass value from new policy (g/L)
ELM	77.04	40.79
ELM-RLS	89.39	55.32

In this fermentation case study, the desired final product quality is the biomass concentration. Among the 100 simulated batches of the process operation, the final biomass concentration is in the range of 32.48 g/L to 62.23 g/L with each batch ending time at 16.5 hours. Based on the model-based modelling techniques using both the proposed technique (ELM-RLS) and the ELM model, the optimal control policy was calculated together with the biomass optimal yield to obtain new substrate feeding policy. The optimal control new policy was later applied to the mechanistic model simulation to attain the desired biomass yield at batch end time. The result obtained is given in Table 5.1 above and it shows how the model predicted final biomass concentration improved from 62.23 g/L to 77.04g/L after optimization of the process model with an ELM and later improved to 89.39 g/L when the proposed modelling technique (ELM-RLS) was used.

Aside obtaining the optimal biomass yield from optimization, the new control policy obtained was reapplied on the mechanistic model simulation to check the biomass yield concentration value. This is also shown in Table 5.1 for both the ELM and the proposed ELM-RLS technique, where it increases from 40.79 g/L on ELM model to 55.32 g/L on ELM-RLS model. The biomass concentration yield obtains on the mechanistic models reapplication of new control policy shows that the optimal control policy on model-based modelling is not optimal when applied to the real process. This is as result of the significance differences between the biomass concentration yield obtained with initial feeding control policy and the optimal control policy (either on the ELM or the proposed ELM-RLS).

Table 5. 2 Optimal Control Results on ELM-RLS Model Simulation

Re-optimization concentration	Optimal biomass value (g/L)	Final biomass from new policy
1st	89.39	55.32
2nd	108.45	65.75
3rd	110.68	69.57
4th	115.21	70.65

The results shown in Table 5.2 are from applying the proposed batch-to-batch optimal control technique for 4 consecutive batches. The biomass concentration yield on the mechanistic model improves significantly from 55.32 g/L of 1st batch optimization to 65.75g/L, 69.57g/L and to 70.65g/L of 2nd, 3rd, and 4th batch optimization respectfully. This signifies that the batch-to batch optimal control technique has almost converge when the previous batch yield is insignificantly increased as in the case of the results of the 3rd and 4th batch.

5.4 Conclusions

A batch-to-batch optimal control strategy of a fed-batch fermentation process is proposed in this chapter using integrated ELM-RLS method model-based. Historical process operation data is used to establish an initial ELM model and to cope with the presence of unknown disturbances, the ELM model is updated after each batch using RLS. The updated ELM with RLS is used to find the optimal control policy for the next batch. The proposed batch-wise modelling and optimisation control strategy is demonstrated on a simulated fed-batch fermentation process for producing baker's yeast. It has been shown that the proposed adaptive optimal control strategy can effectively overcome the effect of unknown disturbances and achieve improved product output operation from batch to batch.

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH WORKS

6.1 Conclusions

The major goal of many manufacturing industries would be to cut the production costs and increases the output yield while retaining or even improving the quality of the desired products. To accomplish this, either the mechanistic, data-driven modelling or even both methods should be deployed in process optimisation and control.

Data-driven modelling techniques are an alternative way to mechanistic modelling and do not demand a full understanding of the process description, unlike the mechanistic modelling that does. Aside from this, mechanistic modelling sometimes is inaccurate due to the unavailability of some model parameters and uncertain reaction pathways, and this results in model-plant mismatches. Moreover, building an accurate model for processes seems a difficult task due to the incomplete knowledge of physicochemical properties.

However, in this research, different data-driven modelling method was explored in modelling both the isothermal fed-batch reactor systems and the baker's yeast fermentation processes in Chapter 3 of this thesis. The data-driven models investigated here for batch process modelling are categorized into two groups: the dynamic and the static models. The explored computational intelligence techniques are ELM, BA-ELM, ILC and the proposed ELM-RLS. A small percentage (25-30%) of the historical process data were used in static modelling with all the modelling methods to show the impacts of insufficient process data in building the models.

BA-ELM were found to be more robust, reliable and provide an accurate model prediction even with the insufficient process data used in building the model. This is due to the re-sampling replications of the original training datasets and the combination of multiple models which can guard against the failure of single models.

To improve the model accuracy and reliability of an ELM model with insufficient historical process data used, Chapter 4 proposed an adaptive batch-to-batch modelling approach that dealt with ways of integrating an ELM with RLS for modelling batch and fed-batch processes. Through the batch-to-batch adaptation of ELM output layer weights with the RLS algorithm, the ELM model can reduce the model-plant mismatches caused by unknown disturbances, insufficient datasets, and unforeseen disturbances. Such a reliable model is very useful for both the batch-to-batch optimal control and for soft sensors developments for processes monitoring.

