

**Multivariate Process Monitoring Of Structural Changes in A CSTR
System**

by

Mohd Hanif Bin Ishak

Dissertation submitted in partial fulfilment of
the requirements for the
Bachelor of Engineering (Hons)
(Chemical Engineering)

MAY 2013

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

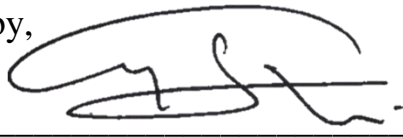
Multivariate Process Monitoring Of Structural Changes in A CSTR System

by

Mohd Hanif Bin Ishak

A project dissertation submitted to the
Chemical Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(CHEMICAL ENGINEERING)

Approved by,



(Ir. Dr. Abdul Halim Shah bin Maulud)

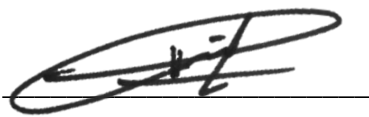
UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

May 2013

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

A handwritten signature in black ink, consisting of a large, stylized loop on the left and several horizontal strokes on the right, positioned above a solid horizontal line.

MOHD HANIF BIN ISHAK

ABSTRACT

Process monitoring traditionally using univariate process monitoring approach where each of individual variables is monitored separately. In this approach process variables interaction is difficult to be monitored and therefore multivariable statistical process monitoring (MSPM) was introduced to cater the drawback of univariate process monitoring. MSPM has a major advantage in detecting change in variables relationship or also known as structural changes. Despite of the advantage, most of studies are focusing on change in variables rather than the variables interaction. In this study, PCA based detection techniques performance including PCA, dynamic PCA and nonlinear PCA has been evaluated under change in reaction kinetic and change in heat transfer coefficient. Hotelling T^2 and SPE chart are employed as the fault detection techniques. The project mainly focusing on fault detectability and fault detection time. All the PCA based approaches are able to detect the structural changes. Nonlinear PCA shows the fastest detectability followed by dynamic PCA and PCA. For highly nonlinear system, Nonlinear PCA are able to detects the fault the fast but the nonlinear PCA not performing the best when encounter with lesser degree of nonlinear data set.

ACKNOWLEDGEMENT

First and foremost, I would like to express my highest gratitude to God for without his blessing and guidance, I would not be able to complete this Final Year Project (FYP). I would like to thank my supervisor, Ir. Dr. Abdul Halim Shah bin Maulud for his invaluable guidance and advice throughout my FYP period that tremendously contribute to this project. I strongly believe the time allocated by him in sharing his thoughts, ideas and experience on the project greatly contribute to completion of Multivariate Process Monitoring of Structural Changes in A CSTR System. Indeed it as a pleasure working with him throughout the project period.

Besides, I would like to express my deepest appreciation to anyone whose name are not mentioned but involved directly or indirectly in the completion of this project. Thank you for all of your support.

TABLE OF CONTENT

CERTIFICATION OF APPROVAL	i
CERTIFICATION OF ORIGINALITY	ii
ABSTRACT.....	iii
ACKNOWLEDGEMENT	iv
TABLE OF CONTENT	v
LIST OF FIGURES	vii
LIST OF TABLES	viii
CHAPTER 1: INTRODUCTION	1
1.1 Background Study	1
1.2 Problem Statement	3
1.3 Significant of the Project.....	4
1.4 Objectives.....	5
1.5 Scope of Study	5
CHAPTER 2: LITERATURE REVIEW	6
2.1 Multivariable Statistical Process Monitoring (MPSM).....	6
2.2 Principle Component Analysis (PCA)	8
2.3 Fault Detection Using PCA.....	9
2.3.1 Hotelling's T^2	9
2.3.2 SPE/Q-Statistic	10
2.4 Dynamic Principle Component Analysis (DPCA)	12
2.5 Nonlinear PCA (Kramer, 1991).....	13
2.5.1 Neural Network.....	13
2.5.2 Autoassociative Neural Network	16
2.5.3 T^2 and SPE Chart for Nonlinear PCA.....	17
CHAPTER 3: METHODOLOGY	18
3.1 Project Methodology	18
3.2 Tool	19
3.3 Gantt Chart	20
3.4 Key Milestone	20

CHAPTER 4: RESULT AND DISCUSSION	22
4.1 Modeling and Simulation	22
4.1.1 CSTR Model and Simulation	22
4.1.2 Fault Detection using PCA	23
4.1.3 Fault Detection using DPCA	23
4.1.4 Fault Detection using NLPCA	26
4.2 SPE and T^2 Statistic for Normal Data	27
4.3 SPE and T^2 Statistic for Structural Fault Data	30
4.3.1 Change in Reaction Kinetics.....	30
4.3.2 Change in Heat Transfer Coefficient	32
4.4 Summary of fault detection time	34
CHAPTER 5: CONCLUSION AND RECOMMENDATION	36
REFERENCES	37
APPENDIX I: CSTR MODEL IN SIMULINK	I
Input Block Diagram For The CSTR Model.....	I
APPENDIX II: MATLAB SOURCE CODE	II
PCA for Normal Data.....	III
PCA for Kinetic Change	VI
PCA for Heat Transfer Coefficient Change	VIII
DPCA (ALL Column) for Normal Data	X
DPCA(ALL Column) for Change In Kinetic	XIII
DPCA(ALL Column) for Change In Heat Transfer Coefficient.....	XVI
Neural Network Training and Simulation for Kinetic and Heat Transfer Coefficient Change.....	XIX
T^2 and SPE for Data From the Neural Network	XXI

LIST OF FIGURES

Figure 1: Interaction of variables that cannot be detected by traditional univariate statistical process control charts	3
Figure 2 Individual control charts for Y1 and Y2 (Runger, 1996a)	7
Figure 3 Joint plot of Y2 vs. Y1 (Runger, 1996a)	7
Figure 4 Hotelling T^2 for 2PC model.....	10
Figure 5 Q residual for 2PC model.....	11
Figure 6: Graphical illustration for fault detection using Q and T^2 statistics (Chiang & Russell, 2001)	11
Figure 7 Neural Network	13
Figure 8: Graph of sigmoid function	14
Figure 9: combined mapping and demapping in NLPCA	16
Figure 10: Project phase/methodology	19
Figure 11: Network structure preview generated by MATLAB.....	26
Figure 12: Function used for layers in the network	26
Figure 13: SPE and T^2 Chart for Normal Operation Data Using PCA approach	27
Figure 14: SPE and T^2 Chart for Normal Data Using DPCA (all column with time lag shift expansion).....	27
Figure 15: SPE and T^2 Chart for Normal Data Using DPCA (two output column with time lag shift expansion).....	28
Figure 16: SPE and T^2 Chart for Normal Data Using NLPCA.....	28
Figure 17: SPE and T^2 Chart for Fault Data due to Change in Reaction Kinetic Using PCA	30
Figure 18: SPE and T^2 Chart for Fault Data due to Change in Reaction Kinetic Using DPCA (all column with time lag shift expansion)	30
Figure 19: SPE and T^2 Chart for Fault Data due to Change in Reaction Kinetic Using DPCA (two output column with time lag shift expansion)	31
Figure 20: SPE and T^2 Chart for Fault data due to Change in Reaction Kinetic Using NLPCA	31
Figure 21: SPE and T^2 Chart for Fault Data due to Change in Reaction Kinetic Using PCA	32
Figure 22: SPE and T^2 Chart for Fault Data due to Change in Heat Transfer Coefficient Using DPCA (all column with time lag shift expansion).....	32

Figure 23: SPE and T ² Chart for Fault Data due to Change in Heat Transfer Coefficient Using DPCA (two output column with time lag shift expansion).....	33
Figure 24: SPE and T ² Chart for Fault Data due to Change in Heat Transfer Coefficient Using NLPCA.....	33

LIST OF TABLES

Table 1: Steady-state and operating condition.....	18
Table 2: Gantt chart	21
Table 3: Input/output variable of the CSTR model	22
Table 4: Explained Variance of Normal Data.....	23
Table 5: Explained Variance of Normal Data with Time Lag Shift of All Variables (DPCA).....	24
Table 6: Explained Variance of Normal Data with Time Lag Shift of Two Output Variables (DPCA).....	25
Table 7: Summary of Fault Detection Time Using T ² and SPE statistic	34

CHAPTER 1: INTRODUCTION

1.1 Background Study

In past decades, chemical industry was focusing only on producing product as much as possible. Nowadays, as the competition in industrial market grows fierce, the objective of industry was shift to produce higher product yield and quality. At the same time, they also are aiming for higher production efficiency, less pollution and waste. All these can be satisfied with a better understanding of the process with a better process control. This implies to a need of attention to condition monitoring strategies.

Process monitoring is commonly based on single variable statistics and it is difficult and time consuming for everyone to find out the problem and evaluate the performance of operation. The existence of multivariable and tremendous amount of data adds up the difficulties of the monitoring process and this is more complicated with the highly interacted nature of chemical process. Because of that, a range of statistically based condition monitoring approach was developed and known as Multivariate Statistical Process Monitoring (MSPM) was introduced.

One of the MSPM objectives is to identify any assignable causes that result in a shift in the process mean that cannot be detected by univariate monitoring approach. A process is said to be in control only when common causes of variation are present. Based on the assumption that data collected are uncorrelated and normally distributed, a multivariate control chart can be utilised to detect abnormal changes in the system that causing shift in process mean.

In process monitoring, there are two types of faults which are faults in variable and faults in structure (Venkat , Raghunathan , Kewen , & Surya, 2002). Variables fault is a change of variable parameter that exceeds the acceptable range of the observed variables. The parameters failures arise when there is disturbance enter the process from environment through one or more variables. An example for such fault is a change in temperature and concentration of reactant from its steady state value in reactor feed. The change might due to disturbances that enter the process.

On the other hand, fault in structure is change of process relationship between variables in a process that depart from in-control region defined by confidence limits calculated from a reference set. Example for structural fault is drift in reaction kinetic which might due to catalyst deactivation and change in heat transfer coefficient due to fouling in heat exchanger.

As large quantity of multivariate data required to be analysed, MSPM have to remove redundancy in the data by introducing a reduced set of statistically uncorrelated variables. Principal component analysis (PCA) is an example of the basic approach applied in MSPM. A good MSPM can provide insight about the stability of the process in individual variables as well as the relationship between the variables (or known as structure).

The purpose of this paper is to investigate few monitoring methods that applicable to promptly detect fault in structural change. The investigation will be using non-isothermal continuous stirred tank reactor (CSTR) system as the case study.

1.2 Problem Statement

Major advantage of multivariate statistical process monitoring (MSPM) is to detect the change in the structure rather than change in process variables.

Figure 1 shows an example of typical situation where two process variables are both inside their control limit in univariate control charts but fails to detect general trend of correlation between these two variables is broken (sample in red).

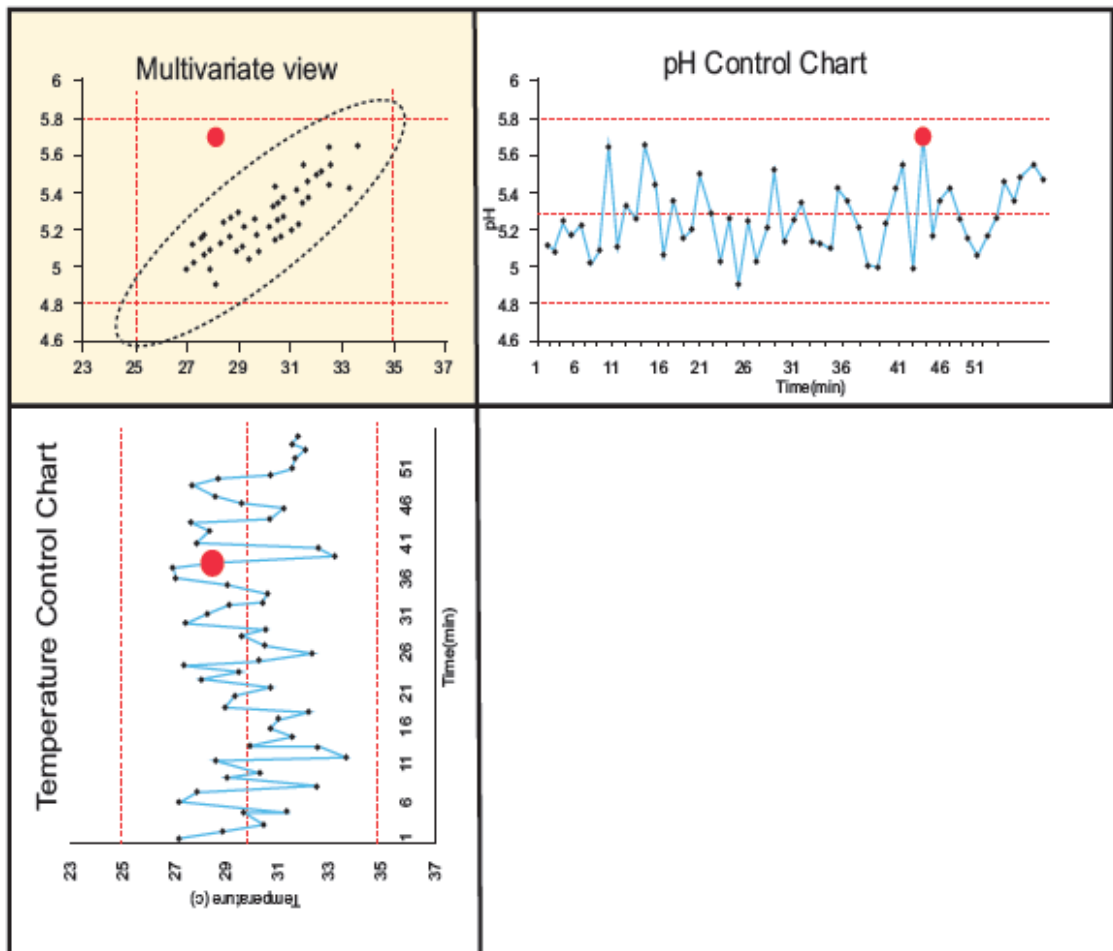


Figure 1: Interaction of variables that cannot be detected by traditional univariate statistical process control charts

However, in past research works most of the case studies concentrate on change in process variable. For example:

- i. (Chen & Liao, 2001) For Tennessee Eastman control problem, 12 out of the 15 are process variable shift.
- ii. (Xiong, Liang, & Qian , 2007) For Polyethylene process catalyser reactor case, only variables change are monitored,
- iii. (Chen, Kruger , Meronk , & Leung, 2004) For debutanizer process case, all the focus are on the variables change.

Despite MSPM are specifically used to detect the structural change, but most techniques are tested on variable change. Therefore there is a need to identify the potential of these techniques for detection of structural change specifically.

1.3 Significant of the Project

The benefit that can be obtained from the detecting structural changes is it will allow the next level in process monitoring i.e. fault identification and fault diagnosis that will lead to identification of the root causes.

Structural fault monitoring could be expand to the level where it able to describe the severity of the fault for example how much catalyst have been deactivated. And therefore will help in production or operation planning on deciding when would be the most economical to change the catalyst.

Nevertheless, the big objectives in the structural fault monitoring must be started with the fault detection i.e. detecting the changes or fault that occurred.

