Design of Input Sequence to Capture Adequate Non Linearity for Empirical Modeling Purposes

By

Siti Kathijah Binti Wahi

7068

Dissertation submitted in partial fulfillment of the requirement for the Bachelor of Engineering (Hons.)
Chemical Engineering

JAN 2009

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

CERTIFICATION OF APPROVAL

Design of Input Sequence to Capture Adequate Non Linearity for Empirical Modeling Purposes

By

Siti Kathijah Binti Wahi

7068

Dissertation submitted in partial fulfillment of the requirement for the Bachelor of Engineering (Hons.)

Chemical Engineering

Approved by,

[illegible]

(Mrs Haslinda Zabiri)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

MAY 2008

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 the original work contained herein have not been undertaken or done by unspecified sources or persons

[illegible]

(SITI KATHIJAH BINTI WAHI)

7068

Bsc. Hons. Chemical Engineering

Universiti Teknologi PETRONAS

I/C 860617-52-5684

ABSTRACT

The objective of this report is to discuss the preliminary research done and basic understanding of the chosen topic, which is Design of Input Sequence to Capture Adequate Non Linearity. The ultimate aim of the project is to find the best input sequence that can capture adequate non linearity and give the best predictive empirical model of Continuous Stir Tank Reactor. The challenge in this project is to find the available input sequence, which has been available in other people research project, applied them in MATLAB Simulink and further tested in various types of Neural Network. Simulation model will be design to test for the best input sequence that will give the best result for prediction of output. Once the result from the simulation has been get, the best input sequence will be test on the real system to prove that the result obtain in the real cases is similar with simulation that had been carried out.

ACKNOWLEDGEMENT

First of all I would like to express most gratitude to Allah, the Almighty for giving us time to undergo and complete this project successfully.

Utmost appreciation to the supervisor, Mrs Haslinda Zabiri, for giving and sharing his knowledge with me and give so much brilliant idea for completion of this project.

I would also like to express my appreciation to Mr Totok, Miss Afny and Miss Fariha, PHD and post graduate student of UTP for their guidance in this project. My appreciation also goes to the rest of Chemical Engineers and support staffs; too many to mention every single one of you here; for the supervision, guidance, care and patience in dealing with me.

I would also like to thank FYPII coordinator, Mr. Mohd Tazli Azizan, Dr Chong Fai Kiat and Mrs Haslinda Zabiri for coordinating the FYPII successfully. This appreciation also goes out to all UTP lecturers, for the advice and knowledge shared.

Last but not least, thanks to all my friends whose had helped either directly or indirectly upon the completion of this project. With full support and cooperation by my family and friends, this project is completed within the time line. Loads of support through good and hard times by other colleague will always be remembered.

Also, an utmost appreciation to those we missed in mentioning here, everyone that involved with us in making this project a success.

A million thanks to each and every one of you.

Contents

ABSTRACT 4

ACKNOWLEDGEMENT 5

List of Figures 8

CHAPTER 1 10

INTRODUCTION 10

1.1 Background of Study 10

1.2 Problem Statement 12

1.3 Significance of Project 12

1.4 Objective and Scope of Study 12

CHAPTER 2 14

LITERATURE REVIEW 14

2.1 Input Sequence 14

2.1.1 Definition 14

2.1.2 Types 15

2.1.3 Type of Model Function 22

2.2 Nonlinearity 27

2.3 Empirical Model 28

2.3.1 Neural Network Model 29

2.3.2 Gaussian Model 31

2.3.3 Wiener Model 32

2.3.4 Hammerstein Model 33

2.3.5 Volterra Model 35

CHAPTER 3 37

METHODOLOGY 37

3.2 Development of CSTR Model 38

3.3 Input Sequence Generation 39

3.4 Neural Network Prediction 40

CHAPTER 4 42

RESULT& DISCUSSION 42

4.1 CSTR Model Development 42

4.2 Input-Output Data Generation 43

4.3 Neural Network Prediction 50

CHAPTER 5 60

CONCLUSION & RECOMMENDATION 60

5.1 Conclusion 60

5.2 Recommendation 61

APPENDIX 62

List of Figures

Figure 1: Schematic of the System Identification Problem 11
Figure 2: Step Input 15
Figure 3: Ramp Input 15
Figure 4: Sine Wave InpuT 16
Figure 5: Rectangular Pulse Input 16
Figure 6: Sawtooth Wave 17
Figure 7: PRBS Signal 18
Figure 8: Gaussian Random Noise 19
Figure 9: Repeating Sequence Stair 19
Figure 10: Control Loop with Nonlinearity 28
Figure 11: Feedforward NN Architecture 30
Figure 12: NARXSP NN Architecture 31
Figure 13: Schematic Project Work Flow 37
Figure 14: CSTR 38
Figure 15: Simulink Library Source Block 40
Figure 16: Simulink Based CSTR Model 42
Figure 17: Cooling Jacket 43
Figure 18: Concentration Profile 43
Figure 19: Temperature Profile 43
Figure 20: Cooling Jacket 43
Figure 21: Concentration Profile 43
Figure 22: Temperature Profile 43
Figure 23: Cooling Jacket 44
Figure 24: Concentration Profile 44
Figure 25: Temperature Profile 44
Figure 26: Cooling Jacket 44
Figure 27: Concentration Profile 44
Figure 28: Temperature Profile 44
Figure 29: Cooling Jacket 45
Figure 30: Concentration Profile 45
Figure 31: Temperature Profile 45
Figure 32: Cooling Jacket 45
Figure 33: Concentration Profile 45
Figure 34: Temperature Profile 45
Figure 35: Cooling Jacket 46
Figure 36: Concentration Profile 46
Figure 37: Temperature Profile 46
Figure 38: Cooling Jacket 46
Figure 39: Concentration Profile 46
Figure 40: Temperature Profile 46
Figure 41: Cooling Jacket 47

Figure 42: Concentration Profile 47
Figure 43: Temperature Profile........ 47
Figure 44: Cooling Jacket........ 47
Figure 45: Concentration Profile 47
Figure 46: Temperature Profile........ 47
Figure 47: Cooling Jacket........ 48
Figure 48: Concentration Profile 48
Figure 49: Temperature Profile........ 48
Figure 50: Cooling Jacket........ 48
Figure 51: Concentration Profile 48
Figure 52: Temperature Profile........ 48
Figure 53: Cooling Jacket........ 49
Figure 54: Concentration Profile 49
Figure 55: Temperature Profile........ 49
Figure 56: Cooling Jacket........ 49
Figure 57: Concentration Profile 49
Figure 58: Temperature Profile........ 49

List of Table

Table 1: NARXSP NN Prediction According to Inputs 50
Table 2: Feed Forward BP Prediction According to Inputs........ 54
Table 3: Baruch1 and Liu Combination Input Prediction........ 58
Table 4: Baruch1 and Soni 1 Combination Input Prediction........ 58

CHAPTER 1

INTRODUCTION

1.1 Background of Study

Detailed mathematical modeling is increasingly being used by companies to gain competitive advantage through such applications as model-based process design, control and optimization. Thus, building various types of high quality models to represent the real systems has become the main concern to most of industries. This activity involves the use of several methods and techniques including model solution techniques, nonlinear systems identification, model verification and validation, optimal design of experiments and etc **(S.P. Asprey).**

This paper will discuss about nonlinear system identification and focus mainly on design of input sequence. Nonlinear system identification involves the following tasks **(Ramesh et al, 2007)**:

- Structure selection – Selection of suitable nonlinear model structure and number of model parameters
- **Input sequence design – Determination of the input sequence u(t) which is injected into the plant to generate the output sequence y(t).**
- Noise modeling – Determination of the dynamic model which generates noise input e(t).
- Parameter estimation – Estimation of the remaining model parameters from the dynamic plant data u(t) and y(t) and the noise input e(t).
- Model validation – Comparison of plant data and model predictions for data not used in model development.

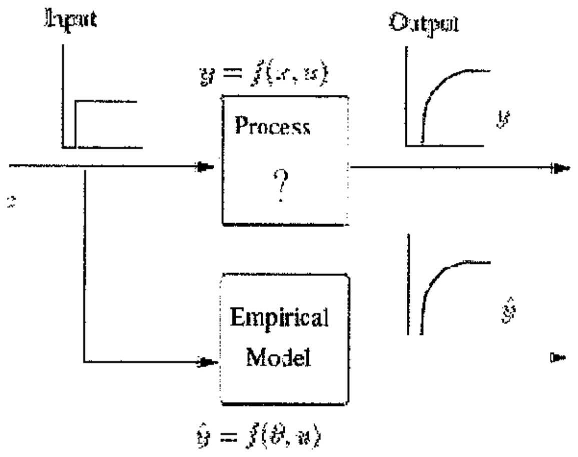


Figure 1: Schematic of the System Identification Problem

Dynamic modeling is used in this paper to describe the behavior of a distributed parameter system in terms of how one qualitative state can turn into another. It can also describe as a medium to express and model the behavior of the system over time. The term dynamics is referring to unsteady state or transient behavior **(Ramasamy M, 2007)**.

Modeling is very essential, in order to fulfill two main objectives; maintain a process at the desired operating conditions, safely and efficiently and satisfy product quality and environmental requirements **(Ramasamy M, 2007)**. There are basically three types of modeling approach **(Ramasamy M, 2007)**; white box (fundamental/physical approach), grey box (semi empirical approach) and black box (empirical approach). For this study, the author employ empirical model approach to model the non linear dynamic behavior of Continuous Stir Tank Reactor and design the optimum input sequence to generate the best output of prediction. Empirical model is intensively used due to the fact that empirical model can give solution to the problem formulation within smaller range of time compare to the first principle modeling. As empirical approach is also known as black box, therefore there's no need of lengthy mathematical equation to describe the real system.

1.2 Problem Statement

Many of process system nowadays exhibit nonlinear behavior. However, most of the widely used, existing empirical model structures are linear. In some cases this basic model formulation may not be able to adequately capture the nonlinear process dynamics. In addition, the input sequence available nowadays has only small operating region. The empirical model may work well to predict only certain non linear behavior by using certain type of input sequence. But if other type of non linear behavior is existed in the process, the modeling will not have an ability to cater and predict the future behavior of the process. Even if the model can give some output of prediction, the output actually doesn't represent the actual behavior of the process. Or the worst thing that would happen is the system can become haywire. Therefore, this project is conduct to design input sequence that will capture adequate non linearity in the process behavior.

1.3 Significance of Project

The best input sequence from this research can be applied in the real system in industries, so that the control of process behavior will be more efficient. By having the best control system, people will easier to know and predict future behavior of the process and do the correction actions before the disturbance upset the process. This will not just saving the money but also time friendly.

1.4 Objective and Scope of Study

The main objectives of this research are:

- To test various input sequence in Continuous Stir Tank Reactor (CSTR) modeling by using MATLAB.
- To select the best input sequence that will generate the best predicted output that represent the actual output behavior of the process.
- To apply the best input sequence to the real system to test the performance of modeling, and check whether the result for simulation test is acceptable or not.

The primary focus of the project is to develop a model by using MATLAB software to get the best input sequence that will give the best predictive model. In order to achieve the objectives, a few tasks and research need to be carried out by collecting all technical details regarding the existing input sequence and by studying the fundamental and application of MATLAB software. The author has also undergone some discussion with expert throughout meeting, training and seminar. The project is assigned to be completed within two semester's time

CHAPTER 2

LITERATURE REVIEW

2.1 Input Sequence

2.1.1 Definition

Input can be defined as a "things that cause" or "stimuli", while output are "things that are affected" or "responses". The word input refers to any variable that influences the process output **(Ramasamy, 2007)**.

There are several common ways to define the word input:
1: The input used to generate the dataset, and it can be tailored to facilitate estimation

2: A state, or a sequence of states, of a point that accepts data.

3: A signal or interference that enters a functional unit, such as a telecommunications system, a computer, or a computer program.

4: Data that is ready for entry into the computer model.

Since the input is implemented in the plant, it should satisfy certain characteristics. First, the input signal must have sufficient energy to excite the full range of nonlinear process dynamics.

Second, the input should be plant-friendly. Since this input signal is implemented by an actuator in the plant, such as a control valve, the input sequence should not have frequent transitions which cause actuator wear. In addition, the magnitude of the input sequence must not be large so that valve saturation is avoided. At the same time the magnitude of the input should be high enough to ensure that the output response lies outside the noise-band for process operation. In addition, a sequence with a length as short as possible is desirable so that system identification does not interfere with normal plant operation. However, these practical requirements are often in conflict with theoretical identification results. It has been shown that the coefficient error variance can be reduced by using a sequence as long as possible. In fact, asymptotic convergence is achieved as the sequence length approaches infinity.

2.1.2 Types

i) Step Input (1st order)

A sudden and sustained of input change **(Seborg et al, 2004)**. It can be describes in Figure 2. This type of input is simple to apply at practice **(Tsai et al, 1986).**

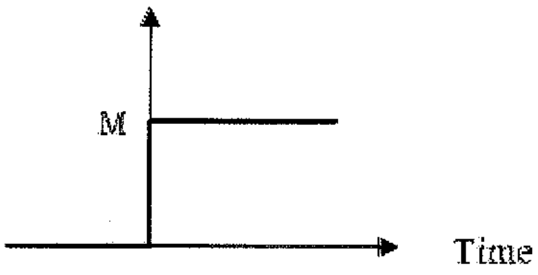


Figure 2: Step Input

$$U_S(t) = \begin{cases} 0 & t < 0 \\ M & t \geq 0 \end{cases}$$

ii) Ramp Input

Input that is gradually changes in upward or downward direction **(Seborg et al, 2004)**. It can be describes in Figure 3.

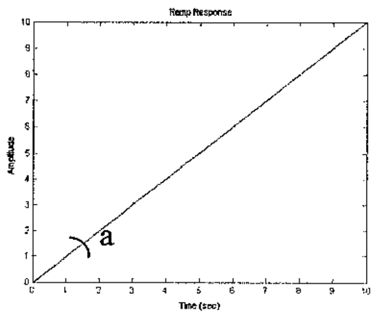
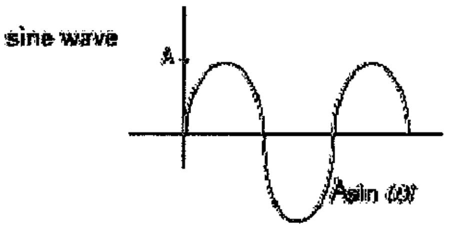


Figure 3: Ramp Input

$$U_R(t) = \begin{cases} 0 & t < 0 \\ at & t \geq 0 \end{cases}$$

iii) Sine Wave Input

A waveform that resembles as a sine curve (Figure 4). Theoretically, this is a simple method by which to obtain a wide range of the frequency responses by applying a wide range of input frequencies. However, in practice, this is not really applicable because the testing period can be extremely long during normal plant operation **(Tsai et al, 1986).**

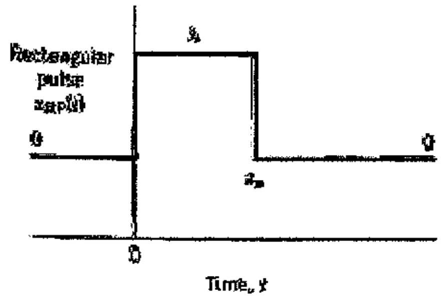


$$U_{sin}(t)=\begin{cases}0 & t<0\\ A\sin\omega t & t\geq 0\end{cases}$$

Figure 4: Sine Wave InpuT

iv) Pulse Input

A step change that limited to certain period of time **(Seborg et al, 2004)** (Figure 5). The pulse type input is any input which has a closed wave form with respect to the steady state reference level. This type of wave form is a popular practical method to obtain dynamic information of the process **(Tsai et al, 1986).**



$$U_{RP}(t)=\begin{cases}0 & t<0\\ h & 0\leq t<t_w\\ 0 & t\geq t_w\end{cases}$$

Figure 5: Rectangular Pulse Input

v) Sawtooth Wave

The sawtooth wave (or saw wave) (Figure 6) is a kind of non-sinusoidal waveform. It is named a sawtooth based on its resemblance to the teeth on the blade of a saw. The sawtooth wave, called the "castle rim function" by **(Trott , 2004)**, is the periodic function given by

$$S(x) = A\,\mathrm{frac}\left(\frac{x}{T} + \phi\right), \tag{1}$$

where $\mathrm{frac}(x)$ is the fractional part $\mathrm{frac}(x) \equiv x - \lfloor x \rfloor$, A is the amplitude, T is the period of the wave, and ϕ is its phase. It therefore consists of an infinite sequence of truncated ramp functions concatenated together.

