ARTIFICIAL INTELLIGENCE BASED METHODOLOGIES FOR MODELING, OPTIMIZATION AND MONITORING OF CHEMICAL PROCESSES

THESIS SUBMITED TO THE

UNIVERSITY OF PUNE

FOR THE DEGREE OF

DOCTOR OF PHILOLOPHY

IN

CHEMICAL ENGINEERING

BY

Somnath Nandi

UNDER THE GUIDANE OF

Dr. B. D. Kulkarni

and

Dr. S. S. Tambe

Chemical Engineering and Process Development Division

National Chemical Laboratory

Dr. Homi Bhabha Road

Pune – 411 008

INDIA

March 2010

CERTIFICATE

This is to certify that the work incorporated in the thesis titled "**Artificial Intelligence Based Methodologies for Modeling, Optimization and Monitoring of Chemical Processes**" submitted by Mr. Somnath Nandi, for the Degree of Doctor of Philosophy, in Chemical Engineering at University of Pune, was carried out by him under our supervision at Chemical Engineering and Process Development Division, National Chemical Laboratory, Pune – 411 008, India. Such material as has been obtained from other sources has been duly acknowledged in the thesis.

Dr. S. S. Tambe **Dr. B. D. Kulkarni**

(Research Co-Guide) **(Research Guide)**

DECLARATION

I hereby declare that the thesis titled "Artificial Intelligence Based Methodologies for Modeling, Optimization and Monitoring of Chemical Processes" submitted by me for the Degree of Doctor of Philosophy is the record of work carried out by me at Chemical Engineering and Process Development Division of National Chemical Laboratory, Pune, India, under the guidance of Dr. B. D. Kulkarni and Dr. S. S. Tambe. The thesis has not formed the basis for the award of any Degree, Diploma, Associateship, Fellowship, Titles in this or any other University or other institutions of Higher Learning. I further declare that the material obtained from other sources has been duly acknowledged in the thesis.

Somnath Nandi

ACKNOWLEDGEMENT

I take this opportunity to express my deep sense of gratitude to my mentor and architect of the thesis Dr. B. D. Kulkarni for introducing me to the fascinating area of artificial intelligence. His silent recognition, accurate observation, and critical analysis of a problem always strengthened me to achieve the goals with perfection.

I am extremely grateful to my co-guide Dr. S. S. Tambe for his encouragement and detailed scrutiny of work at every stage. His positive approaches during struggling phase are worth remembering of lifetime.

My heartfelt thanks are due to Prof. P. Ray, of Department of Chemical Engineering, University of Calcutta for inspiring me to pursue higher studies and encouraging me to face all the technical challenges with patience and zeal.

My sincere thanks to Dr. B. S. Rao and Dr. U. Sridevi, Catalysis Division of NCL for helping me to verify some of the optimized results experimentally.

I should highly appreciate the help rendered to me by all the scientists and staff members of Chemical Engineering and Process Development Division and the facilities provided by NCL library.

I am thankful to Council of Scientific and Industrial Research (CSIR), New Delhi for awarding me a research fellowship and the Director, National Chemical Laboratory for permitting me to submit the work done in form of the thesis.

I would like to express my sincere thanks to Prof. L. K. Kshirsagar, Principal, Maharashtra Institute of Technology, Pune for his constant support and encouragement. Special thanks to Prof. P. B. Jadhav, Coordinator Head, and all other faculty members of the Department of Petroleum and Petrochemical

Engineering, Maharashtra Institute of Technology, Pune for their kind cooperation and support at every stage.

Sincere thanks are due to my wife Sujata for her unfailing support and caring attitude. I am fortunate to be gifted with our loving daughter Diya whose presence is a great motivation at difficult times.

Last but not least, I would like to thank my friends Kaushik, Arindam, Subhash, Bibhas, Rupa, Debasish, Saptarshi, Supriya, Rahul, Imran, Rupali, Jitender, Ramdas, Ananda, Jeni, Yogesh, Jayram, Prakash, Bhupesh for contributing so much in their own ways.

(SOMNATH NANDI)

Dedicated to ARTIFICIAL INTELLIGENCE SYSTEMS GROUP (AISG)

NCL, Pune

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Chemical processes comprise a set of steps that convert raw materials into the desirable products through a chain of physico-chemical transformations. Chemical engineers, technologists and scientists are often required to analyze complex chemical processes and develop mathematical models simulating their steady-state and / or dynamic behavior. The objective of such a modeling effort is to construct, from the theoretical (phenomenological) and empirical knowledge of the process constituents, a mathematical description, which can be used to predict the process behavior under varying operating conditions. A mathematical process model provides information on the system behavior over important ranges of operating variables, and it represents at least the major features of the underlying chemical and physical mechanisms. The process models are mainly used for two purposes, i.e., prediction and optimization. Apart from these, process models are also required for a variety of tasks such as monitoring, fault detection and diagnosis and control.

Conventionally two approaches namely phenomenological and empirical are used to develop process models. In general development of a phenomenological model is difficult, tedious and time consuming since the complete knowledge regarding the underlying kinetics and heat and mass transfer phenomena is usually unavailable. In empirical modeling, the exact form of the data-fitting model needs to be specified which becomes difficult or impossible owing to the nonlinear nature of most chemical processes. In view of the stated drawbacks of phenomenological and conventional empirical modeling approaches, it becomes necessary to explore alternative and novel approaches for chemical process modeling. In recent years Artificial Intelligence (AI) based modeling techniques owing to their numerous advantages have provided an attractive avenue for conducting not only process modeling but also process optimization. Accordingly, the focus of the work presented in this thesis is design, development and application of Artificial Intelligence (AI) based modeling, optimization and monitoring strategies for chemical processes.

The thesis is structured as follows. Chapter 1 provides a broad overview of the AI-based methodologies utilized in the thesis, namely, multilayered perceptron (MLP) neural networks, generalized regression neural networks (GRNN), support vector regression (SVR), fuzzy logic (FL), genetic algorithm (GA), differential evolution (DE) along with the statistical

techniques such as principal component analysis (PCA) and partial least squares (PLS). The application domain of all these techniques has also been provided.

Chapter 2 presents a comparative study of two artificial intelligence based hybrid process modeling and optimization strategies, namely ANN-GA and SVR-GA, for modeling and optimization of benzene isopropylation on HBeta catalytic process. First an artificial neural network model is constructed for correlating process data comprising values of operating and output variables. Next, model inputs describing process operating variables are optimized using genetic algorithm (GA) formalism with a view to maximize the process performance. In the second hybrid methodology, a novel machine learning formalism, namely support vector regression (SVR), has been utilized for developing input-output process models and the input space of these models is optimized again using GAs. Using ANN-GA and SVR-GA strategies, a number of sets of optimized operating conditions leading to maximized yield and selectivity of the benzene isopropylation reaction product, namely cumene, were obtained. The optimized solutions when verified experimentally resulted in a significant improvement in the cumene yield and selectivity.

In Chapter 3, a hybrid strategy integrating PCA and generalized regression neural networks has been presented for modeling and monitoring of batch processes. The proposed PCA-GRNN strategy uses PCA for reducing the dimensionality of the process's input space and the first few principal component scores that explain a large amount of variance in the input data are used to develop a GRNN model correlating inputs and outputs. The effectiveness of the PCA-GRNN strategy for modeling and monitoring of batch processes has been successfully demonstrated by conducting two case studies involving penicillin production and protein synthesis. The results obtained thereby, using both clean and noisy data, demonstrate that the PCA-GRNN methodology is an attractive formalism for modeling and monitoring of nonlinearly behaving batch processes.

Chapter 4 introduces a Generalized Regression Neural Network (GRNN) based strategy for monitoring and modeling of batch processes. The methodology has been successfully utilized to forecast the dynamics of a fed batch bio-reactor for ethanol production. The advantages of the proposed strategy are: (i) it can make accurate output prediction even when the process behavior is nonlinear and (ii) the network training is much faster compared to the classical error-back-propagation strategy for ANN-training. These

advantages enable the proposed strategy as an attactive alternative for implementation in the on-line modeling and monitoring of chemical and biochemical processes.

In Chapter 5, a hybrid strategy integrating the PCA and a fuzzy neural network (FNN) has been presented for modeling and monitoring of a batch process. The proposed PCA-FNN strategy uses PCA for dimensionality reduction of the process input space (operating variables) and the first few principal component scores that explain a large amount of variance in the input data are used to develop an FNN model correlating process inputs and outputs. The effectiveness of the hybrid PCA-FNN strategy for modeling and monitoring of batch processes has been successfully demonstrated in a case study comprising the bio-process of protein synthesis.

A batch polymerization process has been modeled and optimized using ANN-GA hybrid formalism in Chapter 6. A sufficiently generalized ANN model was first developed for the isothermal polymerization of methyl methacrylate (MMA) conducted in the batch mode. This model is subsequently utilized for optimizing the operating condition variables using genetic algorithm. The objective function to be minimized in the optimization represents process operating cost, which in turn depends on the batch time and initiator cost. The results given by the ANN-GA scheme have been compared with those available in the open literature and they reveal a significant improvement over the published results.

In Chapter 7, a gain scheduling based control strategy is proposed for implementing the control action on an unstable process. The sole tunable parameter involved in the control strategy is optimized using the GA formalism. Efficacy of the control strategy has been successfully validated for regulating the dynamics of a fluidized bed reactor (FBR) exactly at an unstable steady-state (USS). The performance of the controller in the presence of deterministic and stochastic load disturbances has also been studied.

Chapter 8 introduces a novel AI-based hybrid process modeling – optimization strategy combining an ANN-based process model with a relatively less studied stochastic optimization formalism, namely, differential evolution (DE). Having built an ANN model based on the historic process data, the input space of the model is optimized using the DE technique. The efficacy of the ANN-DE hybrid formalism has been demonstrated by considering a non-trivial optimization objective involving a CSTR, which in addition to the parameter design, also addresses the issue of tolerance design. The specific optimization

objective considered was minimization of CSTR's annual cost. In this case study, two ANN models were developed using noise-free and noisy steady-state process data. The input space of the ANN model consisting of CSTR's design and operating variables was then optimized using the DE method; simultaneously tolerances associated with the operating variables were also optimized. The solutions obtained thereby have been found to compare favorably with those given by a robust deterministic optimization formalism and those obtained earlier using the ANN-GA method. The ANN-DE approach presented here is sufficiently general and, therefore, can be employed for a variety of process design and optimization problems.

Finally, Chapter 9 provides summary of important results and conclusions drawn from the studies conducted in Chapters 2 to 8 of the thesis. It also provides some suggestions for the future research.

CHAPTER 1

GENERAL INTRODUCTION

1.1 MOTIVATION

Chemical processes comprise a set of steps that convert the raw materials into the desirable products through a chain of physico-chemical transformations. Engineers and scientists are often required to analyze the complex chemical processes and develop mathematical models which simulate their steady-state and / or dynamic behavior. The objective of such a modeling effort is to construct, from theoretical and empirical knowledge of the process, a mathematical description, which can be used to predict the process behavior (Constantinides, 1987). A mathematical model provides information on the process behavior, over important ranges of operating variables, in terms of equations, which reflects at least the major features of the underlying chemical and physical mechanisms. The process models are mainly used for two purposes: prediction and optimization. Apart from these, process models are also required for variety of other tasks such as monitoring, optimization, fault detection and diagnosis and control. The focus of the work compiled in the thesis is on prediction, and monitoring of chemical processes and optimization of process operations.

- Prediction : The primary goal of prediction is: given the values of the causal process variables and parameters, to predict the value(s) of the process response (output) variables. Here, the prediction is made by plugging in the values of the predictor variables (causal variables and parameters) into the mathematical process model.

- Optimization : It corresponds to determining those values of process inputs (predictors), which effect the desired process response. The typical optimization goals comprise maximization of conversion, selectivities of the desired products, operating profit etc., or minimization of the loss and selctivities of undesired products, etc.

- Monitoring : In order to deliver quality products, the process operating variables should precisely follow their specified trajectories precisely However, disturbances arising from the deviations in the specified trajectories, errors in changing raw material, variation in impurities, etc. affect the product quality adversely. Process variability can be reduced by employing an efficient monitoring strategy. Such a system while working online must be capable of quickly identifying any abnormal process behavior so that corrective measures can be taken swiftly.

In a few studies presented in this thesis, an effort has been made to develop strategies, which will ease the process modeling task such that an accurate and efficient process model can be developed in real time. Some novel techniques for optimization are also proposed, which are capable of selecting the best process operating conditions for maximization of a profit function, etc. A couple of process monitoring techniques are proposed and their efficacies are demonstrated for monitoring of running chemical and biochemical processes in more efficient manner. All of these strategies will be very useful to (i) process engineers in their daily operational activities, (ii) research scientists in searching for the best alternative for conducting modeling and optimization tasks, and (iii) a design engineer for fixing of the best design parameter. In essence, the techniques developed and demonstrated in the thesis are expected to be of immense help to chemical process industry.

1.1.1 Process Modeling

Conventionally, two approaches are utilized for the development of chemical process models:

Phenomenological Approach : In this approach (also termed "first principles" approach), the process behavior is described in terms of the appropriate mass, momentum and energy balance equations together with the pertinent chemical engineering principles. In phenomenological modeling, a mathematical model describing the physico-chemical phenomena underlying in the process is first formulated and the model fitting is conducted by estimating the values of the unknown model parameters by utilizing actual process input-output data. For data fitting purposes, the linear / nonlinear regression techniques based on the least squares minimization formalisms are commonly employed.

The distinct advantages of the phenomenological models are : (i) they account for the underlying process phenomenology as closely as possible, and (ii) they can be used in extrapolation i.e., a phenomenological model can be utilized even outside the range spanned by the experimental process input-output data used in the model fitting. A significant drawback of this type of modeling is that in most real-life situations the complete understanding of the physico-chemical phenomena underlying

a process is not available thus making the model development a tedious, costly and difficult task. Moreover, most chemical processes behave nonlinearly and thus they lead to complex nonlinear models consisting of algebraic / differential equations. Most often, these nonlinear models can not be solved using analytical techniques and therefore they require rigorous numerical methods for their solutions.

Empirical Approach : Here, the process behavior is modeled using appropriately chosen empirical equations, for instance, polynomial expressions. Empirical models provide a convenient alternative to the first-principles models. Mostly, they are discrete-time dynamic models comprising, for instance, Hammerstein and Weiner models, Voltera models and polynomial autoregressive moving average with exogenous inputs (ARMAX) or their nonlinear counterparts, such as NARMA models. Empirical models can also be described in terms of multi-variable linear and nonlinear equations. The principal advantage of the empirical modeling approach is that the model can be constructed solely from the process input-output data without explicitly invoking the process phenomenology. Empirical modeling comprises an heuristic procedure wherein an appropriate functional form that possibly fits the process data is selected in advance following which the unknown model parameters are estimated using a suitable function fitting procedure. The main difficulty in the empirical modeling is guessing an appropriate form of the data fitting function. For nonlinear chemical reactions, which are encountered frequently in practice, guessing an appropriate nonlinear form poses enormous difficulties leading to computation-intensive trial-and-error approach. Even after expending such an effort, there is no guarantee that a reasonably fitting model can indeed be found.

In the last two decades, *artificial intelligence* (AI) based novel modeling strategies have become available for modeling complex nonlinear processes. AI is the science and engineering of making “intelligent” systems, especially intelligent computer programs. It is related to the task of using computers to understand the “human intelligence”. Intelligence can be broadly defined as the computational part of our ability to efficiently achieve goals in the world. Since varying kinds and degrees of intelligence occur in people, many animals and some machines, AI does not confine itself to methods that are explicitly biological. There exist four major AI paradigms, namely, *Artificial Neural Networks* (ANNs), *Support Vector*

Regression (SVR), *Genetic Programming* (GP) and *Fuzzy Logic* (FL) that have been used in almost every scientific / engineering / technology discipline for modeling purposes.

***Artificial Neural Networks*:** ANNs are a black-box empirical modeling paradigm where process modeling is possible solely based on the historic process input-output data. For modeling purposes, ANNs utilize a generic nonlinear function and thus the troublesome task of specifying the form of the fitting function gets completely eliminated. ANNs possess certain added advantages due to which process modeling becomes easier, less cumbersome and fast (Tambe et al, 1996; Freeman and Skapura, 1992) compared to the phenomenological approach.

***Support Vector Regression*:** The support vector (SV) machines were largely developed at AT&T Bell Laboratories by Vapnik and co-workers (Vapnik 1995; Scholkopf et al. 1995, 1996). Due to this industrial context, the SV research up to date had a sound orientation towards real-world applications. Initial work focused on optical character recognition (OCR). Within a short period of time, SV classifiers became competitive with the best available systems for both OCR and object recognition tasks (Scholkopf et al., 1996). Recently it is also used in the nonlinear regression and time series prediction applications; excellent performances were soon obtained (Smola and Scholkopf, 2004).

***Genetic Programming*:** Genetic Programming (GP) addresses the problem of automatic synthesis of programming of the computer software (Koza, 1994). Specifically, the GP targets the problem of how to enable a computer to do useful things without instructing it via well-defined stepwise procedure. Genetic programming accomplishes this by breeding a population of computer programs over many generations using the operations of Darwinian natural selection, crossover (recombination), and mutation. In recent years, this approach has found applications also in automatically generating the mathematical process models. Specifically, given process data it has become possible to automatically obtain equations relating the process response (outputs) variables to the process predictor (input) variables (Szpiro, 1997; Kulkarni et al., 1999). The underlying procedure in automatically searching and

optimizing the form and the free parameters of the data fitting model is known as "symbolic regression".

***Fuzzy Logic*:** It essentially represents a multi-valued logic that allows intermediate values to be defined between conventional equations like "yes / no", "true / false", "black / white" etc. Commonly used qualitative notions such as "rather warm" or "pretty cold" can be formulated mathematically using fuzzy logic and processed by computers. In this way, an attempt is made to apply a more human like way of thinking in the programming of computers. Fuzzy Logic (FL) provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy or missing input information. This unique ability of FL has been utilized to model complex nonlinear processes where development of a suitable phenomenological mathematical expression becomes difficult (Bezdek and Pal, 1993; Mendel, 1995).