1.4 Objectives

The objectives of this project to investigate the structural fault detectability using PCA based approaches. To achieve the objective, the sub-objectives this project are as the following:

1. To develop structure change fault case study using CSTR in MATLAB simulation environment.
2. To simulate structural change faults in the model using potential structural change detection approaches.
3. To compare fault detectability performance among potential PCA based approaches.

1.5 Scope of Study

The scope of this project is fault detection in kinetic and heat transfer coefficient changes in non-isothermal CSTR system. The MSPM techniques that will be evaluated on the CSTR system will be using PCA based approaches. Modelling involves in this project will be done using MATLAB software.

CHAPTER 2: LITERATURE REVIEW

2.1 Multivariable Statistical Process Monitoring (MPSM)

Process controllers such as PID controller are designed to maintain operation by cancelling out effect of disturbance. However, there are changes in the process that the controller unable to rectify. These changes are known as faults and it defined as unpermitted deviation of at least one characteristic property or variable of the system (Isermann & Ball, 1996).

In ensuring process operation is at performance specification, the faults need to be detected, diagnosed and removed and these is associated with process monitoring. This process monitoring also called as statistical process control (SPC), but due to confusion with standard process control, some reference use statistical process monitoring (SPM).

The objective of SPM is to ensure success of the planned operation by recognizing anomalies of the behaviour (Chiang & Russell, 2001). The information from SPM not only provides the status of the process but also plant personnel to make appropriate corrective action to eliminate the disturbances. A good monitoring system will result in minimum downtime, lower production cost and higher reliability in operational and safety aspect.

Modern process control system becoming more complex and the current SPM is not adequate to monitor the faults using univariate control chart. Univariate statistical charts (Shewhart, CUSUM, and EWMA) ignore the correlation among other variables and measurements; they do not able to accurately characterize the behaviour of the current industrial process (Chiang & Russell, 2001).

Because of this reason, Multivariable SPM (MSPM) emerges and gaining acceptance in process monitoring as it can provide more accurate information about the process. MPSM can provide monitoring charts that can detect fault and gives warning signal earlier than the classical univariate chart (Chiang & Russell, 2001).

The strongest benefit of MSPM is the ability to exploit relationship between variables (Jackson, 1985). Example of this can be seen by comparing results of Figure 2 and Figure 3. Figure 2 demonstrates univariate charts showing both variables within control limit. Figure 3 using the same data considered jointly to demonstrate that one data record is violating the usual relationship.

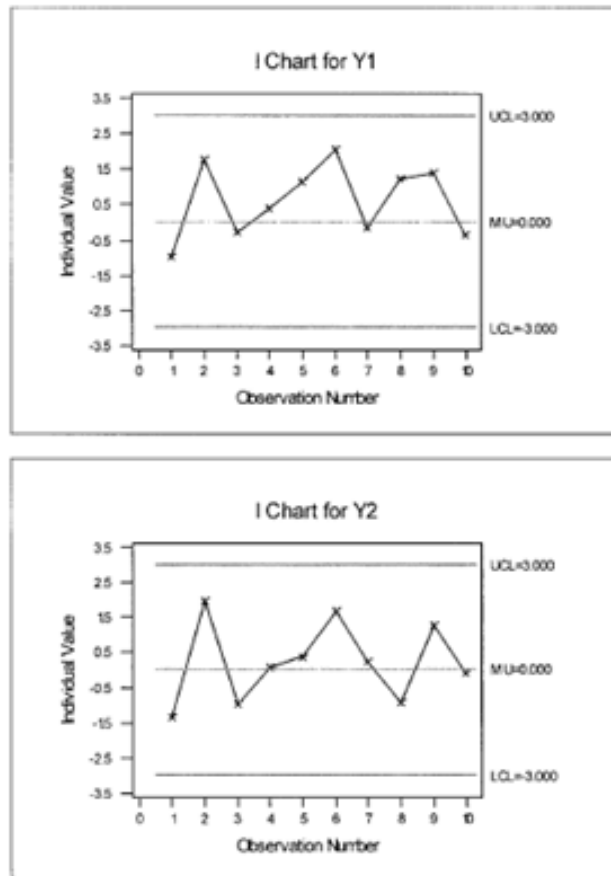


Figure 2 Individual control charts for Y1 and Y2 (Runger, 1996a)

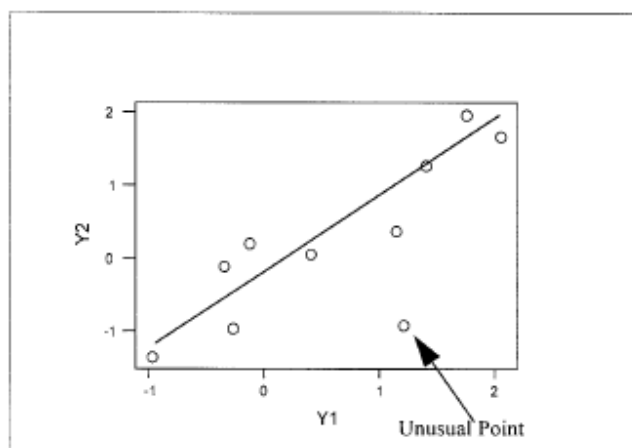


Figure 3 Joint plot of Y2 vs. Y1 (Runger, 1996a)

2.2 Principle Component Analysis (PCA)

PCA is utilised in MSPM as it can greatly simplify data into lower-dimensional space without any loss of variance between variables. It preserves the correlation structure between process variables and captures variability of the data. In addition, PCA could also eliminate the noise effects.

Application of PCA in MSPM can be seen as taking the enormous amount of data from the process, and transforming it into new set of variables called principle components. These principle components are sets of new uncorrelated variables.

PCA determines a set of orthogonal vectors called loading vector and it is ordered by the amount of variance explained in the loading vector direction. PCA decomposition of training X matrix is as follows:

$$X = \hat{X} + E = \sum_{k=1}^L t_k p_k^T + E$$

in which,

$$\hat{X} = \sum_{k=1}^L t_k p_k^T$$

Where,

X is the matrix that store k number of observations and L number of process variables.

\hat{X} is the prediction X on m retained principal components (PCs).

E is the residual matrix that represents the PCA model prediction error.

t is score matrix that describe significant process variation

p is loading matrix that reveal the interrelationship between process variables

Loading matrix, p can be easily determined through the eigenvalue decomposition of sample covariance matrix. After obtaining the loading matrix, the score matrix can be computed as it is given by:

$$t_k = X p_k^T$$

2.3 Fault Detection Using PCA

2.3.1 Hotelling's T^2

Hotelling's T^2 plot can detect small shifts and deviations from normal operation defined by the model. This statistic technique includes contribution of all variables deviation that becomes significant faster than the deviation of an individual variable. This T^2 will measure the variation in each sample within PCA model and only indicate deviation that can be explained by model.

The T^2 statistic can be calculated as the following,

$$T_j^2 = t_j \lambda^{-1} t_j^T$$

Where,

t_j refers to j-th row of t_k score matrix

λ^{-1} is a diagonal matrix of the inverse of the eigenvalue associated with k principal component

The threshold of T^2 statistic can be computed using the equation below

$$T_\alpha^2 = \frac{a(n-1)(n+1)}{n(N-a)} F_\alpha(a, n-a)$$

$n \equiv$ number of observation

$a \equiv$ degree of freedom or the number of sample

$\alpha \equiv$ level of significant or the confidence limit

Figure 4 is the Hotelling T^2 for 2PC model. The dotted line indicates the control limit and the value of T^2 that exceed the line indicate the presence of fault.

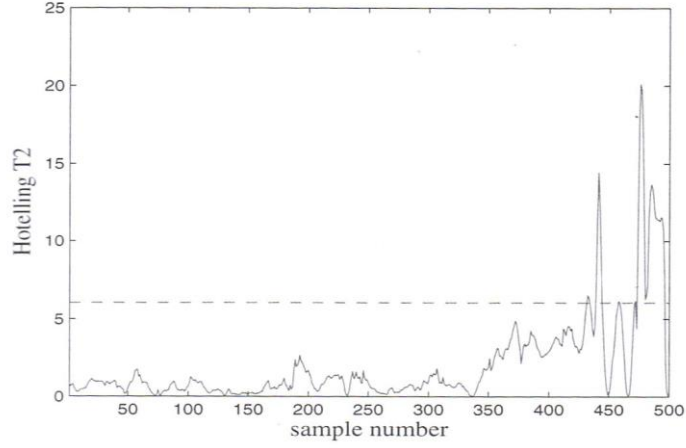


Figure 4 Hotelling T^2 for 2PC model

2.3.2 SPE/Q-Statistic

The squared prediction error (SPE) is also known as Q-statistic indicates how well each sample conforms to PCA model. The SPE chart shows indication of significant deviation that cannot be explained by the model. The SPE can be calculated from the residual matrix which is the sum of squares for each row of the matrix.

$$Q = e_j e_j^T$$

Where e_j is components j row of residual matrix.

After obtaining the SPE, the threshold should be then calculated.

$$Q_\alpha = \theta_1 \left[\frac{h_0 c_\alpha \sqrt{2\theta_2}}{\theta_1} + 1 + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} \right]^{1/h_0}$$

Where $\theta_i = \sum_{j=a+1}^n \sigma_j^{2i}$, $h_0 = 1 - \frac{2\theta_1 \theta_3}{3\theta_2^2}$, and C_α is the normal deviate corresponding to the $(1-\alpha)$.

Figure 5 illustrate the Q residual for 2PC model. The residual that exceed the dotted horizontal line indicate presence of fault.

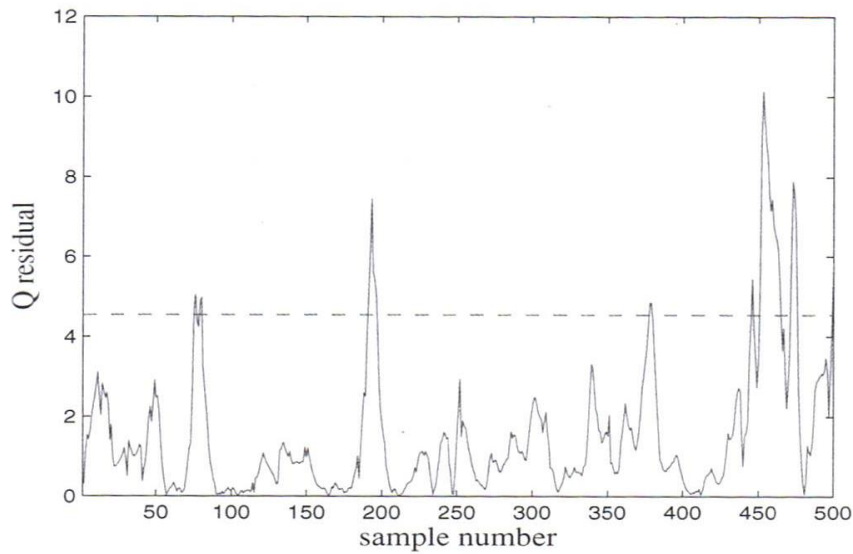


Figure 5 Q residual for 2PC model

Since the T^2 and SPE statistic along their appropriate thresholds can detect different types of faults, thus the advantage of both statistics can be used together.

Figure 6 shows a graphical illustration for fault detection that utilise T^2 and Q statistics. The two statistics produce cylindrical in-control region that 'x' indicated in-control operation data, 'o' data indicate violation of T^2 statistic, and '+' data show Q statistics violation.

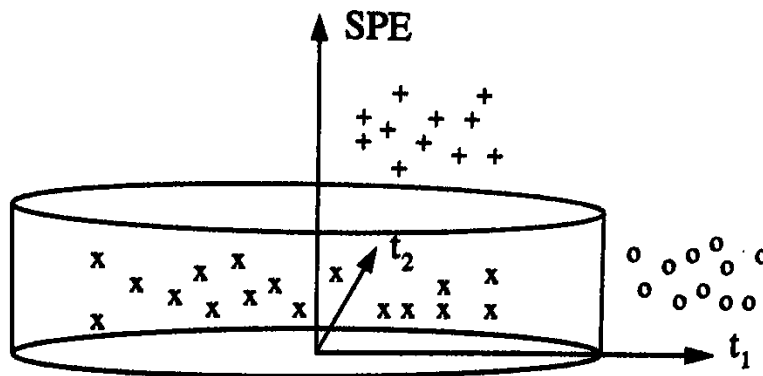


Figure 6: Graphical illustration for fault detection using Q and T^2 statistics (Chiang & Russell, 2001)

2.4 Dynamic Principle Component Analysis (DPCA)

Monitoring system based on PCA approach assumes observations at one time instant statistically independent to observations in the past time. However, in typical industrial processes, assumption is only valid for long sampling time i.e. 2 to 12 hours. Therefore this suggests that a monitoring method with fast sampling require serial correlation data to be considered.(Chiang & Russell, 2001)

Let X be a set of data with n_t observation of p variables.

$$X = [X_1 \quad X_2 \quad \cdots \quad X_p]_{(n_t \times p)}$$

Then, to include the serial correlation of data, it is constructed trajectory matrix applying lime lag shift of order w on each of the columns p of the matrix X .

$$X_t^w = \begin{bmatrix} X_i(1) & X_i(2) & \cdots & X_i(w) \\ X_i(2) & X_i(3) & \cdots & X_i(w+1) \\ \vdots & \vdots & \ddots & \vdots \\ X_i(n_t - w + 1) & X_i((n_t - w + 2)) & \cdots & X_i(n_t) \end{bmatrix}_{(n \times w)}$$

$$X^w = [X_1^w \quad X_2^w \quad \cdots \quad X_p^w]_{(n \times m)}$$

where $n = n_t - w + 1$; $m = pw$

Value for w is made based on compromise between information content and statistical confidence. It is common to select w in the same way as is defined the number of lags to use when constructing an auto-correlation function, $w = n_t/4$ (J. & C, 2005).

Applying the PCA to the above X_t^w matrix is known as dynamic PCA (DPCA). The DPCA can be expected to have higher fault detectability compared to PCA for serially correlated data.

2.5 Nonlinear PCA (Kramer, 1991)

2.5.1 Neural Network

Neural networks are information processing paradigm that is inspired by the way biological nervous system process information. Neural network is also known as artificial neural network (ANN) attempts to recreate the computational mirror of biological neural network. The neural network can be created by modelling a network of model neurons using computer. Neural Network have basic building block which is artificial neuron is often called as nodes. The nodes are connected to each other and their connection is to one another is assigned a value based on their weight: inhibition (maximum being -1.0) or excitation (maximum being +1.0). Higher value of the weight indicates there is strong connection. Within each node's design, a transfer function is built in. The structure of neural network have input nodes, hidden nodes, and output nodes as illustrated in Figure 7

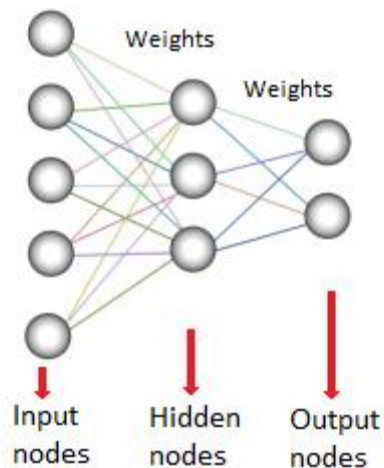


Figure 7 Neural Network

Input nodes take in information in numerical form. The information is presented as activation values, where the each of the nodes will be assigned a number, the higher the number, the greater the activation. This information is then passed throughout the

network. Based on the connection strengths (weights), inhibition or excitation, and transfer functions, the activation value is passed from node to node. Each of the nodes sums the activation values it receives; it then modifies the value based on its transfer function. The activation flows through the network, through hidden layers, until it reaches the output nodes. The output nodes then reflect the input in a meaningful way to the outside of the neural network model.