The proposed ELM-RLS technique was applied to both the isothermal fed-batch reactor process and the baker's yeast fermentation. Based on the results obtained and comparison between the conventional ELM and the proposed ELM-RLS method, the proposed method in both case studies showed better modelling performance compared to the ordinary ELM.

Furthermore, Chapter 5 proposed batch-to-batch optimal control strategy of the fed-batch fermentation process by using the proposed integrated ELM-RLS method model-based obtained in Chapter 4. Historical process operation data is used to establish an initial ELM model and, to cope with the presence of unknown disturbances, the ELM model is updated after each batch using RLS. The updated ELM with RLS is used to find the optimal control policy for the next batch. The proposed batch-wise modelling and optimisation control strategy is demonstrated on a simulated fed-batch fermentation process for producing baker's yeast. It has been shown that the proposed adaptive optimal control strategy can effectively overcome the effect of unknown disturbances and achieve improved product output operation from batch to batch.

6.2 Recommendations for Future Works

From the knowledge gained while working on this research over the years, some of these recommendations and suggestions may be of great importance to further the scope of the research. These are stated as follows:

- All historical process data used for developing the model were simulated and generated from the mechanistic model equations represented in different articles. The model equation representation in terms of the chemical reaction kinetics, mass balances, and physicochemical properties for both case studies was all assumed to be accurate from the journals and simulated with an ODE45 in MATLAB to generate the datasets. To practically demonstrate the developed techniques, real industrial process data should be used, and the modelling and optimisation control strategies developed should be tested on real industrial processes.

- Future work should be done in using the developed batch-to batch optimization control to develop soft sensors for process monitoring and control.

- Future work should be done in integrating the ELM and ILC to improve the model reliability and control. Aside from integrating ELM with ILC, ELM with other statistical techniques should also be worked on.

References

Adnan, R., Ruslan, F.A., Samad, A.M. and Zain, Z.M. (2012) *2012 IEEE International Conference on Control System, Computing and Engineering*. 23-25 Nov. 2012.

Agatonovic-Kustrin, S. and Beresford, R. (2000) 'Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research', *Journal of Pharmaceutical and Biomedical Analysis*, 22(5), pp. 717-727.

Ahmad, Z. and Zhang, J. (2005) 'Bayesian selective combination of multiple neural networks for improving long-range predictions in nonlinear process modelling', *Neural Computing & Applications*, 14(1), pp. 78-87.

Ahn, H., Chen, Y. and Moore, K.L. (2007) 'Iterative learning control: brief survey and categorization', *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6), pp. 1099-1121.

Ahn, H., Lee, K.S., Kim, M. and Lee, J. (2014) 'Control of a reactive batch distillation process using an iterative learning technique', *Korean Journal of Chemical Engineering*, 31(1), pp. 6-11.

Alli, K. and Zhang, J., 2020. Adaptive Modelling of Fed-batch Processes with Extreme Learning Machine and Recursive Least Square Technique. In *ICAART (2)* (pp. 668-674).

Amann, N., Owens, D.H. and Rogers, E. (1996) 'Iterative learning control for discrete-time systems with exponential rate of convergence', *IEE Proceedings - Control Theory and Applications*, 143(2), pp. 217-224 [Online]. Available at: https://digital-library.theiet.org/content/journals/10.1049/ip-cta_19960244.

Bai, M., Huang, J., Hong, M. and Su, F. (2005) 'Fault diagnosis of rotating machinery using an intelligent order tracking system', *Journal of Sound and Vibration*, 280(3), pp. 699-718.

Baron, C.M.C. and Zhang, J. (2017) *International Conference on Informatics in Control, Automation and Robotics*. Springer.

Barton, A.D., Lewin, P.L. and Brown, D.J. (2000) 'Practical implementation of a real-time iterative learning position controller', *International Journal of Control*, 73(10), pp. 992-999.

Bishop, C. (1991) 'Improving the generalization properties of radial basis function neural networks', *Neural Computation*, 3(4), pp. 579-588.

Bishop, C.M. (1995) *Neural Networks for Pattern Recognition*. Oxford University Press.

Bonvin, D. (1998) 'Optimal operation of batch reactors—a personal view', *Journal of Process Control*, 8(5-6), pp. 355-368.

Bristow, D.A., Tharayil, M. and Alleyne, A.G. (2006) 'A survey of iterative learning control', *IEEE Control Systems Magazine*, 26(3), pp. 96-114.

Chen, S., Billings, S.A. and Grant, P.M. (1990) 'Non-linear system identification using neural networks', *International Journal of Control*, 51(6), pp. 1191-1214.