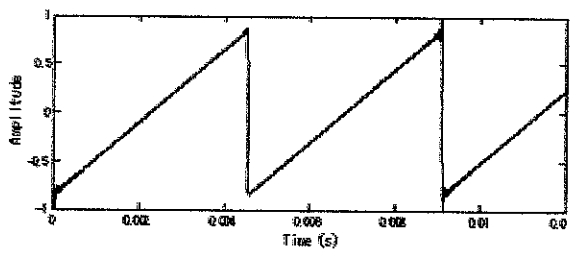


Figure 6: Sawtooth Wave

vi) Pseudorandom Binary Sequence

Pseudo-random Binary Sequence (Figure 7) is based on binary sequences of length N, where N is odd **(K R Godfrey, 1992)**. If the sequence has logic level 1 and 0 and the corresponding signal has voltage level ±V, then the transformation from logic level to voltage level is either 1→+V and 0→-V or 1→-V and 0→+V.

There are several classes of pseudo random binary sequences but one class, called maximum length sequence or m-sequence is very popular because the corresponding signal can easily be generated using shift register circuitry with appropriate feedback.

The advantage of the PRBS input **(Braun et al, 1999)** include ease of implementation and an autocorrelation function. Since the PRBS is periodic and deterministic, it can be designed to process excitation in a control relevant frequency range over a single data cycle. Because the PRBS can be applied to a process multiple times, it provides the user with a convenient means for discarding corrupted segments of data and retaining the most informative data for model estimation and validation.

The PRBS input, however, is not always well suited for nonlinear problems **(Braun et al, 1999)**. Since the PRBS consists of only two levels, the resulting data may not provide sufficient information to identify nonlinear behavior (e.g. y(k) = u2(k)). Additionally, a PRBS signal of too large a magnitude may bias the estimation of the linear kernel. Multi-level pseudo-random sequences (m-level PRS), in contrast, allow the user to highlight nonlinear system behavior while manipulating the harmonic content of the signal to enable unbiased estimation of the linear dynamics in the presence of nonlinearities.

According to (Heaven et al, 1993), in their paper "Application of System Identification to Paper Machine Model Development and Controller Design", the widely used input sequence in industry nowadays is Pseudo-random Binary Sequence. This sequence can be used to determine process dynamics, isolate multivariable process interaction, and develop advanced computer models to evaluate existing and new control strategies.

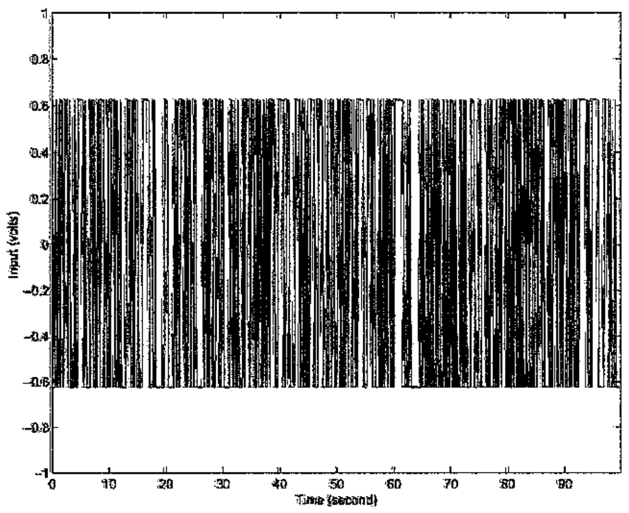


Figure 7: PRBS Signal

vii) Gaussian Random Noise (Figure 8)

White noise that has a probability density graphed as a normal distribution. Gaussian white noise is not a popular input sequence in practice because it results in constant changes in the input especially for unacceptably large input value. However, it is quite popular in the parameter estimation because it provides much information about process dynamics(**Parker et al**).

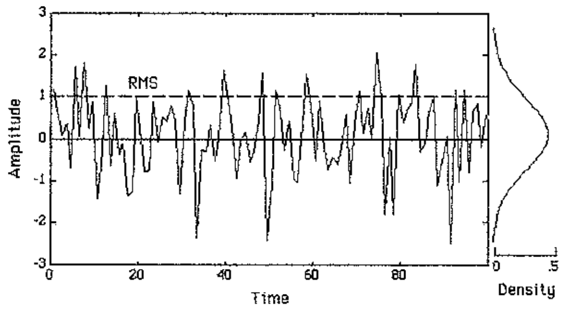


Figure 8: Gaussian Random Noise

viii) Repeating Sequence Stair

Repeat discrete time sequence.

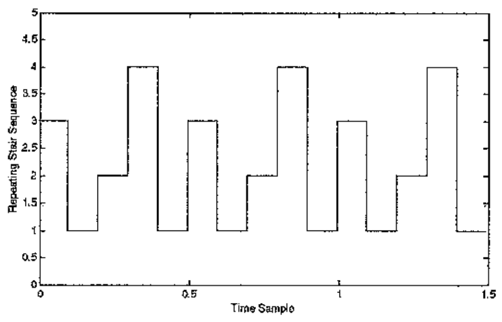


Figure 9: Repeating Sequence Stair

ix) Continuous switching–pace symmetric random sequence(**Soni and Parker, 2004)**

A 4M +4 length deterministic input sequence used to estimate the bias, linear, and diagonal parameters are :

$$u(k)=\begin{cases}\gamma_1 & k=0\\ 0 & 1\le k\le M\\ -\gamma_1 & k=M+1\\ 0 & M+2\le k\le 2M+1\\ \gamma_2 & k=2M+2\\ 0 & 2M+3\le k\le 3M+2\\ -\gamma_2 & k=3M+3\\ 0 & 3M+4\le k\le 4M+3\end{cases}$$

This sequence ensures that the contributions due to the nonlinear sub-diagonal and off-diagonal terms are zero identically, $(u(k - i)u(k - j) = 0\ Vi \neq j\ (i, j \leq M))$. The parameters γ_1 and γ_2 are selected such that $\gamma_2 > \gamma_1$. This is done in order to facilitate sufficient excitation of the model nonlinearities. In addition, the placement of the smaller pulse before the larger pulse guarantees that any residual error from the large pulse response does not corrupt the small pulse output data.

x) Input Design to Estimate Sub-Diagonal Parameter **(Soni and Parker, 2004).**

For this sequence the ratio given by

$$\frac{C_{3SD}}{C_{3OD}}=4.5(\kappa+2)\sigma_u^2$$

is 13102 which ensures selective excitation of the third-order sub-diagonal terms.
In order to derive the sub-diagonal estimator equations, an approach similar to that used for the derivation of the bias, linear, and the diagonal estimators is used.
The only difference is that in subsequent sub-units of the input sequence there is a gap between the two pulses. This gap length increases by one with increasing sub-units, and it is employed to ensure the tailored excitation of all of the sub-diagonal parameters. The first sub-unit of the input sequence is given as:

$$u(k)=\begin{cases}\lambda_1 & k=1\\ \lambda_2 & k=2\\ 0 & 3\le k\le M+1\\ -\lambda_2 & k=M+2\\ -\lambda_1 & k=M+3\\ 0 & M+4\le k\le 2M+2\end{cases}$$

Applying this to the system of interest and removing the bias, linear, and nonlinear diagonal contributions, recovers the residual z(k). The estimates for the third-order sub-diagonal coefficients can he obtained by minimizing the following sum-squared prediction error:

$$J_{xd} = \sum_{k=1}^{2M+2} \{z(k) - \hat{z}(k)\}^2$$

xi) Input Sequence for Constant, linear, and diagonal parameters (2M+ 2-point input sequence) **(Parker et al, 2001)**

u(k) =

γ	**$k = 0$;**
0	**$1 < k < M$;**
$-\gamma$	**$k = M+1$;**
0	**$M+2 < k < 2M+1$**

where

$\gamma > 0$

M=20

xii) Signal of sum of sinusoids with different frequencies **(Baruch et al).**

u(k) =

$\sin(\pi k/25)$	**$0 < k < 26$;**
1.0	**$25 < k < 51$;**
-1.0	**$50 < k < 76$;**
$0.3 \sin(\pi k/25) + 0.1 \sin(\pi k/32) + 0.6 \sin(\pi k/10)$,	**$75 < k < 101$**

xiii) Sequence of pulses with random amplitude and width **(Baruch et al).**

u(k) =

$\sin(\pi k/25)$,	**$0 < k < 251$;**
1.0	**$250 < k < 501$;**
-1.0	**$500 < k < 751$;**
$0.3 \sin(\pi k/25) + 0.1 \sin(\pi k/32) + 0.6 \sin(\pi k/10)$,	**$k < 1001$**

xiv) Signal of random sequence **(Liu et al, 1997).**

u(k) =

$\sin(2\pi k/250)$,	**$k \leq 500$;**
$0.8 \sin(2\pi k/250) + 0.2 \sin(2\pi k/25)$	**$k > 500$**

2.1.3 Type of Model Function

1) Radial Basis Function (RBF)

A radial basis function is a function which acts as an activation functions. They are used in function approximation, time series prediction, and control. An RBF is a weighted sum of translations of a radially symmetric basic function augmented by a polynomial term. The basic function Φ of a positive real r, where r is the distance(radius) from the origin. Popular choices for Φinclude

- The thin-plate spline (for fitting smooth functions of two variable

  $\phi(r) = r^2 \log(r)$

- The Gaussian (mainly for neural networks)

  $\phi(r) = \exp(-cr^2)$

- The multiquadric (for various applications, in particular fitting to topographical data)

  $\phi(r) = \sqrt{r^2 + c^2}$

For fitting functions of three variables, good choices include

- The biharmonic spline + linear polynomial

  $\phi(r) = r$

- The triharmonic spline + quadratic polynomial

  $\phi(r) = r^3$

The primary advantage of RBFs over binary features is that they produce approximate functions that vary smoothly and are differentiable. In addition, some learning methods for RBF networks change the centers and widths of the features as well. Such nonlinear methods may be able to fit the target function much more precisely. The disadvantage of RBF network especially, is greater computational complexity and, often, more manual tuning before learning is robust and efficient.

2) Polynomial Function

A polynomial function is one that has the form

$$y = a_{n}x^{n} + a_{n-1}x^{n-1} + \dots + a_{2}x^{2} + a_{1}x + a_{0}$$

with n denoting a non-negative integer that defines the *degree* of the polynomial. A polynomial with a degree of 0 is simply a constant, with a degree of 1 is a line, with a degree of 2 is a quadratic, and with a degree of 3 is a cubic, and so on.

Historically, polynomial models are among the most frequently used empirical models for fitting functions. These models are popular for the following reasons.

1. Simple form.
2. Well known and understood properties.
3. Moderate flexibility of shapes.
4. A closed family. Changes of location and scale in the raw data result in a polynomial model being mapped to a polynomial model. That is, polynomial models are not dependent on the underlying metric.
5. Computationally easy to use

However, polynomial models also have the following limitations.

1. Poor interpolatory properties. High degree polynomials are notorious for oscillations between exact-fit values.
2. Poor extrapolatory properties. Polynomials may provide good fits within the range of data, but they will frequently deteriorate rapidly outside the range of the data.

3. Poor asymptotic properties. By their nature, polynomials have a finite response for finite $x$values and have an infinite response if and only if the $x$value is infinite. Thus polynomials may not model asymptotic phenomena very well.
4. Have a shape or degree tradeoff. In order to model data with a complicated structure, the degree of the model must be high, indicating and the associated number of parameters to be estimated will also be high. This can result in highly unstable models.

3) Rational Function

A rational function is simply the ratio of two polynomial functions

$$y = \frac{a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0}{b_m x^m + b_{m-1} x^{m-1} + \dots + b_2 x^2 + b_1 x + b_0}$$

with ***n*** denoting a non-negative integer that defines the degree of the numerator and *m* denoting a non-negative integer that defines the degree of the denominator.

When fitting rational function models, the constant term in the denominator is usually set to 1.

A rational function model is a generalization of the polynomial model. It contains polynomial models as a subset in the case when the denominator is a constant.

If modeling using polynomial models is inadequate due to any of the limitations, a rational function model should be considered. The fitting rational function models are also referred to as the Pade approximation.

Rational function models have the following advantages.

1. Moderately simple form.
2. A closed family. As with polynomial models, this means that rational function models are not dependent on the underlying metric.
3. Can take on an extremely wide range of shapes, accommodating a much wider range of shapes than does the polynomial family.
4. Better interpolatory properties than polynomial models. Rational functions are typically smoother and less oscillatory than polynomial models.

5. Excellent extrapolatory powers. Rational functions can typically be tailored to model the function not only within the domain of the data, but also so as to be in agreement with theoretical/asymptotic behavior outside the domain of interest.
6. Excellent asymptotic properties. Rational functions can be either finite or infinite for finite values, or finite or infinite for infinite x values. Thus, rational functions can easily be incorporated into a rational function model.
7. Can often be used to model complicated structure with a fairly low degree in both the numerator and denominator. This in turn means that fewer coefficients will be required compared to the polynomial model.
8. Moderately easy to handle computationally. Although they are nonlinear models, rational function models are a particularly easy nonlinear models to fit.

However, rational function models have the following disadvantages.

1. The properties of the rational function family are not as well known to engineers and scientists as are those of the polynomial family. The literature on the rational function family is also more limited.
2. Unconstrained rational function fitting can, at times, result in undesired nusiance asymptotes (vertically) due to roots in the denominator polynomial.

4) Piecewise Function

In mathematics, a piecewise linear function is describe by

$$f : \Omega \rightarrow V,$$

where ;

V = vector space

Ω = subset of a vector space

A special case is when f is a real-valued function on an interval $[x_1,x_2]$. Then f is piecewise linear if and only if $[x_1,x_2]$ can be partitioned into finitely many sub-intervals, such that on each such sub-interval I, f is equal to a linear function

$$f(x) = a_I x + b_I.$$

The absolute value function $f(x) = | x |$ is a good example of a piecewise linear function. Other examples include the square wave, the sawtooth function, and the floor function.

Important sub-classes of piecewise linear functions include the continuous piecewise linear functions and the convex piecewise linear functions. Splines generalize piecewise linear functions to higher-order polynomials.

A piecewise function is a function that is defined on a sequence of intervals. A common example is the absolute value,

$$|x| = \begin{cases} -x & \text{for } x < 0 \\ 0 & \text{for } x = 0 \\ x & \text{for } x > 0. \end{cases}$$

Many nonlinearities that appear frequently in engineering systems are either piecewise-affine with a saturated linear actuator characteristic or can be approximated as piecewise-affine functions. Piecewise-affine systems are also a class of hybrid systems with a continuous-time state and a discrete-event state. For piecewise-affine systems the discrete-event state is associated with discrete modes of operation.

The continuous-time state is associated with the affine dynamics valid within each discrete mode. Piecewise-affine systems pose challenging problems because of its switched structure.

2.2 Nonlinearity

Nonlinearity by **(Nikolaou and Misra)** definition referring to the absence of linearity can manifest itself in various ways, such as nonlinear dynamics, constraints and changing modes of operation. There are two basic properties that characterize the behavior of a linear system according to **(Hangos and Cameron, 2001)**. The first one is the principle of Superposition. In general, this principle states that the response of a linear system to a sum of N input is the same as the sum of individual input. The second properties for linear system is independence of dynamic response character and process condition. However, if the system does not exhibit any of these properties, it is known as non linear system. The nonisothermal reactor is the popular example of non linear chemical process.

In many real world applications there are nonlinearities and unmodelled dynamics which poses problems when implement practical control strategies. Modern control such as adaptive and optimal control techniques and classical control theories has been commonly applied to deal with these difficulties. Nonlinear model process is dealing with issues like stability, efficient computation, optimization, constraints and others **(Kocijan and Smith)**. Interest in nonlinear feedback control of chemical processes has been steadily increasing over the last several years because both pronounced nonlinear nature of several chemical and increased sensing and computational capabilities afforded by modern sensors, computers, algorithms, and software. Such capabilities have been claimed and at times proven to offer benefits in better operation and control of chemical processes.

Below is example of industrially cases for which nonlinearity is usually present **(Nikolaou and Misra)**:

- Biochemical production of chemicals.
- Non-routine operation situations (e.g., start-ups, shut-downs, change-overs, flares, relief valve emissions). Because a process moves far from a steady state during non-routine operation, nonlinear behavior is usually pronounced.

- Nonlinear distributed process systems

  Example:

  - Control of spatial profiles: etching, crystal growth, packed-bed reactors
  - Control of size distributions: aerosol production and particulate processes.
  - Crystallization, emulsion, polymerization.
  - Cell cultures control of fluid flows: mixing, wave suppression, drag reduction, separation delay,
  - Control of material microstructure: thin-film growth and nano-structured coatings processing.
  - Batch processes: fine chemicals and pharmaceuticals

Figure below show the example of control loop with nonlinearity.