Additional class of weight is known as biases. Biases are values that are added to the sums calculated at each node (except input nodes) during the feedforward phase. Biases are commonly visualized simply as values associated with each node in the intermediate and output layers of a network, but in practice are treated in exactly the same manner as other weights. The use of biases in a neural network increases the capacity of the network to solve problems by allowing the hyperplanes that separate individual classes to be offset for superior positioning. (Leverington, 2009)

The neurons can be having either linear or non-linear transfer function. For non-linear transfer function, the commonly used function is Sigmoid and Log-sigmoid transfer function (LOGSIG) (Dorofki, Ahmed H. Elshafie, Othman, & Othman , 2012).

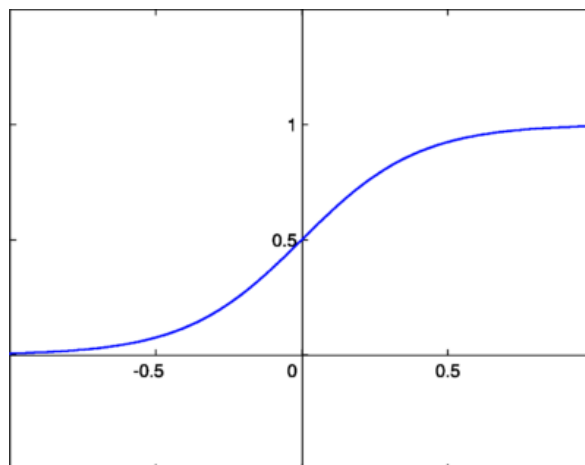


Figure 8: Graph of sigmoid function

To prevent the saturation in output of sigmoid function, the data use in input neuron need to be scaled. Un-scaled input data to sigmoid function will result in useless output information whenever the corresponding output falls under saturated region in sigmoid function. An example of scaling method is to use mean centred at zero with standard deviation of one.

Before building a model, available data need to be partitioned (Frontline Solvers, 2013). Data partitioning will yields mutually exclusive data sets: a training dataset, a validation dataset and test dataset. Training dataset will be used to obtain the network weights. The weight can be determined through linear regression; the training dataset is used to fit the linear regression model, i.e. to compute the regression coefficient.

Validation set on the other hand will be used to find the accuracy of the model after the model is built previously using the training data. For this, the model in put should be using a dataset that was not used in training process but it is a dataset where you know the actual value of target variable. The different between the actual and predicted value is the error in prediction.

Test set is actually just another validation set which often used to fine-tune models. For example, you might try out neural network models with various architectures. And test accuracy of each on the validation dataset to choose among the competing architecture. The accuracy of the model on the test data gives a realistic estimate of the performance of the model on completely unseen/random data.

2.5.2 Autoassociative Neural Network

Nonlinear PCA (NLPCA) includes nonlinear mappings between the original and reduced dimension spaces which are not accounts by the PCA. If non-linear correlation between variables exist, NLPCA are able to explain the data with higher accuracy than PCA, provided there is sufficient data to support formulation of more complex mapping function (Kramer, 1991). NLPCA able to uncovers both linear and nonlinear correlations without restriction on character of the nonlinearities present in the data.

NLPCA is accomplished by training a feedforward neural network to carry out identity mapping on which the network inputs are reproduced at the output layer. There is a “bottleneck” layer that contains fewer nodes than input or output layer and it forces the network to produce reduced representation of the input data and two additional hidden layers (Kramer, 1991). The center of hidden neurons of a bottle-neck neural network can be used to perform nonlinear MPSM (Thissen, Melssen, & Buydens, 2001). The representation of NLPCA can be clearly seen from Figure 9.

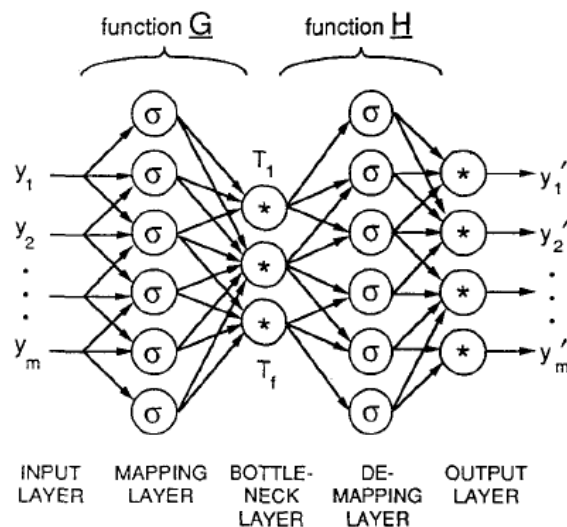


Figure 9: combined mapping and demapping in NLPCA

From this auto-associative neural network, the output of the bottle neck layer will be monitored using the Hotelling T^2 . The error which is the residual between Y and Y' will be monitored using SPE/Q-statistic chart. The analysis can be performed using Matlab software and the neural networks can be implemented easily by using Neural Network Toolbox (*feedforwardnet*).

2.5.3 T² and SPE Chart for Nonlinear PCA

From the auto-associative neural network, output of bottleneck neurons is monitored using T² chart where the residuals from the estimated and the real process are monitored using SPE-chart (Thissen, Melssen, & Buydens, 2001). The charts are constructed similar to regular T² chart and SPE-chart.

T² statistic is calculated as the following,

$$T^2 = \sum_{i=1}^A \frac{t_i^2}{s_{t_i}^2}$$

Where,

A = total number of bottleneck neurons,

t_i = output of neuron i ,

s_{t_i} = variance of output neuron i

On the other hand, SPE-statistics is calculated as the following,

$$SPE = \sum (y_{ji} - y'_{ji})^2$$

Where,

y_{ji} = i th sample to j th neuron of input layer,

y'_{ji} = i th output of j th neuron of output layer.

CHAPTER 3: METHODOLOGY

3.1 Project Methodology

In the first phase, PCA based techniques that have potential in promptly detect faults in structural change will be review and compiled. This will be done through reviewing past research paper works. Then, equations for non-isothermal CSTR system from Chemical Process Modelling and Computer Simulation by Amiya K. Jana will be used for the purpose of modelling the system (Amiya, 2008). The same reference also provides the steady state and operating condition dataset.

In this project, three datasets obtained from simulated system will be used for analysis with the PCA, DPCA and NLPCA. The first dataset will be the output dataset that is simulated with normal operating data. Second dataset will be output of simulated model under drift of heat transfer coefficient. And the third dataset is the output of simulated data on drift in catalyst activity (frequency factor). The parameters of the CSTR considered are shown in Table 1 (Amiya, 2008).

Table 1: Steady-state and operating condition

Notation	Parameters used in the simulation	Value
A_c	Cross-sectional area of the reactor, m ²	4.2822
C_A	Concentration of reactant A in the exit stream, kmol/m ³	8.56303
C_{Af}	Concentration of A in the feed stream, kmol/m ³	10.0
d	Diameter of cylindrical reactor, m	2.335
E	Activation energy, kcal/kmol	11843.0
F_i	Volumetric feed flow rate, m ³ /h	10.0
h	Height of the reactor liquid, m	2.335201
$(-\Delta H)$	Heat of reaction, kcal/kmol	5960.0
R	Universal gas constant, kcal/(kmol)(K)	1.987
α	Frequency factor, h ⁻¹	34930800.0
ρC_p	Multiplication of mixture density and heat capacity, kcal/(m ³)(°C)	500.0
T	Reactor temperature, °C	38.17771
T_f	Feed temperature, °C	25.0
T_j	Jacket temperature, °C	25.0
U_i	Overall heat transfer coefficient, kcal/(m ²)(°C)(h)	70.0

Modelling of the CSTR system is done by using MATLAB software where the steady state input of the system was corrupted with the mean centred equal to zero.

T^2 and SPE chart will be drawn for the reduced dataset after implementing the aforementioned data reduction techniques. The performance of the fault detectability between the techniques will be compared based on the percentage of detectable number of observation. The details on fault will be put in a fault table where there will be the percentage of detection performance of the fault detection techniques demonstrated based on T^2 statistic and Q-statistics. The detection will also be based on specified confidence level. Example of the fault table can be seen in table 2 in (Chen & Liao, 2001).

The discussed methodology is summarized as in Figure 10.

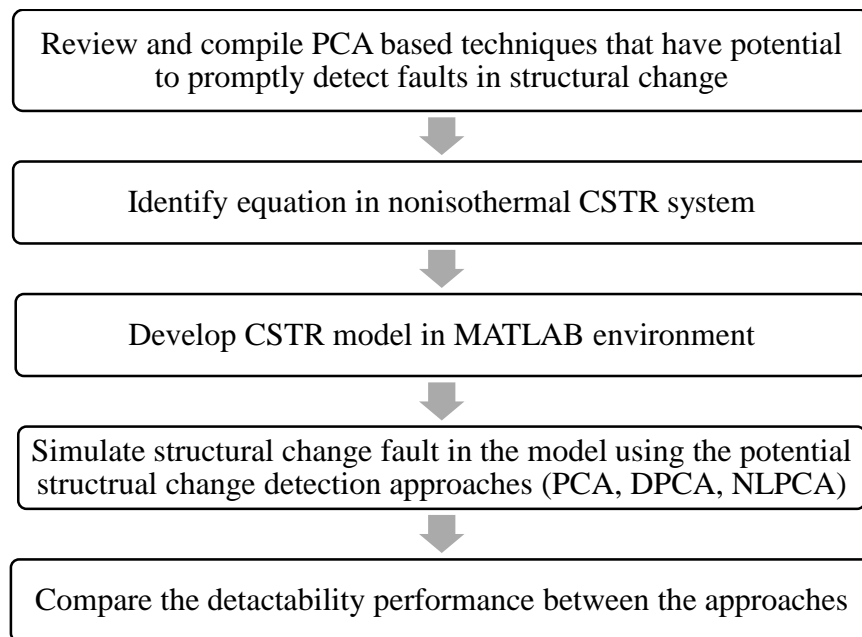


Figure 10: Project phase/methodology

3.2 Tool

MATLAB Software with:

1. Statistic toolbox
2. Neural network toolbox

3.3 Gantt Chart

Table 1 shows the objectives in timelines that should be completed during for FYP II. The first three weeks is allocated for modelling of CSTR system and simulate for normal condition and change in heat transfer coefficient and reaction. Following that, in week 4 the data will be processed and PCA approach will be started and expected to be done in week 7.

Progress report is scheduled for submission in the beginning of week 8. The continuation of the work will be done for DPCA for week 8 and 9 while Autoassociative Neural Network is expected to be started in week 10 and completed within 3 weeks.

Pre-SEDEX presentation will be in week 11 followed by submission of draft report in week 12. Dissertation in soft bound and technical paper are required to be submitted in week 13. Final oral presentation with internal and external examiner was scheduled by coordinator to be in week 14 and submission of final dissertation (hard bound) in week 15.

Only after the title is finalized, the literature review will be started which end by week 6. The literature review will be done alongside with the extended proposal where the objective, problem statement, and methodology would be well identified. The finalized extended proposal is to be submitted at the beginning of week 7.

3.4 Key Milestone

In this project, key milestones was identified as the following:

1. Develop CSTR model and simulate for normal condition, change in heat transfer coefficient and change in reaction kinetic.
2. Data processing and PCA approach for fault detection
3. Fault detection using DPCA
4. Fault detection using Autoassociative Neural Network

Table 2: Gantt chart

NO	DETAIL	WEEK																
		1	2	3	4	5	6	7	Mid Semester Break	8	9	10	11	12	13	14	15	
1	Develop CSTR model and simulate normal condition, change in heat transfer coefficient and drift in reaction kinetics																	
2	Data processing and using PCA for fault detection and analysis																	
	Progress report																	
3	Fault detection using DPCA and analysis																	
4	Fault detection using Autoassociative Neural Network																	
5	Pre-SEDEX																	
6	Submission of Draft Report																	
7	Submission of Dissertation (soft bound)																	
8	Submission of technical paper																	
9	Oral Presentation																	
10	Submission of Project Dissertation (hard bound)																	

● Suggested Milestone

■ Process

CHAPTER 4: RESULT AND DISCUSSION

4.1 Modeling and Simulation

4.1.1 CSTR Model and Simulation

The CSTR system was modelled using Simulink in Matlab software. Screen shot of the model is available in the appendix. The input and output of the model listed in Table 3.

Table 3: Input/output variable of the CSTR model

Input	F_i	Inlet flowrate
	C_{Af}	Inlet concentration
	T_f	Feed temperature
	T_j	Jacket temperature
Output	F_0	Outlet flowrate
	C_A	Outlet concentration
	T_r	Reactor temperature

For the input variables, white noise is added by assuming the real process data exhibit normal distribution. The model is run for time equal to 100. The sampling time for the sampling is 0.01 to obtain 1000 samples. The model is run for the normal data when there is no change in heat transfer coefficient and reaction kinetic.

The model is then simulated for change in heat transfer coefficient with 10% decrease starting time equal to 30 (i.e. at 300th sample). Separately, the model is run for drift in kinetic with 5% decrease starting at time equal to 30. From the simulation, the major responding variable when changing the heat transfer coefficient and reaction kinetics is on the reactor temperature followed by the outlet concentration.

Data from the simulated system is then exported to matrix file (.mat). The data stored in the matrix file is loaded in the main window and normalized with the sample mean and

standard deviation so that normalize variable will be having zero mean and standard deviation equal to one. For the normal condition, the data is normalized using the sample mean and standard deviation while for the drift condition, the data is normalized using the normal condition's mean and standard deviation.

4.1.2 Fault Detection using PCA

The PCA is done for normalized normal condition data to obtain the loading matrix, score matrix and latent. PCA for the fault data is then done by using loading matrix of normal condition data and T^2 calculated using latent (which store variance) of the normal condition data. From the PCA of the normal condition data, the variance explained by the principle component is as following:

Table 4: Explained Variance of Normal Data

No of PC	Explained variance by each of the PC (%)	Cumulative explained variance by the PC (%)
1	19.68	19.68
2	15.07	34.75
3	14.90	49.65
4	14.11	63.76
5	13.80	77.56
6	13.29	90.85
7	9.15	100

From the cumulative explained variance, 5 PC was decided to be retained for the T^2 and SPE calculation. Methodology for calculating the T^2 and SPE is as discussed in literature review section.

4.1.3 Fault Detection using DPCA

For DPCA, the time lag shift is introduced to the input/output variables matrix. The chosen time lag shift was 2 sample lag. For this part, matrix expansion of the time lag was done all variables time lag expansion, and only output variables expansion. For the all variables time lag expansion, additional 2 column matrix was introduces for every variables, while for output only expansion, only 2 column matrix was introduces for output variables.