1.1.2 Process Optimization

Availability of an accurate process model is essential for predicting the process behavior under wide-ranging input conditions. It is however insufficient to possess only the process model since in many real-life situations process design engineers are interested in obtaining the optimal process operating conditions that would maximize the process performance. Accordingly, the process model should be amenable to optimization. The objective of the process improvement (optimization) could involve increasing the annual profit, increasing conversion, selectivities of the desired products, etc. or reducing the overall production cost, lowering selectivities of the undesired products, etc. Depending upon the type of process model (phenomenological / empirical / AI-based) a suitable formalism should be selected for the model optimization. There exist two broad classes of optimization methodologies, namely, *deterministic* and *stochastic* for the optimization of process models.

- ***Deterministic Methodology*:** The most widely employed deterministic optimization strategies use gradient-based techniques. This class of methods finds an optimum by following the local gradient of the objective function. They generate successive results based solely on the previous results. In the gradient-based methods, a gradient is

iteratively evaluated in terms of the first or second order derivative of the objective function (to be minimized) with respect to all the decision variables. In the next step, the decision variables are updated in a direction opposite (negative) of the gradient (Edgar and Himmelblau, 1989). The main drawback of most gradient-descent methodologies is that they require the objective function to be continuous, smooth and differentiable. However, in many real-world problems, the objective function could be noisy, non-smooth and discontinuous. For instance, the objective function based on an ANN model is continuous and differentiable although its smoothness can not be always guaranteed. Another major disadvantage of the gradient-descent methodologies is that they invariably get stuck into local optimum leading to a sub-optimal solution

- ***Stochastic Methodologies*:** These are mostly used in nonlinear optimization. They randomly generate candidate solutions which are manipulated according to a specific algorithm. Here, the emphasis is on sampling the search space as widely as possible while trying to locate the promising regions for further search. In contrast to the traditional deterministic optimization techniques, which manipulate a single candidate solution, the stochastic methods operate on a population of candidate solutions. The size of the population depends on the problem under consideration. This makes it possible for the stochastic techniques to search several areas of the solution space. In the stochastic techniques, randomly generated initial population of solutions is constantly refined so as to find better solutions. In recent years, several AI-based nonlinear search and optimization techniques have been proposed (Deb, 1995). In what follows, two stochastic optimization methods used in the studies presented in this thesis are described.

  ***Genetic Algorithms*:** The Genetic Algorithms (GA) is the most widely used AI-based stochastic nonlinear optimization formalism. These were originally developed as the genetic engineering models mimicking the population evolution in natural systems. Given an objective function, the GAs search its parameter space so as to maximize (or minimize) the function value. The advantage of the GA technique is that it searches the solution space heuristically and hence it is un-affected by the properties (e.g., smoothness, differentiability, continuity, etc.) of the objective function (Goldberg, 1989). Thus, the GA technique can be effectively integrated with the ANN-based process models to search the optimal domains of the process operating parameters.

***Differential Evolution*:** In recent times, differential evolution (DE) is emerging as another efficient AI-based stochastic optimization technique. It has evolved based on the principle of genetic evolution. Similar to the GA, differential evolution technique also scans the solution space probabilistically and hence does not have pre-requisite of smoothness, differentiability, continuity etc. of the objective function (Price and Storn, 1997). Thus for optimization purposes, an ANN-based process model can be effectively be handled with differential evolution technique.

1.1.3 Process Monitoring

The computer-based sophisticated control systems used by today's chemical industry collect and archive process data continuously or at fixed time intervals. These measurements are used for the on-line monitoring of the process and thereby timely detecting and diagnosing any abnormal process behavior. There exist mainly four approaches for monitoring and detecting / diagnosing faults in chemical / biochemical processes. The formalisms used by these approaches are : (i) dynamic phenomenological (first principles - based) model, (ii) knowledge-based expert systems, (iii) artificial neural networks (ANNs) – based models, and (iv) multivariate statistical (MVS) methods namely principal component analysis (PCA) and partial least squares (PLS). Among these, the fourth approach utilizing MVS methods is by far the most popular.

The PCA is a dimensionality reduction technique for compressing noisy and correlated process input measurements into a smaller, informative latent variable space. It forms latent variables (*principal components* or *scores*) that are expressed as the linear combinations of the input variables and the corresponding weights (*loadings*). When measurements are highly correlated, first few principal components (PCs) capture maximum amount of variance in the measurements. Thus, measurements can be represented using fewer PCs without a significant loss of the information content. A PCA-based technique known as multiway principal component analysis (MPCA) has been developed (Wold et al., 1987), whereby dimensionality reduction can be realized for multivariate data emanating from several batches.

The PLS is a PCA-related technique that simultaneously finds latent variables for both the input and output measurement sets following which the latent variables are correlated linearly. The principal advantage of the PLS method is that outputs can be predicted using only a few PCs that capture the maximum variance in the input-output measurements. Multiway version of the PLS (MPLS) has also been proposed to deal with the batch systems for which product quality data are available along with the measurements of input variables. In recent years, several variants of PCA, PLS, MPCA and MPLS have been introduced (Nomikos and MacGregor, 1994). The principal advantage of these formalisms is that the information contained in the original database is extracted in a lower dimensional vectors and matrices; the data reduction achieved thereby is significant, which simplifies the monitoring task substantially.

The PLS and MPLS though are fast and efficient methods, they suffer from a significant drawback that being linear methods they capture only the linear relationships between the principal components of the predictor (input) and response (output) variables. Consequently, they perform poorly in predicting response variables of nonlinearly behaving batch processes, which are abundant in chemical / biochemical industry. To overcome the limitation imposed by the linearity of PLS / MPLS-based models, the neural network PLS (NNPLS) method was proposed (Qin and McAvoy, 1992). In NNPLS, an outer mapping is performed first wherein measurement matrices of the input and output variables are decomposed into their respective scores and loadings matrices. Next. An inner model is defined with the help of a feed-forward neural network (FFNN) model such as the multilayered perceptron (MLP) and radial basis function network (RBFN). The NNPLS method differs from the direct ANN approach in that the input-output data are not directly used to train the FFNN but are preprocessed by the PLS outer transform. This transform decomposes a multivariate regression problem into a number of univariate regressors wherein each regressor is implemented separately using a simple single input – single output (SISO) FFNN. The NNPLS is a powerful technique for monitoring nonlinear batch processes although usage of even a simple SISO neural network does involve some heuristic in choosing an optimal value of the network's structural parameter namely the number of nodes in the hidden layer. Depending on the choice of a training algorithm, the values of algorithm-specific parameters also need to be fixed heuristically. A heuristic search of network's weight

(parameter) space is also required to obtain a globally optimum solution, since none of the above-stated training algorithms guarantees convergence to the globally optimum solution in a single training run. Such a computation–intensive heuristic is usually undesirable especially when the network is trained online and the time duration to take a corrective action in the event of an abnormal process behavior is short. Thus, there exists a significant scope for a generalized formalism which can be fast and accurate for nonlinearly correlating the process output variables with the operating variables.

1.2 RESEARCH AIM

Most chemical processes are inherently nonlinear, and complex multiple interactions among the associated phenomena such as heat and mass transfer make these processes even more difficult to model using phenomenological approaches. Thus, to develop workable models a number of artificial intelligence based nonlinear modeling formalisms are proposed.

Development of a good process model is not sufficient as many a times processes are operated under suboptimal conditions. To secure optimal operating conditions, optimization of a process model is a must. Thus, we intend to use the AI-based gradient-free stochastic methodologies to perform the optimization task in an efficient manner.

Monitoring is an important task as it quickly identifies any abnormal process behavior so that a corrective measure could be taken swiftly to prevent a possible occurrence of accident as also injuries to human life and equipment damage. Multivariate statistical methods are commonly employed for process monitoring; these approaches compress data structure by finding out latent variables without loosing useful information and subsequently predicting the output. The conventional methodologies use a linear mapping which performs poorly for nonlinear chemical / biochemical processes. Thus a nonlinear version of the multivariate statistical method is developed, which will be very useful for monitoring of various types of processes often encountered by chemical industry.

In the thesis an attempt has been made to highlight efficacies of a number of AI-based formalisms namely artificial neural networks (ANN), support vector regression (SVR),

fuzzy logic (FL) for modeling of various chemical and biochemical systems. Some of the industrially important processes are even optimized with help of AI-based gradient-free optimization techniques such as genetic algorithms (GA) and differential evolution (DE). In addition, process monitoring strategies namely principal component analysis (PCA) and partial least squares (PLS) were applied for couple of biochemical processes to demonstrate its usefulness. All these methodologies are discussed in the following section, and their important applications are also listed.

1.3 METHODOLOGIES USED IN THE THESIS

1.3.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) have their origin in the efforts towards developing the computer models of the information processing that takes place in the human nervous system (Rumelhart and McCelland, 1986). In essence, ANNs are simplified mathematical models describing the biological nervous system and its functioning. ANNs are based on the concept that a highly interconnected system of simple processing elements can learn complex interrelationships between the independent and the dependent variables in a data set. They offer an attractive approach to the black-box modeling of highly complex, nonlinear systems having a large number of inputs and outputs (Tambe et al., 1996). Architecturally, ANNs are massively connected parallel structures containing processing elements (PEs) called *neurons* (also known as *nodes*). In the illustrative structure (Figure 1.1) a three-layered ANN is depicted, which performs exclusively data-driven nonlinear function approximation. The layers are known as: an *input* layer, an intermediate (*hidden*) layer and an *output* layer; each layer contains in it a number of neurons. The number of neurons in the input layer is equal to the number of inputs in the system to be modeled while the number of neurons in the output layer equals the number of system outputs. The number of neurons in the hidden layer is determined heuristically. Each neuron in the input and hidden layers is connected to all the neurons in the next layer by means of a weighted connection. The outputs of the hidden and output layer neurons can be calculated by a suitable nonlinear transform equation provided the neuron-specific inputs are known (Bishop, 1994). The ANN architecture depicted in Figure 1.1 belongs to the "feed-forward" class of neural networks. In the feed-forward networks, the

information flow is always in the forward direction, that is from the input layer neurons to the output layer neurons. The other class of ANNs is known as "feedback" networks where output from a neuron is fed back as an additional input. The feedback networks also termed "recurrent networks" are primarily used for modeling dynamical systems. As compared to the feed-forward networks, the recurrent networks have found fewer applications. Feed-forward ANNs have found numerous applications in every scientific, technology and engineering discipline. The principal applications of the ANNs are: (i) nonlinear function approximation (i.e., process modeling), (ii) pattern recognition and classification, (iii) data reduction and compression, (iv) signal processing, and (v) noise reduction.

1.3.1.1 Advantages of ANNs

The distinct advantages of the ANN formalism are:

1. An ANN model can be developed solely from the historic process input-output data.
2. Multi-input-multi-output (MIMO) nonlinear relationships can be approximated easily and simultaneously.
3. A well developed ANN model possesses good generalization ability due to which it can be used to predict values corresponding to a new data set not used in the model development.
4. An ANN can be used to identify/model complex processes in the presence of noisy data or incomplete information.
5. An ANN model can be developed even using qualitative data.
6. ANNs use a generic nonlinear function for function approximation and thus there is no need to specify system-specific data fitting function as done in traditional regression.

If the available data set is large, model-building (training) via ANN approach could be time-consuming however once trained properly they can calculate results from a given input quickly. This unique characteristic makes them potential candidates for an on-line use in the real-time process control.

1.3.1.2 Neural network learning paradigms

In order that an ANN model accurately approximates the nonlinear relationship between its inputs and outputs, it needs to be "trained" using the available example input-output data set. During training, the ANN learns the relationships existing between its inputs and the outputs ANNs training essentially refers to the estimation of connection weights. Broadly speaking there exist two types of algorithms for training an ANN (feed-forward or feed-back), namely, *unsupervised* and *supervised* learning algorithms. In the unsupervised (or self-organizing) learning, the output values predicted by an ANN are not compared with their true (also termed *desired* or *target*) values. Here, the learning is driven by a similarity measure without specifying the target values. The self-organizing network modifies its weights such that the most similar input data vectors are assigned to the same output (cluster) unit. The Kohonen self organizing map (SOM) (Kohonen, 1988) and Adaptive Resonance Theory (ART) (Carpenter and Grossberg, 1988) are the examples of the unsupervised learning.

In the supervised learning, training is accomplished by presenting a set of the input and the corresponding desired output vectors to the network's input and the output layers, respectively. The essential feature of this type of training is the availability of an "external teacher" in the form of desired (target) output vectors. In supervised training, input patterns are applied to the network's input layer sequentially and the outputs of the hidden and output layers are computed. The evaluated network outputs are then compared with their corresponding desired magnitudes (using target output values) and the network weights are adjusted iteratively so as to minimize the value of a pre-specified error function. The widely used algorithm for training the feed-forward networks is the *error-back-propagation* (EBP) algorithm (Rumelhart et al., 1986) which is an example of the supervised learning. The EBP method utilizes the *generalized delta rule* (GDR), which is a gradient descent method for minimizing an error (difference between the target output value and the network predicted value) function (Baldi, 1995). The EBP algorithm is used to train the most widely used feed-forward architecture known as *multilayered perceptron* (MLP). The MLP possesses same structure as shown in Figure 1.1; it can also possess more than one hidden layer. A specialty of the MLP networks is that it uses a nonlinear activation (transfer) function for computing the outputs of the hidden and the output nodes. The phenomenal nonlinear approximation

ability possessed by the MLP network is primarily due to the usage of the nonlinear transfer function such as the sigmoid function.

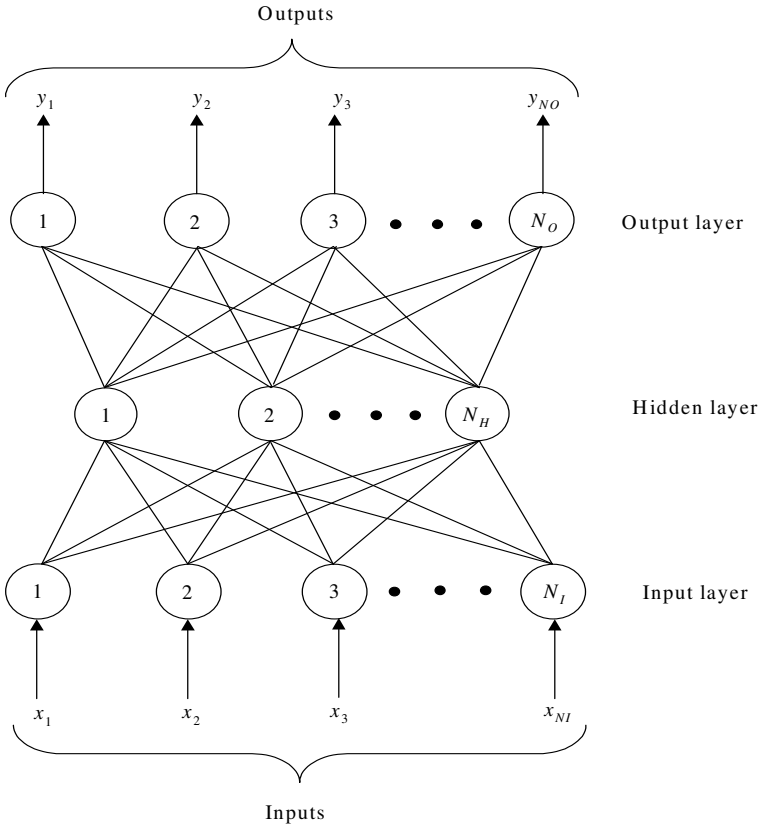


Figure 1.1: Schematic of commonly used artificial neural network

1.3.1.3 ANN applications in chemical engineering

In chemical engineering, ANNs are primarily used for the steady-state/dynamic process modeling, nonlinear process identification and control, fault detection and diagnosis, QSAR (quantitative structure-activity relationships) and QSPR (quantitative structure-property relationships) tasks. Some of the ANN applications in these areas are listed in the Table 1.1 below.

Table 1.1: A representative sample of ANN applications in chemical engineering

Application	Reference
Process Modeling	
• Improvement of catalytic manufacturing process	Chitra (1993)
• Prediction of brewing fermentation	Syu et al. (1994)
• Improvement of penicillin fermentation process	Di Massimo et al. (1992)
• Steady-state modeling of Fischer-Tropsch synthesis	Sharma et al. (1998)
• Degree of mixing in a reactor	Bhagat (1990)
• Naphtha cut point prediction	Wadi (1996)
• Simulation of manufacturing process	Jula (1996)
• Modeling of coker fractionator	Blaesi and Jensen (1992)
• Modeling distillation column for refinery units	Baratti et al. (1995)
• Prediction of product quality in injection moulding	Smith (1993)
• Simulation of membrane separation	Niemi et al. (1995)
• Modeling of distillation column	Baratti et al. (1998)
• Modeling of multiphase photodegradation systems	Pareek et al. (2002)
Process Identification and Control	
• Adaptive optimal controlling	Kim and Lee (1996)
• Nonlinear process control improvement	Sun et al. (1997)
• Process identification and model predictive control of hydroxylation of phenol	Tendulkar et al. (1998)
• Plant identification and control	Barada and Singh (1998)
• Statistical process control	Cheng (1997); Cook and Chiu (1998)

• Quality control	Malkani and Vassiliadis (1995)
• Process control	Cheng (1995)
• Process modeling and controlling	Whittaker and Cook (1995)
• Improving prediction of activated sludge process	Cote et al. (1995)
• Analysis of butane splitter tower and estimation of outlet composition from a distillation tower	Baratti (1994)
• Control of distillation column using a combination of neural feedforward control with MPC	Lee and Park (1992)
• Controlling of an Industrial plant	Ramasamy et al. (1995)
• Control of distillation	Ramachandran and Rhinehart (1995); Willis et al. (1992); Megan and Cooper (1993)
• Control of steel rolling mill	Sbarbaro-Hofer et al. (1992)
• Lime kiln process identification and control	Ribeiro et al. (1993; 1995)
• Modeling dynamic pH control system	Bhat et al. (1990)
• Optimization and control of CSTR fermenter	Normandin et al. (1994)
• PI controller tuning	Chan et al. (1995)
• Real time optimization of crude oil distillation column	Yusof et al. (2003)
• Emission control in palm oil mills	Ahmad et al. (2004)
• Control of industrial polyethylene reactor	Badhe et. al. (2007)
• Identification and control of lab-scale distillation column using LabView	Canete et al. (2008)
Fault Detection and Diagnosis	
• Detection of fouling in a pressurized water reactor in a power plant	Kavaklioglu and Upadhyaya (1994)

• Process fault detection and diagnosis of an nonisothermal CSTR	Vora et al. (1997)
• Detection of coolant boiling in a light water nuclear reactor	Kozma and Nabeshima (1995)
• Fault detection and diagnosis of a heat exchanger coupled with CSTR	Sorsa et al. (1991)
• Equipment/machine fault diagnosis/detection	D'Antone (1994); Gan and Yang (1994)
• Intelligent manufacturing control	Holter et al. (1995)
• Fault diagnosis in complex chemical plants	Hoskins et al. (1991)
• Process monitoring and fault diagnosis of FCCU	Rengaswamy and Venkatasubramaniun (1995)
• Real time fault diagnosis of a nonisothermal CSTR	Kavuri and Venkatasubramaniun (1992)
• Fault diagnosis in batch distillation	Scenna et al. (2000)
• Detection of faults in packed towers	Sharma et al. (2004)
QSAR / QSPR	
• Prediction of fluid properties	Lee and Chen (1993)
• Prediction of polymer properties	Sumpter and Noid (1994)
• PVT data analysis of gases and vapors	Normandin et al. (1993)
• Classification and prediction of inductive and resonance effects of substituents	Kvasnicka et al. (1993)
• Physical properties of palm oil components	Yusof et al. (2003)
Other Important Applications	
• General applications to chemical engineering	Himmelblau (2000)
• Thermodynamics – Prediction of vapor – liquid- equilibrium	Sharma et al. (1999)
• Cost estimation	Bode (1998a, b); De la Garza and Rouhana (1995)

• Job scheduling	Hill and Remus (1994)
• Job shop scheduling	Bugnon et al. (1995); Jain and Meeran (1998); Quiroga and Rabelo (1995); Sabuncuoglu and Gurgun (1996); Sim et al. (1994)
• Process planning	Mei et al. (1995)
• Analysis and optimization of process data	Savkovic-Stevanovic (1994)
• Detection and location of gross errors in process systems	Aldrich and van Deventer (1995)
• Membrane characterization	Yusof et al. (2003)
• Evaluation of drying and degradation Kinetics	Kaminski and Tomczak (2000)
• Monitoring of waste-water quality and prediction of water quality index	Hore et al. (2008)

1.3.2 Support Vector Regression:

Support vector regression (SVR) is based on the statistical learning theory and gaining popularity as alternative to ANNs for data driven modeling. SVR performs structural risk minimization by penalizing model complexity while simultaneously minimizing the training data error. ANNs on the other hand perform empirical risk minimization by minimizing only the training data error. Thus, the overfitting phenomena is almost absent in SVR based models, and these models are sufficiently generalized. SVR provides globally optimal solutions since it is based on the minimization of quadratic objective function possessing a single global minimum. Apart from this, SVR provides robust solutions and allows sparseness of regression function thus describing the function with a limited number of support vectors (subset of training data) (Smola and Scholkopf, 2004).