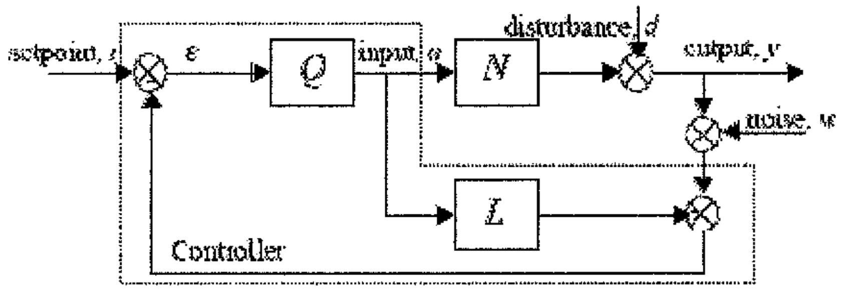


Figure 10: Control Loop with Nonlinearity

2.3 Empirical Model

As in many nonlinear systems it is not an easy task to come up with an accurate enough physical model of the plant, its finally turn to black-box model to describe the systems nonlinear dynamics.

Below are some descriptions of empirical models that have been study by researcher.

2.3.1 Neural Network Model

A neural network is a powerful data modeling tool that is able to capture and represent complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain. Among available structures, neural network have proved to work quite well in the identification of non linear systems on the basis input-output data. Despite neural networks are well known universal approximators, they are quite dependent on the quality of the data set. This feature together with a bounded number of iterations within the training phase leads inexorably to a model mismatch, which in turn is responsible for a static error, in worst case giving rise to instability of the feedback system.

The true power and advantage of neural networks lies in their ability to represent both linear and non-linear relationships and in their ability to learn the relationships directly from the data being modeled. Traditional linear models are simply inadequate when it comes to modeling data that contains non-linear characteristics. The advantages and disadvantages of neural network can be simplified as below:

Advantages:

- Can perform tasks that a linear program cannot.
- When an element of the neural network fails, it can continue without any problem by their parallel nature.
- A neural network learns and does not need to be reprogrammed.
- Can be implemented in any application.
- Can be implemented without any problem.

Disadvantages:

- Needs training to operate.
- The architecture of a neural network is different from the architecture of microprocessors therefore needs to be emulated.
- Requires high processing time for large neural networks.

The most common neural network model is the multilayer perceptron (MLP). This type of neural network is known as a supervised network because it requires a desired output in order to learn. The goal of this type of network is to create a model that correctly maps the input to the output using historical data so that the model can then be used to produce the output when the desired output is unknown.

Applied to control field, NN are essentially nonlinear models that can be useful to solve non-linear control problems. Basically, NN can be classified as static (feedforward) and dynamic (recurrent) **(Smith and Shorten, 2005)**.

- Static: Feedforward Backpropragation Network, Cascade Forward Backpropragation Network
- Dynamic: Recurrent Network - Non linear Autoregressive Network with Exogeneous Inputs (NARX), Bidirectional Recurrent Neural Network, Recurrent Cascade Correlation, Layered Recurrent Neural Network etc.

In this paper, the author used two types of NN which are Feedforward Backpropragation Network and Non linear Autoregressive Network with Exogeneous Inputs Series Parallel (NARXSP)

a) Feedforward Backpropragation Network

Feedforward backpropagation neural networks (FF networks) are the most popular and most widely used models in many practical applications **(Hagan et al, 1996)**. Figure 11 illustrates a FF networks network with three layers:

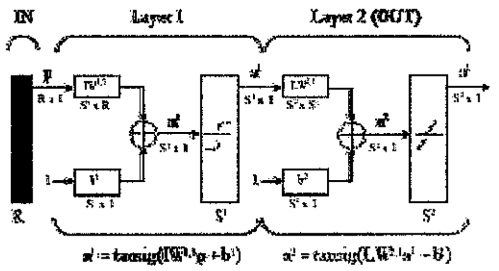


Figure 11: Feedforward NN Architecture

b) Non linear Autoregressive Network with Exogeneous Inputs Series Parallel (NARXSP)

Standard NARXSP architecture is as shown in Figure 12. The true output which is available during the training of the network is used instead of feeding back the estimated output. The advantage of this architecture is that the input to the feedforward network is more accurate. Besides, the resulting network has a purely feedforward architecture, and static back propagation can be used for training **(Zabiri and Mazuki, 2009).**

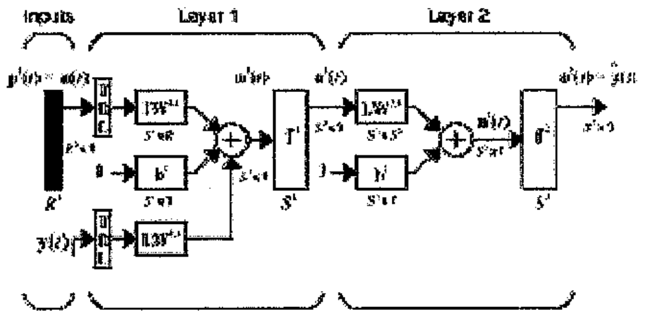


Figure 12: NARXSP NN Architecture

2.3.2 Gaussian Model

Each member of the family may be defined by two parameters, location and scale: the mean ("average", μ) and variance (standard deviation squared) σ2, respectively. Gaussian process models provide a probabilistic non-parametric modeling approach for black-box identification of nonlinear dynamic systems. The Gaussian processes can highlight areas of the input space where prediction quality is poor, due to the lack of data or its complexity, by indicating the higher variance around the predicted mean. It contains noticeably less coefficients to be optimized **(Kocijan and Smith).**

A Gaussian process is an example of the use of a flexible, probabilistic, nonparametric model which directly provides us with uncertainty predictions. A Gaussian process is a collection of random variables which have a join multivariate Gaussian distribution. A common choice is

$$C(x_p, x_q) = v_1 \exp\left[-\frac{1}{2}\sum_{d=1}^{D} w_d (x_p^d - x_q^d)^2\right] + v_0$$

Where $\theta = [w_1 \ldots w_D \; v_0 \; v_1]^T$ are the 'hyperparameters' of the covariance function and D the input dimension.

Consider a set of N D-dimensional input vector $X=[x_1,x_2 \ldots x_N]$ and a vector of output data $Y=[y^1, y^2 \ldots y^N]^T$. The advantage of using this model is we can use the same model to predict the output y* with input x*. Unlike other model, there is no model parameter determination as such within a fixed model structure. With this model, most of the effort consists in tuning the parameter with the covariance function. This is done by maximizing the log-likelihood of the parameter, which is computationally relatively demanding since the inverse of the data covariance matrix have to be calculated at every iteration.

For multistep ahead prediction, the uncertainty of future predictions provides the inputs for estimating further means and uncertainties. Gaussian process can, like neural networks, be used to model static nonlinearities and can therefore be used for modeling dynamic systems if delayed input and output signals are used as regressors **(Kocijan and Smith).**

The Gaussian process model not only can describe the dynamic characteristics of the non-linear system, but at the same time provides information about the confidence in the prediction. The Gaussian process can highlight areas of the input space where prediction quality is poor, due to the lack of data, by indicating the higher variance around the predicted mean.

2.3.3 Wiener Model

(Cervantes et al, 2003) in their research paper describes Wiener Nonlinear Model Predictive to control a CSTR with multiple steady states. This model can represent a process with linear dynamic but a nonlinear gain and can represent many of the memoryless nonlinear systems encountered in industrial processes. Due to the static nature of the nonlinearities, they can be removed from the control problem. However, some computational difficulty is potentially present and due to that an implicit inversion of the non linear static gain is needed. More specifically, the Wiener structure consists of a linear dynamic element followed in series by Static nonlinear element. The linear dynamic element uses a discrete state space model while the nonlinear element uses the Piecewise Linear approximation.

Assuming the system to be controlled can be described by the following discrete-time, nonlinear, state-space model:

$$x(k+1) = g(x(k), \Delta u(k))$$
$$y(k) = h(x(k)) + d(k)$$

Where;

x(k)=vector of state variable

Δu(k)= vector of control movements

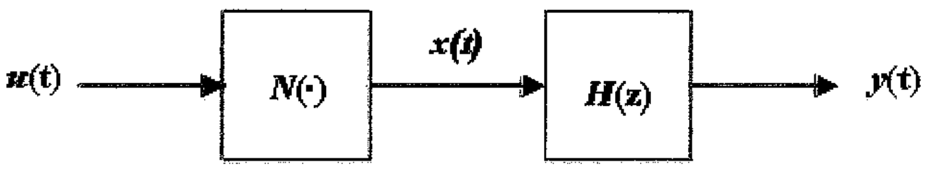
d(k)= vector of additive disturbance variables

y(k)= vector of process output

In general, a wiener model consists of a dynamic linear block in cascade with a static non-linearity at the output. For the static non linear element, Continuous Piecewise Linear (PWL) function y=f (v) is use in this paper **(Cervantes et al, 2003).** The PWL functions have proved to be a very powerful tool in the modeling and analysis of nonlinear systems.

2.3.4 Hammerstein Model

The Hammerstein model **(Ramesh et al, 2007)** consists of a nonlinear static element followed in series by a linear dynamic element. Hammerstein model has been considered as alternatives to linear models in a number of chemical process applications such as distillation column, CSTR, pH Process etc. The structure of the Hammerstein model is shown in figure below.



Where

u(t) = input of the nonlinear static block.

x(t) = output of the nonlinear static block.

Simultaneously;

x(t) = input of the linear dynamic block

y(t) = output of the linear dynamic block.

In **(Ramesh et al, 2007)** research study, they use a new wavenet based nonlinear function to describe the nonlinear static block and Output Error (OE) model is used to describe the linear dynamic block.

The linear block is the Output Error (OE) model, given by the following equation.

$$y(t) = \frac{B(q^{-})}{A(q^{-})}x(t)$$

$$B(q^{-1}) = b_1 q^{-nk} + b_2 q^{-(nk+1)} + \ldots + b_{nb} q^{-(nk+nb+1)}$$

$$A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2} + \ldots + a_{na} q^{-na}$$

where:

nb is the number of coefficients in *B*(*q*-*1*)

na is the number of coefficients in *A*(*q*-*1*)

nk is the delay from input to output

b_1 b_1 , …, b_n are the coefficients of polynomial *B*

a_1 ,a_1 , …, a_n are the coefficients of polynomial *A*

It also consist of Wavenet structure based nonlinear function $x = F(u)$ to represent the static nonlinearity of the Hammerstein model.

$$\begin{aligned} F(u) &= (u-r)PL + as_1 f(bs_1((u-r)Qcs_1)) \\ &+ \ldots + as_k f(bs_k((u-r)Qcs_k)) \\ &+ aw_1 g(bw_1((u-r)Q - cw_1)) \\ &+ \ldots + aw_k g(bw_k((u-r)Q - cw_k)) + d \end{aligned}$$

2.3.5 Volterra Model

One of the commonly used nonlinear dynamic empirical model structures is the Volterra model, and this work develops a systematic approach to the identification of third-order Volterra and Volterra-Laguerre models from process input-output data[10]. Volterra series analysis is an extension of small-signal analysis into the field of weakly nonlinear behavior **(Schetzen M, 1980)**.

The general form of the Volterra model is given as,

$$y(k) = h_0 + \sum_{i=1}^{N} \sum_{j_1=1}^{M} \cdots \sum_{j_N=1}^{M} h_i(j_1, \ldots, j_N) u(k-j_1) \cdots u(k-j_N)$$

Where;

N=model order

M = model memory, the duration over which the past inputs have an effect on the current output, y (k)

h_0 =the bias term,

The Volterra model kernels are given by h i (j_1 , ..., j_N), and the identification problem involves determining the values of these kernels. The Volterra model structure is capable of capturing a variety of nonlinear systems behavior. It has the ability to capture asymmetric output responses to symmetric changes in the input in many chemical engineering systems including reactors and distillation columns.

The Volterra model structure can be considered as a nonlinear extension of the FIR model thereby facilitating its use in on-line applications. One disadvantage of this structure is the number of parameters that must be estimated as the model order increases. This factor has limited the widespread use of higher-order Volterra models for practical applications.

In this work, **(Schetzen M, 1980)** shows that by judiciously exploiting the model structure, input sequences can be tailored so that simplify the task of and minimize the data requirements for identifying the parameters for a third-order Volterra model.

The third-order Volterra model is first decomposed in the following manner **(Soni and Parker, 2004)**,

$$\hat{y}(k) = h_0 + L(k) + D(k) + S(k) + O(k)$$

$$\begin{aligned}
L(k) &= \sum_{i=1}^{M} h_1(i)u(k-i) \\
D(k) &= \sum_{i=1}^{M} h_2(i,i)u^2(k-i) + \sum_{i=1}^{M} h_3(i,i,i)u^3(k-i) \\
S(k) &= 3\sum_{i=1}^{M}\sum_{j=1}^{i-1} h_3(i,i,j)u^2(k-i)u(k-j) \\
&\quad + 3\sum_{i=1}^{M}\sum_{j=1}^{i-1} h_3(i,j,j)u(k-i)u^2(k-j) \\
O(k) &= 2\sum_{i=1}^{M}\sum_{j=1}^{i-1} h_2(i,j)u(k-i)u(k-j) + \\
&\quad 6\sum_{i=1}^{M}\sum_{j=1}^{i-1}\sum_{\ell=1}^{j-1} h_3(i,j,\ell)u^2(k-i)u(k-j)u(k-\ell)
\end{aligned}$$

Where;

L=Linear

D= nonlinear diagonal

S= third-order sub-diagonal

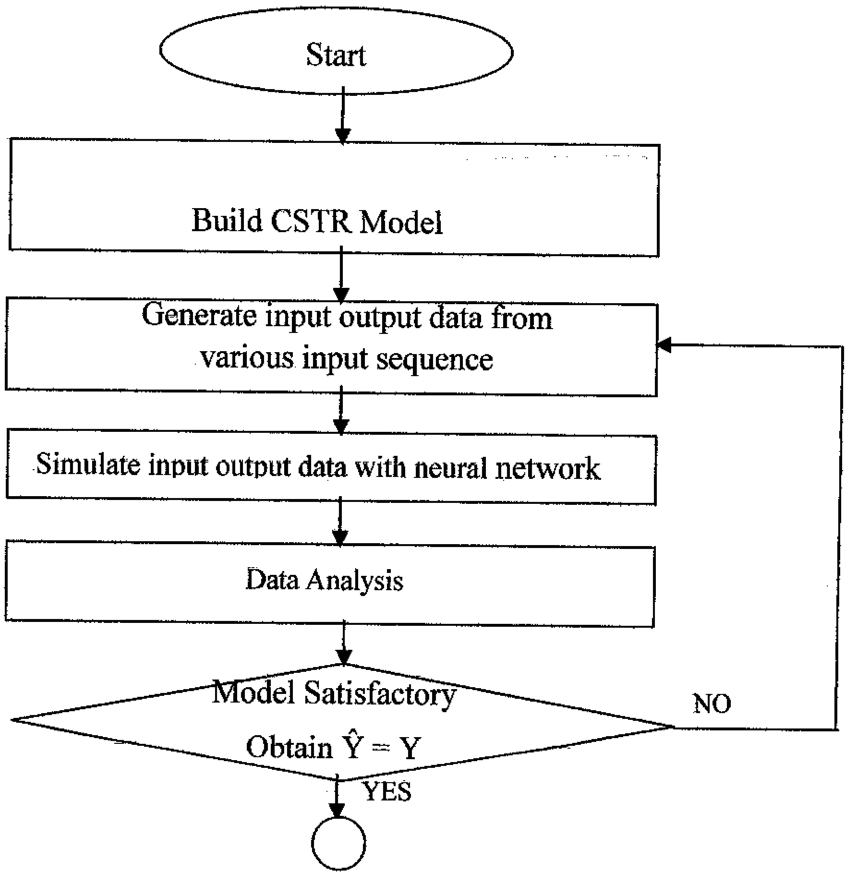
0=nonlinear off-diagonal term

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

Process Identification of non linear process can be conduct as follow;



Start
Build CSTR Model
Generate input output data from various input sequence
Simulate input output data with neural network
Data Analysis
Model Satisfactory Obtain Ŷ = Y
NO
YES

Figure 13: Schematic Project Work Flow

3.2 Development of CSTR Model

To build CSTR model, the author use Simulink based simulation (Figure 12). Simulink is a tool in MATLAB for modeling, simulating and analyzing multidomain dynamic systems, including linear and nonlinear dynamics system. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. The coding for CSTR model and its S-function is attached in APPENDIX.

Simulink based model is very useful and user friendly as the author just need to change the input without changing the overall programming source code of the model.