From the expanded variables matrixes, PCA was done and T^2 and SPE calculation is obtained. From the cumulative explained variance in Table 5 and Table 6, 9 PCs (explained 72.57%) was chosen for all column time lag expansion, and 4 PCs (explained 72.38%) was chosen for only output variables matrix expansion. Then the T^2 and SPE is calculated from the retained PCs.

Table 5: Explained Variance of Normal Data with Time Lag Shift of All Variables (DPCA)

No of PC	Explained variance by each of the PC (%)	Cumulative explained variance by the PC (%)
1	19.17	19.17
2	13.26	32.43
3	8.90	41.33
4	5.48	46.81
5	5.41	52.22
6	5.21	57.44
7	5.19	62.63
8	4.99	67.63
9	4.95	72.57
10	4.65	77.22
11	4.52	81.74
12	4.40	86.14
13	4.18	90.32
14	4.12	94.44
15	3.88	98.33
16	0.82	99.15
17	0.35	99.50
18	0.29	99.79
19	0.11	99.90
20	0.08	99.98
21	0.02	100.00

Table 6: Explained Variance of Normal Data with Time Lag Shift of Two Output Variables (DPCA)

No of PC	Explained variance by each of the PC (%)	Cumulative explained variance by the PC (%)
1	31.42	31.42
2	22.14	53.56
3	9.68	63.24
4	9.14	72.38
5	8.96	81.34
6	8.55	89.88
7	7.67	97.55
8	1.64	99.19
9	0.57	99.76
10	0.18	99.94
11	0.06	100.00

4.1.4 Fault Detection using NLPCA

The feedforward neural network was design with input layer, three hidden layer (including bottleneck) and an output layer. Number of nodes for input and output layer is equal to number of variables, i.e. 7, while the 10 nodes for first and third hidden layer and 3 nodes for the bottleneck layer. The network structure can be clearly seen in Figure 11.

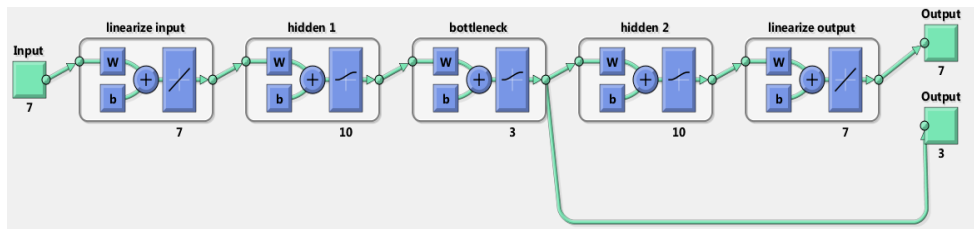


Figure 11: Network structure preview generated by MATLAB

Transfer function for the network layers is as the following:

Figure 12: Function used for layers in the network

Layer	Function
Input	Linear
Hidden layer 1	Logsig
Bottleneck layer	Tansig
Hidden layer 2	Logsig
Output	Linear

The network was trained using normalized data of normal condition input and output data of the CSTR system. Then the fault data of kinetic change and heat transfer coefficient change is run using the trained network, on which the biases and weights from network training is maintained. The output from the bottleneck layer is used to obtained the T^2 statistic and the output from the output layer is utilised to get the SPE statistics.

4.2 SPE and T² Statistic for Normal Data

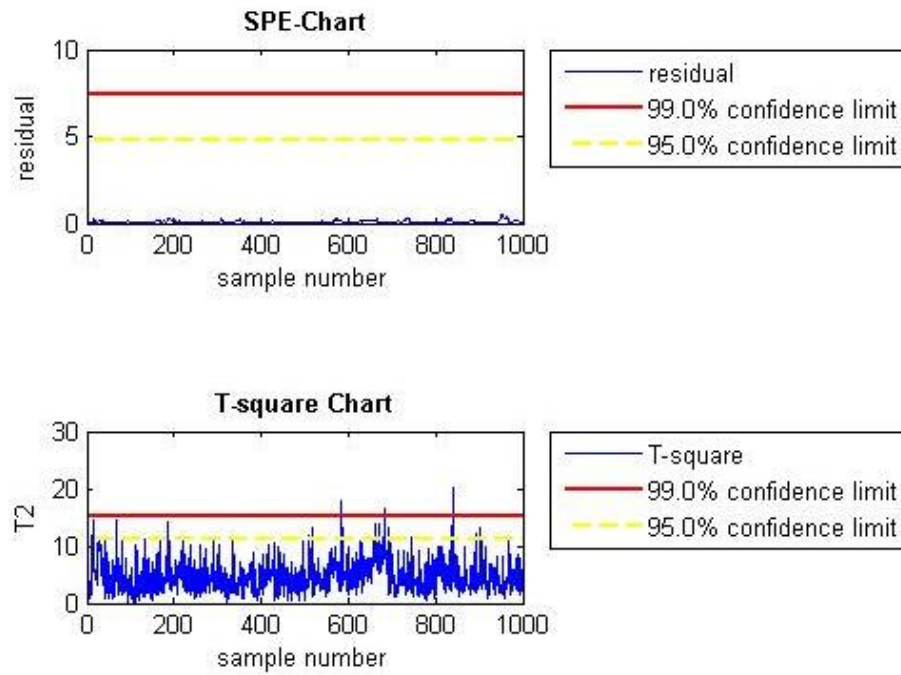


Figure 13: SPE and T² Chart for Normal Operation Data Using PCA approach

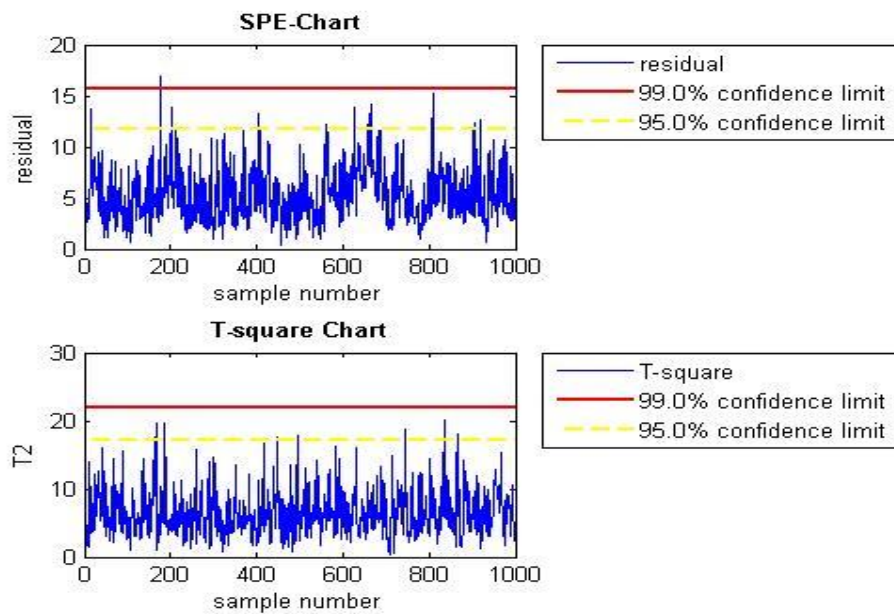


Figure 14: SPE and T² Chart for Normal Data Using DPCA (all column with time lag shift expansion)

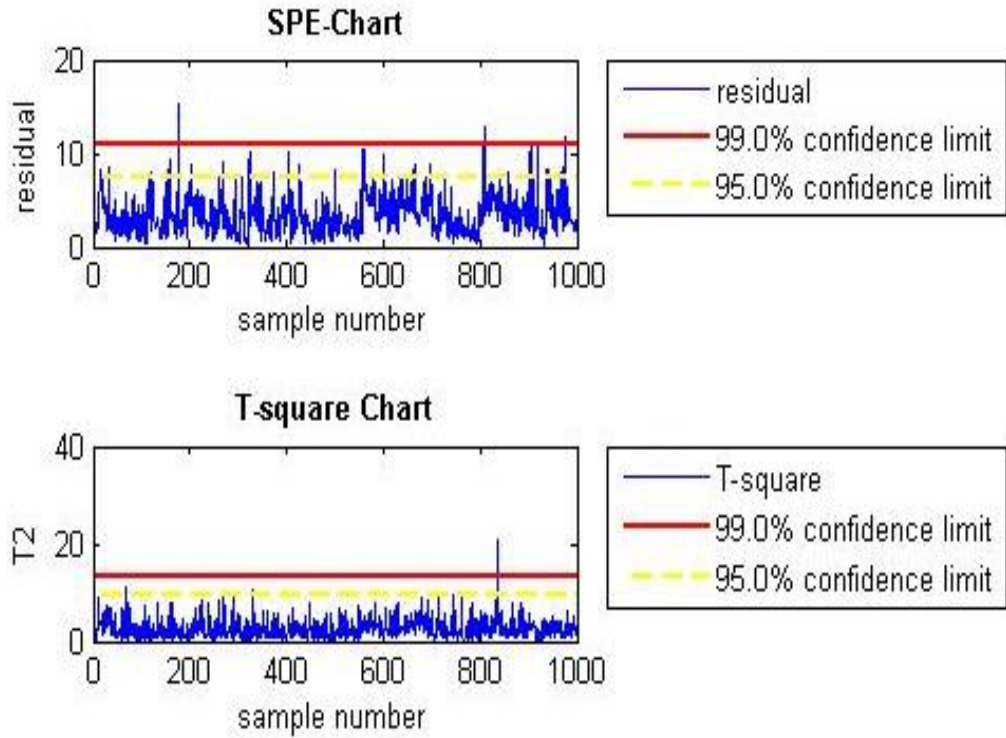


Figure 15: SPE and T^2 Chart for Normal Data Using DPCA (two output column with time lag shift expansion)

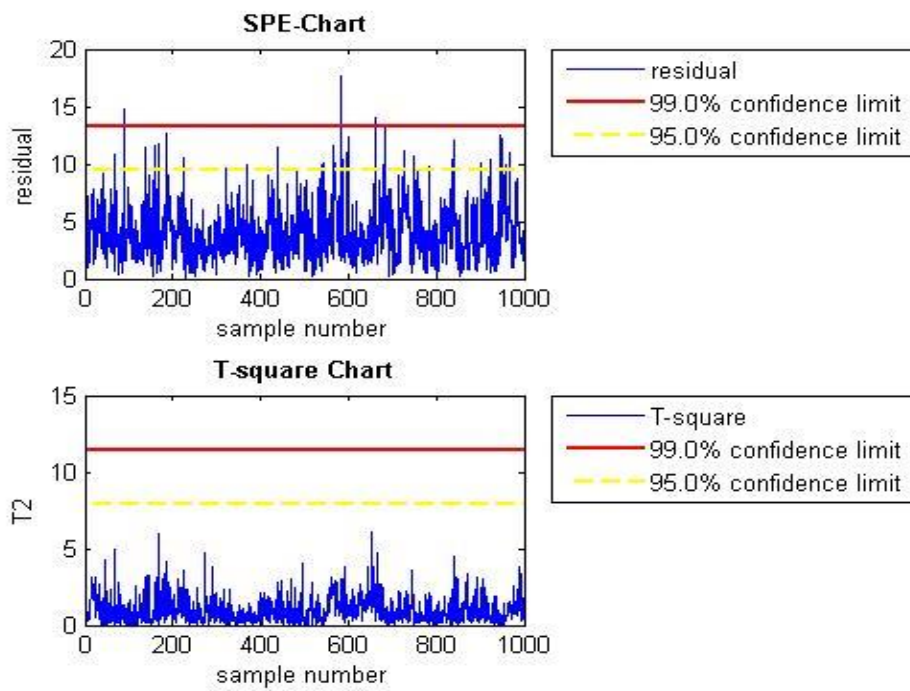


Figure 16: SPE and T^2 Chart for Normal Data Using NLPCA

From the SPE and T^2 charts of normal data for PCA and DPCA, it is observed that there is points that exceed control limit at which 99.0% of confidence limit. There are also points that exceed the warning limits that defined at 95.0% confidence limit. The detection of fault in the normal data might due to variation of input variable that exceed the predefined control limit of 3 standard deviation of the sample mean. This is might due to normal distribution of the added noise to the input variables of the modelled CSTR. Another possible reasoning would be there is multiple input variables that adding positive noise at the same time and signify the variation for a given sample time.

For monitoring the structural changes, the process trending i.e. change in process mean is more important than points that exceed the control limits. To facilitate the detection time determination in this paper, detection time is considered whenever the points exceed the 99.0% confidence limit that preceded with observable change in mean. This guideline is based on the normal data trending in SPE and T^2 charts of PCA, DPCA and NLPCA where the outliers that exceed 99.0% confidence limit does not preceded with any observable change in mean that exceed the warning limit.

4.3 SPE and T² Statistic for Structural Fault Data

4.3.1 Change in Reaction Kinetics

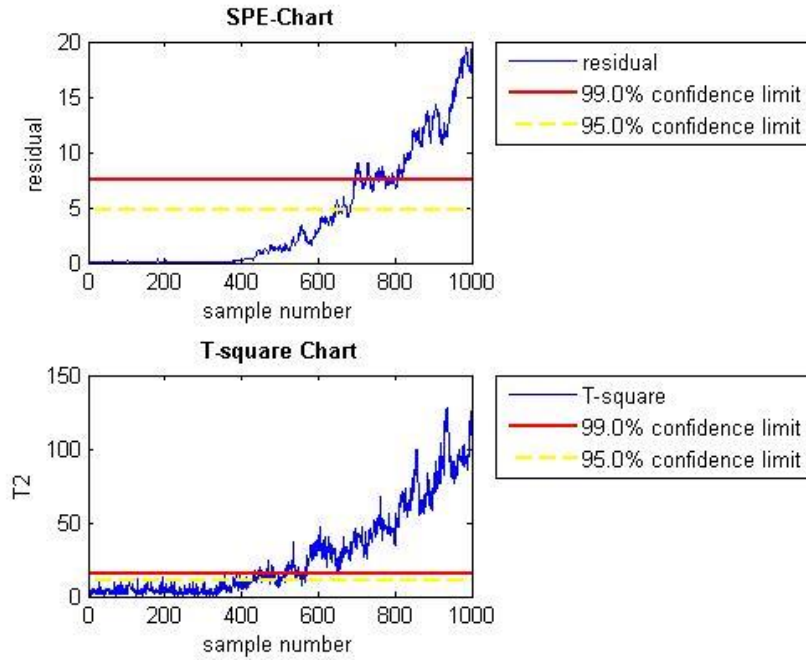


Figure 17: SPE and T² Chart for Fault Data due to Change in Reaction Kinetic Using PCA