Chi Sing, L., Kwok Wo, W., Pui Fai, S. and Lai Wan, C. (1996) 'On-line training and pruning for recursive least square algorithms', *Electronics Letters*, 32(23), pp. 2152-2153.

Choi, S.W., Morris, J. and Lee, I.-B. (2008) 'Dynamic model-based batch process monitoring', *Chemical Engineering Science*, 63(3), pp. 622-636.

Dong-Il, K. and Sungkwun, K. (1996) 'An iterative learning control method with application for CNC machine tools', *IEEE Transactions on Industry Applications*, 32(1), pp. 66-72.

ENGINEERS'HANDBOOK, I. (2006) 'Process Control and Optimization'. CRC press, BG Liptak, Florida, USA.

Freeman, C.T., Lewin, P.L. and Rogers, E. (2005) 'Experimental evaluation of iterative learning control algorithms for non-minimum phase plants', *International Journal of Control*, 78(11), pp. 826-846.

Freeman, C.T., Zhonglun, C., Lewin, P.L. and Rogers, E. (2009) *2009 American Control Conference*. 10-12 June 2009.

Gaensler, T. and Benesty, J. (2004) 'New Insights into the RLS Algorithm', *EURASIP Journal on Advances in Signal Processing*, 2004.

Geladi, P. and Kowalski, B.R. (1986) 'Partial least-squares regression: a tutorial', *Analytica Chimica Acta*, 185, pp. 1-17.

Guang-Bin, H. and Chee-Kheong, S. (2004) *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, 2004.*, 6-9 Dec. 2004.

Havlicsek, H. and Alleyne, A. (1999) *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*. 2-4 June 1999.

Haykin, S.S. (2009) *Neural Networks and Learning Machines*. Prentice Hall.

Hsu, H.-H., Chang, C.-Y. and Hsu, C.-H. (2017) *Big Data Analytics for Sensor-network Collected Intelligence*. Morgan Kaufmann.

Huang, G.-B., Zhu, Q.-Y. and Siew, C.-K. (2006) 'Extreme learning machine: Theory and applications', *Neurocomputing*, 70(1), pp. 489-501.

Huang, G., Huang, G.-B., Song, S. and You, K. (2015) 'Trends in extreme learning machines: A review', *Neural Networks*, 61(Supplement C), pp. 32-48.

Jeong, D.H. and Lee, J.M. (2018) 'Enhancement of modifier adaptation scheme via feedforward decision maker using historical disturbance data and deep machine learning', *Computers & Chemical Engineering*, 108, pp. 31-46.

Jie, Z. (2002) *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*. 8-10 May 2002.

Jie, Z. (2003) *Proceedings of the 2003 IEEE International Symposium on Intelligent Control*. 8-8 Oct. 2003.

Karakuzu, C. (2003) *Modeling and control of industrial baker yeast fermentation using neural networks and fuzzy logic*. Ph. D. thesis. Turkey: University of Kocaeli.

Krishnamoorthy, K. and Tsao, T.-C. (2004) 'Repetitive Learning Control for Precision Machining of Complex Profiles', (47063), pp. 63-69.

Kuc, T.-Y., Lee, J.S. and Nam, K. (1992) 'An iterative learning control theory for a class of nonlinear dynamic systems', *Automatica*, 28(6), pp. 1215-1221.

Kulkarni, S.G., Chaudhary, A.K., Nandi, S., Tambe, S.S. and Kulkarni, B.D. (2004) 'Modeling and monitoring of batch processes using principal component analysis (PCA) assisted generalized regression neural networks (GRNN)', *Biochemical Engineering Journal*, 18(3), pp. 193-210.

Lee, J.H., Lee, K.S. and Kim, W.C. (2000) 'Model-based iterative learning control with a quadratic criterion for time-varying linear systems', *Automatica*, 36(5), pp. 641-657.

Levenspiel, O. (1999) 'Chemical reaction engineering'. John Wiley & Sons.

Li, F., Zhang, J., Oko, E. and Wang, M. (2017) 'Modelling of a post-combustion CO2 capture process using extreme learning machine', *International Journal of Coal Science & Technology*, 4(1), pp. 33-40.

Liu, X., Liu, L., Wang, L., Guo, Q. and Peng, X. (2019) 'Performance sensing data prediction for an aircraft auxiliary power unit using the optimized extreme learning machine', *Sensors*, 19(18), p. 3935.

Liu, Y., Yang, X., Xiong, Z. and Zhang, J. (2005) *International Symposium on Neural Networks*. Springer.

Longman, R.W. (2000) 'Iterative learning control and repetitive control for engineering practice', *International Journal of Control*, 73(10), pp. 930-954.