3.2.1 Continuous Stir Tank Reactor Model

For this research, simulation s done based on continuous stirred-tank reactor (CSTR) process model. Assuming that the process is irreversible, exothermic reaction, A→B, constant volume, and reactor cooled by a single coolant stream. This can be modeled by the following equations:

$$\dot{C}_A(t) = \frac{q(t)}{V}[C_{AO}(t) - C_A(t) - k_o C_A(t) exp\left[\frac{-E}{RT(t)}\right]]$$

$$\dot{T}(t) = \frac{q(t)}{V}(T_0(t)) - \frac{k_o \Delta H}{\rho C_p} C_A(t) exp\left(\frac{-E}{RT(t)}\right) + \frac{\rho_c C_{pc}}{\rho C_p V} q_c(t)\left[1 - exp\left(-\frac{hA}{q_c(t)\rho_c C_{pc}}\right)\right](T_{c0}(t) - T(t))$$

The objective is to control the output Temperature of the model, by manipulating the coolant temperature T_c **(Cervantes et al, 2002).**

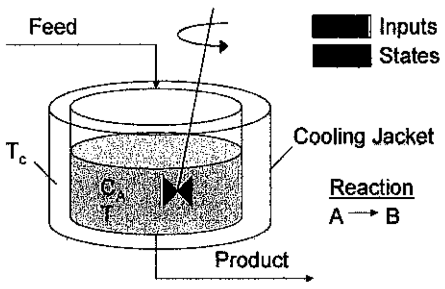


Figure 14: CSTR

Below are the nominal operating condition used in this simulation based on **Cervantes et al, 2003** research.

Nominal CSTR perameter values

Product Concentration	C_A	0.1 mol/l
Reactor Temperature	T	438.54 K
Coolant Flow Rate	Q_C	103.41 l/min
Process Flow Rate	Q	100 m^3/sec
Feed Concentration	C_{Af}	1mol/l
Feed Temperature	T_f	350 K
Inlet Coolant Temperature	T_{CO}	350 K
CSTR Volume	V	100 m^3
Heat Transer Term	HA	7 X 10^5 cal/min K
Reaction rate constant	K_o	7.2 x 10^{10} l/min
Activation Energy Term	E/R	8750 K
Heat of Reaction	ΔH	5 x 10^4 J/mol
Liquid Densities	ρ, ρ_c	1000 kg/m^3
Specific heats	C_p, C_{pc}	0.239 J/kg K

3.3 Input Sequence Generation

Input sequence is generated either from existing simulink library source block or from importing data from excel file or workspace. Input sequence such as PRBS and Gaussian is generated from signal builder while the source block for input such as ramp, sine wave, sawtooth and pulse already exists in the system. For input that is obtained from journals, they are imported from Microsoft Excel before being used in CSTR Simulink Based Model. Figure 15 shows some source block that had been used in modeling.

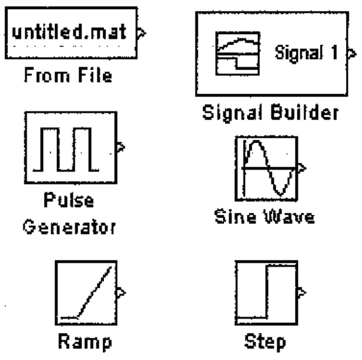


Figure 15: Simulink Library Source Block

3.4 Neural Network Prediction

Input and output from the validated model is taken for further development in Neural Network. This is to determine which input sequence will provide the best input-output data to get the best model in predicting output. Network optimization consists of determining the

- Network architecture
  - Compare Feed forward architecture and NARXSP architecture
- Number of neurons in each layer (input, hidden and output layer)
  - For this study, number of neurons in input layer is fixed to 2 while in output layer, the number of neuron is fixed to 1
  - The number of neurons in hidden layer is varied from 1 to 60 to evaluate the performance of the model prediction fitness
- The appropriate transfer function for each layer
  - 27 combination of transfer function configuration, consists of pure linear (p), log sigmoid(l) and tan sigmoid (t) is utilize in the NN to get the best model. Trial and error method is used in order to determine the optimal combination of the three elements above that gives the least error prediction.

Between those empirical models that mentioned earlier in literature review (Wiener, Hammerstein, Volterra, Gaussian and NN), Neural Network (NN) is selected for prediction of the behavior of process. NN is selected compare to others as it requires less information to run (input and output data only) and require less cost for implementation to the real system. Number of neurons in the hidden layer, the corresponding transfer function for each layer and the network types is varied for the optimization.

Project Gantt Chart:

Task	Date										
	JUL 08	AUG 08	SEP 08	OCT 08	NOV 08	DEC 08	JAN 09	FEB 09	MAR 09	APR 09	MAY 09
Topic Selection	■										
Problem Identification	▨										
Literature Review	■	■	■	■	■						
CSTR Model Development						■	■				
Input sequence generation							■	■			
Neural Network Optimization											
Submission and Presentation											■

CHAPTER 4

RESULT& DISCUSSION

4.1 CSTR Model Development

The author has done a simulation to build CSTR model based on **Cervantes et al, 2002.** Figure 16 shows the simulink based CSTR model.

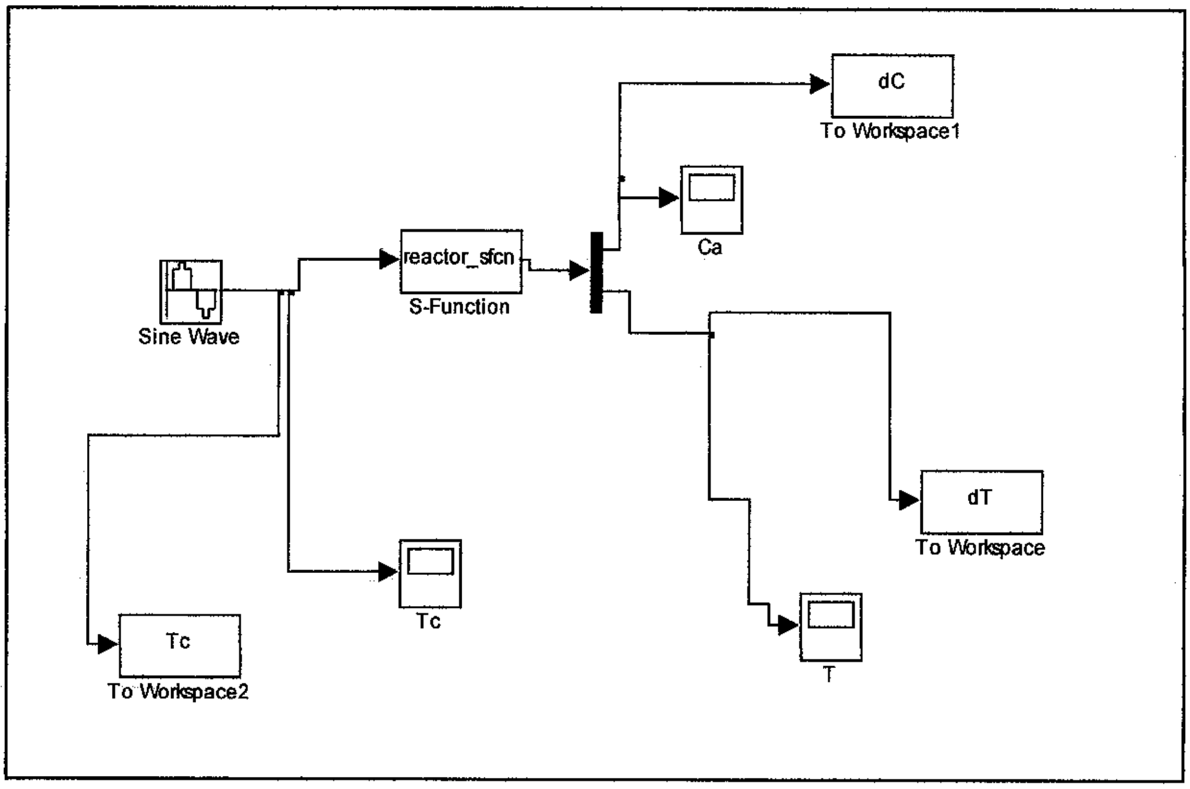


Figure 16: Simulink Based CSTR Model

This Simulink based CSTR model is used with different input sequence block to generate input-output data for further use in neural network model.

4.2 Input-Output Data Generation

Below are the results of running simulink based CSTR model on different input sequence. Generally, when the concentration profile increases, the temperature profile decreases. The following figures show the cooling jacket temperature profile and its corresponding process concentration and temperature profile.

1)Step

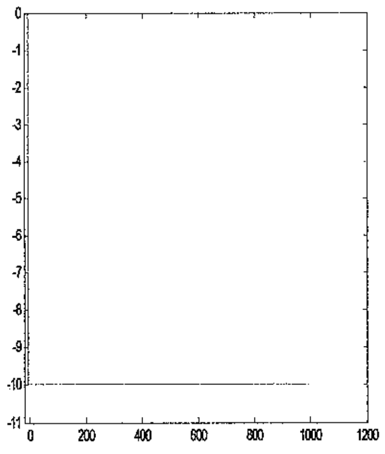
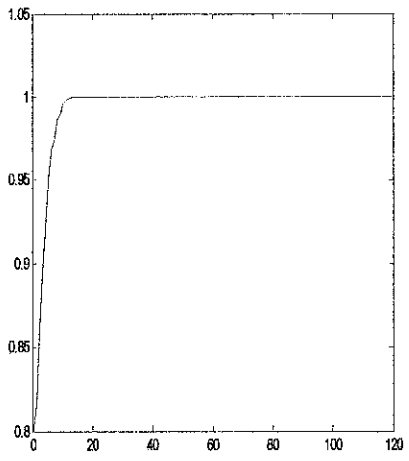
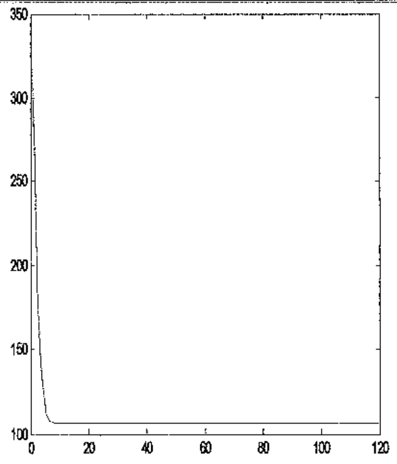
		

Figure 17: Cooling Jacket Temperature Profile	**Figure 18: Concentration Profile**	**Figure 19: Temperature Profile**

2) Ramp

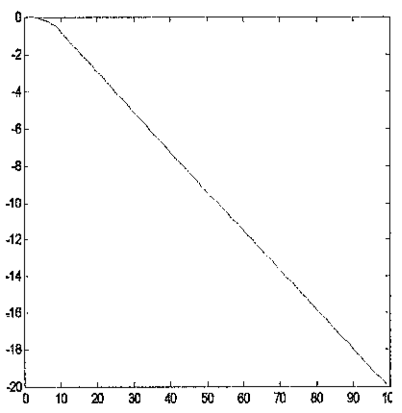
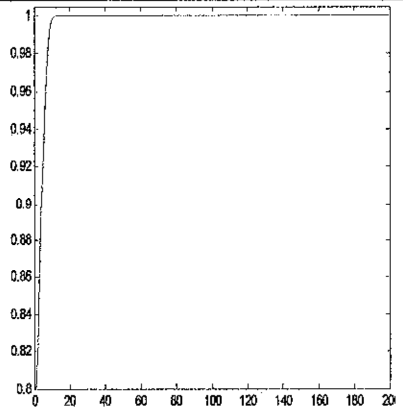
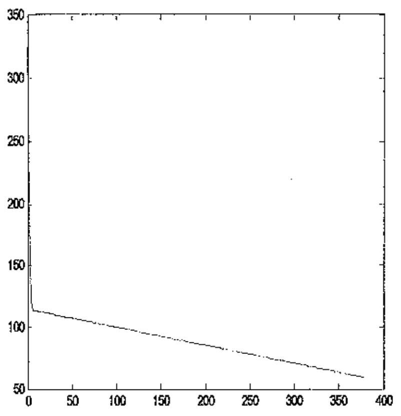
		

Figure 20: Cooling Jacket Temperature Profile	**Figure 21: Concentration Profile**	**Figure 22: Temperature Profile**

3) Pulse

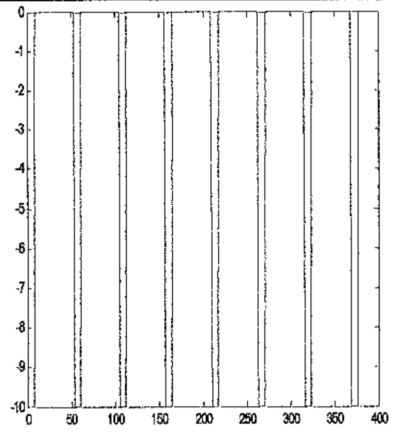
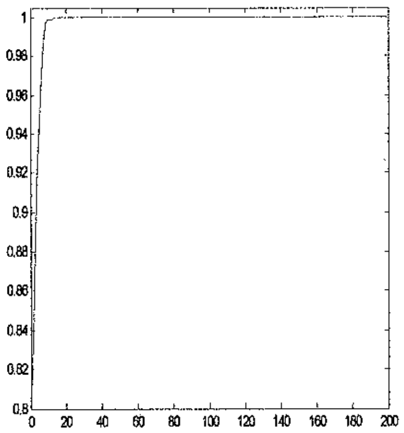
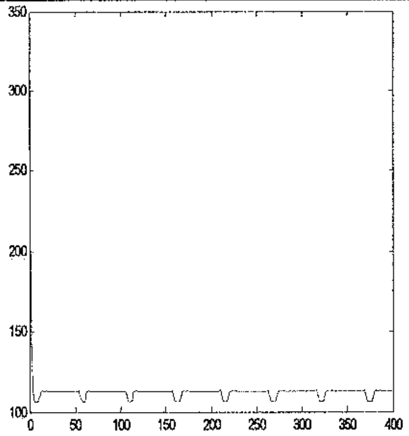
		

Figure 23: Cooling Jacket Temperature Profile	**Figure 24: Concentration Profile**	**Figure 25: Temperature Profile**

4) Sine wave

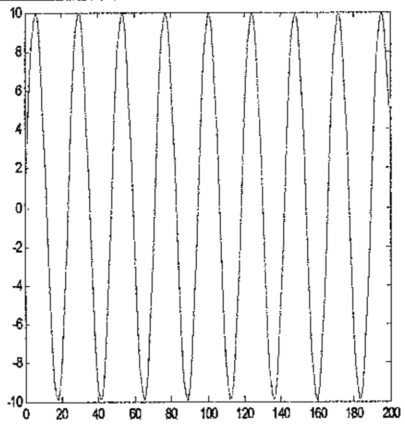
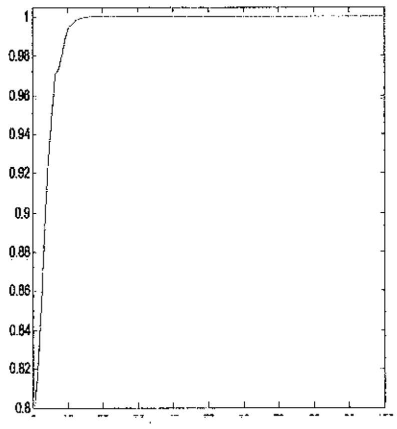
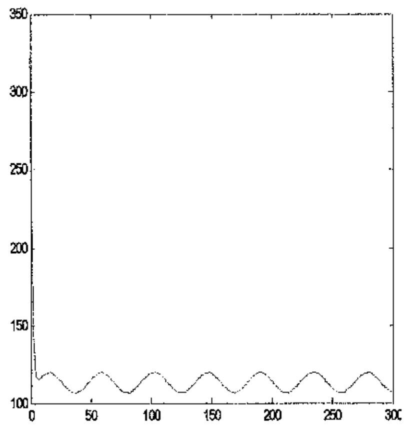
		

Figure 26: Cooling Jacket Temperature Profile	**Figure 27: Concentration Profile**	**Figure 28: Temperature Profile**

5) Gaussian Random Noise

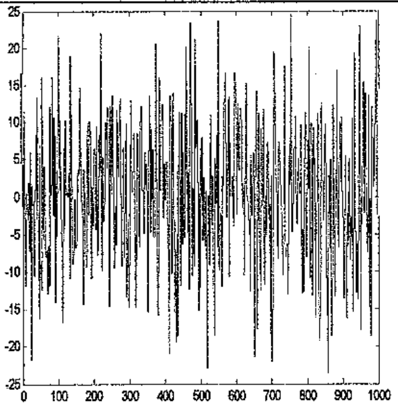
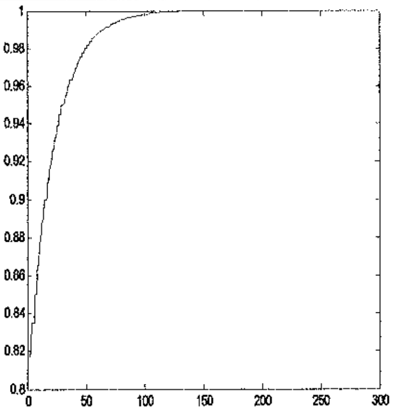
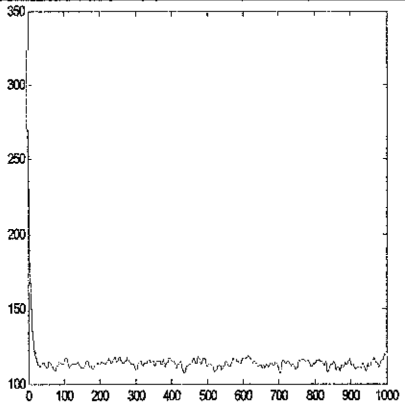
		