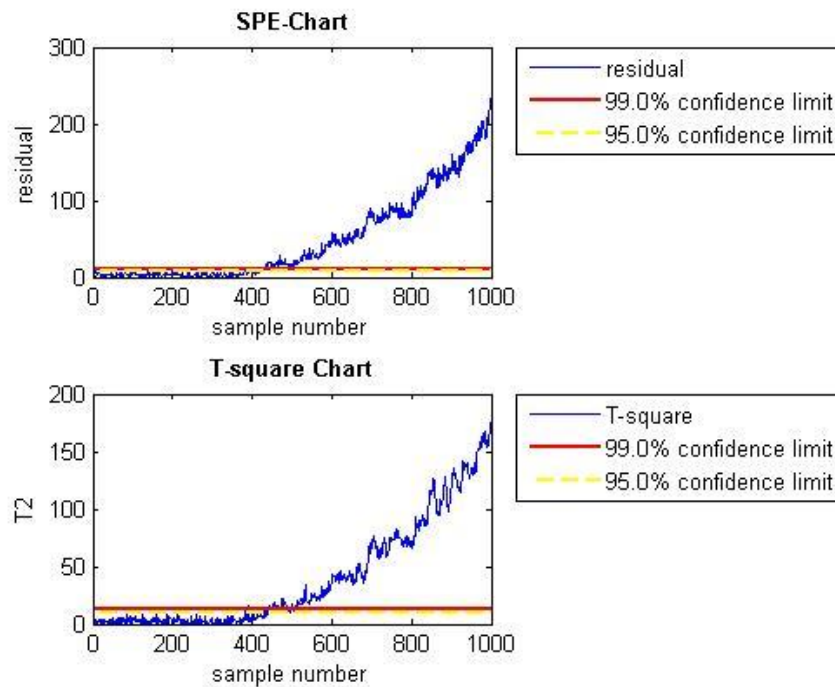


Figure 18: SPE and T² Chart for Fault Data due to Change in Reaction Kinetic Using DPCA (all column with time lag shift expansion)

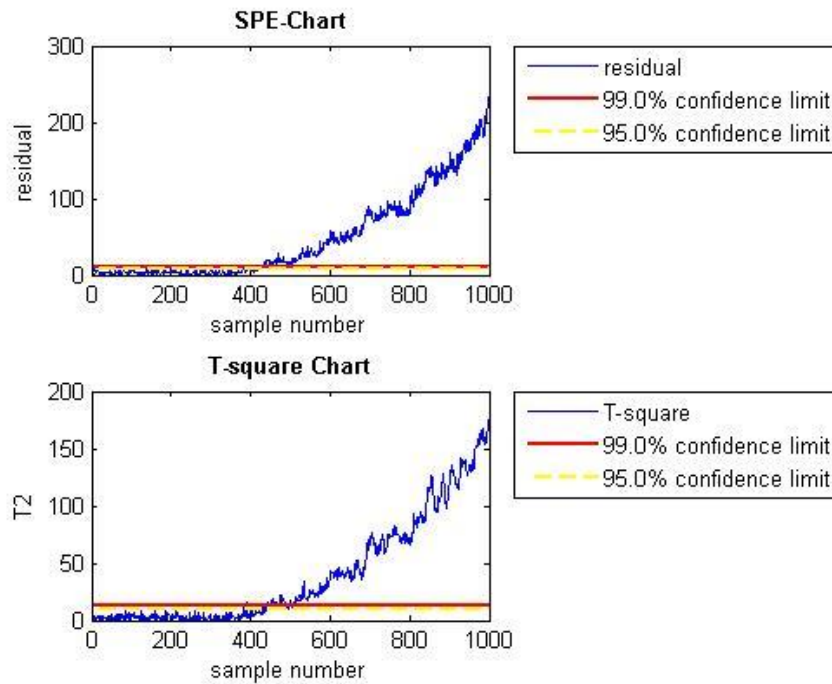


Figure 19: SPE and T^2 Chart for Fault Data due to Change in Reaction Kinetic Using DPCA (two output column with time lag shift expansion)

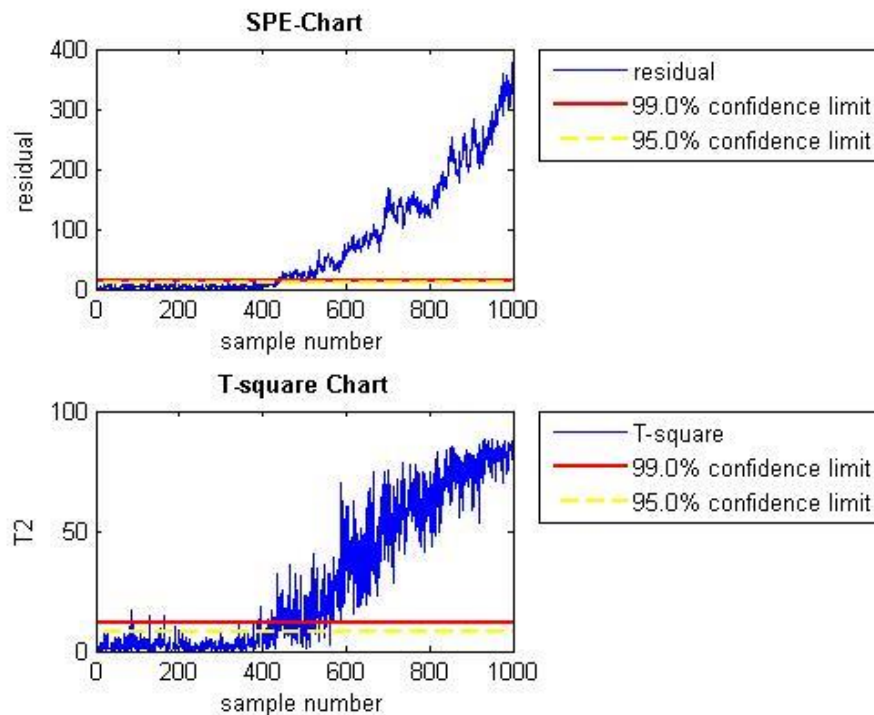


Figure 20: SPE and T^2 Chart for Fault data due to Change in Reaction Kinetic Using NLPCA

4.3.2 Change in Heat Transfer Coefficient

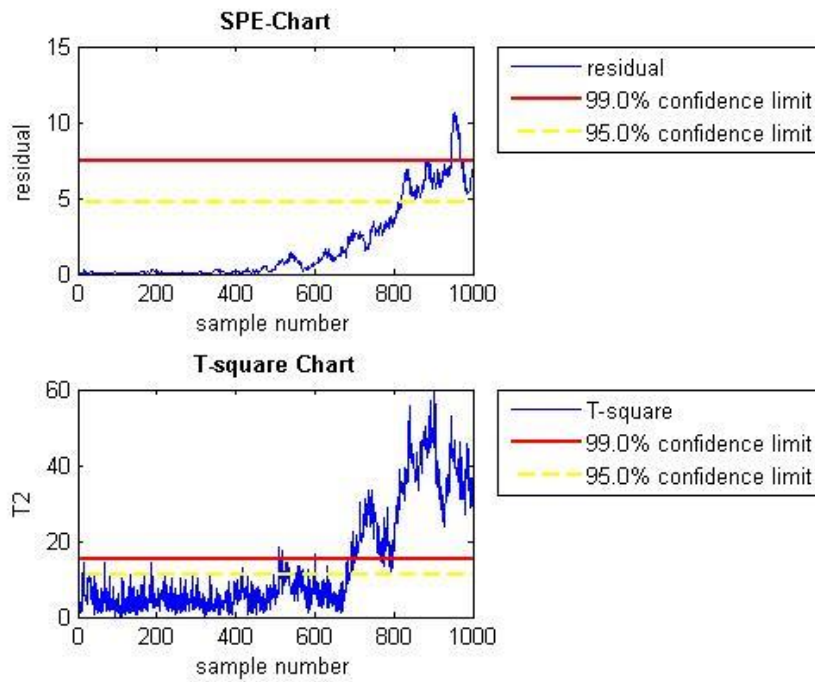


Figure 21: SPE and T^2 Chart for Fault Data due to Change in Reaction Kinetic Using PCA

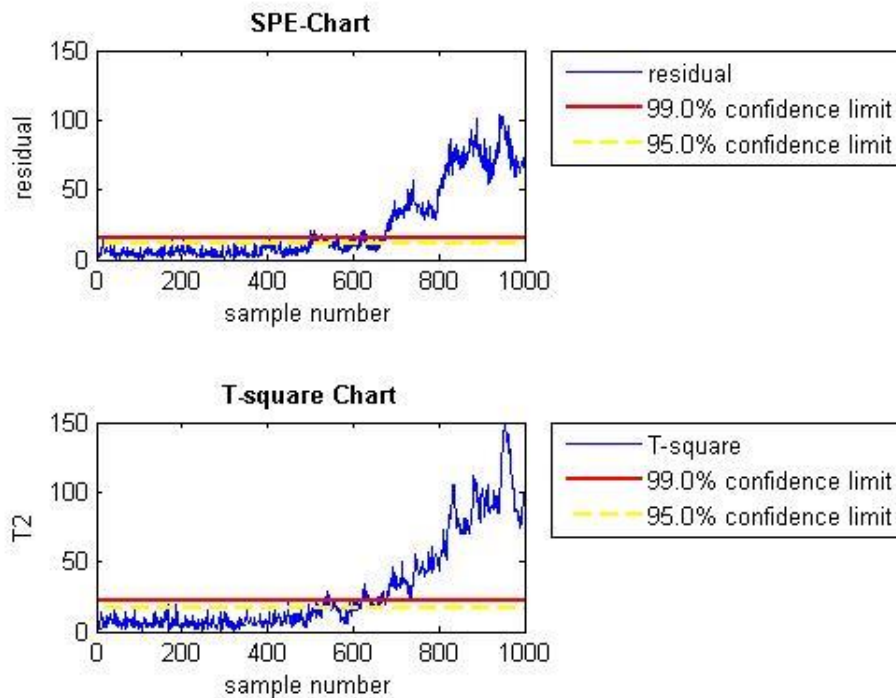


Figure 22: SPE and T^2 Chart for Fault Data due to Change in Heat Transfer Coefficient Using DPCA (all column with time lag shift expansion)

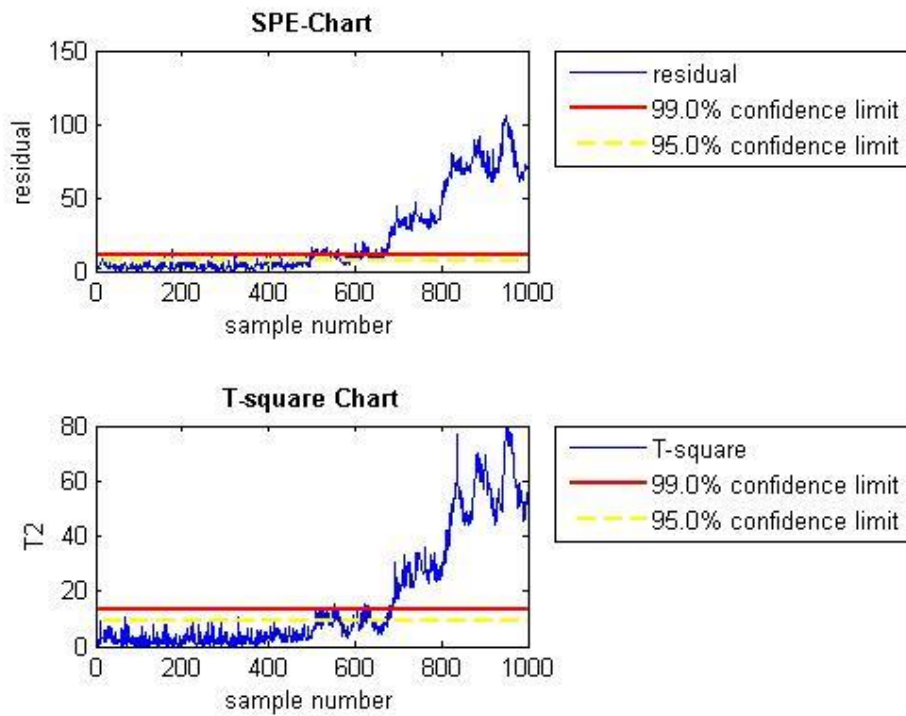


Figure 23: SPE and T^2 Chart for Fault Data due to Change in Heat Transfer Coefficient Using DPCA (two output column with time lag shift expansion)

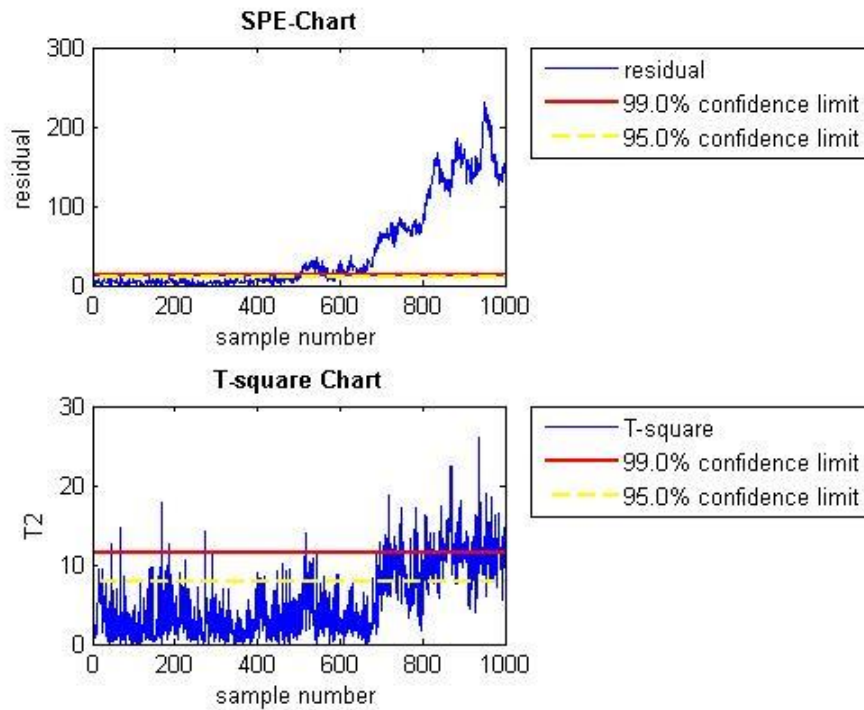


Figure 24: SPE and T^2 Chart for Fault Data due to Change in Heat Transfer Coefficient Using NLPCA

4.4 Summary of fault detection time

The summary of the fault detection for all kinetic and heat transfer coefficient using all the aforementioned techniques are in Table 7.

Table 7: Summary of Fault Detection Time Using T^2 and SPE statistic

		Detection Time (Sample No.)			
		PCA	DPCA (all column)	DPCA (two column)	NLPCA
Reaction Kinetic Change	T^2	446	444	443	436
	SPE	693	430	430	428
Heat Transfer Coefficient Change	T^2	510	535	528	695
	SPE	886	499	499	500

The result of DPCA detection time for all variable time lag shift expansion and only two output variable expansion shows a close fault detection time outcomes especially in SPE statistics. This is due to the independent of input variables and therefore expansion of the matrixes for input variables will give insignificant effect to the fault detection time. On the other hand, the output variables, i.e. temperature and concentration are dependent variables. They have relationship with the previous observation and they are interdependent between past temperature and concentration sample.

For change in reaction kinetics, advanced PCA method using SPE statistics shows better performance compared to T^2 statistics. However, T^2 statistics performance between the approaches is almost similar on which detected first by NLPCA, followed by DPCA and PCA. In this kinetic change, NLPCA perform best although the result is comparable to DPCA.