The support vector regression (SVR) is an adaptation of a recently introduced statistical / machine learning theory based supervised classification paradigm known as, support vector machines (SVM) (Vapnik, 1997). The objective is to build a model to fit a regression function, $y = f(x)$, such that it accurately predicts the outputs $\{y_i\}$ corresponding to

a new set of input examples, $\{x_i\}$. To obtain the correct fit, SVR considers the linear estimation function in the high dimensional feature space (Vapnik, 1999).

Support vectors (SV) obtained in SVR are known as the most informative data points that compress the information content of the training set, thereby representing the entire SVR function. Thus, support vectors (SVs) are a subset of the input data set. However, with an increase in the input dimensions, the dimensions in the high dimensional feature space further increases by many folds, and thus transformation becomes computationally intractable. Such a problem can be overcome by defining appropriate kernel functions in place of the dot product of the input vectors in high dimensional feature space. The advantage of a kernel function is that the dot product in the feature space can now be computed without actually mapping x_i into high dimensional feature space, thus allowing calculations to be done in the input space itself.

1.3.2.1 Advantages of SVR:

1. SVR uses structural risk minimization by penalizing model complexity while minimizing the training data error. This results in a model possessing better generalization ability. ANN on the other hand minimizes only the empirical risk training set error.
2. SVR provides globally optimal minimal solutions by solving a quadratic objective function possessing a single minimum.
3. SVR allows computations in the input space itself, thus avoiding cumbersome calculations.
4. SVR defines a regression function that is robust and allows sparseness of regression function.

1.3.2.2 Applications of SVR:

The SVR has found diverse applications such as optical character recognition (OCR) (Vapnik, Golowich and Smola 1997), object recognition tasks (Scholkopf, Burges and Vapnik 1996, Blanz et al. 1996, Scholkopf 1997), and regression and time series prediction applications, (Muller et al. 1997, Drucker et al. 1997, Stitson et al. 1999). In Table 1.2 a list of few selected applications of support vector regression in chemical engineering and process systems engineering is provided.

Table 1.2: Chemical engineering applications of support vector regression

Applications	References
• Soft sensor development for fed-batch bioreactors	Desai et al. (2006)
• Prediction of gas hold-up in bubble column reactors	Gandhi et al. (2007); Gandhi et. al. (2008)
• Modeling and optimization of pilot plant study of cumene synthesis	Nandi et al., (2004)
• Modeling and optimization of plasma reactor	Istadi and Amin, (2006)
• Development of Correlations for overall gas hold-up, volumetric mass transfer coefficient, effective interfacial area of Newtonian and non-Newtonian fluids	Gupta et al., (2009)
• Catalyst Development	Valero et al., (2009)
• Analysis of LDA time-series	Gandhi et al., (2008)
• Applications of SVR in Chemistry	Ivanciuc, (2007)
• Constrainted run to run optimization for batch processes	Li et al., (2006)

1.3.3 Fuzzy Logic:

The modeling of many systems involves the consideration of some uncertain and qualitatively defined variables. These variables and their influences on the system are defined in linguistic terms. This form of uncertainty can be handled in a rational framework of 'fuzzy set theory'. Just as the human brain, which can interpret imprecise and incomplete information and still comes out with appropriate solutions, the fuzzy logic provides a systematic method to deal with information represented linguistically. It performs numerical computations by using linguistic labels represented by membership functions.

In contrast to a classical set, a fuzzy set is a set without a crisp boundary. The transition from 'belongs to a set' to 'not belongs to a set' is gradual, and this transition is characterized by membership functions that gives fuzzy sets their enormous flexibility in modeling commonly used linguistic expressions such as 'selectivity of desired product is quite high' or 'weight average molecular weight distribution of polymer is uniform', or 'reactor temperature is low' etc. It should be noted that the fuzziness generally originates from the uncertain and imprecise nature of abstract thoughts and concepts. The effect of several variables like temperature and pressure can only be described in terms linguistic terms such as *very high*, *high*, *fairly high*, *medium*, *fairly low* and *low*. These linguistic terms represented by the fuzzy numbers (say) *A*, *B*, *C*, *D*, *E* and *F* are defined by the membership functions as represented in Fig. 1.2. The membership function (MF) is a curve that defines how each point in the input space is mapped to a "membership value" (or degree of membership) between 0 and 1.

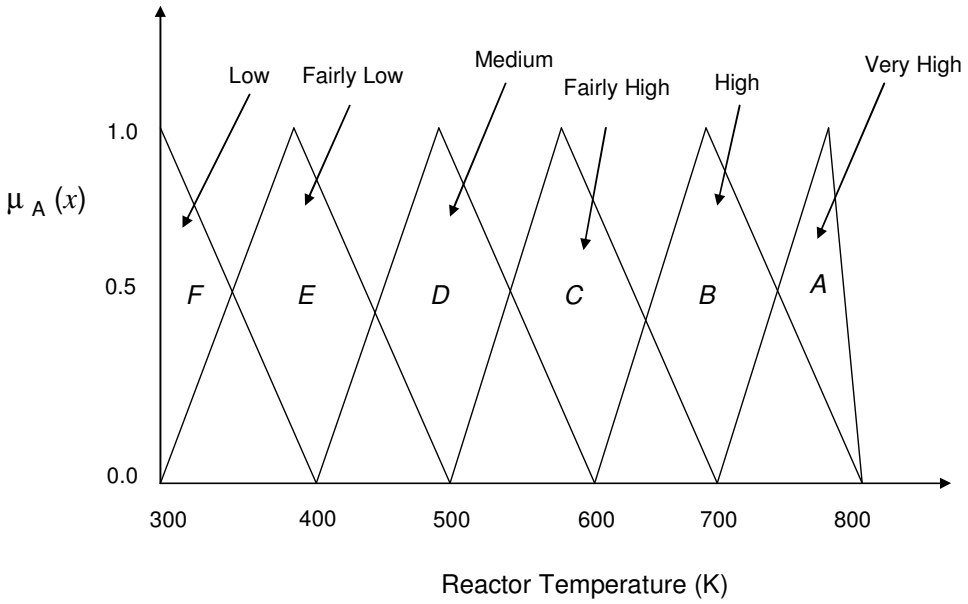


Figure 1.2: Membership function indicating definition of linguistic grades relative to temperature

A fuzzy variable, *A*, is commonly expressed as a pair of data:

$$A = \left\{ \left(x, \mu_A (x) \right|_x \in X \right) \right\} \tag{1.1}$$

where, x is the element, X is collection of objects denoted by x; and $\mu_A(x)$ is called the membership function for the fuzzy set A, which defines the degree of an element belonging to a set. The membership function maps X to the membership space, M, where $M = \{0, 1\}$.

A common definition of a fuzzy system is that which emulates a human expert. In this paradigm the knowledge of human expertise would be put in the form of a set of fuzzy linguistic rules. These rules would produce an approximate decision, just like a human being would do in similar situations. Refer to Figure 1.3 where a block diagram of a fuzzy system is shown. As seen, the human operator observes quantities by making an observation of the inputs, i.e., reading a meter or measuring a chart, and performs a definite action, e.g., pushes a knob, turns a switch, reduce opening of some valve, shutting down or start up of any operation etc. which leads to a crisp action, shown in the figure by the output variable $y(t)$. The human expert can be replaced by a combination of a fuzzy rule based system (FRBS) and a block termed defuzzifier. The sensory (crisp or numerical) data is fed into the FRBS where physical quantities are represented or compressed into logistic variables with appropriate membership functions. These linguistic variables are then used in the "antecedents" (the "IF") of a set of fuzzy rules with an inference engine to result in a new set of fuzzy linguistic variables or "consequents" (the "THEN") as variables are then denoted in the figure by z, combined and changed to a crisp (numerical) output $y^*(t)$ which represents an approximation to actual output $y(t)$ (Jamshidi, 1997).

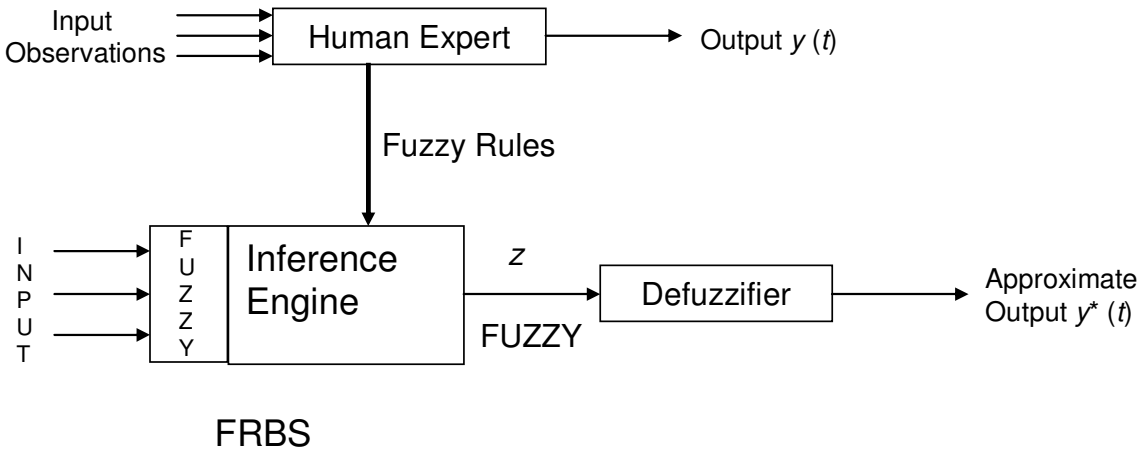


Fig. 1.3: Schematic representation of fuzzy systems

Fuzzy logic and neural networks are complementary to each other, in order to utilize strength of both; they can be combined to an integrated system. The integrated system will then have advantages of both neural networks (namely, learning abilities and connectionist structure) and the fuzzy systems (humanlike 'if – then' rules and ease of incorporating expert knowledge and judgment available in linguistic terms). The complexity and nonlinearity of the underlying system are handled through a neural network while the imprecision associated with the system parameters are incorporated through fuzzy logic.

An approach to integrate fuzzy logic and neural network is to simply fuzzify some of the neural network system parameters and retain the basic properties and architectures of the neural network model. In such models, a crisp neuron becomes fuzzy and response of the neuron to its next layer activation signal is of a fuzzy relation. The learning mechanisms and interpretation capability of the neural network system is enhanced by the fuzzy representations of the knowledge domain (Ni et. al, 1996; Ronen et. al., 1998; Rahman et. al., 2002).

1.3.3.1 Applications of Fuzzy Logic

There are many chemical engineering processes, where the quality characteristics of the product cannot be measured objectively either on-line due to the lack of proper sensors or off-line due to the absence of any measuring devices. In these cases, a human expert is employed to assign the product quality characteristics to certain predefined categories (classes), based on his / her experience and perception. The procedure of employing a human expert to perform the classification usually requires interruption of the process in order to collect a sample. Furthermore, this way of classifying the product quality is very subjective and may lead to significant errors, especially when the same expert is not always employed to perform the classification. Over the last few years fuzzy logic theory has emerged as a useful tool for modeling processes which are too complex for conventional quantitative techniques or when the available information from the process is qualitative, inexact or uncertain. Another advantage of FL is that in contrast to the classical system modeling, where both the order and the type (i.e. linear or nonlinear) of the model are important, in fuzzy modeling we are mostly concerned about the order of the model (Pedrycz, 1993), which is actually the number of rules included in the rule base (Sugeno & Yasukawa, 1993). Given that the

estimation of a human expert who is employed to classify the quality properties of a product can only be qualitative (fuzzy), it naturally arises that fuzzy logic theory could be the proper tool for quality parameter prediction (Raptis et. al., 2000).

Table 1.3: Some applications of fuzzy logic to chemical engineering

Applications	References
• Fault diagnosis of chemical processes	Chang et. al. (2002); Dash et al. (2003); Chang and Chang (2003); Chen and Chang (2006)
• Fermentation process	Yamada et al. (1991); Alfafara et al. (1993); Nucci et al. (2005)
• Wine and brewery distillate maturation process modeling	Whitnell et. at. (1993); Tsekouras et al. (2002)
• Control of biochemical reactor	Sousa and Almeida (2001); Campello et. al. (2003); Galluzzo et. al. (2008)
• Modeling of anaerobic digester	Polit et al. (2002)
• Tissue softness estimation	Tsekouras et al. (2002)
• Wastewater treatment process	Muller et al. (1997); Manesis et. al. (1998); Huang and Wang (1999); Galluzzo and Cosenza (2009); Pai et. al. (2009)
• Modeling and control of FCC unit	Sharma and Rengaswamy (2000); Azeem et. al. (2006); Taskin et. al. (2006); Osofisan and Obafaiye (2007)
• Drug synthesis and production	Horiuchin and Hiraga (1999)
• Temperature control	Lu (1996); Honda et. al. (1998); Rao et. al. (1999); Skogestad (2000); Abilov et. al. (2002); Karer et. al. (2007); Causa et. al.

	(2008)
• Polymerization reactor	Mahesh et. al. (1993); Vicente et. al. (2003); Altinten et. al. (2003); Hanai et. Al. (2003); Ali and Al-Humaizi (2005); Antunes et. al. (2005); Ghasem (2006); Altinten et. al. (2006)
• Crystallization operation	Sheikhzadeh et. al. (2008a, 2008b)
• Granulation operation	Cameron et. al. (2005)
• Industrial Applications	Sugeno (1985); Jamshidi (1997); King and Mamdani (1997); Ravi et. al. (1998)
• Uncertainty analysis	Kraslawski (1989); Kumar and Schuhmacher (2005); Mitra et. al. (2009)

1.3.4 Genetic Algorithms

Genetic Algorithms (GAs) are computerized search and optimization algorithms based on the mechanics of natural genetics and natural selection. They utilize the "survival-of-the-fittest" paradigm of biological evolution along with the "genetic propagation of the characteristics" for robustly searching the solution (parameter) space of an optimization problem (Goldberg, 1989). In particular, given an objective function, the GAs search its parameter space stochastically (randomly) with a view of minimizing or maximizing the function value. GAs simultaneously evaluate many points in the decision variable space and hence they are more likely to converge at the globally optimum solution. Moreover, GAs do not require the objective function to be smooth, continuous and differentiable and thus they can be used in situations where objective function properties are not known in advance. In GA implementation, first a population of probable (candidate) solutions to an optimization problem is generated randomly. The population so created is iteratively refined using the principles of natural selection and genetics. This refinement is conducted using a number of genetic operators. At the end of an iteration, a new population (generation) of candidate

solutions gets formed. Within an iteration, GAs explore different areas of the decision variable space, and then direct the search to regions where there is a high probability of finding better solutions. How a candidate solution performs at fulfilling the optimization goal is ascertained by evaluating its fitness (goodness) using a prespecified fitness function. Invariably, the new generation of probable solutions fares better at the optimization task involving objective function maximization or minimization. Upon evolving over many generations, the GA arrives at an optimal solution.

1.3.4.1 Overview of GA implementation

A simple GA-implementation procedure comprises following components:

1. Representation / Encoding
2. Initialization
3. Evaluation of fitness
4. Genetic operations

Representation: Commonly, the candidate solutions (also termed *strings* or *chromosomes*) are represented using binary digits (binary coding) i.e., in terms of *zero* and *one*. Some GA variants also work on the real values of the decision variables directly i.e., without coding them in binary.

Initialization: Initialization refers to the generation of the initial population of solutions as well as the choice of some GA-specific parameters such as the population size. The preferred characteristics of an initial population are diversity and reasonable levels of fitness values. Depending upon application, the initial population may be generated randomly or can be chosen carefully from the candidates based on the user's experience. The optimal choice of the population size depends upon the nature of domain, representation and evolution schemes and the genetic operators used in the simulation.