MacGregor, J.F. and Kourti, T. (1995) 'Statistical process control of multivariate processes', *Control Engineering Practice*, 3(3), pp. 403-414.

MacKay, D.J. (1992) 'A practical Bayesian framework for backpropagation networks', *Neural computation*, 4(3), pp. 448-472.

Moore, K.L. (1999) 'Iterative Learning Control: An Expository Overview', in Datta, B.N. (ed.) *Applied and Computational Control, Signals, and Circuits: Volume 1*. Boston, MA: Birkhäuser Boston, pp. 151-214.

Mukherjee, A. and Zhang, J. (2008) 'A reliable multi-objective control strategy for batch processes based on bootstrap aggregated neural network models', *Journal of Process Control*, 18(7-8), pp. 720-734.

Noor, R.A.M., Ahmad, Z., Don, M.M. and Uzir, M.H. (2010) 'Modelling and control of different types of polymerization processes using neural networks technique: A review', *The Canadian Journal of Chemical Engineering*, 88(6), pp. 1065-1084.

Norrlöf, M. and Gunnarsson, S. (2002) 'Time and frequency domain convergence properties in iterative learning control', *International Journal of Control*, 75(14), pp. 1114-1126.

Nwankpa, C., Ijomah, W., Gachagan, A. and Marshall, S. (2018) 'Activation functions: Comparison of trends in practice and research for deep learning', *arXiv preprint arXiv:1811.03378*.

Oh, S.-K. and Lee, J.M. (2016) 'Iterative learning model predictive control for constrained multivariable control of batch processes', *Computers & Chemical Engineering*, 93, pp. 284-292.

Rezaeizadeh, A. and Smith, R.S. (2018) 'Iterative Learning Control for the Radio Frequency Subsystems of a Free-Electron Laser', *IEEE Transactions on Control Systems Technology*, 26(5), pp. 1567-1577.

Ruppen, D., Benthack, C. and Bonvin, D. (1995) 'Optimization of batch reactor operation under parametric uncertainty — computational aspects', *Journal of Process Control*, 5(4), pp. 235-240.

Sonnleitner, B. and Käppeli, O. (1986) 'Growth of Saccharomyces cerevisiae is controlled by its limited respiratory capacity: formulation and verification of a hypothesis', *Biotechnology and bioengineering*, 28(6), pp. 927-937.

Tao, K.M., Kosut, R.L. and Aral, G. (1994) *Proceedings of 1994 American Control Conference - ACC '94*. 29 June-1 July 1994.

Tayebi, A. (2004) 'Adaptive iterative learning control for robot manipulators', *Automatica*, 40(7), pp. 1195-1203.

Tian, Y., Zhang, J. and Morris, J. (2001) 'Modeling and optimal control of a batch polymerization reactor using a hybrid stacked recurrent neural network model', *Industrial & engineering chemistry research*, 40(21), pp. 4525-4535.

Togai, M. and Yamano, O. (1985) *1985 24th IEEE Conference on Decision and Control.* 11-13 Dec. 1985.

Wallén, J., Norrlöf, M. and Gunnarsson, S. (2011) 'A framework for analysis of observer-based ILC', *Asian Journal of Control*, 13(1), pp. 3-14.

Wang, Y., Cao, F. and Yuan, Y. (2011) 'A study on effectiveness of extreme learning machine', *Neurocomputing*, 74(16), pp. 2483-2490.

Wigren, T. (1993) 'Recursive prediction error identification using the nonlinear wiener model', *Automatica*, 29(4), pp. 1011-1025.

Xiong, Z. and Zhang, J. (2003) 'Product Quality Trajectory Tracking in Batch Processes Using Iterative Learning Control Based on Time-Varying Perturbation Models', *Industrial & Engineering Chemistry Research*, 42(26), pp. 6802-6814.

Xiong, Z. and Zhang, J. (2004) 'Batch-to-batch optimal control of nonlinear batch processes based on incrementally updated models', *IEE Proceedings - Control Theory and Applications*, 151(2), pp. 158-165.

Xiong, Z. and Zhang, J. (2005) 'A batch-to-batch iterative optimal control strategy based on recurrent neural network models', *Journal of Process Control*, 15(1), pp. 11-21.

Xiong, Z., Zhang, J., Wang, X. and Xu, Y. (2004a) 'Run-to-Run Iterative Optimization Control of Batch Processes Based on Recurrent Neural Networks', in Yin, F.-L., Wang, J. and Guo, C. (eds.) *Advances in Neural Networks - ISNN 2004: International Symposium on Neural Networks, Dalian, China, August 19-21, 2004, Proceedings, Part II.* Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 97-103.

Xiong, Z., Zhang, J., Wang, X. and Xu, Y. (2004b) Berlin, Heidelberg. Springer Berlin Heidelberg.