For change in heat transfer coefficient, again the advanced PCA method shows better performance using SPE statistics. However for T^2 statistics shows that PCA demonstrate best performance compared to T^2 of DPCA and NLPCA. In overall, DPCA demonstrating the best in detecting fault in heat transfer coefficient through SPE statistics.

From the detection time result, we can see that DPCA approach are able to detect the structural faults almost as the same as the PCA approach when compare using T^2 chart. The less significant differences are due to the assumption that the observation statistically independent to observation of past time is true and therefore expansion for time lag shift does not have significant effect on the T^2 statistic i.e. monitoring the variance between data samples. In structural fault monitoring, data observation can be safely assumed to be independent of past observation since long sampling time internals are applied.

On the other hands, DPCA through SPE detect faults faster compared to PCA. This is due to the introduction of time-lag shift to the original data matrix where it will eventually make the errors between the projected space and original data to be compounded when calculating the SPE. The residual between the projected and original data are then signified in the monitoring charts.

It is also observable that the NLPCA shows superiority in detecting fault due to reaction kinetic change comparing to PCA and DPCA. It is however less effective when it comes to detecting fault due to change in heat transfer coefficient. The main reasons might due to nonlinearity of the data in kinetic change dataset is more prominent, on which nonlinear PCA work best.

PCA approach through T^2 statistics shows better performance compared to DPCA and NLCPA in detecting change in heat transfer coefficient. This might due to the nonlinearity of the dataset on which PCA model are able to characterize it better than DPCA and NLPCA.

From the general overview of the result, it can be observed that NLPCA shows the best performance in detecting structural fault whenever the dataset contains highly nonlinearity between the variables. While DPCA have its own strength when encounter different degree of nonlinearity of dataset. Although DPCA might not perform the best compared to other approaches, there is an observable consistency of the comparable result from DPCA (compared to other approaches) when giving different dataset. SPE statistics also prove in this project to detect faults faster compared to T^2 statistics.

CHAPTER 5: CONCLUSION AND RECOMMENDATION

The objective of this work is to investigate structural fault detectability using PCA based approaches. The significant of the study is to fill the gap of knowledge in fault detection that is specifically for structural fault. The structural change in CSTR model was successfully simulated using Simulink in MATLAB and the data obtained was used as feeding data to PCA based monitoring approaches i.e. PCA, DPCA, and NLPCA.

Based on fault detection, the NLPCA shows the fastest detection time followed by DPCA and PCA. The NLPCA is demonstrated the most robust structural fault detection when encounter nonlinear system. On the other hand, PCA is better in characterizing data that contains lesser degree of nonlinear dataset. For DPCA, expansion of dependent variable is sufficient in monitoring structural fault. The differences between results obtained through PCA and DPCA mainly in SPE statistics where DPCA are able to signify the errors by compounding the errors.

Therefore, it can be concluded that structural fault can be detected using PCA based techniques and the objectives of the project are successfully achieved.

Nevertheless, the suggested work for future is as below:

1. Integrate CUSUM (cumulative sum) and EWMA (exponentially weighted moving average) directly to NLPCA to increase the sensitivity and robustness of process monitoring.
2. The continuation of the structural fault detection using other MPSM techniques and proceeded with fault identification and fault diagnosis.

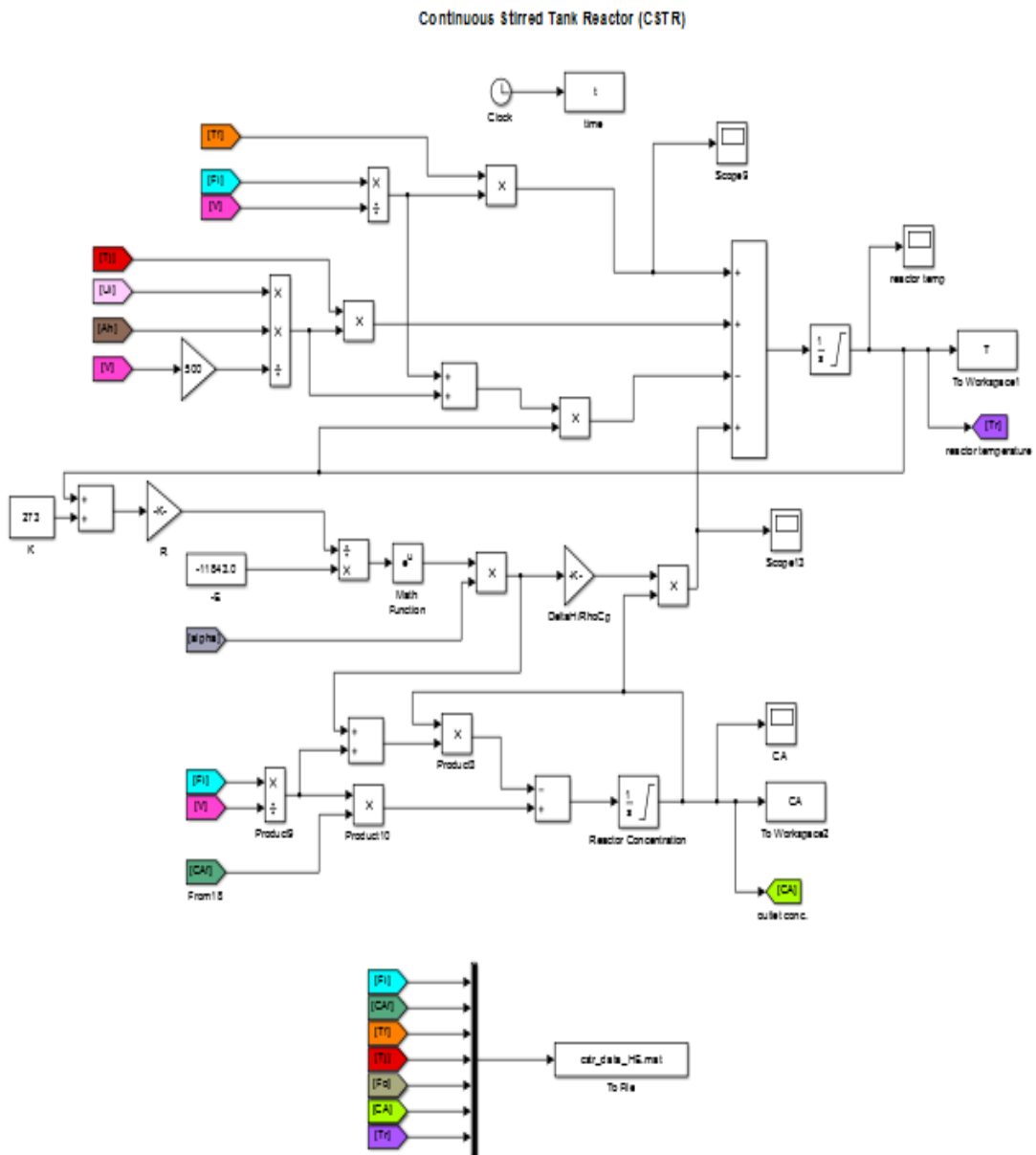
REFERENCES

- Amiya, J. K. (2008). *Chemical Process Modelling and Computer Simulation*. New Delhi: Prentice-Hall of India Private Limited.
- Chen, J., & Liao, C.-M. (2001). Dynamic process fault monitoring based on neural network and PCA. *ournal of Process Control*(12), 277.
- Chen, Q., Kruger , U., Meronk , M., & Leung, A. (2004). Synthesis Of T^2 and Q statistics for process monitoring. *Control Engineering Practice*, 753.
- Chiang , L., & Russell, E. (2001). *Fault detection and diagnosis in industrial system*. Great Britain: Springer.
- D. Dong, & T. J. McAvoy. (1993). Nonliner Principle Component Analysis-Based on Principal Curve and Neural Networks. *Computer Chem Engineering*, 65-78.
- Dorofki, M., Ahmed H. Elshafie, Othman, J., & Othman , A. K. (2012). Comparison of Artificial Neural Netwrok Transfer Function Abiliites to Simulate Extreme Runoff Data. *2012 International Conference on Environmental, Energy and Biotechnology*. Singapore.
- Edward C. Malthouse. (1998, January 1). Limitations of Nonlinear PCA as Performed with Generic Neural Networks. *IEEE Transactions on Neural Networks*, 9.
- Frontline Solvers. (2013). *Standard Data Partition*. (Frontline Systems, Inc.) Retrieved April 12, 2013, from <http://www.solver.com/xlminer/help/standard-data-partition>
- Isermann, R., & Ball, P. (1996). Trends in the application of model based fault detection and diagnosis of technical process. *Proc. of the 13th IFAC World Congress, N*, 1-12. Picataway, Newjersey: IEEE Press.
- J., M., & C, V. (2005). Faule Detectio Using Dynamic Principal Component Analysis by Average Estimation. *2nd International Conference on Electirical and*

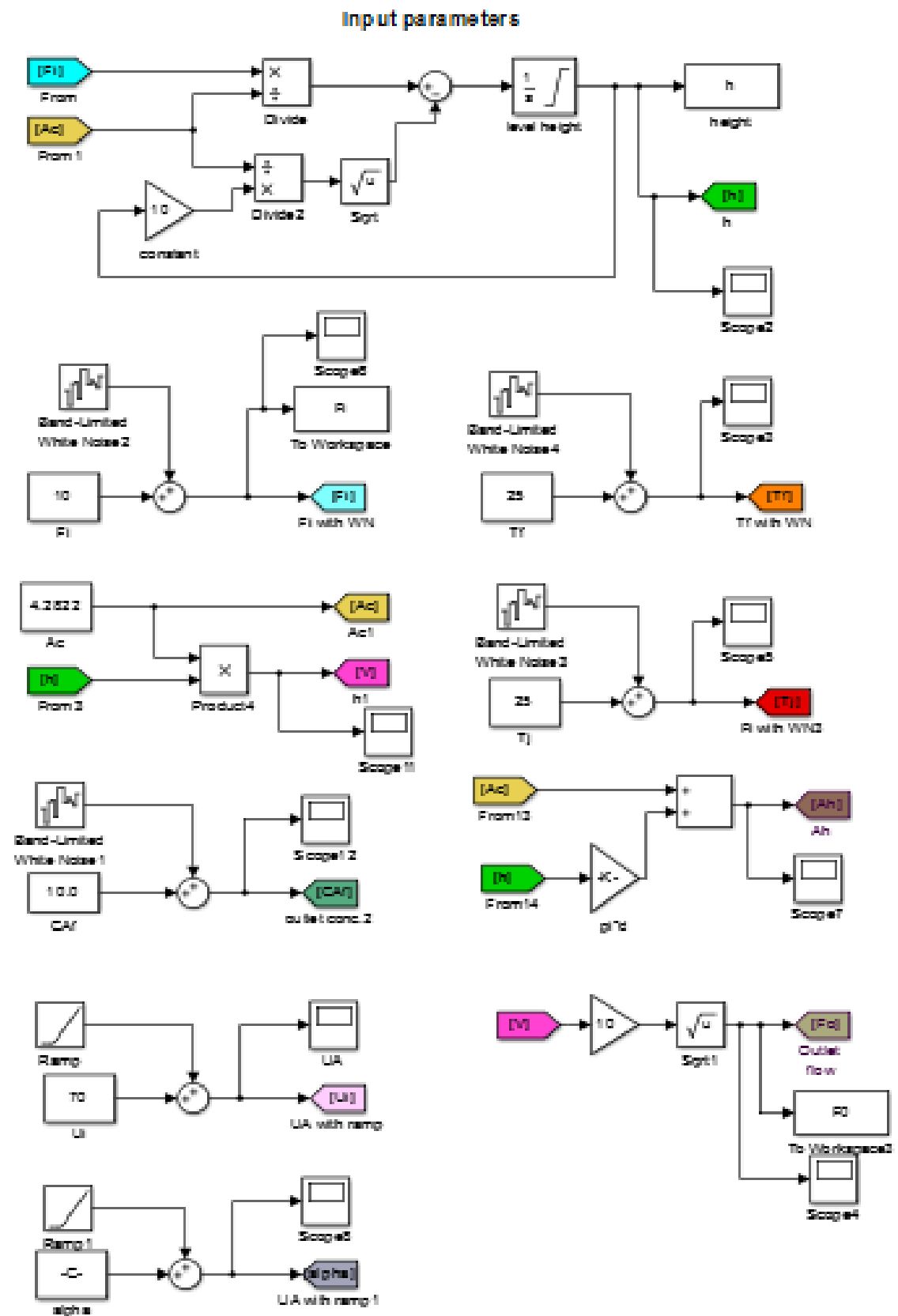
Electronics Engineering (ICEEE) and XI Conference on Electrical Engineering (CIE 2005) . Mexico City, Mexico.

- Jackson, J. E. (1985). *Multivariate Quality Control*.
- Kramer, M. A. (1991). Nonlinear Principal Component Analysis Using Autoassociative Neural Networks. *AIChE Journal*, 37(2), 233-242.
- Leverington, D. (2009). *A Basic Introduction to Feedforward Backpropagation Neural Networks*. (Texas Tech Univeristy) Retrieved April 2013
- Mostafa Noruzi Nashalji, Mahdi Aliyari Shoorehdeli, & Mohammad Teshnehlab. (n.d.). Fault Detection of the Tennessee Eastman Process Using Improved PCA and Neural Classifier. *International Journal of Electrical & Computer Science*, 9(9), 481-486.
- Qingchao Jiang, & Xuefeng Yan. (2012). Chemical process monitoring based on weighted principal component analysis and its application. *Chemometrics and Intelligent Laboratory Systems*, 119, 11-20.
- Thissen, U., Melssen, M. J., & Buydens, L. M. (2001). Nonlinear process monitoring using bottle-neck neural networks. *Analytica Chimica Acta* (466), 371-383.
- Venkat , V., Raghunathan , R., Kewen , Y., & Surya, K. N. (2002). A review of process fault detection and diagnosis. *Computer and Chemical Engineering*, 293-311.
- Xiong, L., Liang, J., & Qian , J. (2007). Multivariate Statistical Process Monitoring of an Industrial Polypropylene Catalyzer Reactor with Component Analysis and Kernel Density Estimation. *Chinese Journal of Chemical Engineering*, 527.
- Yunpeng, H., Huanxin , C., Xionshuang, Y., Cheng, Z., & Junlong, X. (2012). Chiller sensor fault detection using a self-adaptive principal component analysis method. *Energy and Building*, 54, 252-258.