Evaluation of fitness: Once a population of candidate solutions is created, the solutions need to be evaluated to determine their fitness in the environment. For an optimization problem, the "environment" refers to the objective function. Depending on how low (for minimization

problem) or how high (for maximization) the objective function value for an individual solution is, its fitness should have a proportionally high value. In many real life cases several objectives need to be fulfilled simultaneously. One way of handling multiple objectives is to define an aggregated new objective function that is a weighted-sum of all the objectives. Here, the choice of the weights reflects the relative importance of multiple optimization objectives.

Genetic operations: Genetic operators provide the means by which sufficiently good strings (solutions) are selected from the population and subsequently their genetic components or the building blocks are altered to produce a new population (the "offspring") of solutions. There exist three genetic operators: (i) *reproduction*, (ii) *crossover*, and (iii) *mutation*.

- *Reproduction*: This operator selects good strings from the current population of solutions to form the mating pool of parents. The essential feature of reproduction operation is to choose solutions possessing above-average fitness values (parents) for producing a new generation of solutions. This step ensures that the fitter individuals in the current population are allotted more opportunities to produce offspring. The *Roulette Wheel* (RW) selection method is a commonly used method of parent selection and is easy to implement. The RW is a noisy scheme (Goldberg, 1989) however, and therefore its stable version called *Stochastic Remainder Selection* (SRS) (Deb, 1995) is commonly used. Another selection scheme, called *Tournament Selection* (TS), where probability of the selection of a candidate (parent) remains fairly constant across generations (Goldberg, 1989), is also employed in specific cases.
- *Crossover*: It refers to the random recombination of the parts of two chromosomes (parents) to produce two new chromosomes (offspring). Many types of crossover operations are reported in the GA literature. The single point crossover (shown in Figure 1.4) is the most common though other types (multi-point crossover) also have been used. In the single point crossover two strings are picked from the mating pool at random and the crossover site is also selected randomly. Next, the parent strings are cut at the crossover point and cut portions are exchanged mutually between the parent strings to form two offspring strings. From an optimization point of view, the crossover operator

tends to improve the combinatorial diversity in the new solution population by using the building blocks present in the current population.

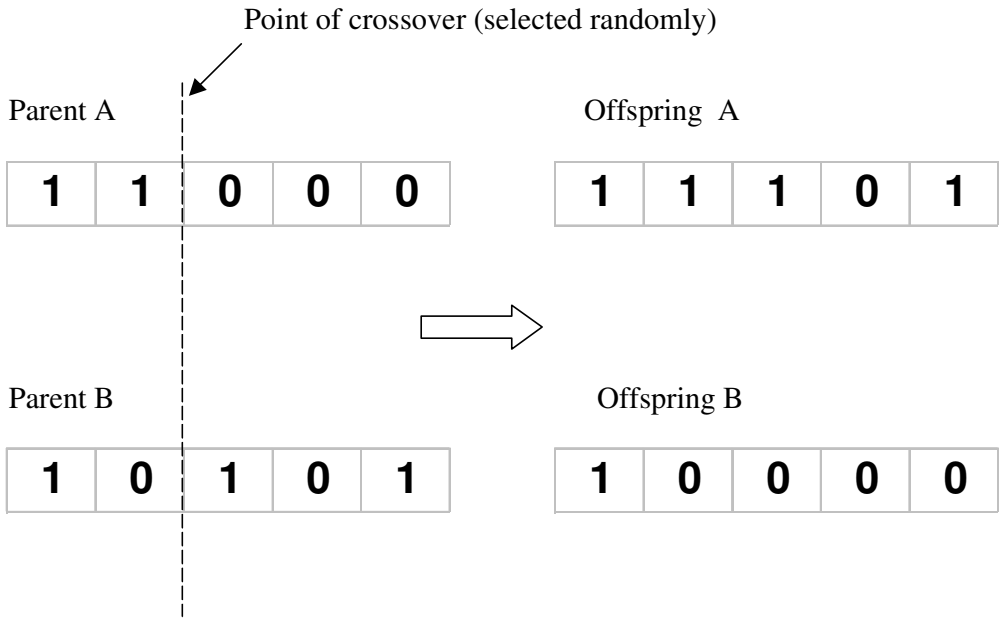


Figure 1.4: Schematic showing single point crossover in binary strings

- *Mutation*: To be effective, the GA needs an influx of characteristics extraneous to the population created using the crossover operation. This is provided by the mutation operation. For a simple GA using the binary encoding, the mutation refers to changing of 1 to 0 and vice versa (see Figure 1.5). The mutation operation on binary coded string is performed with a low probability (≈ 1% bits are mutated). This is due to the fact, that with a high probability of mutation, the good features brought forth by the selection and crossover operations get completely destroyed. The mutation operation is needed to alter a candidate solution locally in the hope of creating a better string i.e., the mutation operation helps in searching the local regions in the parameter space more thoroughly in order to home on to the global optima.

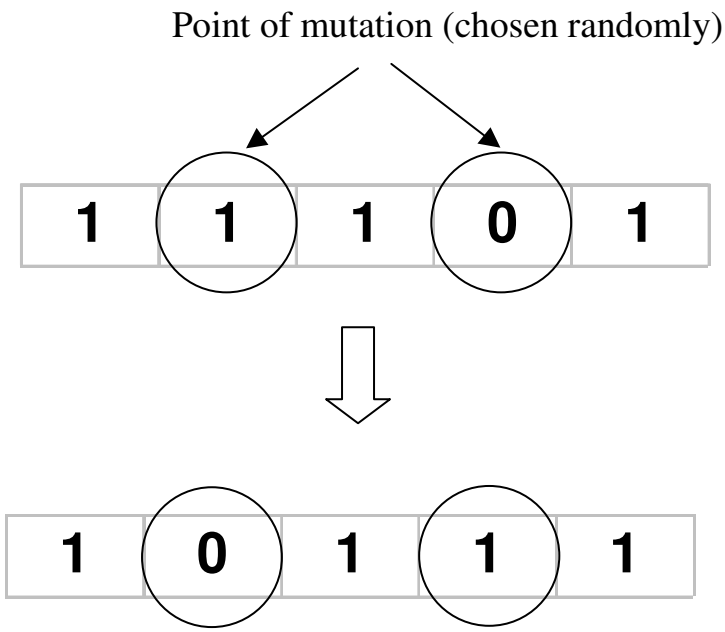


Figure 1.5: Schematic showing mutation operation in a binary string

1.3.4.2 Advantages of using GAs

1. GAs search the solution space randomly yet systematically and hence the objective function need not satisfy the criteria like smoothness, continuity and differentiability. Also, GAs only require objective function values and not its first or second order derivatives (as in gradient based deterministic optimization methods used commonly).
2. Most often GAs reach the global optimum of an objective function through a heuristic search or by computer experimentation.
3. GAs are capable of handling well highly nonlinear and noisy objective functions. In such cases the traditional gradient based methods are known to be inefficient.
4. GAs are amenable to parallel processing. This facilitates the use of parallel computers for the search procedure thereby reducing the optimization time significantly.

1.3.4.3 Applications of GAs

Some of the important chemical engineering applications of the GAs are listed in the table below:

Table 1.4: Chemical engineering applications of GAs

Application	Reference
• Function optimization	DeJong (1975); Bersini and Renders (1994); Petridis et al. (1998)
• Image processing	Fitzpatrick et. al. (1984); Chiu and Liu (1996)
• Traveling salesman problem	Goldberg and Lingle (1985); Grefenstette (1985)
• System identification	Das and Goldberg (1988); Etter et. al. (1982); Kristinsson and Dumount (1988); Smith and DeJong (1981)
• System identification and control	Kristinsson and Dumont (1992); Lennon and Passino (1999)
• Molecular design	Venkatasubramanian et al. (1994)
• Noisy data interpretation	Herdberg (1994)
• Design of neural networks	Marti (1992)
• Design and tuning of controllers	Varsek et al. (1993)
• Design of fuzzy net controllers	Kim et al. (1995)
• Hybrid fuzzy / genetic controllers	Karr and Gentry (1993)
• Chemical flowshop sequencing	Cartwright and Long (1993)
• Model structure and controller structure identification	French et al. (1997)
• Solving non-convex trim loss problem	Ostermark (1999)
• Scheduling problems	Norman and Bean (1999); Syswerda (1991); Nakano (1991)
• Design and operation optimization of an industrial dryer	Hugget et al. (1999)
• Optimal control of microfiltration	Perrot et al. (1998)
• For waste management	Garrard and Fraga (1998)

• Generation of fuzzy rules	Cho et al. (1997); Homaifar and McCormick (1995); Park and Langholz (1994); Nelles (1997)
• Emission control in palm oil mills	Ahmad et al. (2004)
• Optimization of epoxy-polymerization	Deb et. al. (2004); Mitra et. al. (2004); Majumdar et. al. (2005)
• Soft sensor for polymerization reactor	Bhat et al. (2006)
• Control of polymerization reactor	Agarwal et. al. (2007); Badhe et al. (2007)
• Biotechnology and food industry	Masuduzzaman and Rangaiah (2009)
• Feed optimization of catalytic cracking	Tan et. al., (2009)
• Petroleum refining and petrochemicals	Masuduzzaman and Rangaiah (2009)
• Chemical engineering and processing industries	Bhaskar et. al, (2000); Rajesh et. al. (2001); Tarafder et. al., (2005); Tarafder et. al., (2007); Rangaiah (2009)

1.3.5 Differential Evolution:

Price and Storn first introduced the *Differential Evolution* (DE) algorithm (Price and Storn, 1997). Like genetic algorithms (GA), DE also belongs to the class of stochastic population based optimization algorithms. Similar to evolutionary strategies (ES), DE too uses floating-point encoding. Among DE's advantages are its simple structure, ease of use, speed and robustness. The overall structure of differential evolution algorithm is discussed below:

Let us consider minimization of objective function $f(X)$

$$\min\left(f\left(X \right)\right) \tag{1.2}$$

By optimizing the values of its parameters:

$$X = \left(x_1, x_2, ..., x_D \right), \qquad X \in \Re^D \tag{1.3}$$

where X denotes a vector composed of D objective function parameters. For maximization problems, mathematical manipulation is done on objective function in order to convert the

problem to a minimization problem. Usually, the parameters of the objective function are also subject to lower and upper boundary constraints, x^L and x^U respectively:

$$x_j^L \le x_j \le x_j^U \qquad j=1,2,...,D \tag{1.4}$$

As with all evolutionary optimization algorithms, DE operates on a population, P_g of candidate solutions, not just a single solution. These candidate solutions are the individuals of the population. In particular, DE maintains a population of constant size that consists of *NP* real-valued vectors, X_{ig}, where *i* indexes the population and *g* is the generation to which the population belongs.

$$P_g = X_{i,g} \qquad i=1,2,...,NP, \quad g=1,2,...,g_{\max} \tag{1.5}$$

Each vector $X_{i,g}$ contains *D* real parameters (chromosomes of individuals):

$$X_{i,g} = x_{j,i,g} \qquad i=1,2,...,NP, \quad j=1,2,...,D \tag{1.6}$$

The working principle of differential evolution is represented in Fig. 1.6 as shown below.

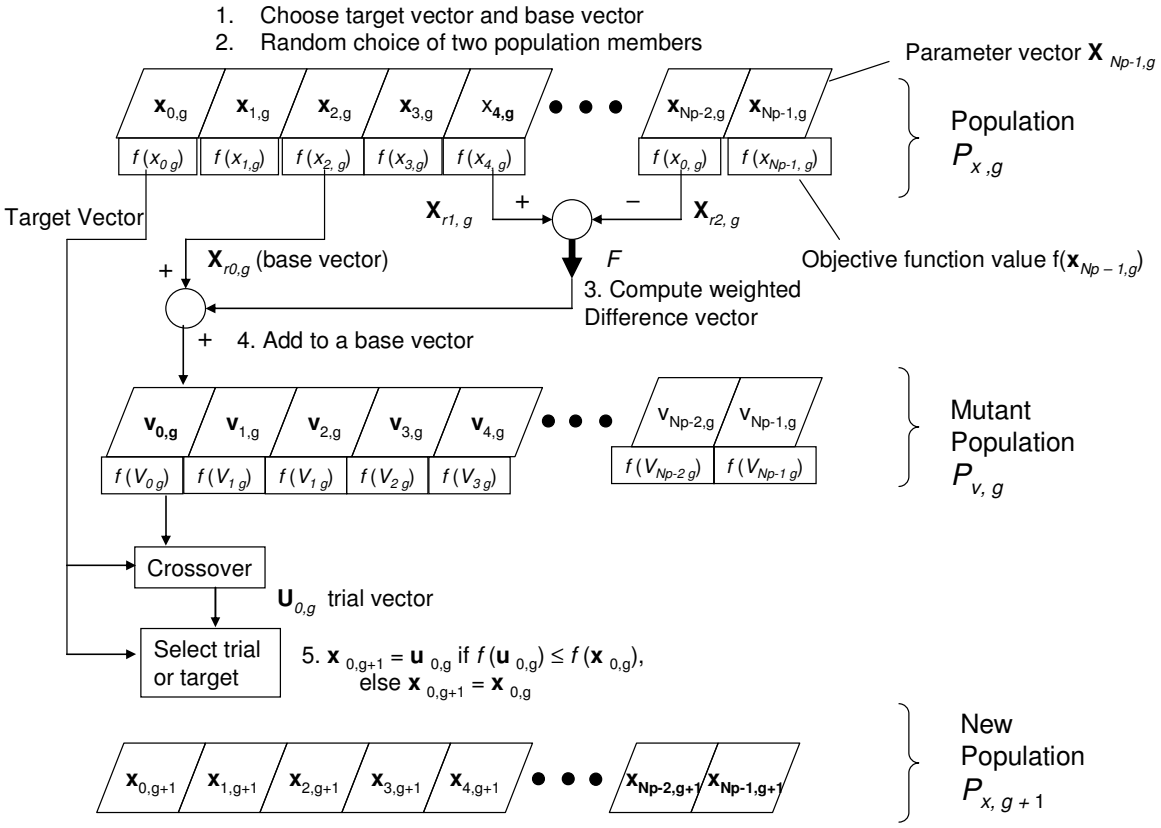


Figure 1.6: Schematic representation of working principle of differential evolution

Initialization

In order to establish a starting point for optimum seeking, the population must be initialized. The preferred characteristics of an initial population are diversity and reasonable levels of the fitness values. Generally initial population is generated randomly though can be chosen carefully based on prior experience. The population size should be such that it represents diversity of the entire search space, and at the same time it should not be too big so that difficult for computation.

Mutation and Crossover

DE's self-referential population reproduction scheme is different from other evolutionary algorithms. From 1st generation onwards, vectors in the current population, P_g are randomly sampled and combined to create candidate vectors for the subsequent generation, P_{g+1}. The population of candidate or trial vectors, $P_g' = U_{i,g} = u_{j,i,g}$, is generated as follows:

$$u_{j,i,g} = \begin{cases} v_{j,i,g} = x_{j,r3,g} + F \times \left(x_{j,r1,g} - x_{j,r2,g} \right) & \text{if } rand_j[0,1] \le CR \vee j = k \\ x_{j,i,g} \quad \text{otherwise} \end{cases} \tag{1.7}$$

where,

$i = 1, 2, \ldots, NP, \quad j = 1, 2, \ldots, D$

$k \in \{1, 2, \ldots, D\}$ random parameter index, chosen once for each i

$r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$ randomly selected, except : $r_1 \neq r_2 \neq r_3 \neq i$

$$CR \in [0,1], \quad F \in (0, 1+] \tag{1.8}$$

The randomly chosen indexes, r_1, r_2 and r_3 are different from each other and also different from the running index, i. New random integer values for r_1, r_2 and r_3 are chosen for each value of the index i, i.e., for each individual. The index k refers to a randomly chosen chromosome, which is used to ensure that each individual trial vector, $U_{i,g}$ differs from its counterpart in the previous generation, $X_{i,g}$ by at least one parameter. A new random integer value is assigned to k prior to the construction of each trial vector, i.e., for each value of index, i. The vector mutation scheme is illustrated in Fig. 1.7.

In DE formulation, F and CR are control parameters like, NP both values remain constant during the search process. F is a real-valued factor in the range (0, 1+] that scales the differential variations. The upper limit of F has been empirically determined. CR is a real-valued crossover constant in the range of [0, 1] which controls the probability that a trial vector parameter will come from the randomly chosen, mutated vector, $V_{i,g}$ instead of from the current vector, $X_{i,.g}$Figure 1.7 gives a pictorial representation of DE's crossover operation. Generally, both F and CR affect convergence velocity and robustness of the search process. Their optimal values are dependent on objective function characteristics and on the population size, NP. Usually, suitable values of F, CR and NP can be found by trial-and-error after a few tests using different values. Reasonable values to start with are $NP = 5 \times D \ldots 30 \times D$, $F = 0.9$ and $CR = 0.9$.

Selection

DE's selection scheme also differs from other evolutionary algorithms. The population for the next generation, P_{g+1}, is selected from the current population, P_g, and the child population, P_g', according to the following rule:

$$X_{i,g+1} = \begin{cases} U_{i,g} & \text{if } f\left(U_{i,g}\right) \le f\left(X_{i,g}\right) \\ X_{i,g} & \text{otherwise} \end{cases} \tag{1.9}$$

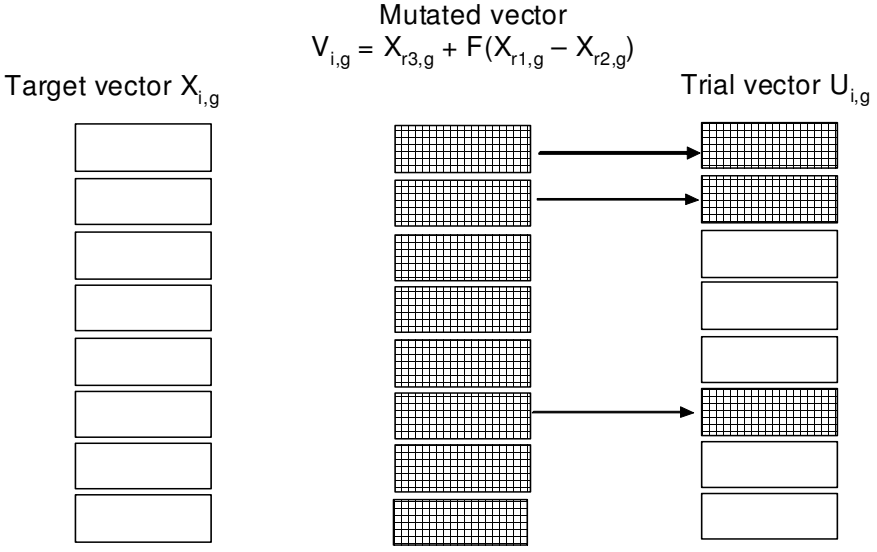


Fig. 1.7: Pictorial representation of crossover operation in differential evolution

Thus, each individual of the temporary population is compared with its own counterpart in the current population. Assuming that the objective function is to be minimized, the vector with lower or equal objective function value wins a place in the next generations population. As a result, all the individuals of the next generation are as good or better than their counterparts in the current generation.