Xiong, Z.H., Xu, Y.M., Wang, X. and Zhang, J. (2007) 'Integrated tracking control strategy for batch processes using a batch-wise linear time-varying perturbation model', *IET Control Theory & Applications*, 1(1), pp. 179-188.

Xu, J.-X., Chen, Y., Lee, T.H. and Yamamoto, S. (1999) 'Terminal iterative learning control with an application to RTPCVD thickness control', *Automatica*, 35(9), pp. 1535-1542.

Yuxin, C., Le-Ngoc, T., Champagne, B. and Changjiang, X. (2004) 'Recursive least squares constant modulus algorithm for blind adaptive array', *IEEE Transactions on Signal Processing*, 52(5), pp. 1452-1456.

Yüzgeç, U., Türker, M. and Hocalar, A. (2009) 'On-line evolutionary optimization of an industrial fed-batch yeast fermentation process', *ISA Transactions*, 48(1), pp. 79-92.

Zhang, J. (1999) 'Developing robust non-linear models through bootstrap aggregated neural networks', *Neurocomputing*, 25(1), pp. 93-113.

Zhang, J. (2001) 'Developing robust neural network models by using both dynamic and static process operating data', *Industrial & engineering chemistry research*, 40(1), pp. 234-241.

Zhang, J. (2004) 'A reliable neural network model based optimal control strategy for a batch polymerization reactor', *Industrial & Engineering Chemistry Research*, 43(4), pp. 1030-1038.

Zhang, J. (2008) 'Batch-to-batch optimal control of a batch polymerisation process based on stacked neural network models', *Chemical Engineering Science*, 63(5), pp. 1273-1281.

Zhang, J., Martin, E.B., Morris, A.J. and Kiparissides, C. (1997) 'Inferential estimation of polymer quality using stacked neural networks', *Computers & Chemical Engineering*, 21, pp. S1025-S1030.

Zhang, J., Morris, A.J., Martin, E.B. and Kiparissides, C. (1998) 'Prediction of polymer quality in batch polymerisation reactors using robust neural networks', *Chemical Engineering Journal*, 69(2), pp. 135-143.

Appendices

Appendix A

1. Principal Component Analysis

Principal Component Analysis (PCA) is the analysis of inconsistency in a particular set of data by shrinking the dimension of the specific dataset through inspecting number of direct groupings that can be used to represent the original variables without loss of key information. In other words, it eliminates redundancy by finding a combination of features that captures important discrepancies between the original data features and brings out strong concise patterns in the dataset. The primary objectives of PCA are data summarisation, grouping of variables and fault detection which has extensive applications in both industries and research institutions.

The PCA decompose the data into principal components (PCs) with each component representing the total inconsistency to original data. The first PC have the greatest discrepancy followed by the subsequent PCs. Therefore, the first and second PCs captured virtually all the variability to the original data with other PCs as mere random noise (MacGregor and Kourti, 1995). All these can be explained and represented mathematically as follows:

2. Principal component

Let Y be a $K \times l$ matrix with each of its column representing the variables of the process and each row representing specific sample time datasets. The dataset Y can be decomposed into the summation of l outer product of vectors as:

$$Y = p_1 t_1^T + p_2 t_2^T + \cdots + p_l t_l^T \qquad 2.1$$

In Eq (2.1), $p_i \epsilon R^k$ is called the ith score vector and $t_i \epsilon R^l$ is called the ith load vector. The score vectors of Y are also called the principal components of Y. Eq (2.1) can be written in the following matrix form:

$$Y = PT^T \qquad 2.2$$

where $P = [p_1 \, p_2 \, ... p_l \,]$ is called the score matrix and $T = [t_1 \, t_2 \, ... t_l \,]$ is called the loading matrix. Score and loading vectors are mutually orthogonal if and only if $i \neq j, \; p_i^T p_j = 0$ and $t_i^T t_j = 0$ 2.3

$$t_i^T t_j = 1 \qquad \text{for } \; i = j \tag{2.4}$$

Multiplying both sides of Eq (2.1) by t_i, we can get the following

$$Yt_i = p_1 t_1^T t_i + p_2 t_2^T t_i + \cdots + p_l t_l^T t_i \tag{2.5}$$

Substituting Eq (2.3) and Eq (2.4) into Eq (2.5) gives

$$p_i = Yt_i \tag{2.6}$$

Eq (2.6) represents the *i*th score vector p_i as the forecast of the dataset Y on the corresponding loading vector. Thus, the length p_i reveals the deviation of Y in the direction of t_i . If the length of the score vectors are arranged in descending order as follows:

$\| \, p_1 \, \| > \| \, p_2 \, \| > \cdots > \| \, p_l \, \|$

The loading vector t_1 represents the largest variation in Y while t_2 represents the second largest direction of variation in Y, and t_l represents the smallest direction of variation in Y. According to (MacGregor and Kourti, 1995) the discrepancy in Y dataset are captured in the first few PC's decomposition, provided there is a direct correlation between the variables of Y. The first two or three PCs are enough to explain the discrepancies of Y from the original dataset without considering other decomposed PC's. Therefore, the PCA decomposition of Y can be denoted as follows:

$$Y = p_1 t_1^T + p_2 t_2^T + \cdots + p_k t_k^T + \cdots + E \tag{2.7}$$

where *E* is the residual matrix representing the variation of Y in the directions of t_{k+1} to t_m. In many practical applications, *k* is usually much smaller than *l*. Since *E* mainly contains noise, ignoring E has the effect of noise filtering and will not cause any significant loss of useful information. Therefore, the data set Y can be approximated as :

$$Y = p_1 t_1^T + p_2 t_2^T + \cdots + p_k t_k^T = \textstyle\sum_{i=1}^{l} p_i t_i^T \tag{2.8}$$

The PCA of Y is equivalent to the eigenvector analysis of the covariance matrix of Y and $Y^T Y$ and the loading vectors of Y are the eigenvectors of $Y^T Y$ such that if the eigenvalues of $Y^T Y$ is arranged in descending order in the following sequence: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_l$, then the corresponding eigenvectors $t_1, t_2, \ldots , \; t_l$ are the loading vectors of Y.

PCA can also be solved with singular value decomposition (SVD) technique. The SVD concept involves determining the discrepancy in Y which can be expressed as:

$$Y = U\sum V^T \qquad 2.9$$

where

$$U = [u_1\, u_2\ \ldots\ u_n] \in\ R^{n\times n}$$

$$U = [v_1\, v_2\ \ldots\ v_m] \in\ R^{m\times m}$$

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_m \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in R^{nxm}$$

In the above equation, $\sigma_1 > \sigma_2 > \ldots > \sigma_m$ are the singular values of Y. the singular values of Y are the square root of the eigenvalues of its covariance matrix Y^TY.

Where

$$\sigma_1 = \sqrt{\lambda_1}$$

$$\sigma_2 = \sqrt{\lambda_2}$$

$$\ldots$$

$$\sigma_m = \sqrt{\lambda_m}$$

The columns in U are orthogonal and of unit length, so are the columns in *V*. Therefore, Eq (2.9) can be represented as:

$$Y = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \ldots + \sigma_m u_m v_m^T \qquad 2.10$$

If v_i is denoted as t_i and $\sigma_i u_i$ as p_i, then Eq (2.10) becomes Eq (2.1) which implies that $\sigma_1 u_1$ is the first score vector while v_1 is the loading vector of Y.

The principal components of Y vary with the scales used for the variables in Y. According to Zhang (2018, p.15), "when applying PCA concept, the ideal practice is to scale the data to zero mean and unit variance in order not to obtain different numerical values because same variable can have different unit.”

3. Linear Regression

Regression analysis is one of the widely used analytical modelling methods that explore the relationship between a dependent variable and one or more independent variables using a linear function. The regression analysis concept can either be linear (when it involves a dependent variable and one independent variable) or multiple linear (when it involves a dependent variable and two or more independent variables) regression (MLR). The method involves building a model from the available data to forecast desired responses and the built model served as the summary representation of the whole available data. This technique is mainly used for prediction in time series modelling or finding the fundamental relationships that exist within the variables.

Linear regression models are one of the most widely used to be fitted to a new dataset because it is simple to interpret and easy to train, they are often used as a baseline for evaluating other more complex regression models. Some of the linear regression types are as follows:

a. Single variable least squares

Theoretically, the fitting of a straight line gives no great difficulty. By selecting any two points on the straight line $y = mx + C$, both the slope (m) and the intercept (C) coefficients are calculated. For instance, considering these two points (x_1, y_1) and (x_2, y_2), the slope can be found as:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \qquad 2.11$$

Once m has been obtained, the intercept, C, can be found by solving the equation of the straight line:

$$C = y_1 - mx_1 \qquad 2.12$$

While this procedure provides exact, correct coefficients using points that lie on a straight line, experimental points rarely (if ever) do so. This is indicated by the expression generally used to represent a regression model with a single predictor variable:

$$y = mx + c + \varepsilon \qquad 2.13$$

Where the additional term in Eq (2.13), ε, signifies that for a fixed value of the predictor variable the observed value of the response will not lie exactly on the line $y = mx + C$, but will be in error by an amount ε.