APPENDIX I: CSTR MODEL IN SIMULINK



Input Block Diagram For The CSTR Model



APPENDIX II: MATLAB SOURCE CODE

PCA for Normal Data

```
clear all; clc;

load('cstr_data.mat')
cstr=cstr';
cstr=cstr(1:1000,2:8);
mn=mean(cstr); %mean
sd=std(cstr); %standard deviation

save('cstr_data_HE.mat','mn','sd','-append');
save('cstr_data_CAT.mat','mn','sd','-append');

[cstr_row, cstr_column]=size(cstr); %state column size for normalization
loop

for i=1:cstr_column, %normalization loop
    norm_column=(cstr(:,i)-mn(i))./sd(i);
    cstr_norm(:,i)=norm_column;
end
i=0;

figure(1)
for i=1:7,
    subplot(8,1,i);
    plot(cstr_norm(:,i))
    line('xData', [0 1000], 'yData', [3 3], 'LineStyle', '--', ...
        'LineWidth', 2, 'Color','b');
    line('xData', [0 1000], 'yData', [-3 -3], 'LineStyle', '--', 'LineWidth', 2,
        'Color','b');
end
i=0;

[coeff,score,latent,tsquare,explained,mu] = princomp(cstr_norm); %principal component
analysis

save('cstr_data_HE.mat','coeff','latent','-append');
save('cstr_data_CAT.mat','coeff','latent','-append');

[coeff_row,coeff_column]=size(coeff);

for i=1:7, %convert latent to square matrix
    latent_mat(i,i)=latent(i,1);
end
i=0;

mn_score=mean(score); %mean of score matrix
sd_score=std(score); %standard deviation of score
matrix
save('cstr_data_HE.mat','mn_score','sd_score','-append');
save('cstr_data_CAT.mat','mn_score','sd_score','-append');

[score_row, score_column]=size(score); %state column size for
normalization loop
for i=1:score_column, %normalization loop
    nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);
    score_norm(:,i)=nSCORE_column;
end
```

```

i=0;

no_princomp=5;          %no of retained component. no_princomp=5, explained 77.5619%

score=score(:,1:no_princomp);

score_square=(score_norm.^2);
%[coeff,score,latent,tsquare] = princomp(score_norm);
for i=1:1000,
    column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1)');
    tsquare(i,1)=sum(column_t(i,:));
end
i=0;

%r=pcares(cstr_norm,no_princomp);   %pcares return residual from PCA
%q=r.*r;

%[r_row, r_column]=size(r);

backprojection=score_norm*(coeff)';

r=cstr_norm-backprojection;

q=r.*r;

[r_row, r_column]=size(r);

SPE=sum(q');

figure(2)

subplot (2,1,1);
plot (SPE)          %SPE Chart

chisquare_99=chi2inv(0.99,cstr_column-1);
chisquare_95=chi2inv(0.95,cstr_column-1);

theta1=sum(latent((no_princomp+1):cstr_column,1));
theta2=sum(latent((no_princomp+1):cstr_column,1).^2);
theta3=sum(latent((no_princomp+1):cstr_column,1).^3);
g=theta2/theta1;
h=(theta1^2)/theta2;
h0=1-((2*theta1*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

SPE_threshold99=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_99*((2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0)) %SPE limit alternative 1
SPE_threshold95=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_95*((2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0))

%SPE_threshold99=g*chisquare_99      %SPE limit alternative 2
%SPE_threshold95=g*chisquare_95

%SPE_threshold99=g*h*((1-(2/(9*h)))+(z_99*((2/(9*h))^0.5)))^3) %SPE limit alternative
3
%SPE_threshold95=g*h*((1-(2/(9*h)))+(z_95*((2/(9*h))^0.5)))^3)

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...

```

```

'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

finv_99=finv(0.99,no_princomp,(score_row-no_princomp)); %F alpha (no_princomp, (no
of sample - no_princomp))

finv_95=finv(0.95,no_princomp,(score_row-no_princomp)); %F alpha (no_princomp, (no
of sample - no_princomp))

thold_99=((no_princomp)*(score_row-1)*(score_row+1))/(score_row*(score_row-
no_princomp))*finv_99;

thold_95=(no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp))*finv_95;

subplot (2,1,2);

plot(tsquare) %T2 chart

line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

'-----end of program-----'

```

PCA for Kinetic Change

```
clear all; clc;

load('cstr_data_CAT.mat')
cstr=cstr';
cstr=cstr(1:1000,2:8);

[cstr_row, cstr_column]=size(cstr);           %state column size for normalization
loop

for i=1:cstr_column,                         %normalization loop
    norm_column=(cstr(:,i)-mn(i))./sd(i);
    cstr_norm(:,i)=norm_column;
end
i=0;

figure(1)

for i=1:7,
    subplot (8,1,i);
    plot(cstr_norm(:,i))
    line('xData', [0 1000], 'yData', [3 3], 'LineStyle', '--', ...
        'LineWidth', 2, 'Color','b');
    line('xData', [0 1000], 'yData', [-3 -3], 'LineStyle', '--', 'LineWidth', 2,
        'Color','b');
end
i=0;

score=cstr_norm*coeff;
[coeff_row,coeff_column]=size(coeff)

for i=1:7, %convert latent to square matrix
    latent_mat(i,i)=latent(i,1);
end
i=0;

[score_row, score_column]=size(score);       %state column size for
normalization loop
for i=1:score_column,                       %normalization loop
    nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);
    score_norm(:,i)=nSCORE_column;
end
i=0

no_princomp=5;                             %no of retained component
score_square=(score_norm.^2)
%[coeff,score,latent,tsquare] = princomp(score_norm);
for i=1:1000,                               %T-square
    loop
        column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1))';
        tsquare(i,1)=sum(column_t(i,:));
    end
end
i=0

%r=pcares(cstr_norm,no_princomp);           %pcares return residual from PCA
%q=r.*r;

backprojection=score_norm*(coeff)';
r=cstr_norm-backprojection;
q=r.*r;
[r_row, r_column]=size(r);
SPE=sum(q');
```

```

figure(2)
subplot (2,1,1);
plot (SPE)          %SPE Chart

chisquare_99=chi2inv(0.99,cstr_column-1);
chisquare_95=chi2inv(0.95,cstr_column-1);

theta1=sum(latent((no_princomp+1):7,1));
theta2=sum(latent((no_princomp+1):7,1).^2);
theta3=sum(latent((no_princomp+1):7,1).^3);
g=theta2/theta1;
h=(theta1^2)/theta2;
h0=1-(2*theta1*theta3)/(3*(theta2^2));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

SPE_threshold99=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_99*(2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0)) %SPE limit alternative 1
SPE_threshold95=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_95*(2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0))

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')
finv_99=finv(0.99,no_princomp,(score_row-no_princomp)) %F alpha (no_princomp, (no
of sample - no_princomp))

finv_95=finv(0.95,no_princomp,(score_row-no_princomp)) %F alpha (no_princomp, (no
of sample - no_princomp))

thold_99=((no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp)))*finv_99;

thold_95=(no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp))*finv_95;

subplot (2,1,2);
plot(tsquare) %T2 chart
line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

'-----end of program-----'

```

PCA for Heat Transfer Coefficient Change

```
clear all; clc;
load('cstr_data_HE.mat')
cstr=cstr';
cstr=cstr(1:1000,2:8);

[cstr_row, cstr_column]=size(cstr);           %state column size for normalization
loop

for i=1:cstr_column,                          %normalization loop
    norm_column=(cstr(:,i)-mn(i))./sd(i);
    cstr_norm(:,i)=norm_column;
end
i=0;

figure(1)

for i=1:7,
    subplot(8,1,i);
    plot(cstr_norm(:,i))
    line('xData', [0 1000], 'yData', [3 3], 'LineStyle', '--', ...
        'LineWidth', 2, 'Color','b');
    line('xData', [0 1000], 'yData', [-3 -3], 'LineStyle', '--', 'LineWidth', 2,
        'Color','b');
end
i=0;

score=cstr_norm*coeff;
[coeff_row,coeff_column]=size(coeff)

for i=1:7, %convert latent to square matrix
    latent_mat(i,i)=latent(i,1);
end
i=0;

[score_row, score_column]=size(score);       %state column size for
normalization loop
for i=1:score_column,                        %normalization loop
    nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);
    score_norm(:,i)=nSCORE_column;
end
i=0

no_princomp=5;          %no of retained component

score=score(:,1:no_princomp);

score_square=(score_norm.^2)
%[coeff,score,latent,tsquare] = princomp(score_norm);
for i=1:1000,
    column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1))';
    tsquare(i,1)=sum(column_t(i,:));
end
i=0;

backprojection=score_norm*(coeff)';
r=cstr_norm-backprojection;
q=r.*r;
[r_row, r_column]=size(r);

SPE=sum(q');

figure(2)
subplot(2,1,1);
plot(SPE)          %SPE Chart
```

```

chisquare_99=chi2inv(0.99,cstr_column-1);
chisquare_95=chi2inv(0.95,cstr_column-1);

thetal=sum(latent((no_princomp+1):7,1));
theta2=sum(latent((no_princomp+1):7,1).^2);
theta3=sum(latent((no_princomp+1):7,1).^3);
g=theta2/thetal;
h=(thetal^2)/theta2;
h0=1-((2*thetal*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

SPE_threshold99=thetal*((1-(theta2*h0*(1-h0)/(thetal^2)) +
((z_99*((2*theta2*(h0^2))^0.5))/(thetal)))^(1/h0)) %SPE limit alternative 1
SPE_threshold95=thetal*((1-(theta2*h0*(1-h0)/(thetal^2)) +
((z_95*((2*theta2*(h0^2))^0.5))/(thetal)))^(1/h0))

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

finv_99=finv(0.99,no_princomp,(score_row-no_princomp)) %F alpha (no_princomp, (no
of sample - no_princomp))

finv_95=finv(0.95,no_princomp,(score_row-no_princomp)) %F alpha (no_princomp, (no
of sample - no_princomp))

thold_99=((no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp)))*finv_99;

thold_95=(no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp))*finv_95;

subplot (2,1,2);
plot(tsquare) %T2 chart
line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

'-----end of program-----'

```


DPCA (ALL Column) for Normal Data

```
clear all; clc;
load('cstr_data.mat')
cstr=cstr';
cstr=cstr(1:1000,2:8);

[cstr_row, cstr_column]=size(cstr);
%cstr_tlshift=cstr(3:cstr_row,cstr_column);

no_lag=2;

for i=1:cstr_column,

    if i==1;
        cstr_tlshift(:,i)= cstr((no_lag+1):cstr_row,i);

        for n=1:no_lag,
            cstr_tlshift(:,i+n)=cstr((no_lag+1-n):(cstr_row-n),i);
        end
        n=0;

    else
        cstr_tlshift(:,(i*(no_lag+1)-no_lag))= cstr((no_lag+1):cstr_row,i);

        for n=1:no_lag,
            cstr_tlshift(:,(i*(no_lag+1)-no_lag+n))=cstr((no_lag+1-n):(cstr_row-n),i);
        end
    end
    n=0;
end

save('dpca_all_column.mat','cstr_tlshift');

%-----DPCA-----%

clear all; clc;
load('dpca_all_column.mat')

cstr=cstr_tlshift;

mn=mean(cstr); %mean
sd=std(cstr); %standard deviation

save('dpca_all_column_CAT.mat','mn','sd','-append');
save('dpca_all_column_HE.mat','mn','sd','-append');

[cstr_row, cstr_column]=size(cstr); %state column size for normalization
loop

for i=1:cstr_column, %normalization loop
    norm_column=(cstr(:,i)-mn(i))./sd(i);
    cstr_norm(:,i)=norm_column;
end
i=0;

[coeff,score,latent,tsquare,explained,mu] = princomp(cstr_norm); %principal component
analysis

save('dpca_all_column_CAT.mat','coeff','latent','-append');
save('dpca_all_column_HE.mat','coeff','latent','-append');
```

```

[coeff_row,coeff_column]=size(coeff);

for i=1:cstr_column, %convert latent to square matrix
    latent_mat(i,i)=latent(i,1);
end
i=0;

mn_score=mean(score); %mean of score matrix
sd_score=std(score); %standard deviation of score
matrix
save('dpca_all_column_CAT.mat','mn_score','sd_score','-append');
save('dpca_all_column_HE.mat','mn_score','sd_score','-append');

[score_row, score_column]=size(score); %state column size for
normalization loop

for i=1:score_column, %normalization loop
    nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);
    score_norm(:,i)=nSCORE_column;
end
i=0;

no_princomp=9; %no of retained component. 9 no princomp explained 72.5726%
save('dpca_all_column_CAT.mat','no_princomp','-append');
save('dpca_all_column_HE.mat','no_princomp','-append');

score=score(:,1:no_princomp);

score_square=(score_norm.^2);
%[coeff,score,latent,tsquare] = princomp(score_norm);
for i=1:score_row,
    column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1)');
    tsquare(i,1)=sum(column_t(i,:));
end
i=0;

backprojection=score_norm*(coeff)';
r=cstr_norm-backprojection;
q=r.*r;

[r_row, r_column]=size(r);
SPE=sum(q');

figure(2)
subplot(2,1,1);
plot(SPE) %SPE Chart

chisquare_99=chi2inv(0.99,cstr_column-1);
chisquare_95=chi2inv(0.95,cstr_column-1);

theta1=sum(latent((no_princomp+1):cstr_column,1));
theta2=sum(latent((no_princomp+1):cstr_column,1).^2);
theta3=sum(latent((no_princomp+1):cstr_column,1).^3);
g=theta2/theta1;
h=(theta1^2)/theta2;
h0=1-((2*theta1*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

SPE_threshold99=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_99*(2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0)) %SPE limit alternative 1
SPE_threshold95=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_95*(2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0))

```

```

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

finv_99=finv(0.99,no_princomp,(score_row-no_princomp)); %F alpha (no_princomp, (no
of sample - no_princomp))

finv_95=finv(0.95,no_princomp,(score_row-no_princomp)); %F alpha (no_princomp, (no
of sample - no_princomp))

thold_99=((no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp)))*finv_99;

thold_95=(no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp))*finv_95;

subplot (2,1,2);
plot(tsquare) %T2 chart
line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

'-----end of program-----'

```

DPCA(ALL Column) for Change In Kinetic

```
clear all; clc;
load('cstr_data_CAT.mat')
cstr=cstr';
cstr=cstr(1:1000,2:8);

[cstr_row, cstr_column]=size(cstr);
%cstr_tlshift=cstr(3:cstr_row,cstr_column);
no_lag=2;

for i=1:cstr_column,
    if i==1;
        cstr_tlshift(:,i)= cstr((no_lag+1):cstr_row,i);

        for n=1:no_lag,
            cstr_tlshift(:,i+n)=cstr((no_lag+1-n):(cstr_row-n),i);
        end
        n=0;

    else
        cstr_tlshift(:,(i*(no_lag+1)-no_lag))= cstr((no_lag+1):cstr_row,i);

        for n=1:no_lag,
            cstr_tlshift(:,(i*(no_lag+1)-no_lag+n))=cstr((no_lag+1-n):(cstr_row-n),i);
        end

    end
    n=0;

end

save('dpca_all_column_CAT.mat','cstr_tlshift','-append');
%-----DPCA-----%
clear all; clc;

load('dpca_all_column_CAT.mat')
cstr=cstr_tlshift;

[cstr_row, cstr_column]=size(cstr);           %state column size for normalization
loop

for i=1:cstr_column,                          %normalization loop
    norm_column=(cstr(:,i)-mn(i))./sd(i);
    cstr_norm(:,i)=norm_column;
end
i=0;

score=cstr_norm*coeff
[coeff_row,coeff_column]=size(coeff)

for i=1:cstr_column, %convert latent to square matrix
    latent_mat(i,i)=latent(i,1);
end
i=0;

[score_row, score_column]=size(score);       %state column size for
normalization loop
for i=1:score_column,                        %normalization loop
    nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);
    score_norm(:,i)=nSCORE_column;
end
i=0
```

```

score_square=(score_norm.^2)

for i=1:score_row, %T-
square loop
    column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1))';
    tsquare(i,1)=sum(column_t(i,:));
end
i=0

%r=pcares(cstr_norm,no_princomp); %pcares return residual from PCA
%q=r.*r;

backprojection=score_norm*(coeff)'; %back projection from the score matrix

r=cstr_norm-backprojection; %residual
q=r.*r; %the Q/SPE statistics

[r_row, r_column]=size(r);

SPE=sum(q');
figure(2)
subplot(2,1,1);
plot(SPE) %SPE Chart

chisquare_99=chi2inv(0.99,cstr_column-1);
chisquare_95=chi2inv(0.95,cstr_column-1);

theta1=sum(latent((no_princomp+1):score_column,1));
theta2=sum(latent((no_princomp+1):score_column,1).^2);
theta3=sum(latent((no_princomp+1):score_column,1).^3);
g=theta2/theta1;
h=(theta1^2)/theta2;
h0=1-((2*theta1*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

SPE_threshold99=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_99*((2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0)) %SPE limit alternative 1
SPE_threshold95=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_95*((2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0))

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

finv_99=finv(0.99,no_princomp,(score_row-no_princomp)) %F alpha (no_princomp, (no
of sample - no_princomp))

finv_95=finv(0.95,no_princomp,(score_row-no_princomp)) %F alpha (no_princomp, (no
of sample - no_princomp))

thold_99=((no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp)))*finv_99;

thold_95=(no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp))*finv_95;