The interesting point concerning DE's selection scheme is that a trial vector is only compared with one individual, not with all the individuals in the current population. Such a scheme can be viewed as a *binomial tournament* selection, as its basic principle is similar to the general tournament selection rule with tournament size equal to 2. Here the binomial tournaments are kept between each trial individual and the corresponding current population member – the winners of a one-to-one competition enter the next stage of the optimization.

1.3.5.1 Advantages of Differential Evolution:

1. DE requires objective function values only and not its derivatives and hence the objective function need not satisfy the criteria like smoothness, continuity and differentiability.
2. DEs reach the global optimum or near global optimum of an objective function through a systematic heuristic search.
3. DE possesses a simple structure, other advantages are ease of use, speed and robustness.

1.3.5.2 Applications of Differential Evolution:

The various applications of DE include: digital filter design (Storn, 1995), fuzzy decision making (Wang et al., 1998), design of pressure vessel (Lampinen and Zelinka, 1999), flow shop scheduling problem (Onwubolu, 2001) etc. A few of DE's important applications related to chemical engineering field are listed in the table given below:

Table 1.5: Important applications of differential evolution related to chemical engineering

Applications	References
• Batch fermentation process	Chiou and Wang, (1999)
• Optimization of fermentation process	Wang et al., (2001)
• Polymerization reactor optimization	Lee et al, (1999)
• Estimation of heat transfer parameters in trickle bed reactor	Babu and Sastry, (1999)
• Optimization of alkylation reaction	Babu and Chaturvedi, (2000)
• Optimization of thermal cracking operation	Babu and Angira, (2001)
• Optimization of low pressure chemical vapor deposition reactors	Lu and Wang, (2001)
• Auto-thermal ammonia synthesis reactor	Babu et al., (2002)
• Optimization of adiabatic styrene reactor	Gujarathi and Babu (2009)
• Optimal design of heat exchangers	Babu and Mohiddin, (1999); Babu and Munawar, (2000); Babu and Munawar, (2007)
• Kinetic parameter estimation	Wang et al. (2001)
• Optimization of nonlinear chemical processes	Babu and Angira (2006), Hang and Wang (2002)
• Adiabatic styrene reactor	Babu et. al. (2005)
• Phase equilibrium and other thermodynamic problems	Srinivas and Rangaiah (2006); Srinivas and Rangaiah (2007)

1.3.6 Principal Component Analysis (PCA):

Principal components analysis (PCA) is a multivariate, statistical technique that can be used to examine data variability. It is frequently applied to datasets which are large, difficult to interpret, and where complex inter-relationships between variables are difficult to identify and visualize. Multivariate techniques can consider a number of factors which control data variability simultaneously and therefore offer significant advantages over univariate techniques, where errors associated with repeated statistical testing can occur (Kozub and MacGregor, 1992). In simple terms PCA is a data reduction technique whereby new variables (*principal components* or *factors*) are calculated from linear combinations of the original variables. The first principal component, or factor, accounts for the greatest variability in the data, and there can be an infinite number of new factors with each accounting for less data variability than the previous (Dong and McAvoy, 1996). Factor loadings are correlation coefficients between the original variables and factors and are frequently used in the literature to investigate those processes which control data variability. Factor scores indicate how strongly individual samples are associated with each of the factors, and thus can be used to investigate similarity between samples, where samples with a similar composition will have similar scores and may therefore have similar contaminant sources and/or behaviour.

PCA has proven to achieve excellent results in feature extraction and data reduction in large datasets (Karhunen and Joutsensalo, 1995). The basic idea of PCA is to reduce the dimensionality of a dataset in which there are a large number of interrelated variables, while the current variation in the dataset is maintained as much as possible. This reduction is accomplished by transforming the original set of variables to a new set of variables that are uncorrelated and ordered by their significance, so that the first few variables retain most of the variation present in all of the original data (Nomikos and MacGregor, 1994). PCA has many applications in image understanding and pattern recognition that includes pattern matching (Smith, 2002), artificial neural networks (Karhunen and Joutsensalo, 1995), speech analysis, visual learning, and active vision (Pinkowski, 1997). In feature recognition, PCA has been extensively used to identify face features. In general, usage of PCA enhances computational efficiency since it is accompanied with the reduced dimension of data (Nomikos and MacGregor, 1994).

1.3.6.1 Applications of PCA:

Principal component analysis is majorly used in chemometrics for dimensionality reduction purposes. Following is the brief list of its important applications to chemical and allied engineering fields.

Table 1.6: Some of the important chemical engineering applications of PCA

Applications	References
• Rapid thermal annealing of semiconductors	Li, Yue, Valle-Cervantes, and Qin (2000)
• Monitoring of a spray dryer	Shao, Jia, Martin, and Morris (1999)
• Process Monitoring	Nomikos and MacGregor, (1994); Li et. al. (2000); Misra et. al. (2002)
• Monitoring of polymerization reactor	Kozub and MacGregor (1992); Liu (2007)
• Model identification	Kramer (1991); Dong and McAvoy, (1996); Yoon and MacGregor, (2000); Narasimhan and Shah (2008)
• Improved process understanding	Kosanovich et al. (1996); Wentzell et. al. (1997)
• Disturbance detection and isolation	Ku et. al. (1995); Yoon and MacGregor, (2000); lennox and Sandoz (2002), Venkatasubramanian et. al. (2003)
• Product and process monitoring	Karhunen and Joutsensalo (1995); Kourti and MacGregor (1996)
• Statistical control for batch processes	Lennox et al. (2000)
• Environmental related applications	Zitko (1994); Sanchiz et. al. (1998); Shaw (2003); Komulainen (2004); Ellis et. al. (2006); Reid and Spancer (2009)

1.3.7 Partial Least Squares (PLS):

In the linear PLS model, a causal relationship is assumed to exist between two variable blocks. The data is divided into these groups of variables, into *x*-variables (explanatory) and *y*-variables (response). In PLS modeling *x* and *y* sets are modelled separately with a principal component analysis (PCA) type of models. In this procedure the original data sets are projected onto principal component axes. Each of the principal components captures as much as possible the variance which has not been explained by the former components. In this manner the original data set is projected onto a new co-ordinate space where the objects are described with scores and the variables by loadings. The principal components are orthogonal and linear. In comparison with PCA the PLS solution is rotated so that the covariance (or correlation if *X* and *Y* is auto-scaled) between columns of *X* and *Y* is maximized.

Besides the reduction of dimensions, the PLS also enables modeling of collinear variables. In PLS the collinearity between variables represents a stabilizing advantage rather than a problem. Although PLS is a bilinear regression method, it has been shown that with PLS some non-linear relationships can be modelled. In this case study, the purpose of variable reduction was to identify the most important descriptor variables to ensure proper understanding of the phenomena. The variable selection was based on maximal squared covariance between *X* and *Y*. Only the variables with significant covariance were selected into the final PLS model. Another way to select important variables used was the amount of variance used by the model.

1.3.7.1 Applications of PLS:

Partial least squares are used to map input – output data. Following are some of the important PLS applications in chemical engineering and processing industries

Table 1.7: A sample of PLS applications in chemical and allied engineering

Applications	References
• Chemometrics	Lindberg et al., (1983); Wold et al., (1984); Geladi, Kowalski, (1986); Fuller et al., (1988); Haaland and Thomas, (1988); Fuller et. al. (1988a,b); Helland et. al. (1992); Qin

	and McAvoy (1996); Wold et. al. (2001)
• Process modeling	Wold et. al. (1989); Piovoso and Owens, (1991)
• Dynamic modeling	Ricker, (1988); Wise and Ricker, (1990); MacGregor et al., (1991)
• Process Monitoring	MacGregor et al., (1991); Negiz and Cinar, (1992); Nomikos and MacGregor, (1994); Nomikos and MacGregor, (1995); Kourti and MacGregor, (1996)
• Online quality improvement for binary distillation column	Chen, McAvoy, and Pivoso (1998)
• System identification	Pearson et. al. (1992); Qin (1998)
• Mineral floatation process	Dayal and MacGregor (1997)
• Quality control	MacGregor et. al., (1991); Lennox et. al. (2000);
• Online application to dearomatization process	Komulainen et al. (2004)
• Product composition profile in batch distillation	Zamprogna et al. (2004)

1.4 OUTLINE OF THE THESIS

The aim of the thesis is to design and develop various AI-based formalisms for chemical and bio-chemical engineering problems. All the symbols and references of are provided at the end of each chapter separately. Below the main contributions of each of these chapters are summarized and put into context of the thesis.

Chapter 2 "***Hybrid Process Modeling and Optimization Strategies Integrating Neural Networks/Support Vector Regression and Genetic Algorithms: Study of Benzene Isopropylation on HBeta Catalyst***". This chapter presents a comparative study of two artificial intelligence based hybrid process modeling and optimization strategies,

namely ANN-GA and SVR-GA, for modeling and optimization of benzene isopropylation on Hbeta catalytic process. In the ANN-GA approach, an artificial neural network model is constructed for correlating process data comprising values of operating and output variables. Next, model inputs describing process operating variables are optimized using genetic algorithms (GAs) with a view to maximize the process performance. In the second hybrid methodology, a novel machine learning formalism, namely *support vector regression* (SVR), has been utilized for developing process models and the input space of these models is optimized again using GAs. The SVR-GA is a new strategy for chemical process modeling and optimization. The major advantage of the two hybrid strategies is that modeling and optimization can be conducted exclusively from the historic process data wherein the detailed knowledge of process phenomenology (reaction mechanism, rate constants, etc.) is not required. Using ANN-GA and SVR-GA strategies, a number of sets of optimized operating conditions leading to maximized yield and selectivity of the benzene isopropylation reaction product, namely cumene, were obtained. The optimized solutions when verified experimentally resulted in a significant improvement in the cumene yield and selectivity.

Chapter 3 "***Modeling and Monitoring of Batch Processes Using Principal Component Analysis (PCA) Assisted Generalized Regression Neural Networks (GRNN)***". Multivariate statistical methods namely, principal component analysis (PCA) and partial least squares (PLS), which perform dimensionality reduction and regression, respectively, are commonly used in batch process modeling and monitoring. While PCA is used to monitor whether process input variables are behaving normally, the PLS is used for predicting the process output variables. A significant drawback of the PLS is that it is a linear regression formalism and thus makes poor predictions when relationships between process inputs and outputs are nonlinear. In this chapter, a formalism integrating PCA and generalized regression neural networks (GRNNs) is introduced for batch process modeling and monitoring. The advantages of this PCA-GRNN hybrid methodology are: (i) process outputs can be predicted accurately even when input-output relationship is nonlinear, and (ii) unlike other commonly used artificial neural network paradigms (such as multilayer perceptron), training of GRNN

is a one-step procedure, which helps in faster development of nonlinear input-output models. The effectiveness of the PCA-GRNN strategy has been successfully demonstrated by conducting two case studies involving penicillin production and protein synthesis.

Chapter 4 "***Forecasting Batch Process Dynamics using Generalized Regression Neural Networks***". Process monitoring plays an important role in quality control and process safety. In this chapter, we have proposed a Generalized Regression Neural Network (GRNN) based strategy for monitoring and modeling of batch processes. The methodology has been successfully utilized to forecast the dynamics of a fed batch bio-reactor for ethanol production. We have presented a GRNN-based process monitoring and modeling strategy to forecast dynamics of a process. The strategy has been shown to provide excellent results when applied to a fed batch bio-reactor for production of ethanol.

Chapter 5 "***Hybrid PCA-Fuzzy Neural Network Formalism for Batch Process Monitoring***". Batch processes due to their complex nonlinear nature, are difficult to model phenomenologically and, therefore, multivariate statistical methods namely, principal component analysis (PCA) and partial least squares (PLS) are commonly used to model and monitor batch processes. In particular, PCA is used to monitor whether process operating (predictor) variables are behaving normally and PLS is used for predicting the process output (response) variables. A significant drawback of the PLS is that it is a linear regression formalism and thus it makes poor predictions when relationships between process predictor and response are nonlinear. To overcome this difficulty, a formalism integrating PCA and fuzzy neural networks (FNNs) has been developed for modeling and monitoring of batch processes. The proposed methodology is generic in nature and has been successfully validated for modeling and monitoring of the protein synthesis process.

Chapter 6 "***Artificial Neural Network Assisted Genetic Algorithm Based Formalism for Optimization of Batch Polymerization Reactor***". Polymerization reactions are often carried out in batch or semi-batch mode. Optimization of these processes is necessary for cost minimization as well as from environmental and safety considerations. A novel methodology integrating artificial neural networks (ANNs) and genetic

algorithms (GAs) is utilized for optimizing a free–radical batch solution polymerization of methyl metahcrylate (MMA) to produce poly-MMA. The study is concerned with the development of optimal temperature policy and the initial initiator concentration to reduce the residual monomer concentration to the desired level and simultaneously produce the polymer with the desired number average molecular weight.

Chapter 7 "***Controlling Nonlinear Dynamics of a Non-isothermal Fluidized Bed Reactor***". Nonisothermal fluidized bed catalytic reactors in certain parameter regions exhibit complex nonlinear dynamical behavior such as multiplicity, oscillations and chaos. For the consecutive exothermic reaction $A \rightarrow B \rightarrow C$ with B as the desired component, maximization of yield takes place if the reactor is operated at the middle unstable steady state (USS). A novel gain scheduling based nonlinear control methodology is demonstrated here to control the reactor exactly at the unstable steady state for achieving product maximization. Here, optimal value of the sole tuning parameter is obtained using genetic algorithm (GA).

Chapter 8 ***"Artificial Neural Network Assisted Novel Optimization Strategy for Process Parameters and Tolerances"*** This chapter presents robust process optimization strategies integrating artificial intelligence based modeling technique, i.e., *artificial neural networks* (ANN) with a novel AI-based optimization formalism namely, *differential evolution* (DE). Firstly an ANN based process model was developed from the process input-output data and subsequently design and operating variables were optimized using the DE formalism. The efficacy of the methodology was demonstrated using a case study involving a non-isothermal continuous stirred tank reactor (CSTR). The task addressed in this study is complex since it involves the issue of optimal tolerance design while performing the conventional parameter design. Results obtained using the ANN-DE method has been compared with those published earlier using the ANN-GA hybrid formalism.

Chapter 9 ***"Conclusion and Future Scope"*** This is the final chapter, which concludes the thesis providing concise summary of results obtained and suggestions for the future research.

1.5 REFERENCES

1. Abilov, A. G, Zeybek, Z., Tuzunalp, O. and Telatar, Z. "Fuzzy Temperature Control of Industrial Refineries Furnaces Through Combined Feedforward / Feedback Multivariable Cascade Systems", ***Chem. Engg. & Proces.***, **41**, 87 – 98 (2002).

2. Agarwal, N., Rangaiah, G. P., Ray, A. K. and Gupta, S. K. "Design Stage Optimization of an Industrial Low-Density Polyethylene Tubular Reactor for Multiple Objectives Using NSGA-II and its Jumping Gene Adaptations", ***Chem. Engg. Sci.***, **62**, 2346 – 2365 (2007).

3. Ahmad, A. L., Azid, I. A., Yusof, A. R. and Sheethamaru, K. N., "Emission Control in Palm Oil Mills Using Artificial Neural Network and Genetic Algorithm", ***Comp. & Chem. Engg.***, **28,** 2709 - 2715 (2004).

4. Aldrich, C., and S. van Deventer, "Comparison of Different Artificial Neural Nets for the Detection and Location of Gross Errors in Process Systems", ***Ind. Engg. Chem. Res.*, 34**, 216 - 224 (1995).

5. Alfafara, C.G., Miura, K., Shimizu, H. Shioya, S., Suga, K. and Suzuki, K. "Fuzzy Control of Ethanol Concentration and its Application to Maximum Glutathione Production in Yeast-batch Culture" ***Biotechnol.& Bioengg.,*** **41**, 493 - 501 (1993).

6. Ali, E. and Al-humaizi, K "Advanced Control Algorithms for a Chemical Polymerization Process", ***ASCE Jour.***, **5**, 35 – 53 (2005).

7. Alessandri, A. and Parisini, T. "Nonlinear Modelling of Complex Large-Scale Plants Using Neural Networks and Stochastic Approximation", ***IEEE Trans. Syst., Man, Cybernatics – A*** **27**, 750 - 757 (1997).

8. Altinten, A., Erdogan, S., Hapoglu, H., Aliev, F. and Alpbaz, M. "Application of Fuzzy Control Method with Genetic Algorithm to a Polymerization Reactor at Constant Set Point", ***Chem. Engg. Res. & Des.***, **84**, 1012 – 1018 (2006).

9. Antunes, A.J.B., Pereira, J.A.F.R and Fileti, A.M.F, "Fuzzy Control of a PMMA Batch Reactor: Development and Experimental Testing", ***Comp. & Chem. Engg.***, **30**, 268 - 276 (2005).

10. Babu, B. V. and Sastry, K. K. N., "Estimation of Heat Transfer Parameters in a Trickle Bed Reactor using Differential Evolution and Orthogonal Collocation", ***Comp. & Chem. Engg.***, **23**, 327 – 339 (1999).

11. Babu, B. V. and Mohiddin, S. B., "Autothermal Design of Heat Exchangers using Artificial Intelligence based Optimization", Proceedings of CHEMCON – 1999, Punjab University, Chandigarh, December 20 – 23, (1999).

12. Babu, B. V. and Munawar, S. A., "Differential Evolution for the Optimal Design of Heat Exchangers", Proceedings of All India Seminar on Chemical Engineering Progress on Resource Development : A Vision 2010 and Beyond, Bhubaneshwar, March 13, (2000), available at http://bvbabu.50megs.com/custom.html/#28.