b. The principle of least squares

To predict responses for the model equation (2.14) below,

$$y = \alpha + \beta x + \varepsilon \qquad 2.14$$

it is natural to select a predicting equation of the form:

$$\hat{y} = \hat{\alpha} + \hat{\beta} x + \varepsilon \qquad 2.15$$

Where $\hat{\alpha}$ and $\hat{\beta}$ are the estimates of the true intercept, $\hat{\alpha}$ and slope, $\hat{\beta}$, respectively. A measured of how well $\hat{y}$ predicts the response variable y is the magnitude of the residual, r, the difference between y and $\hat{y}$:

$$r = y - \hat{y} = y - (\hat{\alpha} + \hat{\beta} x) \qquad 2.16$$

Ideally, one would like the magnitudes of the residuals to be as small as possible, close to zero. Provided that $\varepsilon = 0$ for all responses, if not, all observed responses will not lie on a straight line and no prediction equations in the above form will fit the data points exactly. Hence no linear prediction equations can make all the residuals zero. In this case, to find the values of intercept, $\hat{\alpha}$ and slope, $\hat{\beta}$, which would minimise the sum of the squared residuals for all the observations.

The sum of squared residuals can be written as:

$$J = \sum_{i=1}^{N} r_i^2 = \sum_{i=1}^{N} (y_i - \hat{\alpha} - \hat{\beta} x_i)^2 \qquad 2.17$$

Differentiating equation (2.17) with respect to $\hat{\alpha}$ and $\hat{\beta}$, we have

$$\frac{dJ}{d\hat{\alpha}} = -2\sum_{i=1}^{N}(y_i - \hat{\alpha} - \hat{\beta} x_i) = 2N\hat{\alpha} - 2\sum_{i=1}^{N} y_i + 2\hat{\beta}\sum_{i=1}^{N} x_i \qquad 2.18$$

$$\frac{dJ}{d\hat{\beta}} = -2\sum_{i=1}^{N}(y_i - \hat{\alpha} - \hat{\beta} x_i)x_i = 2\sum_{i=1}^{N} y_i x_i + 2\hat{\alpha}\sum_{i=1}^{N} x_i + 2\hat{\beta}\sum_{i=1}^{N} x_i^2 \qquad 2.19$$

At the optimum value of equation (2.18) and (2.19), the gradients should be zero. Equating the two equations to zero and solving for $\hat{\alpha}$ and $\hat{\beta}$, we have:

$$\hat{\beta} = \frac{\sum_{i=1}^{N} y_i x_i - (\sum_{i=1}^{N} y_i - \sum_{i=1}^{N} x_i)/N}{\sum_{i=1}^{N} {x_i}^2 - (\sum_{i=1}^{N} x_i)^2/N} \qquad 2.20$$

$$\hat{\alpha} = \frac{\sum_{i=1}^{N} y_i - \hat{\beta} \sum_{i=1}^{N} x_i}{N} \qquad 2.21$$

c. Multi-variable Least Squares

Consider multi-input single output (MISO) models, the extension to multi-input multi- output (MIMO) models is straightforward. Consider the following model:

$$y = \sum_{i=1}^{n} \theta_i x_i + \varepsilon \qquad 2.22$$

where y represents the response variable, $x_1, x_2, \dots, x_n$ are the n predictor variables and ε is the random error term. The prediction equation will be of the form:

$$\hat{y} = \sum_{i=1}^{n} \hat{\theta}_i x_i \qquad 2.23$$

where$\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_n$are suitable estimate of the unknown regression coefficients $\theta_1, \theta_2, \dots, \theta_n$. The model prediction residual of the equation (2.23) is

$$e = y - \hat{y} = y - \sum_{i=1}^{n} \hat{\theta}_i x_i \qquad 2.24$$

For N observations, the following matrix equation can be formulated

$$\hat{Y} = X\hat{\theta} \qquad 2.25$$

$$E = Y - \hat{Y} = Y - X\hat{\theta} \qquad 2.26$$

Where $Y \epsilon R^{N\times 1}$ is a vector observed responses, $\hat{Y} \epsilon R^{N\times 1}$ is a vector of model predictions, $\hat{X} \epsilon R^{N\times n}$ is matrix predictors, $E \epsilon R^{N\times 1}$ is a vector for model residuals and $\hat{\theta} \epsilon R^{n\times 1}$ is a vector of model parameters.