```

```

subplot (2,1,2);

plot(tsquare)      %T2 chart
line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
     'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
     'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
      'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')
datestr(clock,0)
'-----end of program-----'

```

DPCA(ALL Column) for Change In Heat Transfer Coefficient

```
clear all; clc;
load('cstr_data_HE.mat')
cstr=cstr';
cstr=cstr(1:1000,2:8);

[cstr_row, cstr_column]=size(cstr);

%cstr_tlshift=cstr(3:cstr_row,cstr_column);

no_lag=2;

for i=1:cstr_column,

    if i==1;
        cstr_tlshift(:,i)= cstr((no_lag+1):cstr_row,i);

        for n=1:no_lag,
            cstr_tlshift(:,i+n)=cstr((no_lag+1-n):(cstr_row-n),i);
        end
        n=0;

    else
        cstr_tlshift(:,(i*(no_lag+1)-no_lag))= cstr((no_lag+1):cstr_row,i);

        for n=1:no_lag,
            cstr_tlshift(:,(i*(no_lag+1)-no_lag+n))=cstr((no_lag+1-n):(cstr_row-n),i);
        end

    end
    n=0;

end

save('dpca_all_column_HE.mat','cstr_tlshift','-append');

%-----DPCA-----%

clear all; clc;
load('dpca_all_column_HE.mat')
cstr=cstr_tlshift;

[cstr_row, cstr_column]=size(cstr);           %state column size for normalization
loop

for i=1:cstr_column,                           %normalization loop
    norm_column=(cstr(:,i)-mn(i))./sd(i);
    cstr_norm(:,i)=norm_column;
end
i=0;

score=cstr_norm*coeff;

[coeff_row,coeff_column]=size(coeff);

for i=1:cstr_column, %convert latent to square matrix
    latent_mat(i,i)=latent(i,1);
end
i=0;
```

```

[score_row, score_column]=size(score);           %state column size for
normalization loop
for i=1:score_column,                           %normalization loop
    nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);
    score_norm(:,i)=nSCORE_column;
end
i=0;

score_square=(score_norm.^2);

for i=1:score_row,                             %T-
square loop
    column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1))';
    tsquare(i,1)=sum(column_t(i,:));
end
i=0;

%r=pcares(cstr_norm,no_princomp); %pcares return residual from PCA
%q=r.*r;

backprojection=score_norm*(coeff)'; %back projection from the score matrix

r=cstr_norm-backprojection; %residual
q=r.*r; %the Q/SPE statistics

[r_row, r_column]=size(r);

SPE=sum(q');

figure(2)

subplot (2,1,1);
plot (SPE) %SPE Chart

chisquare_99=chi2inv(0.99,cstr_column-1);
chisquare_95=chi2inv(0.95,cstr_column-1);

theta1=sum(latent((no_princomp+1):score_column,1));
theta2=sum(latent((no_princomp+1):score_column,1).^2);
theta3=sum(latent((no_princomp+1):score_column,1).^3);
g=theta2/theta1;
h=(theta1^2)/theta2;
h0=1-((2*theta1*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

SPE_threshold99=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_99*((2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0)); %SPE limit alternative 1
SPE_threshold95=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_95*((2*theta2*(h0^2))^0.5)/(theta1)))^(1/h0));

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

```



```

finv_99=finv(0.99,no_princomp,(score_row-no_princomp));    %F alpha (no_princomp, (no
of sample - no_princomp))

finv_95=finv(0.95,no_princomp,(score_row-no_princomp));    %F alpha (no_princomp, (no
of sample - no_princomp))

thold_99=((no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp)))*finv_99;

thold_95=(no_princomp)*(score_row-1)*(score_row+1)/(score_row*(score_row-
no_princomp))*finv_95;

subplot (2,1,2);
plot(tsquare)      %T2 chart

line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

datestr(clock,0)
'-----end of program-----'

```

Neural Network Training and Simulation for Kinetic and Heat Transfer Coefficient Change

```
clear all; clc;
load('cstr_data.mat');
cstr=cstr(2:8,1:1000)';

mn_normal=mean(cstr); %mean
sd_normal=std(cstr); %standard deviation

[cstr_row, cstr_column]=size(cstr); %state column size for normalization
loop
for i=1:cstr_column, %normalization loop
    norm_column=(cstr(:,i)-mn_normal(i))./sd_normal(i);
    cstr(:,i)=norm_column;
end
i=0;
cstr=cstr';
ncstr_normal_input=cstr;
save('ann_data.mat','ncstr_normal_input','-append');

net1=feedforwardnet;
net1.numLayers=5;
net1.numInputs=1;
%net1.inputconnect(1,1:7)=1

net1.layers{1}.name='linearize input';
net1.layers{2}.name='hidden 1';
net1.layers{3}.name='bottleneck';
net1.layers{4}.name='hidden 2';
net1.layers{5}.name='linearize output';

net1.outputConnect=[0 0 0 0 1];
net1.layerconnect(2,1)=1;
net1.layerconnect(3,2)=1;
net1.layerconnect(4,3)=1;
net1.layerconnect(5,4)=1;

net1.biasconnect=[1;1;1;1;1];

net1.layers{1}.dimensions=7;
net1.layers{2}.dimensions=10;
net1.layers{3}.dimensions=3;
net1.layers{4}.dimensions=10;
net1.layers{5}.dimensions=7;

net1.layers{1}.transferFcn='purelin';
net1.layers{2}.transferFcn='logsig';
net1.layers{3}.transferFcn='tansig';
net1.layers{4}.transferFcn='logsig';
net1.layers{5}.transferFcn='purelin';

net1.layers{3}.initFcn='initnw';
net1.layers{4}.initFcn='initnw';
net1.layers{5}.initFcn='initnw';
%view(net1)

rng(580301); %random seed applied
net2=trainlm(net1,cstr,cstr);
%y=net2(xxxx); %xxxx is the new input and y is output%
%view(net2)

net2.outputConnect(1,3)=1; %output of bottleneck layer, appear in top 3
row in output matrix
```

```

cstr_normal_output=net2(cstr);

%-----RUN FOR KINETIC CHANGE-----%
clear norm_column; clear cstr;
load('cstr_data_CAT.mat');
cstr=cstr(2:8,1:1000)';

for i=1:cstr_column, %normalization loop
    norm_column=(cstr(:,i)-mn_normal(i))./sd_normal(i);
    cstr(:,i)=norm_column;
end
i=0;
cstr=cstr';
ncstr_CAT_input=cstr;
save('ann_data.mat','ncstr_CAT_input','-append');

cstr_cat_output=net2(ncstr_CAT_input); %network output, 3 row from
top is the bottleneck output
save('ann_data.mat','cstr_normal_output','cstr_cat_output','-append');

%-----RUN FOR HE COEFFICIENT CHANGE-----%
clear norm_column; clear cstr;
load('cstr_data_HE.mat');
cstr=cstr(2:8,1:1000)';

for i=1:cstr_column, %normalization loop
    norm_column=(cstr(:,i)-mn_normal(i))./sd_normal(i);
    cstr(:,i)=norm_column;
end
i=0;
cstr=cstr';
ncstr_HE_input=cstr;
save('ann_data.mat','ncstr_HE_input','-append');

%net2.outputConnect(1,3)=1;

cstr_HE_output=net2(cstr); %network output, 3 row from top is the
bottleneck output

save('ann_data.mat','cstr_normal_output','cstr_cat_output','cstr_HE_output','-
append');
%fyp2_ann_pca

```

T² and SPE for Data From the Neural Network

```
load('ann_data_final.mat');

cstr_normal_output=cstr_normal_output';
mn_bottle=mean(cstr_normal_output);
sd_bottle=std(cstr_normal_output);
var_bottle=var(cstr_normal_output);

for i=1:10, %normalization loop %bottleneck only first 3 column
    ncstr_normal_output_column=(cstr_normal_output(:,i)-mn_bottle(i))./sd_bottle(i);
    ncstr_normal_output(:,i)=ncstr_normal_output_column;
end
i=0;

ncstr_normal_square=(ncstr_normal_output(:,1:3).^2); %ncstr_normal_square for T2 calc.

for i=1:1000,
    column_normal_t(i,:)=ncstr_normal_square(i,:)./(var_bottle(1:3));
    tsquare_normal(i,1)=sum(column_normal_t(i,:));
end
i=0;

finv_99=finv(0.99,3,(1000-3)) %F alpha (no_princomp, (no of sample - no_princomp))
finv_95=finv(0.95,3,(1000-3)) %F alpha (no_princomp, (no of sample - no_princomp))

thold_99=((3)*(1000-1)*(1000+1)/(1000*(1000-3)))*finv_99;
thold_95=((3)*(1000-1)*(1000+1)/(1000*(1000-3)))*finv_95;

figure(1)
subplot (2,1,2);

plot(tsquare_normal) %T2 chart

line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

r=ncstr_normal_input'-cstr_normal_output(:,4:10);
q=r.*r;
SPE=sum(q');

figure(1)
subplot (2,1,1);
plot (SPE) %SPE Chart

v=var(SPE);
m=mean(SPE);
H=(2*(m^2))/v;
chisquare99=chi2inv(0.99,H);
chisquare95=chi2inv(0.95,H);
```

```

sigma99=(v/(2*m))*(chisquare99)*((2*(m^2))/v);
sigma95=(v/(2*m))*(chisquare95)*((2*(m^2))/v);

SPE_threshold99=sigma99;
SPE_threshold95=sigma95;

thetal=sum(var_bottle(1,(3+3+1):10));           %3+3+1   (var bottle layer + var
princomp + 1)
theta2=sum(var_bottle(1,(3+3+1):10).^2);
theta3=sum(var_bottle(1,(3+3+1):10));
g=theta2/thetal;
h=(thetal^2)/theta2;
h0=1-((2*thetal*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);

chisquare_99=chi2inv(0.99,7-1);
chisquare_95=chi2inv(0.95,7-1);

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

%-----PCA ANN KINETIC CHANGES-----%

cstr_cat_output=cstr_cat_output';
for i=1:10,      %normalization loop %bottleneck only first 3 column
    ncstr_cat_output_column=(cstr_cat_output(:,i)-mn_bottle(i))./sd_bottle(i);
    ncstr_cat_output(:,i)=ncstr_cat_output_column;
end
i=0;

ncstr_cat_square=(ncstr_cat_output(:,1:3).^2);

for i=1:1000,
    column_cat_t(i,:)=ncstr_cat_square(i,:)./(var_bottle(1:3));
    tsquare_cat(i,1)=sum(column_cat_t(i,:));
end
i=0;

finv_99=finv(0.99,3,(1000-3))      %F alpha (no_princomp, (no of sample - no_princomp))
finv_95=finv(0.95,3,(1000-3))      %F alpha (no_princomp, (no of sample - no_princomp))
thold_99=((3)*(1000-1)*(1000+1)/(1000*(1000-3)))*finv_99;
thold_95=((3)*(1000-1)*(1000+1)/(1000*(1000-3)))*finv_95;

figure(2)
subplot (2,1,2);

plot(tsquare_cat)      %T2 chart

line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','y');

```

```

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
      'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

r_cat=ncstr_CAT_input'-cstr_cat_output(:,4:10);
q_cat=r_cat.*r_cat;
SPE_cat=sum(q_cat');

figure(2)
subplot (2,1,1);
plot (SPE_cat)           %SPE Chart

line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
    ...
    'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
    ...
    'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
      'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

%-----PCA ANN HEAT TRANSFER COEFFICIENT CHANGE-----%
cstr_HE_output=cstr_HE_output';
for i=1:10, %normalization loop %bottleneck only first 3 column
    ncstr_HE_output_column=(cstr_HE_output(:,i)-mn_bottle(i))./sd_bottle(i);
    ncstr_HE_output(:,i)=ncstr_HE_output_column;
end
i=0;

ncstr_HE_square=(ncstr_HE_output(:,1:3).^2);

for i=1:1000,
    column_HE_t(i,:)=ncstr_HE_square(i,:)./(var_bottle(1:3));
    tsquare_HE(i,1)=sum(column_HE_t(i,:));
end
i=0;

finv_99=finv(0.99,3,(1000-3)) %F alpha (no_princomp, (no of sample - no_princomp))
finv_95=finv(0.95,3,(1000-3)) %F alpha (no_princomp, (no of sample - no_princomp))
thold_99=((3)*(1000-1)*(1000+1)/(1000*(1000-3)))*finv_99;
thold_95=((3)*(1000-1)*(1000+1)/(1000*(1000-3)))*finv_95;

figure(3)
subplot (2,1,2);

plot(tsquare_HE) %T2 chart

line('XData', [0 1000], 'YData', [thold_99 thold_99], 'LineStyle', '-', ...
    'LineWidth', 2, 'Color','r');

line('xData', [0 1000], 'yData', [thold_95 thold_95], 'LineStyle', '--', ...
    'LineWidth', 2, 'Color','y');

```

```

legend('T-square','99.0% confidence limit','95.0% confidence limit',...
      'Location','NorthEastOutside')

title('T-square Chart','FontWeight','bold')
xlabel('sample number')
ylabel('T2')

r_HE=ncstr_HE_input'-cstr_HE_output(:,4:10);
q_HE=r_HE.*r_HE;
SPE_HE=sum(q_HE');

figure(3)
subplot (2,1,1);
plot (SPE_HE)           %SPE Chart
line('XData', [0 1000], 'YData', [SPE_threshold99 SPE_threshold99], 'LineStyle', '-',
...
     'LineWidth', 2, 'Color','r');
line('xData', [0 1000], 'yData', [SPE_threshold95 SPE_threshold95], 'LineStyle', '--',
...
     'LineWidth', 2, 'Color','y');

legend('residual','99.0% confidence limit','95.0% confidence limit',...
      'Location','NorthEastOutside')

title('SPE-Chart','FontWeight','bold')
xlabel('sample number')
ylabel('residual')

```