Figure 29: Cooling Jacket Temperature Profile	**Figure 30: Concentration Profile**	**Figure 31: Temperature Profile**

6) PRBS

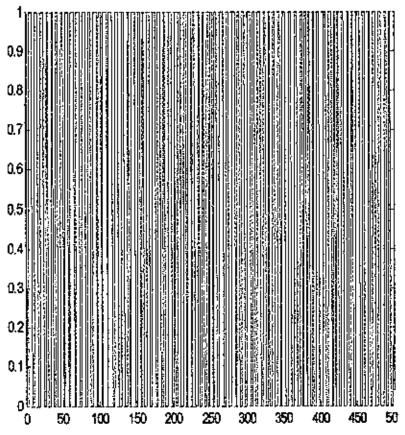
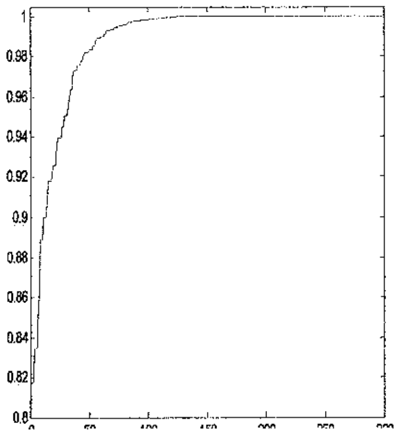
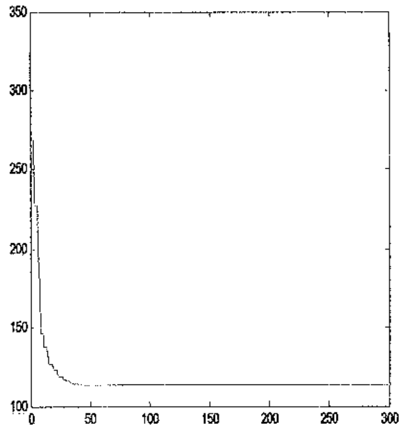
		

Figure 32: Cooling Jacket Temperature Profile	**Figure 33: Concentration Profile**	**Figure 34: Temperature Profile**

7) Repeating Sequence Stair

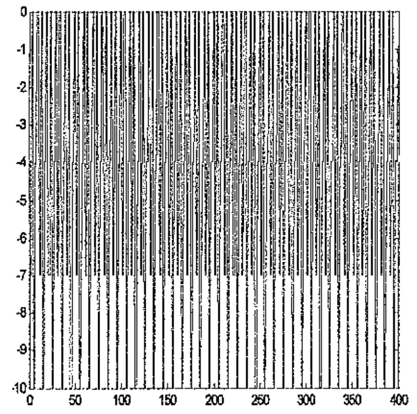


Figure 35: Cooling Jacket Temperature Profile

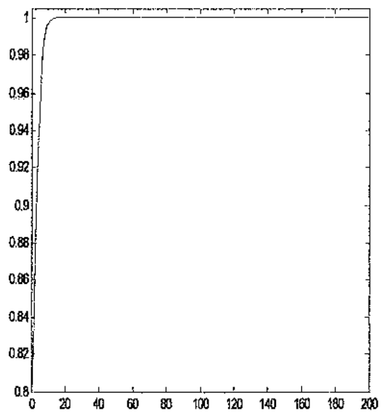


Figure 36: Concentration Profile

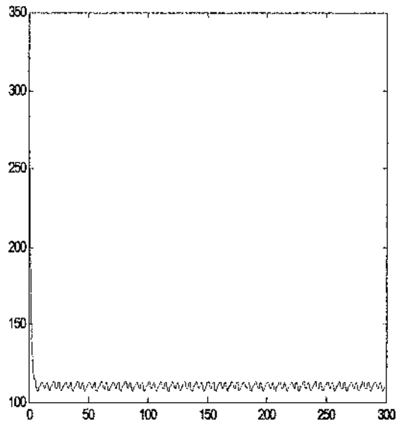


Figure 37: Temperature Profile

8) Sawtooth Wave

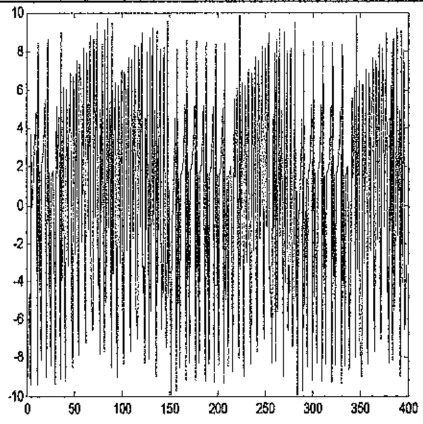


Figure 38: Cooling Jacket Temperature Profile

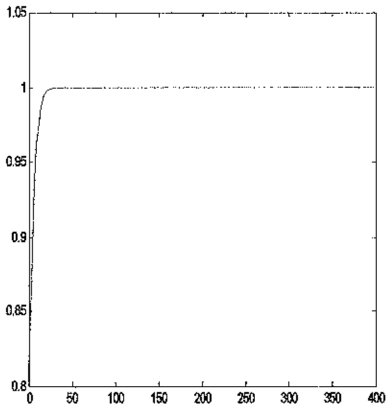


Figure 39: Concentration Profile

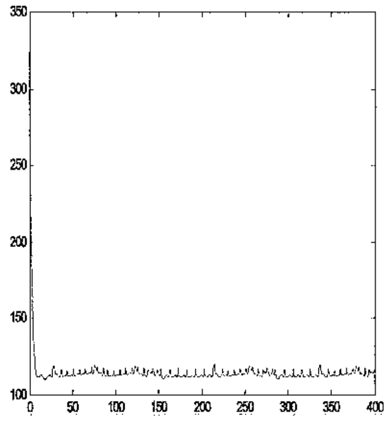


Figure 40: Temperature Profile

9) Baruch 1

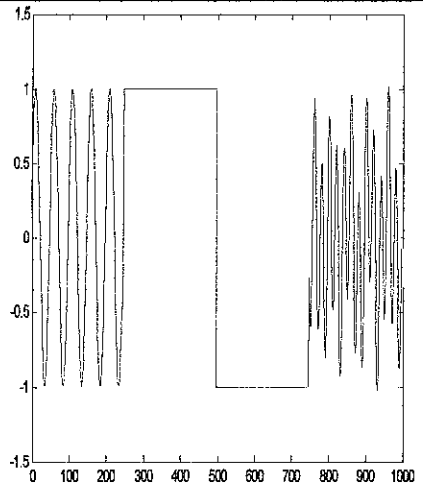
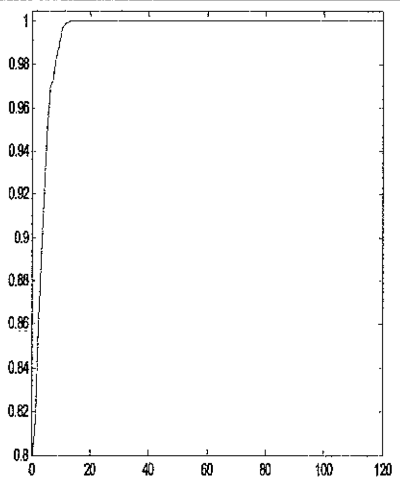
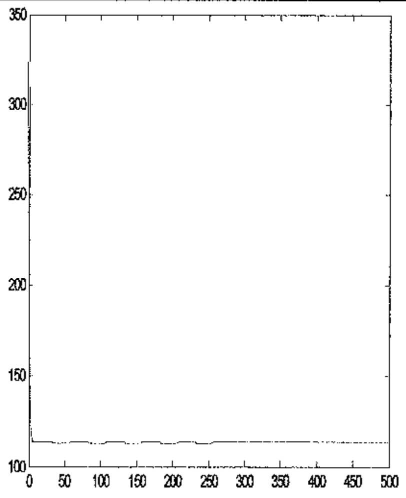
		

Figure 41: Cooling Jacket Temperature Profile	**Figure 42: Concentration Profile**	**Figure 43: Temperature Profile**

10) Baruch 2

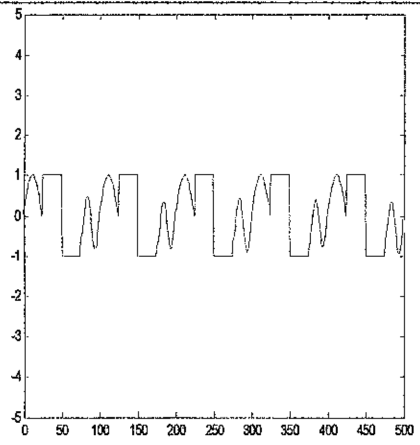
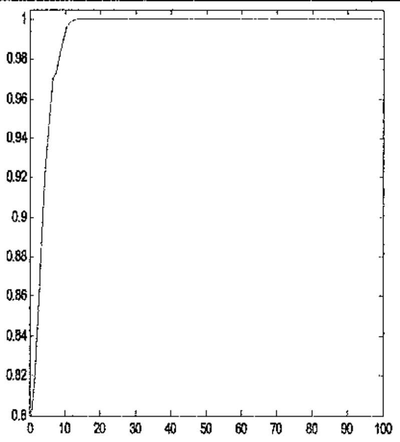
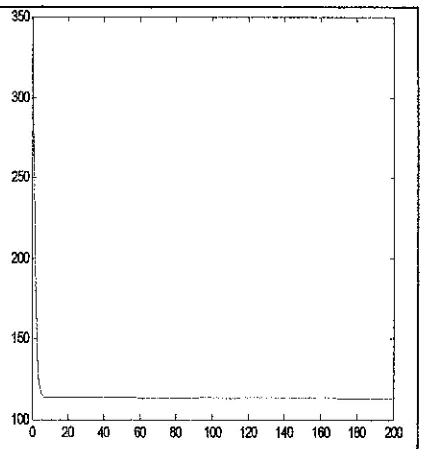
		

Figure 44: Cooling Jacket Temperature Profile	**Figure 45: Concentration Profile**	**Figure 46: Temperature Profile**

11) Liu

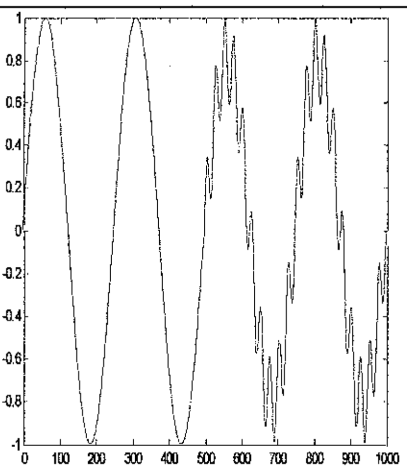
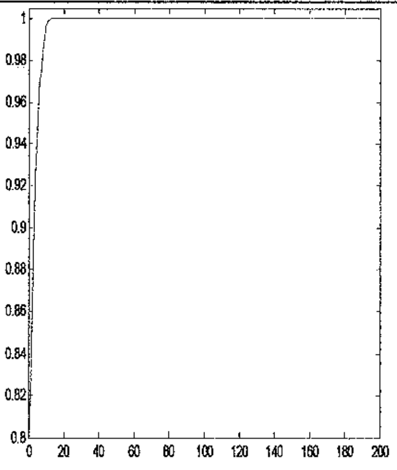
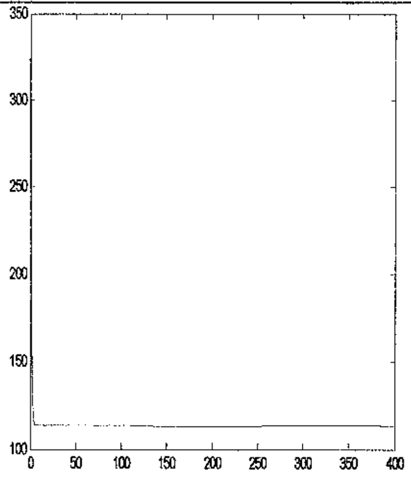
		

Figure 47: Cooling Jacket Temperature Profile	**Figure 48: Concentration Profile**	**Figure 49: Temperature Profile**

12) Parker

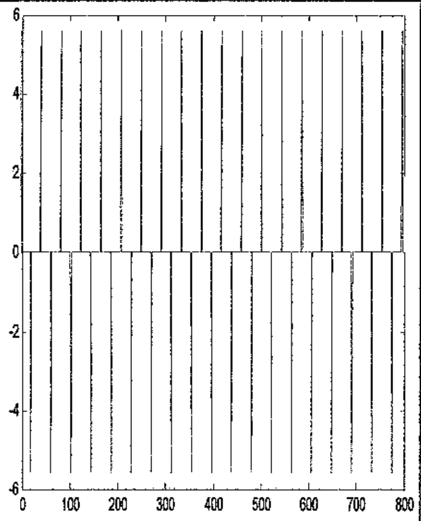
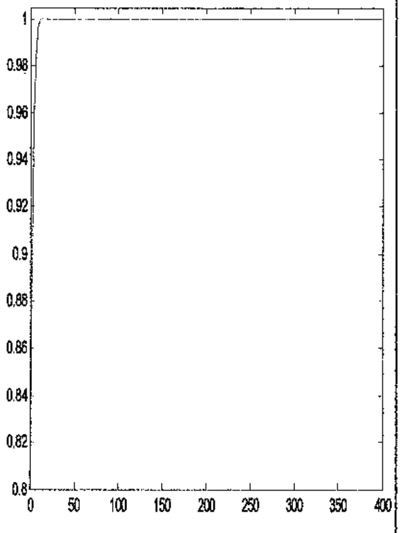
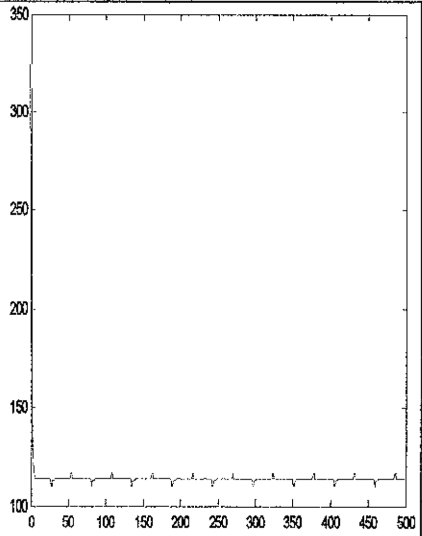
		

Figure 50: Cooling Jacket Temperature Profile	**Figure 51: Concentration Profile**	**Figure 52: Temperature Profile**

13) Soni 1

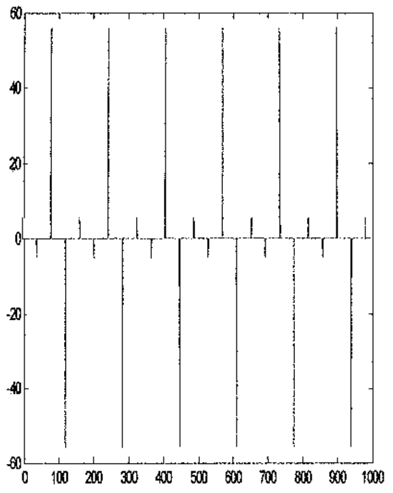
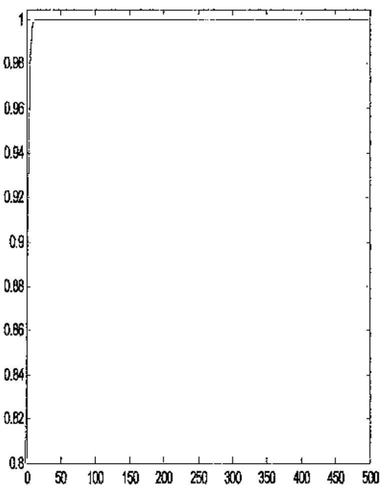
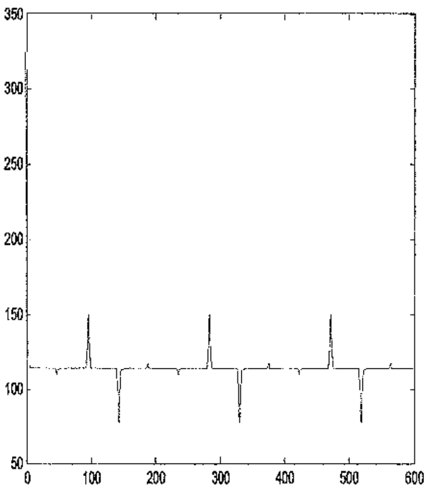
		

Figure 53: Cooling Jacket Temperature Profile	**Figure 54: Concentration Profile**	**Figure 55: Temperature Profile**

14) Soni 2

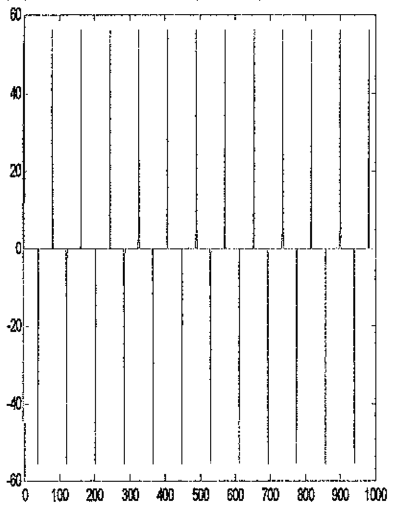
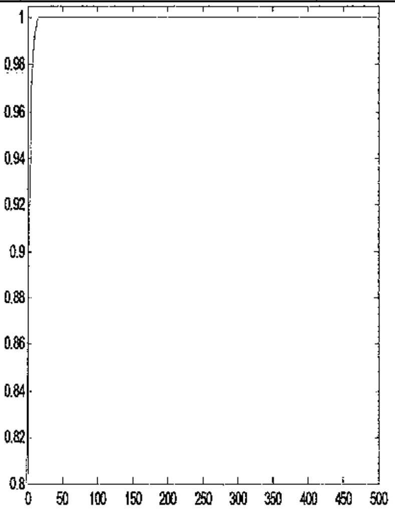
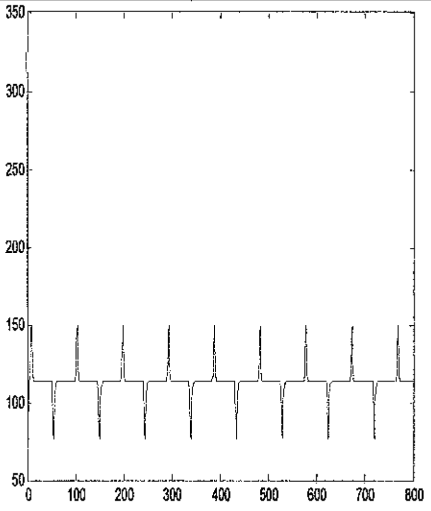
		

Figure 56: Cooling Jacket Temperature Profile	**Figure 57: Concentration Profile**	**Figure 58: Temperature Profile**

4.3 Neural Network Prediction

The two types of NN for modeling CSTR outlet temperature are applied to simulate data generated using the validated and proven CSTR model from the different input sequences. The thirteen cases of different input sequences are investigated.