The sum of squared residuals can be written as

$$J = E^T E = \left(Y - X\hat{\theta}\right)^T \left(Y - X\hat{\theta}\right)$$

$$= Y^T Y - 2Y^T X\hat{\theta} + \hat{\theta}^T X^T X\hat{\theta} \qquad 2.27$$

Differentiating J with respect to $\hat{\theta}$, we have

$$\frac{dJ}{d\hat{\theta}} = -2X^T Y + 2X^T X\hat{\theta} \qquad 2.28$$

When the model parameters are at their peak figures, the above slope vector should be a zero vector. Equating Eq (2.28) to a zero vector, we have the following matrix equation

$$-2X^TY + 2X^TX\hat{\theta} = 0 \qquad 2.29$$

Which is known as the normal equation. Solving the normal equation for $\hat{\theta}$, we have

$$\hat{\theta} = (X^TX)^{-1}X^TY \qquad 2.30$$

The above equation also applies to multi-input multi-output models where Y is a matrix of observed responses.

Appendix B

1. Principal Component Regression

Principal component regression (PCR) is a regression technique that basically compute principal components (PCs) and provides maximum discrepancy of variables in a dataset using least squares principle to optimize the predictive ability of the model. Thus, it only extracts the score vectors to describe the observed variability in the independent variables without considering the dependent variable.

According to Zhang (2018, p.18) when the number of independent variables correlated is large compared to number of observations due to noise, the matrix $X^T X$ will be of full rank and likely to be singular. Thus, in such case, using the least squares concept or multiple linear regression will result to significant errors in model parameters correlations and PCR will be helpful in providing solution to the problem of full rank.

The predictor matrix, X, is substituted with its score's matrix, T, which comprises of the major score vectors.

$$T = XP \qquad 2.31$$

with the multiple linear regression formula of the form:

$$Y = TB + E \qquad 2.32$$

where B is a vector of model parameters. The least square estimation of B is given as

$$B = (T^T T)^{-1} T^T Y \qquad 2.33$$

when the estimated model parameter vector B in Eq (2.33) is substituted into the model parameter vector $\hat{\theta}$ in the following Eq (2.34), it resulted to:

$$\hat{\theta} = PB = P(T^T T)^{-1} T^T Y \qquad 2.34$$

Thus, the least model error on the testing data is selected when the PCR models are tested on the testing data set and the number of principal components to be used can be determined from cross validation as well.

2. Partial Least Square Regression

Partial least square regression (PLS-R) is another form of statistical technique that combine both MLR and PCR techniques. PLS-R is different from PCR only that it takes cognisance of dependent variable while capturing the observed discrepancy in the independent variables and tries to achieve maximum correlation (dimension reduction) between the response and independent variables. Detail explanation on PLS-R techniques can be read on journal presented by (Geladi and Kowalski, 1986).

PLS-R is particularly useful when there is multicollinearity (i.e. independent variables is large and there exist high collinearity) between X and Y. In other word, when predictor matrix has more variables than observations. The mathematical representation of PLS-R is given as follows:

The outer relationship for the input data matrix *X* can be written as:

$$X = TP^T + E = \sum_{h=1}^{a} t_h P_h^T + M \qquad 2.35$$

The outer relationship for the output data matrix Y can be written as:

$$Y = UQ^T + F = \sum_{h=1}^{a} u_h q_h^T + N \qquad 2.36$$

If enough eigenvectors (i.e., sufficiently large *a*) are used, then both *M* and *N* can be made zero. The main goal of PLS-R modelling is to forecast the relationship (by predicting Y from X) between *X* and *Y* while reducing the magnitude of N as minimum as desired. The inner relationship in PLSR can be represented as:

$$\hat{u}_h = b_h t_h \qquad 2.37$$

where $b_h = {t_h^T u_h}/{t_h^T t_h}$. Here b_h can be referred to same as the regression parameter in MLR or PCR. According to Zhang (2018, p.19) "If equation (2.35) and (2.36) is used to calculate the principal components, then the resulting model is not the most desirable model. This is because the principal components of *X* and Y are calculated under the condition that X and Y are independent and, therefore, there may not exists a strong relationship between u_h and t_h. The objective of modelling is to maximally explain the output Y using input X".

3. Nonlinear Regression

Nonlinear regression is a statistical modelling method that helps to explain the nonlinear relationships in dataset into linear or explainable form. The nonlinear regression models are commonly regarded as parametric because all the modelling equations representing the model are described as a nonlinear equation (nonlinear refers to a fit of function that is not linear in function of the parameters). This type of regression is always useful where data has strong nonlinear trends and cannot be easily transformed into a linear space.

One area that the nonlinear regression is applicable is through the extension of the independent variables by incorporating nonlinear function terms in the original independent variables. The nonlinear transformations term includes logarithmic transformation, sinusoidal transformation, and polynomial transformation and sometimes a nonlinear model can be transformed directly into linear model by applying some mathematical transformations such as log or semi-log transformation. Some of this nonlinear regression can be of the form:

- Nonlinear regression through least squares optimisation
- Orthogonal least squares regression.