13. Babu, B. V. and Chaturvedi, G. "Evolutionary Computation Strategy for Optimization of an Alkylation Reaction", Proceedings of CHEMCON – 2000, Science City, Calcutta, December 18 – 21 (2000).

14. Babu, B. V. and Angira, R. "Optimization of Thermal Cracker Operation using Differential Evolution", Proceedings of CHEMCON – 2001, CLRI, Chennai, December 19 – 22 (2001).

15. Babu, B. V. and Munawar, S. V., "Optimal Design of Shell-and-Tube Heat Exchangers using Different Strategies of Differential Evolution" PreJournal.com – The Faculty Lounge, Article No. 003873, also available at http://bvbabu.50megs.com/custom.html/#35 (2003).

16. Babu, B. V., Angira, R. and Nilekar, A., "Differential Evolution for Optimal Design of an Auto-Thermal Ammonia Synthesis Reactor", in ***New Optimization Techniques in Engineering***, Ed. Onwubolu, G. C. and Babu, B. V., Springer – Verlag, Berlin Heidelberg (2004).

17. Badhe, Y.P., Lonari, J., Tambe S.S. and Kulkarni, B.D. "Improve Polyethylene Process Control and Product Quality", ***Hydrocarb. Proc.***, 53 – 60 (2007).

18. Baldi, P. "Gradient Descent Learning Algorithms Overview: A General Dynamical Systems Perspective", ***IEEE Transactions on Neural Networks***, **6**, 182 - 195 (1995).

19. Banga, J. R., Alonso, A. A and. Singh, R. P. "Stochastic Dynamic Optimization of Batch and Semicontinuous Bioprocesses", ***Biotech. Progr.***, **13**, 326 - 335, (1997).

20. Bajpai, R. K. and Reuss, M. "Evaluation of Feeding Strategies in Carbon Regulated Secondary Metabolic Production Through Mathematical Modeling", ***Biotech. & Bioengg***., **23**, 714 - 738, (1981).

21. Balsa-Canto, E., Banga, J. R., Alonso, A. A. and Vassiliadis, V. S. "Dynamic Optimization of Chemical and Biochemical Processes Using Restricted Second Order Information", ***Comp. & Chem Engg.***, **25**, 539 - 546 (2001).

22. Barada, S. and Singh, S. "Generating Optimal Adaptive Fuzzy-Neural Models of Dynamical Systems with Applications to Control", ***IEEE Trans. On Sys.,Man, and Cybernetics – Part C: Appl. and Reviews***, **28**, 371 - 391 (1998).

23. Baratti, R., A. Servida, and G. Vacca, *in* ***Proc. 1st Natl. Conf. on Chaos and Fractals in Chem. Engg.,*** Eds. G. Biardi, M. Giona, and A. Giona, 275, World Scientific, Singapore (1994).

24. Baratti, R., Vacca, G. and Servida, A. "Neural Network Modeling of Distillation Control", ***Hydrocarb. Process***., **74** (6), 35-38 (1995).

25. Bersini, H., and B. Renders, *in* ***Proc. of the IEEE Int. Symp. on Evoln. Comput***., 312, (1994).

26. Bezdek, J., and Pal, S. K. ***Fuzzy Models for Pattern Recognition***, IEEE Press, Piscataway, New York (1993).

27. Bhagat, P. "An Introduction to Neural Nets", ***Chem. Engg. Prog.***, **86**, 55 -61 (1990).

28. Baratti, R., Vacca, G. and Servida, A. "Neural Network Modeling of Distillation Columns", ***Hydrocarb. Process.***, **107** (1998).

29. Bhaskar, V., Gupta, S. K. and Ray, A. K., "Applications of Multi-objective Optimization in Chemical Engineering", ***Rev. in Chem. Engg.***, **16**, 1 – 54 (2000).

30. Bhat, N., P. Jr. Minderman, T. McAvoy, and N. S. Wang, "Modeling Chemical Processes via Neural Computation", ***IEEE Contr. Sys. Mag.***, **10**, 24 - 30 (1990).

31. Bhat, S.A., Saraf, D.N., Gupta, S. and. Gupta, S.A "Use of Agitator Power as a Soft Sensor for Bulk Free-radical Polymerization of Methyl Methacrylate in Batch Reactors", ***Ind. Engg. Chem. Res.***, **45**, 4243 – 4255 (2006).

32. Bishop, C. M., "Neural Networks and Their Applications", ***Rev. Sci. Instru.***, **65**, 1803 - 1834 (1994).

33. Blanz V., Scholkopf B., Bulthoff H., Burges C., Vapnik V., and Vetter T. "Comparison of View-based Object Recognition Algorithms Using Realistic 3D Models. In: von der Malsburg C., von Seelen W., Vorbr¨uggen J.C., and Sendhoff B. (Eds.), ***Artificial Neural Networks ICANN'96***, Berlin. Springer Lecture Notes in Computer Science, Vol. 1112, pp. 251–256 (1996).

34. Blaesi, J., and B. Jensen, ***INTECH***, 34, Dec (1992).

35. Boruvka, L., Vecek, O., Jenlika, "Principal Component Analysis as a Tool to Indicate the Origin of Potentially Toxic Elements in Soil", ***Geoderma*** **128**, 289 – 300 (2005).

36. Bugnon, B., K. Stoffel, and M. Widmer, "FUN: A Dynamic Method for Scheduling Problems", ***Euro. J. Opern. Res., 83***, 271 – 282 (1995).

37. Cameron, I. T., Wang, F. Y., Imanuel, C. D., and Stepanek, F. "Process Systems Modeling and Application in Granulation: A Review", ***Chem. Engg. Sci.***, **60**, 3723 – 3750 (2005).

38. Carpenter, G. A. and Grossberg S. "The ART of Adaptive Pattern Recognition by a Self-organizing Neural Network", ***Computer***, **21**, 77 - 88 (1988).

39. Cartwright, H. M. and Long, R. A. "Simultaneous Optimization of Chemical Flowshop Sequencing and Topology Using Genetic Algorithms", ***Ind. Eng. Chem. Res.***, **32**, 2706 (1993).

40. Causa, J., Karer, G., Nunez, A., Saez, D., Skrjanc, I. and Zupancic, B. "Hybrid Fuzzy Predictive Control on Genetic Algorithms for the Temperature Control of a Batch Reactor", ***Comp. & Chem. Engg.***, **32**, 3254 - 3263 (2008).

41. Cauwenberghs, G., *Analog VLSI Autonomous Systems for Learning and Optimization*, Ph.D. Thesis, California Institute of Technology (1994).

42. Chan, K., S. Leong, and G. Lin, ***Artf. Intell. Engng***., **9**, 167 (1995).

43. Chang, S-Y. and Chang, C-T. "Fuzzy Diagnosis Method for Control Systems with Coupled Feedforward and Feedback Loops", ***Chem. Engg. Sci.***, **58**, 3105 – 3128 (2006).

44. Cheng, C. S. "A Neural Network Approach for the Analysis of Control Chart Patterns" ***Int. J. of Prod. Res***., **35**, 667 - 697 (1997).

45. Chen, G., McAvoy, T. J. and Piovoso, M. J. "A Multivariate Statistical Controller for Online Quality Improvement", ***Jour. of Proc. Contr.***, **8**, 139 – 149 (1998).

46. Chen, Y-C. and Teng, C-C. "A Model Reference Control Structure Using Fuzzy Neural Network", ***Fuzzy Sets and Syst.***, **73**, 291-312, (1995).

47. Cheng, C. S., "A Multi-layer Neural Network Model for Detecting Changes in the Process Mean" ***Comp. & Ind. Engg***., **28**, 51 - 61 (1995).

48. Chitra, S. P. "Use Neural Networks for Problem Solving", ***Chem. Engg. Prog.***, **89** (4), 44 - 52 (1993).

49. Chiu, C-C. and Liu, P-T. "Image Reconstruction of a Perfectly Conducting Cylinder by the Genetic Algorithm", ***IEE Proc. Microw. Antennas Propag***., **143**, 249 - 253 (1996).

50. Cho, H-J., Cho, K-B. and Wang, B-H. "Fuzzy-PID Hybrid Control: Automatic Rule Generation Using Genetic algorithms**",** ***Fuzzy Sets and Systems***, **92**, 305 - 316 (1997).

51. Chiou, J. P. and Wang, F. S., "Hybrid Method of Evolutionary Algorithms for Static and Dynamic Optimization Problems with Applications to Fed-batch Fermentation Process", ***Comp. & Chem. Engg.***, **23**, 1277 – 1291 (1999).

52. Constantinides, A. ***Applied Numerical Methods with Personal Computers***, McGraw Hill, Singapore (1987).

53. Cook D. F. and Chiu, C. C. "Using Radial Basis Function Neural Networks to Recognize Shifts in Process Parameters", ***IIE Trans***., **30**, 227 - 234 (1998).

54. Cote, M., Grandjean, B. P. A., Lessard, P. and Thibault, J. "Dynamic Modelling of the Activated Sludge Process: Improving Prediction Using Neural Networks", ***Water Res***., **29**, 995 - 1004 (1995).

55. D'Antone, I., ***Microproc. and Microcomp***., **40**, 305 (1994).

56. Das, R. and Goldberg, D. E. *in* ***Proc. 19th annual Pittsburgh Conf. Modeling and Simulation***, (1988).

57. Dash, S., Rengaswamy, R. and Venkatsubramanian, V, "Fuzzy-Logic Based Trend Classification for Fault Diagnosis of Chemical Processes", ***Comp. & Chem. Engg***, **27**, 347 - 362 (2003).

58. Dayal, B. S. and MacGregor, J. F. "Recursive Exponentially Weighted PLS and its Applications to Adaptive Control and Prediction, ***Jour. of Proc. Contr.***, **7**, 169–179 (1997).

59. Deb, K., ***Optimization for Engineering Design: Algorithms and Examples***, Prentice Hall, New Delhi, (1995).

60. Deb, K., Mitra, K., Dewri, R and Majumdar, S. "Towards a Better Understanding of the Epoxy Polymerization Process Using Multi-objective Evolutionary Computation", ***Chem. Engg. Sci.***, **59**, 4261 – 4277 (2004).

61. De Jong, K. A., "An Analysis of the Behavior of a Class of Genetic Adaptive Systems," Ph.D. dissertation (CCS), Univ. Mich., Ann Arbor, MI (1975).

62. De la Garza J. M., K. G. Rouhana, "Neural Networks Versus Parameter-based Applications in Cost Estimating", ***Cost Engng***., **37**, 14 - 18 (1995).

63. Dennis, J. E., and R. B. Schnabel, *in* ***Optimization Handbooks in OR & MS***, vol. 1, Eds. G. L. Nemhauser, A. H. G. Kan Rinnooy, and M. J. Todd, (1989).

64. Desai, K., Badhe, Y., Tambe, S. S. and Kulkarni, B. D. "Soft Sensor Development for Fed-batch Bioreactors Using Support Vector Regression" ***Biochem. Eng. J.*** **27**, 225 – 232 (2006).

65. Di Massimo, C., G. Montague, M. Willis, M. Tham, and A. Morris, "Towards Improved Penicillin Fermentation via Artificial Neural Network", ***Comp. & Chem. Engg***., **16**, 283 – 291 (1992).

66. Dong, D. and McAvoy, T. J. "Nonlinear Principal Component Analysis Based on Principal Curves and Neural Networks", ***Comp. & Chem. Engg.***, **20**, 65 – 78 (1996).

67. Drucker H., Burges C.J.C., Kaufman L., Smola A., and Vapnik V. "Support Vector Regression Machines. In: Mozer M.C., Jordan M.I., and Petsche T. (Eds.), ***Advances in Neural Information Processing Systems*** 9, MIT Press, Cambridge, MA, pp. 155–161 (1997).

68. Edgar, T. F., and Himmelblau, D. M. ***Optimization of Chemical Processes***, McGraw-Hill, Singapore (1989).

69. Ellis, R.N., Kroonenberg, P.M., Harch, B.D., and Basford, K.E. "Non-linear Principal Components Analysis: an Alternative Method for Finding Patterns in Environmental Data", ***Environmetrics***, **17**, 1 – 11 (2006).

70. Etter, D. M., M. J. Hicks, and K. H. Cho, *in* ***Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing***, vol. 2, 635 (1982).

71. Freeman, J., and Skapura, D. ***Neural Networks: Algorithms, Applications and Programming Techniques***, Addision-Wesley, Reading, MA (1991).

72. French, I. G., C. S. Cox, and C. K. S. Ho, "Genetic Algorithms in Model Structure and Controller Structure Identification", *in* ***Proc. Instn. Mech. Engrs.***, vol. 211, part I, 333 - 343 (1997).

73. Fitzpatrick, J. M., Grefenstette, J. J. and Van Gucht, D. "Image Registration by Genetic Search" *in* ***Proc. IEEE Southeastcon '84***, 460 – 464 (1984).

74. Fuller, M. P., Ritter, G. L. and Draper, C. S. "Partial Least - Squares Quantitative Analysis of Infrared Spectroscopic Data. Part I: Algorithm Implementation; Part II: Application to Detergent Analysis", ***Appl. Spectrosc.*** **42**, 217 – 236 (1988).

75. Galluzzo, M. and Cosenza, B. "Control of Biodegradation of Mixed Wastes in a Continuous Bioreactor by a Type-2 Fuzzy Controller", ***Comp. & Chem. Engg.***, **33**, 1475 – 1483 (2009).

76. Gan, R., and Yang, D. ***Comp. & Ind. Engg***., **27**, 437 (1994).

77. Gandhi, A. B., Joshi, J. B., Jayaraman, V. K. and Kulkarni, B. D. "Development of Support Vector Regression (SVR)-Based Correlation for Prediction of Overall

Gas Hold-up in Bubble Column Reactors for Various Gas-Liquid Systems", ***Chem. Eng. Sci***,. **62**, 7078 – 7092 (2007).

78. Gandhi, A. B., Joshi, J. B., Kulkarni, A. A., Jayaraman, V. K. and Kulkarni, B. D. "SVR-Based Prediction of Point Gas Hold-up for Bubble Column Reactor Through Recurrence Quantification Analysis of LDA Time-series", ***Inter. Jour. of Multiph. Flow,*** **34** (12), 1099 - 1107 (2008).

79. Garrard, A., and E. S. Fraga, "Mass Exchange Network Synthesis Using Genetic Algorithms", ***Comp. & Chem. Engg***., **22**, 1837 - 1850 (1998).

80. Geladi, P., and Kowalski, B. R. "Partial Least-Squares Regression: A Tutorial," ***Anal. Chim. Acta***, **185**, 1-17, (1986).

81. Ghasem, N.M. "Design of a Fuzzy Logic Controller for Regulating the Temperature in Industrial Polyethylene Fluidized Bed Reactor", ***Chem. Engg. Res. & Des.***, **84**, 97 – 106 (2006).

82. Goldberg, D. E., ***Genetic Algorithms in Search, Optimization and Machine Learning***, Addison-Wesley, NY (1989).

83. Goldberg, D. E., and Lingle, Jr. R. "Alleles, Loci and Travelling Salesman Problem" *in* ***Proc. Int. Conf. GAs and Their Appl.***, 154 - 159 (1985).

84. Grefenstette, J. J., Gopal, R., Rosmaita, B. and Van Gucht, D. *in* ***Proc. Int. Conf. GAs and Their Appl.***, 160 (1985).

85. Gupta, P. P., Merchant, S. S., Bhat, A. U., Gandhi, A. B., Bhagwat, S. S., Joshi, J. B., Jayaraman, V. K and Kulkarni, B. D. "Development of Correlations for Overall Gas Hold-up, Volumetric Mass Transfer Coefficient, and Effective Interfacial Area in Bubble Column Reactors using Hybrid Genetic Algorithm – Support Vector Regression Technique: Viscous Newtonian and Non-Newtonian Liquids", ***Ind. Eng. Chem. Res***., **48**, 9631 – 9654 (2009).

86. Gujarathi, A. M. and Babu, B. V., "Optimization of Adiabatic Styrene Reactor : A Hybrid Multiobjective Differential Evolution Approach", ***Ind. Engg. Chem. Res.,*** **48**, 11115 – 11132 (2009).

87. Haaland, D. M. and Thomas, E. V. "Partial Least Squares Methods for Spectral Analysis: 1. Relation to Other Quantitative Calibration Methods and the

Extraction of Qualitative Information. 2. Application to Simulated and Glass Spectral Data", ***Anal. Chem.***, **60**, 1193 – 1208 (1988).

88. Henson, M. A., "Nonlinear Model Predictive Control: Current Status and Future Directions", ***Comp. Chem. Eng.***, **23**, 187 - 202 (1998).

89. Helland, K., Berntsen, H. E., Borgen, O. S. and Martens, H. "Recursive Algorithm for Partial Least Squares Regression", ***Chemo. & Intel. Lab. Syst.***, **14**, 129 – 137 (1992).

90. Himmelblau, D.M. "Applications of Artificial Neural Networks in Chemical Engineering", ***Kor. J. Chem. Eng***,. **17**, 373 – 392 (2000).

91. Holland, J. H., ***Adaptation in Neural and Artificial Systems***, University of Michigan Press, Ann Arbor (1975).

92. Homaifar, A., and McCormick, E. "Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms", ***IEEE Trans. on Fuzzy Systems***, **3**, 129 - 139 (1995).

93. Hoskins, J., K. Kaliyur, and D. Himmelblau, "Fault Diagnosis in Complex Chemical Plants Using Artificial Neural Networks" ***AIChE Jour.***, **37**, 137 - 141 (1991).

94. Hugget, A., P. Sebastian, and J-P. Nadeau, "Global Optimisation of a Dryer by Using Neural Networks and Genetic Algorithms" ***AIChE Jour.***, **45**, 1227 – 1238 (1999).

95. Istadi, N., Amin, A. S. "Hybrid Artificial Neural Network-Genetic Algorithm Technique for Modeling and Optimization of Plasma Reactor", ***Ind. Eng. Chem. Res.***, **45**, 6655 – 65562 (2006).

96. Ivanciuc, O. "Applications of Support Vector Machines in Chemistry", ***Rev. in Comput. Chem.***, **23**, 291 - 400 (2007).

97. Jain, A. S. and Meeran, S. "Job-shop Scheduling Using Neural Networks' ***Int. J. Prod. Res***., **36**, 1249 – 1272 (1998).