Figures in Table 1 show the results for outlet CSTR temperature behavior for NARXSP model in comparison to the actual behavior of the CSTR model.

Neural Network is considered giving the best performance if:

- Has low root mean squared error (RMSE)
- Has high correct directional change (CDC)
- Follow well the behavior of the signal

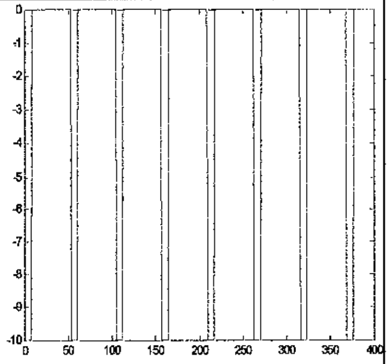
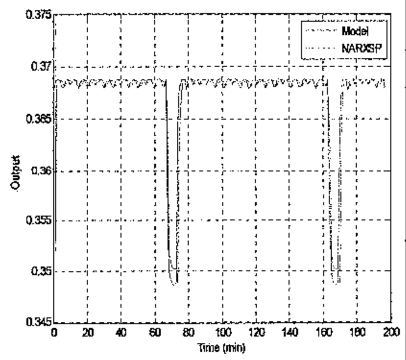
RMSE is the mean, standard deviation, and correlation coefficient that show whether the NN network architecture are highly correlated or not with the input-output data of CSTR model.

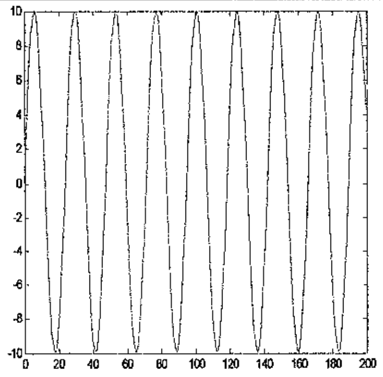
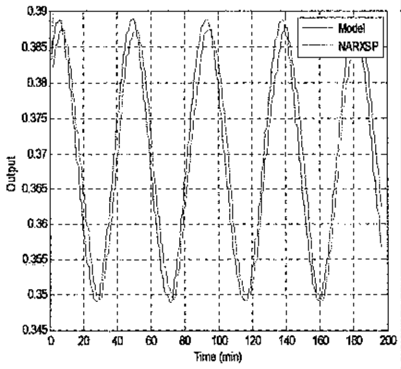
The CDC meanwhile is the correct direction of change in a variable that measure the capability of the model to follow the trend of data given.

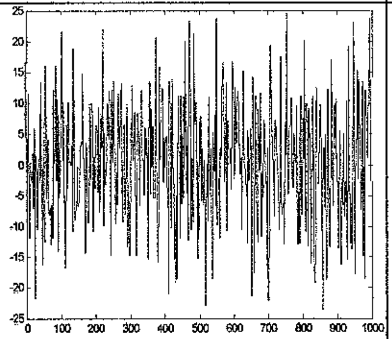
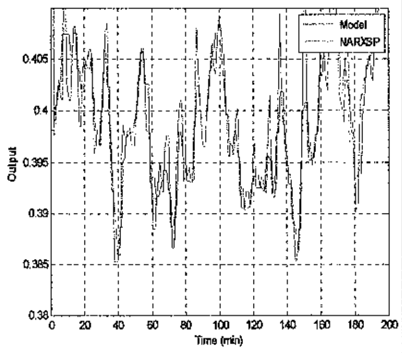
Table 1 shows the NARXSP neural network prediction depending on their input. Also, the best configuration of transfer function and its RMSE and CDC calculated is put into Table 1.

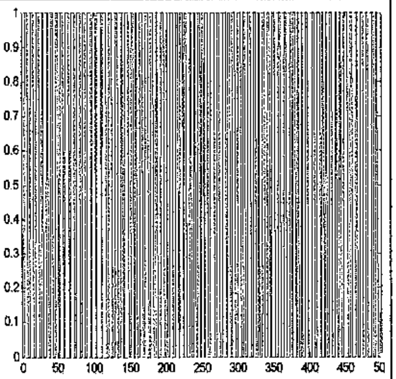
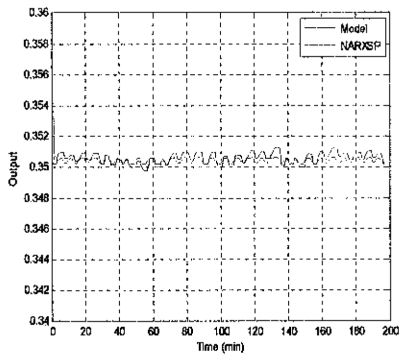
Table 1: NARXSP NN Prediction According to Inputs

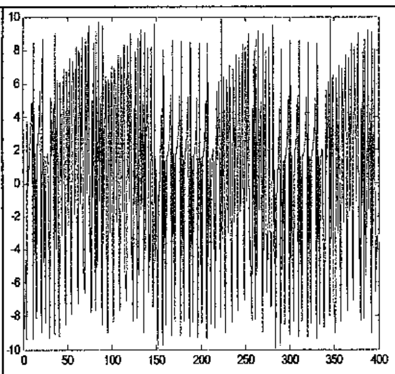
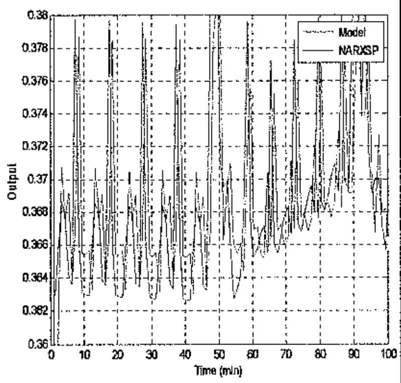
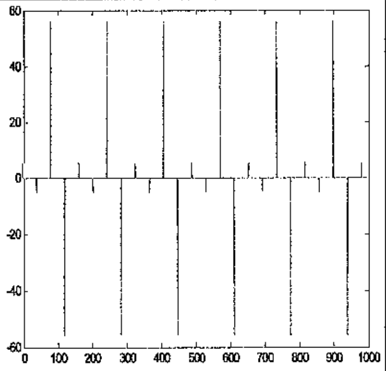
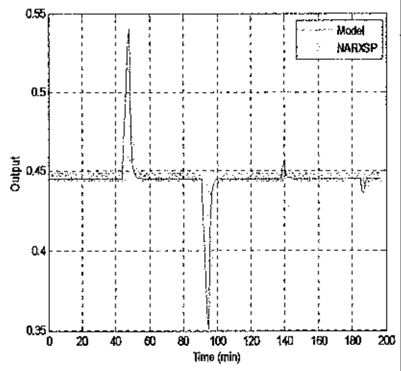
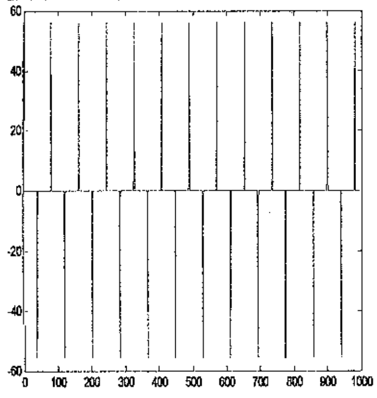
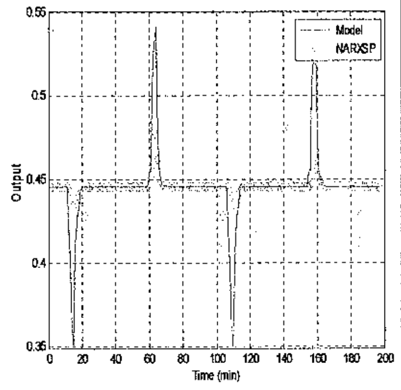
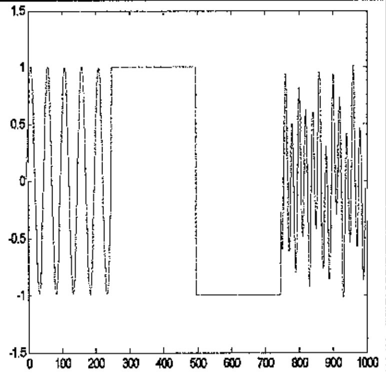
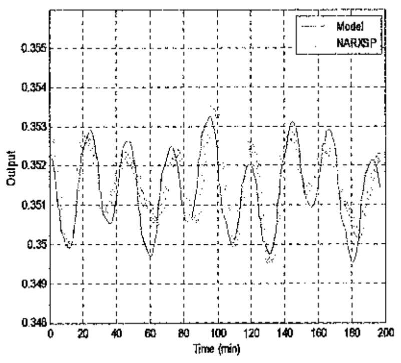
Type	INPUT	NN	Analysis

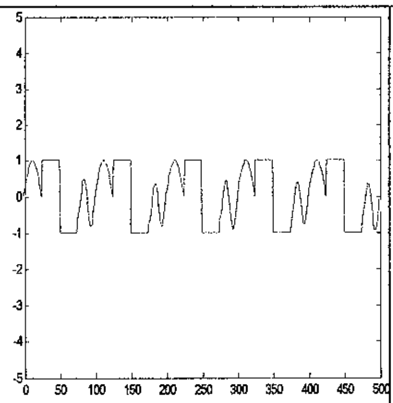
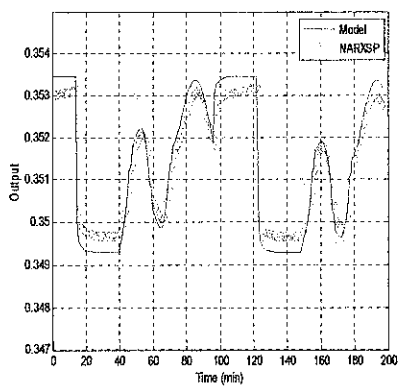
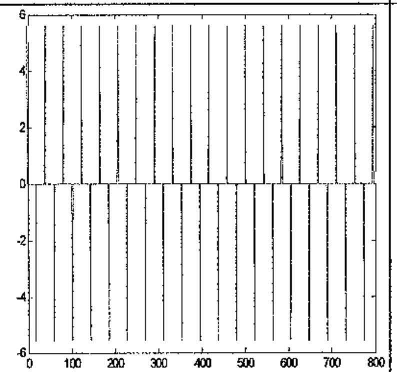
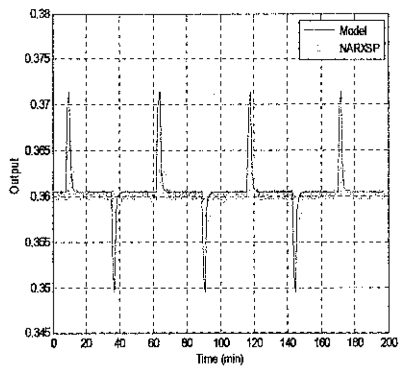
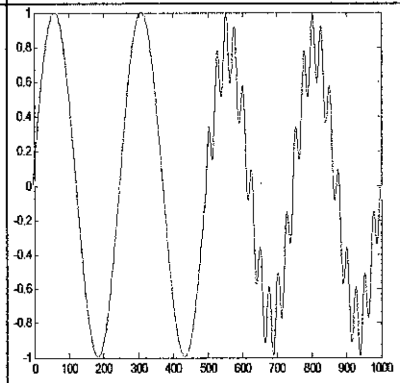
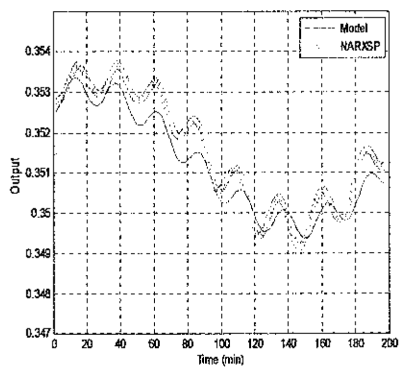
Pulse
TTT
RMSE = 0.0108
CDC = 54.315
			The optimal number of neuron for this type of input sequence is 2. Tansig, tansig, tansig is the best combination of its transfer function. It has low RMSE and average CDC. NN prediction follows well the behavior of pulse temperature output. However, there is slightly deviation during the early seconds.

Sine
PLP
RMSE = 0.0154
CDC = 89.340
			The optimal number of neuron for this type of input sequence is 35. Purelin, logsig, purelin is the best combination of its transfer function. It has low RMSE and high CDC. The NARXSP NN follows the data very well.

Gaussian
PLT
RMSE = 0.0285
CDC = 77.157
			The optimal number of neuron for this type of input sequence is 35. Purelin, logsig, tansig is the best combination of its transfer function. It has low RMSE and above average CDC. Slight deviation especially at the peak.

PRBS
PTP
RMSE = 6.2641 E-004
CDC = 86.294
			The optimal number of neuron for this type of input sequence is 2. Purelin, tansig, purelin is the best combination of its transfer function. It has very low RMSE and high CDC. However, the NARXSP NN can only merely match the temperature behavior of the process with a lot of deviation.