98. Jula, P., Houshyar, A. Severance, F. L. and Sawhney, A. ***Comp. & Ind. Engg***., **31**, 417 (1996).

99. Jayaraman, V. K., Kulkarni, B. D., Gupta, K., Rajesh, J. and Kusumaker, H. S. "Dynamic Optimization of Fed-batch Bioreactors Using the Ant Algorithm", ***Biotech. Progr.***, **17**, 81 - 88, (2001).

100. Karhunen J. and Joutsensalo J. "Generalizations of Principal Component Analysis, Optimization Problems and Neural Networks", ***Neural Networks***, **8**, 549 – 562 (1995).

101. Kavaklioglu, K., and Upadhyaya, B. "Monitoring Feedwater Flow Rate and Component by means of Artificial Neural Networks" ***Nucl. Tech.***, **107**, 112 – 123 (1994).

102. Kavuri, S. and Venkatasubramaniun, V. "Neural Network Decomposition Strategies for Process Fault Diagnosis" ***Comp. & Chem. Engg.***, **16**, 299 – 312 (1992).

103. Kim, J. Moon, Y. and Zeigler, P. "Designing Fuzzy Net Controllers Using Genetic Algorithms" ***IEEE Contrl. Sys.***, 66 – 72 (1995).

104. Karr, C. L. and Gentry, E. J. "Fuzzy Control of pH Using Genetic Algorithms" ***IEEE Trans. on Fuzzy Syst.***, **1**, 46 – 53 (1993).

105. Kohonen, T., ***Self-organization and Associative Memory*** (3rd Edn.), Springer, Verlag, Berlin (1988).

106. Kosanovich, K. A., Dahl, K. S. and Piovoso, M. J. "Improved Process Understanding Using Multi-way Principal Component Analysis," ***Ind. Engg. Chem. Res.***, **35**, 138 – 146 (1996).

107. Kourti, T. and MacGregor, J. F. "Multivariate SPC Methods for Process and Product Monitoring," ***Jour. of Qual. Tech.***, **28**, 409 – 428, (1996).

108. Koza, J. R. ***Genetic Programming II***, MIT Press, Cambridge, Massachusettes (1994).

109. Kozma, R. and Nabeshima, K. ***Ann. Nucl. Ener.***, **7**, 483 (1995).

110. Kozub, D. J. and MacGregor, J. F. "State Estimation for Semi-batch Polymerization Reactors," ***Chem. Engg. Sci.***, **47**, 1047 – 1062 (1992).

111. Kramer, M. A. "Nonlinear Principal Component Analysis Using Autoassociative Neural Networks" ***AIChE Jour.***, **37**, 233 – 243 (1991).

112. Kristinsson, K. and Dumont, G. A. "System Identification and Control Using Genetic Algorithms" ***IEEE Trans. on Sys., Man and Cybern.***, **22**, 1033 - 1042 (1992).

113. Ku, W., Storer, R. H. and Georgakis, C. "Disturbance Detection and Isolation by Dynamic Principal Component Analysis", ***Chemo. & Intel. Lab. Syst.***, **30**, 179 – 196 (1995).

114. Kulkarni, B. D., Tambe, S. S., Dahule, R. K. and Yadavalli, V. K. "Consider Genetic Programming for Process Identification", ***Hydrocarb. Process.*** **78** (7), 89 - 97 (1999).

115. Kumar, V. and Schuhmacher, M. "Fuzzy Uncertainty Analysis in System Modelling", ***Comp. Aided Chem. Engg.***, **20**, 391 – 396 (2005).

116. Kvasnicka, V., Sklenak, S. and Pospichal, J. "Neural Network Classification of Inductive and Resonance Effects of Substituents", ***J. Am. Chem. Soc.***, **115**, 1495 – 1503 (1993).

117. Lampinen, J. and Zelinka, I., "Mechanical Engineering Design Optimization by Differential Evolution", ***New Ideas in Optimization***, Eds. Crone, D., Dorigo, M. and Glover, F., McGraw Hill Inc., UK, pp. 127 – 146 (1999).

118. Lennox, B. and Sandoz, D. "Recent Experiences in the Industrial Exploitation of Principle Component Based Fault Detection Methods", In ***Proceedings of the IEEE International Symposium on Computer Aided Control System Design*** (pp. xxxv–xliv) (2002).

119. Lennox, B., Hiden, H. G., Montague, G. A., Kornfeld, G. G. and Goulding P. R., "Application of Multivariate Statistical Process Control to Batch Operations," ***Comp. & Chem. Engg.***, **24**, 291 – 296 (2000).

120. Li, W., Yue, H. H., Valle-Cervantes, S. and Qin, S. J. "Recursive PCA for Adaptive Process Monitoring", ***Jour. of Proc. Contr.***, **10**, 471 – 486 (2000).

121. Lindberg, W., Persson, J. and Wold, S. "Partial Least Squares Method for Spectrofluorimetric Analysis of Mixtures of Humic and Liguinsulfonate", ***Anal, Chem***. **55**, 643 – 648 (1983).

122. Liu, J. "On-line Soft Sensor for Polyethylene Process with Multiple Production Grades", ***Contr. Engg. Pract.***, doi:10.1016/j.conengpractice.2005.12.005 (2007).

123. Lee, M. H., Han, C. and Chang, K. S. "Dynamic Optimization of a Continuous Polymer Reactor using a Modified Differential Evolution", ***Ind. Engg. Chem. Res.,*** **38**, 4825 – 4831 (1999).

124. Lennon, W. K. and Passino, K. M. ***Engg. Appl. of Artf. Intell.***, **12**, 185 (1999).

125. Li, G.-P., Yan, W.-W., Shao, H.-H., "Constrained Run-to-Run Optimization for Batch Process Based on Support Vector Regression Model", ***Jour. of Shanghai Jiaotong Univ. (Science)***, **11**, 478 - 483 (2006).

126. Lu, J. C. and Wang, F. C., "Optimization of Low Pressure Chemical Vapor Deposition Reactors using Hybrid Differential Evolution", ***Can. Jour. of Chem. Engg.***, **79**, 246 – 254 (2001).

127. MacGregor, J. F., Marlin, T. E., Kresta, J. and Skagerberg, B. "Multivariate Statistical Methods in Process Analysis and Control", in ***Proc. of the Chemical Process Control Conference, CPC-IV***, Eds. S. Padre Island, Texas, 18-22 Feb (1991).

128. Maeda, Y., Hirano, H. and Kanata, Y. "Learning Rule of Neural Networks for Neuro-Controller" ***Neural Networks*** **8**, 251 - 259 (1995).

129. Maeda, Y. and De Figueiredo, R. J. P. "Learning Rules for Neuro-Controller" ***IEEE Trans. Neural Networks*** **8**, 1119 - 1127 (1997).

130. Mahesh, M. M, Madhavan, K. P. and Chidambaram, M. "Fuzzy Control of a Semibatch Copolymerization Reactor", ***Chem. Engg. & Process.***, **32**, 327 - 331 (1993).

131. Majumdar, S., Mitra, K. and Raha, S. "Optimized Species Growth in Epoxy Polymerization with Real Coded NSGA-II", ***Polymer***, **46**, 11858 – 11869 (2005).

132. Malkani, A. and Vassiliadis, C. A. "Parallel Implementation of the Fuzzy ARTMAP Neural Network", ***Exprt. Syst.***, **12**, 39 - 53(1995).

133. Marti, L., *in* ***Proc. IEEE Intl. Jt. Conf. on Neural Net.***, vol. IV, 537 – 542 (1992).

134. Masuduzzaman and Rangaiah, G. P., "Multi-objective Optimization Applications in Chemical Engineering", in ***Multiobjective Optimization: Techniques and Applications in Chemical Engineering***, Ed. Rangaiah, G. P., pp. 27 – 60. World Scientific, Singapore (2009).

135. McKay, B., Willis, M. and Barton, G. W. "Steady-State Modelling of Chemical Process Systems Using Genetic Programming", ***Comp. & Chem. Engg.***, **21**, 981 - 996 (1997).

136. Megan, L., and D. Cooper, Presented in the AIChE Annual Meeting, St. Louis, MO, Paper 145f (1993).

137. Mei, J., Zhang, H-C. and Oldham, W. J. B. "A Neural Network Approach for Datum Selection in Computer-Aided Process Planning", ***Comp. in Indus.***, **27**, 53 - 64 (1995).

138. Mendel, J. M., "Fuzzy Logic Systems for Engineering: A Tutorial", ***Proc. of IEEE*****, 83**, 345 – 377 (1995).

139. Misra, M., Yue, H. H., Qin, S. J. and Ling, C. "Multivariate Process Monitoring and Fault Diagnosis by Multi-scale PCA", ***Comp. & Chem. Engg.***, **26**, 1281 – 1293 (2002).

140. Mitra, K., Majumdar, S. and Raha, S. "Multiobjective Dynamic Optimization of a Semi-Batch Epoxy Polymerization Process", ***Comp. & Chem. Engg.***, **28**, 2583 – 2594 (2004).

141. Mitra, K., Gudi, R. D. Patwardhan, S. C.and Sardar, G. "Towards Resilient Supply Chains: Uncertainty Analysis Using Fuzzy Mathematical Programming", ***Chem. Engg. Res. & Des.***, **87**, 967 – 981 (2009).

142. Müller A., Marsilli-Libelli S., Aivasidis A., Lloyd T., Kroner S. and Wandrey C. "Fuzzy Control of Disturbances in a Wastewater Treatment Process", ***Wat. Res.*****,** **31**, 3157 – 3167 (1997).

143. Muller K.-R., Smola A., Ratsch G., Scholkopf B., Kohlmorgen J., and Vapnik V. "Predicting Time Series with Support Vector Machines. In: Gerstner W.,

Germond A., Hasler M., and Nicoud J.-D. (Eds.), ***Artificial Neural Networks ICANN'97,*** Berlin. Springer (1997).

144. Najim, K., Rusnak, A., Meszaros, A. and Fikar, M. "Constrained Long - Range Predictive nonlinear systems", ***Int. J. of Syst. Sci.***, **28**, 1211 (1997).

145. Nakano, R., "Conventional Genetic Algorithm for Job Shop Problems" *in* ***Proc. Fourth Int. Conf. Genetic Algo.***, 474 (1991).

146. Nandi, S., Ghosh, S., Tambe, S. S., and Kulkarni, B. D. "Artificial Neural-Network-Assisted Stochastic Process Optimization Strategies," ***AIChE Jour***, **47**, 126 - 141 (2001).

147. Nandi, S., Mukherjee, P., Tambe, S. S. Kumar, R. and Kulkarni, B. D. "Reaction Modeling and Optimization Using Neural Networks and Genetic Algorithms: Case Study Involving TS-1 Catalyzed Hydroxylation of Benzene", ***Ind. & Engg. Chem. Res.***, **41**, 2159 - 2169, (2002).

148. Nandi, S., Badhe, Y., Lonari, J., Sridevi, U., Rao, B. S., Tambe,S. S. and Kulkarni, B. D. "Hybrid Process Modeling and Optimization Strategies Integrating Neural Networks / Support Vector Regression and Genetic Algorithms: Study of Benzene Isopropylation on HBeta Catalyst", ***Chem. Eng. J.*** **2004**, **97**, 115 – 129 (2004).

149. Narasimhan, S. and Shah, S. L. "Model Identification and Error Covariance Matrix Estimation from Noisy Data Using PCA", ***Contr. Engg. Prac.***, **16**, 146 – 155 (2008).

150. Nelles, O., *in* ***Fuzzy Evolutionary Computation*** Ed. W. Pedrycz, Kluwer Academy Press (1997).

151. Niemi, H., Bulsari, A. and Palosaari, S. "Simulation of Membrane Separation Using Neural Networks", ***J. Mem. Sci.***, **102**, 185 – 191 (1995).

152. Nomikos, P. and MacGregor, J. F. "Monitoring Batch Processes Using Multiway Principal Component Analysis", ***AIChE Jour.***, **40**, 1361 – 1375 (1994).

153. Nomikos, P. and MacGregor, J. "Multi-way Partial Least Squares in Monitoring Batch Processes", ***Technometrics***, **37**, 41 – 51 (1995).

154. Negiz, A. and Cinar, A. "On the Detection of Multiple Sensor Abnormalities in Multivariate Processes", In ***Proc. of ACC,*** Chicago, IL (1992).

155. Norman, B. A. and Bean, J. C. "A Genetic Algorithm Methodology for Complex Scheduling Problems", ***Naval Res. Logistics***, **46**, 199 – 211 (1999).

156. Normandin, A., Grandjean, B. and Thibault, J. "PVT Data Analysis Using Neural Network Models", ***Ind. Engg. Chem. Res.***, **32**, 970 – 975 (1993).

157. Normandin, A., Thibault, J. and Grandjean, B. P. A. "Optimizing Control of a Continuous Stirred Tank Fermenter Using a Neural Network", ***Bioproc. Engg.***, **10**, 109 – 113 (1994).

158. Nucci, E. R., Silva, R. G., Gomes, T. C., Giordano, R. C. and Cruz, A. J. G. "A Fuzzy Logic Algorithm for Identification of the Harvesting Threshold During PGA Production by *Bacillus Megaterium*", ***Braz. J. Chem. Eng,*** **l2**, 521 - 527 (2005).

159. Quiroga, L. A. and Rabelo, L. C. "Learning from Examples: A review", ***Comp.& Ind. Engg.***, **29**, 561 – 565 (1995).

160. Onnwubolu, G. C. "Optimization using Differential Evolution", The University of the South Pacific Institute of Applied Science Technical Report, TR – 2001/05 (2001).

161. Osofisan, P.B. and Obafaiye, O.J. "Fuzzy Logic Modeling of the Fluidized Catalytic Cracking Unit of a Petrochemical Refinery", ***The Pacific Jour. of Sci.& Technol.***, **8**, 59 – 67 (2007).

162. Ostermark, R., "Solving a Nonlinear Nonconvex Trim Loss Problem with a Genetic Hybrid Algorithm", ***Comp. & Opern. Res.***, **26**, 623 – 635 (1999).

163. Pai, T.Y. , Wan, T.J., Hsu, S.T., Chang, T.C., Tsai, Y.P., Lin, C.Y., Su, H.C. and Yu, L.F. "Using Fuzzy Inference System to Improve Neural Network for Predicting Hospital Wastewater Treatment Plant Effluent", ***Comp. & Chem. Engg.***, **33**, 1272 – 1278 (2009).

164. Pareek, V.K., Brungs, M.P., Adesina A.A. and Sharma, R. "Artificial Neural Network Modeling of a Multiphase Photodegradation System", ***J. Photochem. Photobiol. A: Chem.***, **149**, 139 – 146 (2002).

165. Park, D., Kandel, A. and Langholz, G. ***IEEE Trans. on Syst., Man and Cybern.***, **24**, No. 1, (1994).

166. Pearson, R. K., Ogunnaike, B. A. and Doyle, F. J. "Identification of Discrete Convolution Models for Nonlinear Processes", AIChE Annual Meeting, Paper 125b, Miami, 1-6 Nov. (1992).

167. Pedrycz, W. ***Fuzzy Control and Fuzzy Systems***, Research Studies Press Ltd, UK (1993).

168. Perrot, N., Me, L., Trystram, G., Trichard, J-M. and Decloux, M. ***Fuzzy Sets and Systems***, **94**, 309 (1998).

169. Petridis, V., Kazarlis, S. and Bakirtzis, A. ***IEEE Trans. on Syst., Man and Cybern.***, **28**, 629 (1998).

170. Pinkowski B. "Principal Component Analysis of Speech Spectrogram Images",. ***Pattern Recogn.***, **30**, 777 – 787 (1997).

171. Polit M., Estaben M. and Labat P. "A Fuzzy Model for an Anaerobic Digester, Comparison with Experimental Results", ***Artif. Intell.,*** **15**, 385 – 390 (2002).

172. Price, K. and Storn, R., "Differential Evolution – A Simple Evolution Strategy for Fast Optimization", ***Dr. Dobb's Jour.***, **22**, 18 – 24 (1997).

173. Qin, S. J. and McAvoy, T. J. "Nonlinear PLS Modelling Using Neural Networks", ***Comp. & Chem. Engg.***, **16**, 379 – 391 (1992).

174. Qin, S. J. "A Recursive PLS Algorithm for System Identification", AIChE Annual Meeting, November 7-12, St. Louis, (1993).

175. Qin, S. J. and McAvoy, T. J. "Building Nonlinear FIR Models via a Neural Net PLS Approach", ***Comp. & Chem. Engg.*** **20**, 147 – 159 (1996).

176. Qin, S. J. "Recursive PLS Algorithms for Adaptive Data Modeling", ***Comp.& Chem. Engg.***, **22**, 503 – 514 (1998).

177. Rahman, M. S. and Wnag, J., "Fuzzy Neural Network Model for Liquefaction Prediction", ***Soil Dyn. & Earthquak. Engg.***, **22**, 685 – 694 (2002).

178. Rajesh, J. K., Gupta, S. K., Rangaiah, G. P and Ray, A. K., "Multi-objective Optimization of Industrial Hydrogen Plants", ***Chem. Engg. Sci.***, **56**, 999 – 1010 (2001).

179. Ramachandran, S. and Rhinehart, R. "A Very Simple Structure for Neural Network Control of Distillation", ***J. Proc. Contr.***, **5**, 115 -128 (1995).

180. Ramaswamy, S., Deshpande, P. Paxton, G. and Hajare, R. "Consider Neural Nework for Process Fault Identification", ***Hydrocarb. Process.***, **74** (6), 59 -62 (1995).

181. Rangaiah, G. P. Ed. ***Multiobjective Optimization: Techniques and Applications in Chemical Engineering***, World Scientific, Singapore (2009).

182. Rao, A., Jayaraman, V.K., Kulkarni, B.D., Japanwala, S. and Shevgaonkar, P. "Improved Controller Performance with Simple Fuzzy Rules", ***Hydrocarb. Proces.***, 97-100 May (1999).

183. Raptis, C.G, Siettos, C.I., Kiranoudis, C.T. and Bafas, G.V. "Classification of Aged Wine Distillates Using Fuzzy and Neural Network Systems", ***Jour. of Food Engg.***, **46**, 267 – 275 (2000).

184. Ravi, V., Reddy, P. J. and Dutta, D. "Application of Fuzzy Nonlinear Goal Programming to a Refinery Model", ***Comp. & Chem. Engg.***, **22**, 709 - 712 (1998).

185. Reid, M.K. and Spencer, K.L. "Use of Principal Components Analysis (PCA) on Estuarine Sediment Datasets: The Effect of Data Pre-treatment", ***Environ. Pollu.***, **157**, 2275 – 2281 (2009).