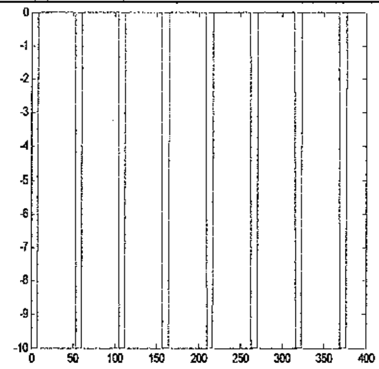
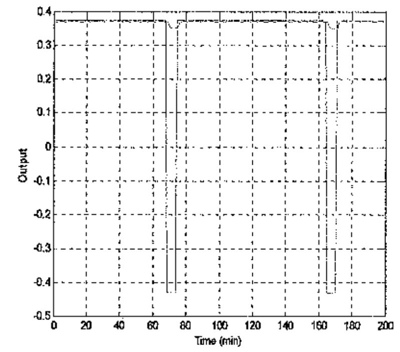
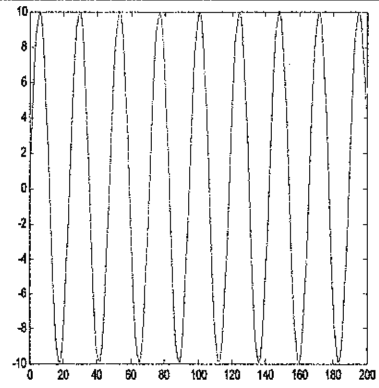
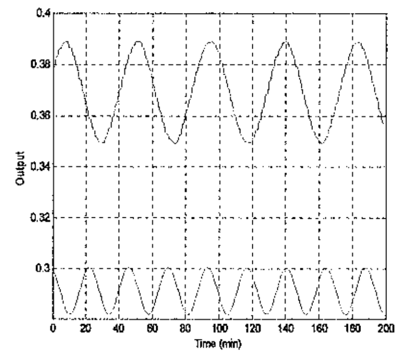
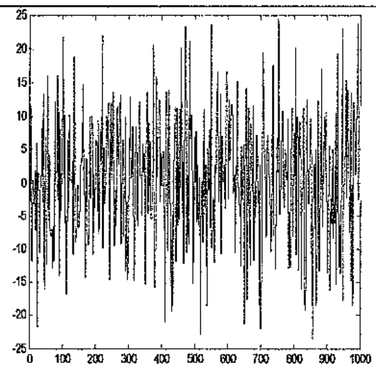
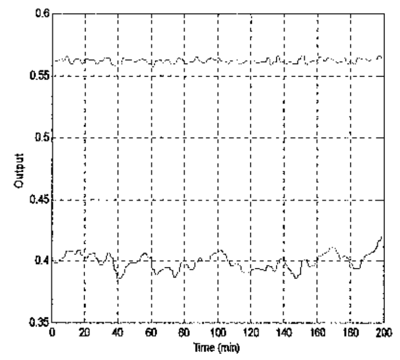
Sawtooth PTT RMSE = 0.0230 CDC = 58.883			The optimal number of neuron for this type of input sequence is 2. Purelin, tansig, tansig is the best combination of its transfer function. It has low RMSE and average CDC. Most of the time NARXSP NN cannot follow well the trend of data.
Soni1 TTT RMSE = 0.0282 CDC = 92.893			The optimal number of neuron for this type of input sequence is 25. Tansig, ansig, tansig is the best combination of its transfer function. It has low RMSE and very high CDC. However, the NARXSP NN cannot follow the behavior and unable to capture the sharp edges.
Soni2 PTP RMSE = 0.0245 CDC = 92.386			The optimal number of neuron for this type of input sequence is 6. Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and very high CDC. However, considerable deviation occurs at sudden peaks.
Baruch1 PTL RMSE = 0.0161 CDC = 80.203			The optimal number of neuron for this type of input sequence is 35. Purelin, tansig, logsig is the best combination of its transfer function. It has low RMSE and high CDC. The NARXSP can also follow well the behavior with only slight deviation

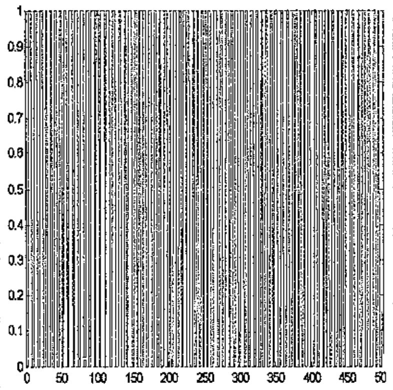
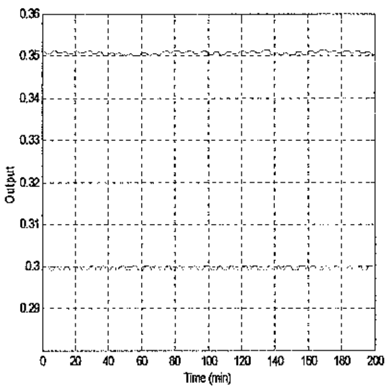
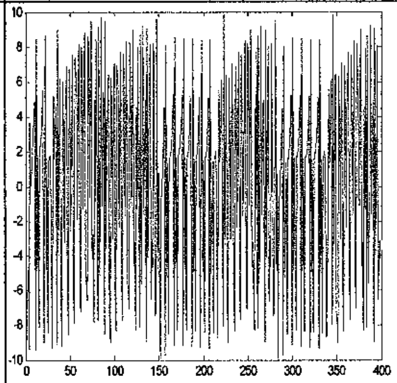
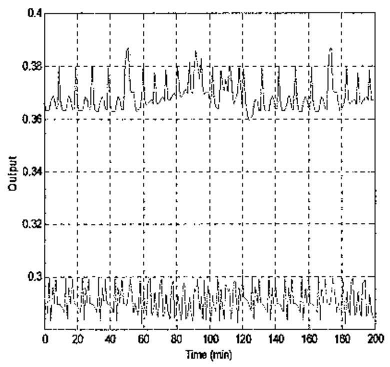
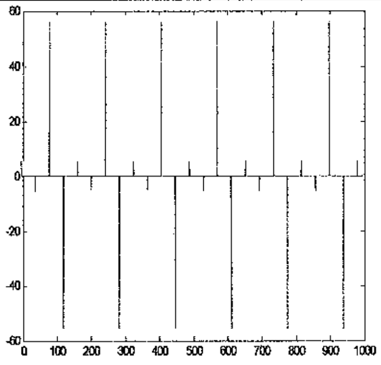
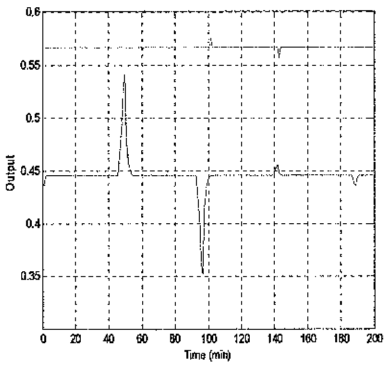
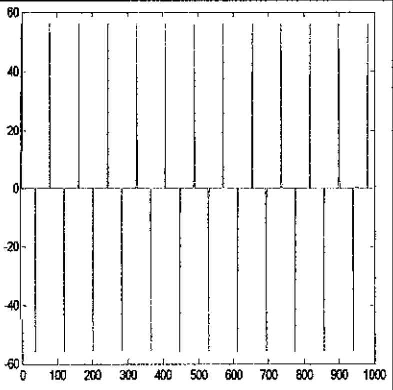
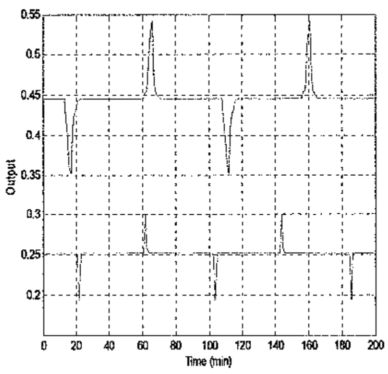
Baruch2 PTL RMSE = 0.0174 CDC = 91.371			The optimal number of neuron for this type of input sequence is 35. Purelin, tansig, logsig is the best combination of its transfer function. It has low RMSE and very high CDC. The NARXSP can merely follow the trend of the data driven model with this type of input sequence.
Parker PPL RMSE = 0.0136 CDC = 89.848			The optimal number of neuron for this type of input sequence is 50. Purelin, purelin,logsig is the best combination of its transfer function. It has low RMSE and high CDC. Normally, neural network with higher number of neuron will require longer simulation time.
Liu LPP RMSE = 0.0030 CDC = 80.203			The optimal number of neuron for this type of input sequence is 1. Logsig, Purelin, Purelin is the best combination of its transfer function. It has very low RMSE and high CDC.There is also a slight deviation when the signal changes direction but NARXSP NN considerably follow the behavior well.

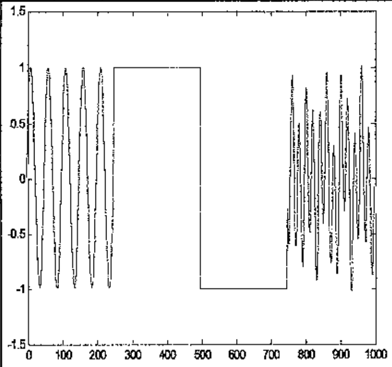
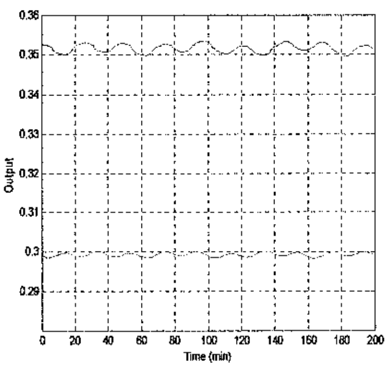
Comparable Correct Directional Change (CDC) values are obtained for all networks as indicated by the satisfactory directional change tracking. Almost all data present low RMSE value. The NARXSP NN follows well the process behavior with acceptional deviation.

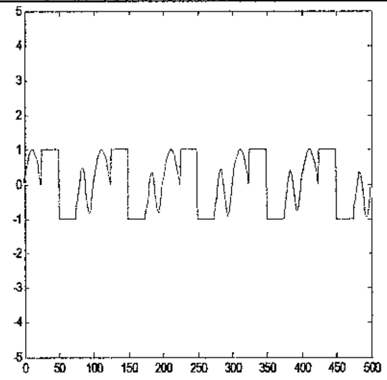
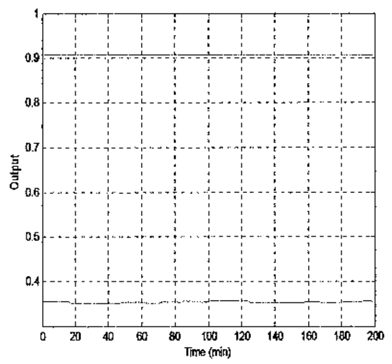
Comparison then made with Feed Forward (FF) Back Propagation(BP). Table 2 shows the Feed Forward neural network prediction depending on their input. Also, the best configuration of transfer function and its RMSE and CDC calculated is put into Table 2.

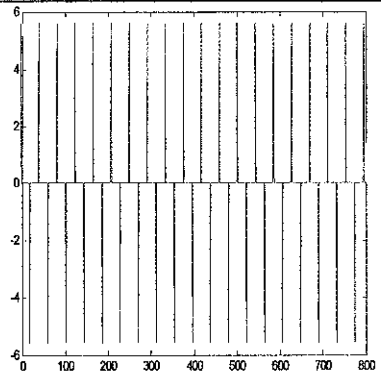
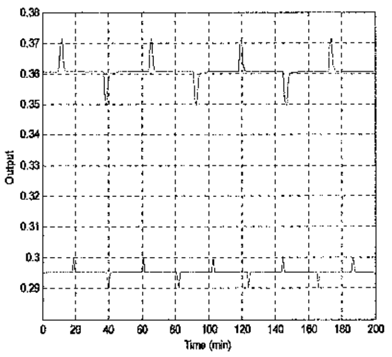
Table 2: Feed Forward BP Prediction According to Inputs

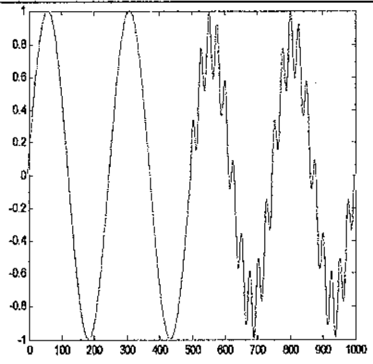
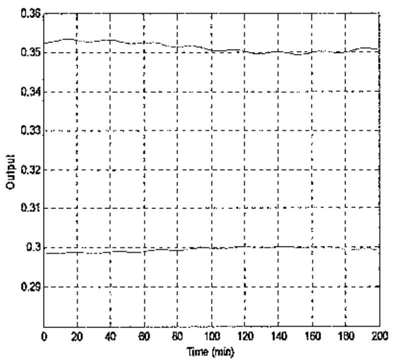
Type	INPUT	NN	Analysis
Pulse LLT rmse = 0.1923 CDC = 1.0050			The optimal number of neuron for this type of input sequence is 35. Logsig, logsig, tansig is the best combination of its transfer function. It has low RMSE and very low CDC. FF BP cannot follow the shape of process change. It was overshoot at some point.
Sine PTP rmse = 0.0805 CDC = 49.748 7			The optimal number of neuron for this type of input sequence is 2. Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and average CDC. The FF BP predict smaller sinusoidal pattern and very far from the real data plot.
Gaussian PPL Rmse= 0.1628 CDC = 55.276			The optimal number of neuron for this type of input sequence is 2. Purelin, purelin, logsig is the best combination of its transfer function. It has low RMSE and average CDC. Also, the FF BP prediction is very far from the real temperature. Moreover, the shape is not following well the shape of the CSTR temperature.

PRBS PTP rmse = 0.0510 CDC = 18.091			The optimal number of neuron for this type of input sequence is 8. Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and low CDC. The shape predicted not follows well the actual process and very far with CSTR model output.
Sawtooth PTP rmse = 0.0782 CDC = 65.829			The optimal number of neuron for this type of input sequence is 50. Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and above average CDC. FF BP predict quite well but it is very far from CSTR model output.
Soni1 PPL Rmse= 0.1217 CDC = 2.5126			The optimal number of neuron for this type of input sequence is 30. Purelin, logsig, purelin is the best combination of its transfer function. It has low RMSE and very low CDC. There is imprecise matching between the Feed Forward BP with the CSTR model output.
Soni2 PTP rmse = 0.1952 CDC = 3.5176			The optimal number of neuron for this type of input sequence is 35. Purelin,logsig,purelin is the best combination of its transfer function. It has low RMSE and very low CDC. There is inaccurate matching between the Feed Forward BP with the CSTR model output.

Baruch1
PTP
rmse = 0.0523
CDC = 48.241
			The optimal number of neuron for this type of input sequence is 20. Purelin,logsig,purelin is the best combination of its transfer function. It has low RMSE and average CDC. The FF BP prediction is not closely match the CSTR temperature. Large deviation occurs.

Baruch2
PTT
rmse = 0.0523
CDC = 21.608
			The optimal number of neuron for this type of input sequence is 35. Purelin, logsig, purelin is the best combination of its transfer function. It has low RMSE and high CDC. The same observation can be seen when Baruch 2 input is applied. The FF BP also cannot correctly match the CSTR model.

Parker
PTP
rmse = 0.0656
CDC = 4.5226
			The optimal number of neuron for this type of input sequence is 2. Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and low CDC.The FF BP slightly follow the temperature pattern but the difference are quite wide.

Liu
PTP
rmse = 0.0520
CDC = 69.849
			The optimal number of neuron for this type of input sequence is 35. Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and above average CDC. There is an apparent deviation at the early seconds when the signal changes direction.

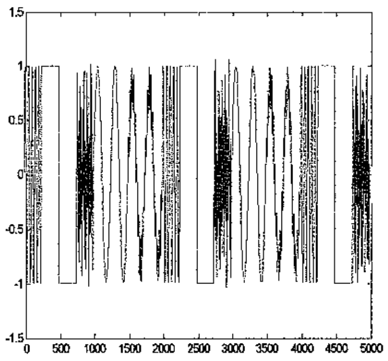
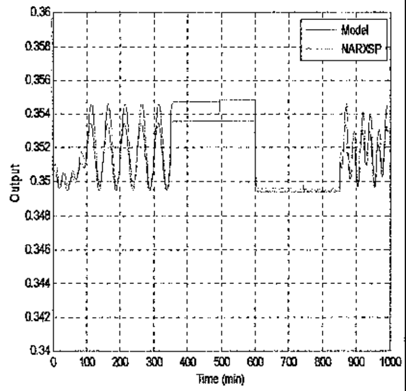
FeedNeural Network cannot predict the behavior of the output very well. Low correct directional change (CDC) value shows that Neural Network cannot trend the behavior of the signal when it is changing direction. Feed forward BP failed to track the temperature behavior dexterously when it is unable to follow the shape of the data driven CSTR model.

Results clearly indicate that good sequence of input, proper selection of the NN type, together with optimum configuration of the corresponding network architectures, can efficiently and accurately model the process temperature behavior. Only NARXSP NN with real output fed to the network feature tracks the process behavior as efficient and as accurate as the CSTR model. Generally, it is obvious that NARXSP-based stiction model gives the best performance compare to Feed Forward Neural Network. Narxsp Neural Network gives the best performance, due to low root mean squared error (RMSE) and it follow well the behavior of the signal. It is widely accepted that NARXSP structure always results in excellent performance since the actual output available during training is fed back to the network as part of the inputs for prediction **(Gomm et al, 1996)**.In most cases, test using high number of neurons gives long computational time compare to test using low number of neurons.

Consequently, a significant disadvantage of this mode of operation, termed the predictor mode, is the inability of the model to be used independently from the plant **(Zabiri and Mazuki, 2009).** An alternative as proposed by **(Gomm et al, 1996)** which is to use the trained NARXSP network in the parallel (feedback) architecture, where the predicted output from the network is being delayed and fed back along with the input to the network. This alternative mode of operation is called model mode.

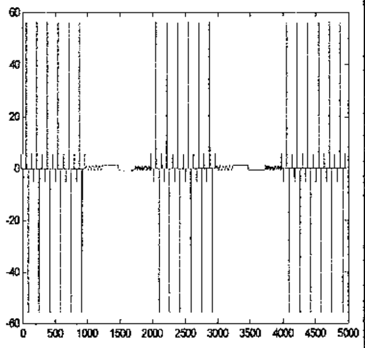
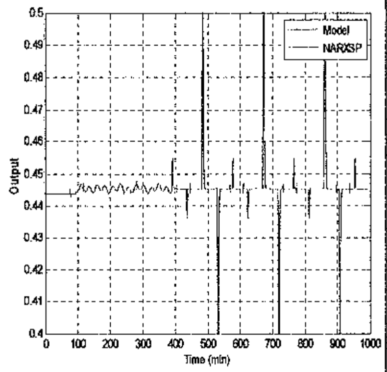
The best input sequence selected, which are Baruch1 input, Sine Wave input and Liu Input. As an initiative, from the best inputs, combinations are made between any two best inputs. Then, the effect of NARXSP NN prediction for the process behavior is observed.

Table 3: Baruch1 and Liu Combination Input Prediction

Type	INPUT	NN	Analysis
Baruch1 Liu PTT RMSE = 0.0174 CDC = 55.166			Purelin, tansig, tansig is the best combination of its transfer function. It has low RMSE and average CDC. There is an apparent deviation especially at the peak of sinusoidal pattern of behavior. Clear deviation at time of 350 sec to 600 sec reaction.

Subsequently, combination between one best input and one input with poor performance is made and further tested in NARXSP NN model. For this purposes, Baruch 1 and Soni 1 input is combined and produce new pattern of input sequences.

Table 4: Baruch1 and Soni 1 Combination Input Prediction

Type	INPUT	NN	Analysis
Soni1 Baruch1 TTP RMSE = 0.0100 CDC = 82.648			Purelin, tansig, purelin is the best combination of its transfer function. It has low RMSE and above average CDC. There is an apparent deviation at the early seconds when the signal changes direction.

From the graphs obtained combination of the best input (Baruch 1) and input with poor performance (Soni 1) give better performance compare to the combination of two best input (Baruch 1 and Liu). This shows that, it is not necessary that the combination of both good inputs will lead to a better prediction of model.

CHAPTER 5

CONCLUSION & RECOMMENDATION

5.1 Conclusion

The author is successful in building the CSTR model with various input sequence. The CSTR model created is able to generate input output data structure for NN purpose. The input-output data that generated from simulink-based model can be run in neural network to determine the best input sequence. In this study, a black box Neural Network-based modeling approach is proposed in modeling the temperature behavior of process. For different types of input sequences, the NARXSP-based model is practically good in predicting the actual process temperature profile. Numerical evaluations showed that with optimized model structures, NARXSP model is able to predict temperature behavior in CSTR to sufficient accuracy. It is also found that parallel (feedback) network trained using the series-parallel form (NARXSP) is able to provide multi-steps ahead prediction with sufficient accuracy. The best input sequences that can represent the best empirical model are Baruch 1, Sine and Liu inputs. The NN model ability in predicting the actual behavior of the process condition depends on its type. Generally, NARXSP neural network give the best prediction compare to feed forward neural network.

The project is successful within the time limit.

5.2 Recommendation

For area of improvement, study on more variety of input sequence from journals should be made to get a better input sequence. Besides that, research can be improve by using other types of neural network with different architecture such as Elman NN, Layered NN, NARX NN and Cascade NN, instead of using only NARXSP and Feed forward neural network. Additionally, for better analysis, study can alternatively be conduct with different types of empirical model such as Volterra Model, Gaussian Model, Hammerstein Model, and Wiener Model to replace the functions of Neural Network empirical model.

APPENDIX

Coding of MATLAB for CSTR Model:

```
function dx = reactor(t,x,Tj)
%model for reactor
% Concentration of A in CSTR (lbmol/ft^3)
Ca = x(1);
% Temperature in CSTR (F)
T = x(2);
% Parameters:
% Volumetric Flowrate (m^3/sec)
q = 100;
% Volume of CSTR (m^3)
V = 100;
% Density of A-B Mixture (kg/m^3)
rho = 1000;
% Heat capacity of A-B Mixture (J/kg-K)
Cp = .239;
% Heat of reaction for A->B (J/mol)
mdelH = 5e4;
% E - Activation energy in the Arrhenius Equation (J/mol)
% R - Universal Gas Constant = 8.31451 J/mol-K
EoverR = 8750;
% Pre-exponential factor (1/sec)
k0 = 7.2e10;
% U - Overall Heat Transfer Coefficient (W/m^2-K)
% A - Area - this value is specific for the U calculation (m^2)
UA = 5e4;
% Feed Concentration (mol/m^3)
Caf = 1;
% Feed Temperature (K)
Tf = 350;


% Compute x:
dCa= (q/V*(Caf - Ca) - k0*exp(-EoverR/T)*Ca);
dT = (q/V*(Tf - T) + mdelH/(rho*Cp)*k0*exp(-EoverR/T)*Ca + UA/V/rho/Cp*(Tj-
T));
dx=[dCa;dT];
```

Coding for S-Function in Simulink :

```
function [sys,x0,str,ts]=reactor_sfcn(t,x,u,flag,Cinit,Tinit)
switch flag
    case 0
%           [sys,x0,str,ts] = mdlInitializeSizes(Cinit,Tinit);    %initialize


        str=[];
        ts=[0 0];


        s=simsizes;
        s.NumContStates=2;
        s.NumDiscStates=0;
        s.NumOutputs=2;
        s.NumInputs=1;
        s.DirFeedthrough=0;
        s.NumSampleTimes=1;
sys=simsizes(s);
x0=[Cinit,Tinit];


    case 1 %derivatives
        Tj=u;
        sys=reactor(t,x,Tj);
    case 3 %output
        sys=x;
    case {2 4 9} %2:discrete,4:calcTimeHit,9:termination
        sys=[];


    otherwise error(['unhandled flag=',num2str(flag)]);
end
```

Sample of NARXSP Neural Network Coding:

```
clear;clc;
%  Extract data from M-file
A = xlsread('sinenormalize');

P_tr = A(1:500,1)';
T_tr = A(1:500,2)';
P_v = A(501:800,1)';
T_v = A(501:800,2)';
P_te = A(801:1000,1)';
T_te = A(801:1000,2)';

% converting vector to cell
P_tr = con2seq(P_tr); T_tr = con2seq(T_tr);
P_v = con2seq(P_v); T_v = con2seq(T_v);
P_te = con2seq(P_te); T_te = con2seq(T_te);

% create the training matrix
boy = 3;
pt = [P_tr(boy:end);T_tr(boy:end)]; tt = T_tr(boy:end);
ptv = [P_v(boy:end);T_v(boy:end)]; ttv = T_v(boy:end);
pte = [P_te(boy:end);T_te(boy:end)]; tte = T_te(boy:end);

de = 2;
d1 = [1:de];
d2 = [1:de];
% naming TF
p = 'purelin'; t = 'tansig'; l = 'logsig';
narx_net = newnarxsp(minmax(pt),d1,d2,[2 2 1],{t,t,p});
narx_net.trainFcn = 'trainrp';
narx_net.trainParam.show = 10;
narx_net.trainParam.epochs = 600;

for k=1:de,
  Pi{1,k}=P_tr{k};
end
for k=1:de,
  Pi{2,k}=T_tr{k};
end

val.P=ptv; val.T=ttv;
test.P=pte; test.T=tte;
% [net tr] = train(net,P_tr,T_tr,[],[],val,test);
narx_net = train(narx_net,pt,tt,Pi,[],val,test);

a = sim(narx_net,pte,Pi);
a = cell2mat(a);
tte = cell2mat(tte);

% Actual min max of the data set
T_temax = 1; T_temin = 0.0000;

T_te = tte;
```

```
 % Unnormalized data set
[row1,col1] = size(T_te);
unnorm_Tte = zeros(1,1:col1);
for j = 1:col1;
    unnorm_Tte(j) = T_te(j)*(T_temax-T_temin)+T_temin;
    j = j+1;
end
unnorm_a = zeros(1,1:col1);
for jj = 1:col1;
    unnorm_a(jj) = a(jj)*(T_temax-T_temin)+T_temin;
    jj = jj+1;
end

figure(4)
time = 1:length(T_te);
plot(time,unnorm_Tte,'-', time,unnorm_a,'-r'),...
xlabel('Time (min)'), ylabel('Output'),...
axis([0 1000 0.4 0.5]),...
legend('Model','NARXSP')
grid on;

% rmse calculation
[row1,col1] = size(T_te);
error_col = zeros(1,1:col1);
for i = 1:col1;
    error_col(i) = (unnorm_a(i) - unnorm_Tte(i))^2;
    i = i+1;
end
sum_error = sum(error_col);
rmse = sqrt(sum_error/col1)

%%% CDC calculation
d1=zeros(1,col1-1);
ii=2;
for iii=1:col1-1
    ai=unnorm_Tte(:,ii) - unnorm_Tte(:,ii-1);
    bi=unnorm_a(:,ii) - unnorm_a(:,ii-1);
    ci=ai*bi;
    d1(:,ii-1)=ci;
    ii=ii+1;
    iii=iii+1;
end
Dt1=zeros(1,col1-1);
jjj=1;
for jjjj=1:col1-1
    if d1(:,jjj)>0
        Dt1(:,jjj)=1;
    else
        Dt1(:,jjj)=0;
    end
    jjj=jjj+1;
    jjjj=jjjj+1;
end

[row2,col2] = size(Dt1);
CDC = (sum(Dt1))*(100/(col2))
```

Sample of Feed Forward Neural Network Coding:

```
clear
%  Extract data from M-file
A = xlsread('sinenormalize');

% Determine size of XY matrix
[row,col] = size(A) ;

%  Allocating input and target columns for T, V, and TS

P_tr = A(1:500,1)';
T_tr = A(1:500,2)';
P_v = A(501:800,1)';
T_v = A(501:800,2)';
P_te = A(801:1000,1)';
T_te = A(801:1000,2)';
P_te = B(1:1000,1)';
T_te = B(1:1000,2)';

% naming TF
p = 'purelin'; t = 'tansig'; l = 'logsig';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Setup network %%%%%%%%%%%%%%%%%%%%%%%%%%
% net=newcf(minmax(P_tr), [37 20
1],{'logsig','logsig','logsig'},'trainrp','learngdm','mse');
net=newff(minmax(P_tr), [2 2 1],{p,t,p},'trainrp','learngdm','mse');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Setup network %%%%%%%%%%%%%%%%%%%%%%%%%%

%
net.trainParam.show=100;
net.trainParam.epochs=500;
net.trainparam.goal=1e-4;

% Train network with early stopping
rand('seed',419877);
net = init(net);

%% Set up the validation and testing sets in a structure form
val.P=P_v; val.T=T_v;
test.P=P_te; test.T=T_te;
% [net tr] = train(net,P_tr,T_tr,[],[],val,test);
%[net tr] = train(net,P_tr,T_tr,[],[],[],[]);

% Simulate network
a = sim(net, P_te);


% figure(1)
% [slope,intercept,R] = postreg(a,T_te);

% Actual min max of the data set
T_temax = 1; T_temin = 0;
```

```
% Unnormalized data set
[row1,col1] = size(T_te);
unnorm_Tte = zeros(1,1:col1);
for j = 1:col1;
    unnorm_Tte(j) = T_te(j)*(T_temax-T_temin)+T_temin;
    j = j+1;
end
unnorm_a = zeros(1,1:col1);
for jj = 1:col1;
    unnorm_a(jj) = a(jj)*(T_temax-T_temin)+T_temin;
    jj = jj+1;
end

figure(2)
time = 1:length(T_te);
plot(time,unnorm_Tte,'-', time,unnorm_a,'-r'),...
xlabel('Time (min)'), ylabel('Output'),...
axis([0 1000 -0.3 2.5]),...
legend('model','Feed forward')
grid on;

% rmse calculation
[row1,col1] = size(T_te);
error_col = zeros(1,1:col1);
for i = 1:col1;
    error_col(i) = (unnorm_a(i) - unnorm_Tte(i))^2;
    i = i+1;
end
sum_error = sum(error_col);
rmse = sqrt(sum_error/col1)

%%% CDC calculation
d1=zeros(1,col1-1);
ii=2;
for iii=1:col1-1
    ai=unnorm_Tte(:,ii) - unnorm_Tte(:,ii-1);
    bi=unnorm_a(:,ii) - unnorm_a(:,ii-1);
    ci=ai*bi;
    d1(:,ii-1)=ci;
    ii=ii+1;
    iii=iii+1;
end
Dt1=zeros(1,col1-1);
jjj=1;
for jjjj=1:col1-1
    if d1(:,jjj)>0
        Dt1(:,jjj)=1;
    else
        Dt1(:,jjj)=0;
    end
    jjj=jjj+1;
    jjjj=jjjj+1;
end

[row2,col2] = size(Dt1);
CDC = (sum(Dt1))*(100/(col2))
```

REFERENCES

S.P. Asprey, Imperial College, London, U.K. S. Macchietto, Imperial College, London, U.K. , Dynamic Model Development: Methods, Theory and Applications

K. Ramesh, N. Aziz and S.R. Abd Shukor, Nonlinear Identification of Wavenet Based Hammerstein Model – Case Study on High Purity Distillation Column, School of Chemical Engineering, Engineering Campus, Universiti Sains Malaysia, 2007

M Ramasamy, Chemical Process Dynamic Control lecture note, Universiti Teknologi Petronas, 2007

E. M. Heaven, T. M. Kean, I. M. Johnsson, M. A. Manness, K. M. Vu and R.N. Vyse, Application of System Identification to Paper Machine Model Development and Controller Design, Second IEEE Conference and Control Applications, Sept 13-16, 1993

K R Godfrey, Introduction to Binary Signal Used In System Identification, University of Warwick UK, 1992

Michael Nikolaou and Pratik Misra, Linear Control of Nonlinear Processes: Recent Developments and Future Directions

Katalin M. Hangos, I. T. Cameron, Process Modelling and Model Analysis, Academic Press, 2001

Juˇs Kocijan1,2 and Roderick Murray-Smith3,4, "Nonlinear Predictive Control with a Gaussian,Process Model"

R. Murray-Smith, R. Shorten (Eds.): Switching and Learning, LNCS 3355, pp. 185–200, 2005. _c Springer-Verlag Berlin Heidelberg 2005

Ania Lusso´ n Cervantes, Osvaldo E. Agamennoni1, Jose´ L. Figueroa "Use of Wiener Nonlinear MPC to Control CSTR with Multiple Steady State", 2003

Schetzen, M., The Volterra and Wiener theories of nonlinear systems, Wiley, New York, 1980.

Ania Lusso´ n Cervantes, Osvaldo E. Agamennoni1, Jose´ L. Figueroa "A Nonlinear Model Predictive Control System based on Wiener Piecewise Linear Models", 2002

M. W. Braun, D. E. Rivera A. Stenman, W. Foslien, and C. Hrenya. Proceedings of the American Control Conference San Diego, California - June 1999", "Multi-level Pseudo-Random Signal Design and 'Model-on-Demand' Estimation Applied to Nonlinear Identification of a RTP Wafer Reactor"

Tulleken, H. J. A. , "Generalized Binary Noise Test-signal Concept for Improved Identification Experiment Design. Automatica", (1990).

Jonas Martensson, Henrik Jansson, Hakan Hjalmarsson, "Input Design for Identification of Zeros", Department of Signals, Sensors and Systems, KTH SE-100 44 Stockholm, Sweden

J.-P Costa, E. Thierry and T. Pitarque, "A Restricted Class of Input Sequence For Volterra Filter Identification", 1999

Abhishek *S.* Soni and Robert *S.* Parker "Control Relevant Identification for Third-Order Volterra Systems: A Polymerization Case Study", Proceeding of the 2004 American Control Conference Boston, Massachusetts June 30 -July 2, 2004

Seborg, D.E., Edgar, T.F. and Mellichamp, D.A.,Process Dynamics and Control, 2nd Ed., John Wiley, 2004.

Trott, M. The Mathematica GuideBook for Programming. New York: Springer-Verlag, 2004.

T.H Tsai, J.W. Lane, C.S. Lin, Modern Control Techniques for the Processing Industries, Marcel Dekker, Inc., 1986.

Robert S. Parker, Douglas Heemstra, Francis J Doyle III, Ronald K Pearson, Babatunde A Ogunnaike, "The Identification of Nonlinear Models for Process Control Using Tailored Plant Friendly Input Sequences", 2001.

I. Baruch, F. Thomas, R. Garrido, and E. Gortcheva, "A Hybrid Multimodel Neural Network for Nonlinear Systems Identification", 1996.

G.P Liu, V. Kadirkamanathan, S.A. Billings, "On-line Identification of Nonlinear Systems Using Volterra Polynomial Basis Function Neural Networks", 1997

Hagan M.T., Demuth H.B., Beale M.H., Neural Network Design, PWS Publishing Company, Boston, MA, 1996

H. Zabiri and N. Mazuki, "A Black-Box Approach in Modeling Valve Stiction", International Journal of Natural Sciences and Engineering, 2009

Gomm, J. B., D. Williams, J. T. Evans, "Development of a Neural-Network Predictive Controller", Liverpool John Moores, University,UK., 1996.