186. Ribeiro, B., Dourado, A. and Costa, E. *in* ***Proc. of Int. Conf. on ANNs and GAs (ICANNGA' 1993)***, Innsbruck, Austria, Eds. R. F. Albrecht et al., 117, Springer-Verlag, Wien (1993).

187. Ribeiro, B., Dourado, A. and Costa, E. *in* ***Proc. of Int. Conf. on ANNs and GAs (ICANNGA' 1995)***, Ales, France, Eds. D. Pearson et al., 408, Springer-Verlag, Wien (1995).

188. Ricker, N. L. **"**Use of Biased Least-Squares Estimators for Parameters in Descrete-Time Pulse Response Models" ***Ind. Engg. Chem. Res.*****, 27**, 343 – 350 (1988).

189. Ronen, M., Shabtai, Y. and Guterman, H. "Rapid Process Modeling — Model Building Methodology Combining Unsupervised Fuzzy-Clustering and Supervised Neural Networks", ***Comp. & Chem. Engg.***, **22**, S1005 - S1008 (1998).

190. Rumelhart, D. E., and J. L. McCelland, ***Parallel Distributed Processing***, MIT Press, Cambridge, MA (1986).

191. Rumelhart, D., Hinton, G., and Williams, R. "Learning Representations by Backpropagating Errors", ***Nature***, **323**, 533 - 536 (1986).

192. Savkovic-Stevanovic, J., "Design of Neural Controller by Inverse Modeling", ***Comp. Chem. Engg.***, **18**, 1149 – 1155 (1994).

193. Sbarbaro-Hofer, D., D. Neumerkel, and K. Hunt, *In* ***IEEE Int. Symp. Intell. Contr.***, ISIC-92, Aug. 11-13, 1992, Glasgow, UK (1992).

194. Scholkopf B., Burges C. and Vapnik, V. "Incorporating Invariances in Support Vector Learning Machines. In: von der Malsburg C., von Seelen W., Vorbruggen J. C., and Sendhoff B. (Eds.), ***Artificial Neural Networks ICANN'96***, pp. 47–52, Berlin, Springer Lecture Notes in Computer Science, Vol. 1112, (1996).

195. Scholkopf B. "Support Vector Learning", R. Oldenbourg Verlag, Munchen. Doktorarbeit, TU Berlin. Available at: http://www.kernel-machines.org (1997).

196. Sharma, B. K., Sharma, M. P. Roy, S. K., Kumar, S., Tendulkar, S. B., Tambe, S. S. and Kulkarni, B. D. "Fischer-Tropsch Synthesis with Co/SiO_2-Al_2O_3 Catalyst and Modeling Using Neural Networks", ***Fuel***, **77**, 1763 – 1776 (1998).

197. Sharma, R. Singhal, D., Ghosh R. and Dwivedi, A. "Potential Applications of Artificial Neural Networks to Thermodynamics: Vapor–Liquid Equilibrium Predictions", ***Comp. &. Chem. Eng.***, **23**, 385 – 390 (1999).

198. Sharma, R., Singh, K., Singhal, D., and Ghosh, R. "Neural Network Applications for Detecting Process Faults in Packed Towers", ***Chem. Engg. & Proc.***, **43**, 841 - 847 (2004).

199. Sanchiz, C., Benedito, V., Garcia-Camascosa, A.M. and Pastor, A. "A Multivariate Analysis of Heavy Metal Contents and Physico-chemical Characteristics of Sediments from Marine Macrophyte Meadows", ***Fresen. Environ. Bull.***, **7**, 429 – 436 (1998).

200. Seng,, T.L., Khalid, M. and Yusof, R. "Adaptive GRNN Modeling of Dynamic Plants", http://www.cairo.utm.my/publications/lsteo_agrnn.pdf. (2001)

201. Shaw, P. J. A. ***Multivariate Statistics for the Environmental Sciences***, Arnold, London (2003).

202. Sim, S. K., Yeo, K. T. and Lee, W. H. "An Expert Neural Network System for Dynamic Job Scheduling", ***.Int. J. Prod. Res.***, **32**, 1759 – 1773 (1994).

203. Smith, T. and, DeJong K. A., *in* ***Proc. 12th Annu. Pittsburgh Conf. Modeling and Simulation***, pp. 955 (1981).

204. Smith L. "A Tutorial on Principal Component Analysis. Available from http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf , (2002).

205. Smola, A. J. and Scholkopf, B., "A Tutorial on Suport Vector Regression", ***Stat. & Comput.***, **14**, 199 – 222 (2004).

206. Sorsa, T., Koivo, H. and Koivisto, H. "Neural Networks in Process Fault Diagnosis", ***IEEE Trans. Sys. Man Cybern.***, **21**, 815 – 825 (1991).

207. Sousa Jr, R. and Almeida, P.I.F. "Design of Fuzzy System for the Control of a Biochemical Reactor in Fed-batch Culture" ***Proc. Biochem.***, **37**, 461 - 469 (2001).

208. Specht, D.F. "A General Regression Neural Network", ***IEEE Trans. on Neu. Net.***, **2**, 568 - 576, (1991).

209. Srinivas, M. and Rangaiah, G. P. "An Integrated Stochastic Method for Global Optimization of Continuous Functions", ***Comp. Aided Chem. Engg.***, **21**, 439 – 444 (2006).

210. Srinivas, M. and Rangaiah, G. P. "A Study of Differential Evolution and Tabu Search for Benchmark, Phase Equilibrium and Phase Stability Problems", ***Comp. & Chem. Engg.***, **31**, 760 – 772 (2007).

211. Sridevi, U., Rao, B. K. B., Pradhan, N. C., Tambe, S. S., Satyanarayana C. V., and Rao, B. S., "Kinetics of Isopropylation of Benzene over HBeta Catalyst", ***Ind. Engg. Chem. Res***., **40**, 3133-3138 (2001).

212. Storn, R. "Differential Evolution Design of an IIR – filter with Requirements for magnitude and Group Delay", International Computer Science Institute, TR – 95 – 018 (1995).

213. Sun, G., Dagli, C. H. and Thammano, A. "Dynamic Neuro-Fuzzy Control of the Nonlinear Process" ***Comp. & Ind. Engg.***, **33**, 413 – 416 (1997).

214. Syswerda, G., *in* ***Proc. Fourth Int. Conf. Genetic Algo***., 502 (1991).

215. Sugeno, M "An Introductory Survey of Fuzzy Control", ***Inf. and Sci.***, **36**, 59-83, (1985).

216. Sugeno M. and Yasukawa T., "A Fuzzy-logic-based Approach to Qualitative Modeling", ***IEEE Trans. on Fuzzy Systems***, **1**, 7 – 31 (1993).

217. Szpiro, G. G., "Forcasting Chaotic Time-Series with Genetic Algorithms", ***Phys. Rev. E***., **55**, 2557 – 2568 (1997).

218. Tambe, S. S., Kulkarni, B. D. and Deshpande, P. B. ***Elements of Artificial Neural Networks with Selected Applications in Chemical Engineering, and Chemical & Biological Sciences***, Simulation & Advanced Controls Inc., Louisville, KY (1996).

219. Tan, K. C., Phang, K. P. and Yang, Y. J. "Feed Optimization for Fluidized Catalytic Cracking using a Multi-objective Evolutionary Algorithm", in ***Multiobjective Optimization: Techniques and Applications in Chemical Engineering***, Ed. Rangaiah, G. P., pp. 277 – 300. World Scientific, Singapore (2009).

220. Tarafder, A., Rangaiah, G. P and Ray, A. K., "Multi-objective Optimization of an Industrial Styrene Monomer Manufacturing Process", ***Chem. Engg. Sci.***, **60**, 347 – 363 (2005).

221. Tarafder, A., Rangaiah, G. P and Ray, A. K., "A Study of Finding Many Desirable Solutions in Multi-objective Optimization of Chemical Processes", ***Comp. & Chem. Engg.***, **31**, 1257 – 1271 (2007).

222. Taskin, H., Kubat, C., Uygun, O. and Arslankaya, S. "FUZZYFCC: Fuzzy Loic Control of a Fluidized Catalytic Cracking Unitr (FCCU) to Improve Dynamic Performance", ***Comp. & Chem. Engg.***, **30**, 850 – 863 (2006).

223. Tendulkar, S. B., Tambe, S. S., Chandra, I., Rao, P. V. , Naik, R. V. and Kulkarni, B. D. "Hydroxylation of Phenol to Dihydroxybenzenes: Development of Artificial Neural-Network-Based Process Identification and Model Predictive Control Strategies for Pilot Plant Scale Reactor", ***Ind. Eng. Chem. Res***., **37**, 2081 – 2085 (1998).

224. Tong, H., ***Nonlinear Time Series: A Dynamic System Approach***, Claredon Press, Oxford (1990).

225. Tsekouras, G., Sarimveis, H., Raptis, C. and Bafas**,** G. "A Fuzzy Logic Approach for the Classification of Product Qualitative Characteristics", ***Comp. & Chem. Engg.***, **26**, 429 - 438 (2002)

226. Vapnik, V. ***The Nature of Statistical Learning Theory***, Springer Verlag, New York (1995).

227. Vapnik, V., Golowich, S. and Smola, A. "Support Vector Method for Function Approximation, Regression Estimation and Signal Processing," ***Adv. in Neu. Inform. Pro. Syst.***, **9**, 281-287, (1996).

228. Vapnik V., Golowich S., and Smola A. "Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing. In: Mozer M.C., Jordan M.I., and Petsche T. (Eds.) ***Advances in Neural Information Processing Systems*** 9, MA, MIT Press, Cambridge. pp. 281–287 (1997).

229. Vapnik, V. ***Statistical Learning Theory***, John Wiley, New York (1998).

230. Varsek, A., Urbancic, T. and Filipic, B. "Genetic Algorithms in Controller Design and Tuning", ***IEEE Trans. on Sys., Man and Cybern.***, **23**, 1330 – 1339 (1993).

231. Venkatasubramanian, V., Chan, K. and Caruthers, J. M. "Computer-aided molecular design using genetic algorithms", ***Comp &. Chem. Engg.***, **18**, 833 – 844 (1994).

232. Venkatsubramanian, V. and Sundaram, A. *in* ***Encyclopedia of Computational Chemistry***, Ed. P. R. Schleyer, Vol. 2, 1115, John Wiley, Chichester (1998).

233. Vincente, M., Leiza, J. R. and Ausa, J. M. "Maximizing Production and Polymer Quality (MWD and Composition) in Emulsion Polymerization Reactors with Limited Capacity of Heat Removal", ***Chem. Engg. Sci.***, **58**, 215 – 222 (2003).

234. Vora, N., Tambe, S. S. and Kulkarni, B. D. "Counterpropagation Neural Networks for Fault Detection and Diagnosis", ***Comp. & Chem. Engg.***, **21**, 177 – 185, (1997).

235. Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N. and Yin, K. "A Review of Process Fault Detection and Diagnosis Part III: Process History Based Methods", ***Comp. & Chem. Engg.***, **27**, 327 – 346 (2003).

236. Wadi, I., ***Oil and Gas J***., **25**, 67 (1996).

237. Wang, F. S., Jing, C. H. and Tsao, G. T., "Fuzzy-decision-making Problems on Fuel Ethanol Production using Genetically Engineered Yeast", ***Ind. Engg. Chem. Res.***, **37**, 3434 – 3443 (1998).

238. Wang, F. S., Su, T. L. and Jang, H. J., "Hybrid Differential Evolution for Problems of Kinetics Parameter Estimation and Dynamic Optimization of an Ethanol Fermentation Process", ***Ind. Engg. Chem. Res.***, **40**, 2876 – 2885 (2001)

239. Wentzell, P. D., Andrews, D. T., Hamilton, D. C., Faber, K. and Kowalski, B. R. "Maximum Likelihood Principal Component Analysis", ***Jour. of Chemo.***, **11**, 339 – 366 (1997).

240. Whittaker, A. D. and Cook, D. F. "Neural Network Process Modeling of a Continuous Correlated Process" ***Int. J. of Prod. Res***., **33**, 1901 (1995).

241. Willis, M., G., Montague, C., Di Massimo, M. Tham, and A. Morris, "Artificial Neural Networks in Process Engineering", ***Automatica***, **28**, 1181 (1992).

242. Wise, B. M. and Rcker, N. L. “The effect of biased regression on the identification of FIR and ARX models”, AIChE Annual Meeting, Chicago, IL, November (1990).

243. Wold, S., Ruke, A., Wold, H. and Dunn III, W. J. “The Collinearity Problem in Linear Regression, The Partial Least Squares (PLS) Approach to Generalized Inverses”, ***SlAM ,I. Sci. Stat. Computs.* 5**, 735 – 743 (1984).

244. Wold, S., Esbensen, K. and Geladi, P. “Principal Component Analysis”, ***Chemo. & Intel. Lab. Syst.***, **2**, 37-52, (1987).

245. Wold, S., Kettaneh-Wold, N. and Skagerberg, B. “Nonlinear PLS Modeling”, ***Chemo. & Intell. Lab. Syst.***, **7**, 53 – 65 (1989).

246. Wold, S., Sjostrom, M. and Eriksson, L. “PLS - Regression: A Basic tool of Chemometrics”, ***Chemo. & Intel. Lab. Syst.***, **58**, 109 – 130 (2001).

247. Yadavalli, V. K., Dahule, R. K., Tambe, S. S. and Kulkarni, B. D. “Obtaining Functional Form for Chaotic Time Series Evolution Using Genetic Algorithm”, ***Chaos***, **9**, 789 (1999).

248. Yamada, Y., Haneda, K., Murayama, S. and Shiomi, S. “Application of Fuzzy System to Coenzyme Q10 Fermentation”, ***Jour. of Chem. Engg. Japan***, **24**, 94 - 99 (1991).

249. Yoon, S. and MacGregor, J. F. “Statistical and Causal Model-based Approaches to Fault Detection and Isolation”, ***AIChE Jour.***, **46**, 1813 – 1819 (2000).

250. Zadeh, L.A. "Fuzzy Sets," ***Info. & Ctl.***, **8**, 338 – 353 (1965).

251. Zadeh, L.A. "Fuzzy Algorithms," ***Info. & Ctl.***, **12**, 94 – 102 (1968).

252. Zamprogna, E., Barolo, M. and Seborg, D.E. “Estimating Product Composition Profiles in Batch Distillation via Partial-least-squares Regression”, ***Contr. Engg. Prac.* 12**, 917 – 929 (2004).

253. Zitko, V. “Principal Components Analysis in the Evaluation of Environmental Data”, ***Marine Pollu. Bull.***, **28**, 718 – 722 (1994).

CHAPTER 2

HYBRID PROCESS MODELING AND OPTIMIZATION STRATEGIES INTEGRATING NEURAL NETWORKS / SUPPORT VECTOR REGRESSION AND GENETIC ALGORITHMS: STUDY OF BENZENE ISOPROPYLATION ON HBETA CATALYST

Published in ***Chemical Engineering Journal***, Vol. **97** (No. 2 – 3), 115 – 129 (2004).

Abstract

This chapter presents a comparative study of two artificial intelligence based hybrid process modeling and optimization strategies, namely ANN-GA and SVR-GA, for modeling and optimization of benzene isopropylation on HBeta catalytic process. In the ANN-GA approach, an artificial neural network model is constructed for correlating process data comprising values of operating (input) and output variables. Next, model inputs describing process operating variables are optimized using genetic algorithms (GAs) with a view to maximize the process performance. The GA possesses certain unique advantages over the commonly used gradient-based deterministic optimization algorithms. In the second hybrid methodology, a novel machine learning formalism, namely support vector regression (SVR), has been utilized for developing process models and the input space of these models is optimized again using GAs. The SVR-GA is a new strategy for chemical process modeling and optimization. The major advantage of the two hybrid strategies is that modeling and optimization can be conducted exclusively from the historic process data wherein the detailed knowledge of process phenomenology (reaction mechanism, rate constants, etc.) is not required. Using ANN-GA and SVR-GA strategies, a number of sets of optimized operating conditions leading to maximized yield and selectivity of the benzene isopropylation reaction product, namely cumene, were obtained. The optimized solutions when verified experimentally resulted in a significant improvement in the cumene yield and selectivity.

2. 1 INTRODUCTION

Availability of a process model is a prerequisite to process optimization. Conventionally, two approaches namely *phenomenological* (first principles) and *empirical*, are employed for chemical process modeling. In phenomenological modeling, the detailed knowledge of the reaction kinetics and associated heat and mass transport phenomena are required to represent mass, momentum, and energy balances. The advantages of a phenomenological model are: (i) since it represents physico-chemical phenomenon underlying the process explicitly, it provides a valuable insight into the process behavior, and (ii) it possesses extrapolation ability. Owing to the complex nature of many chemical processes, the underlying physico-chemical phenomenon is seldom fully understood. Also, collection of the requisite phenomenological information is costly, time-consuming and tedious, and therefore development of phenomenological process models poses considerable practical difficulties. Moreover, nonlinear behavior being common in chemical processes, it leads to complex nonlinear models, which in most cases are not amenable to analytical solutions; thus, computationally intensive numerical methods must be utilized for obtaining solutions. Difficulties associated with the construction and solution of phenomenological models necessitates exploration of alternative modeling formalisms. Modeling using empirical (regression) methods is one such alternative. In conventional empirical modeling, appropriate linear or nonlinear models are constructed exclusively from the process input-output data without invoking the process phenomenology. A fundamental deficiency of the conventional empirical modeling approach is that the structure (functional form) of the data-fitting model must be specified a priori. Satisfying this requirement, especially for nonlinearly behaving processes is a cumbersome task since it involves selecting heuristically an appropriate nonlinear model structure from numerous alternatives.

In the last decade, *artificial neural networks* (ANNs) and more recently *support vector regression* (SVR) have emerged as two attractive tools for nonlinear modeling especially in situations where the development of phenomenological or conventional regression models becomes impractical or cumbersome. The most widely utilized ANN paradigm is the *multi-layered perceptron* (MLP) that approximates nonlinear relationships existing between an input set of data (causal process variables) and the corresponding output (dependent variables) data set. The advantages of an ANN-based model are: (i) it can be constructed solely from the historic process input-output data (example set), (ii)

detailed knowledge of the process phenomenology is unnecessary for the model development, (iii) a properly trained model possesses excellent generalization ability owing to which it can accurately predict outputs for a new input data set, and (iv) even multiple input-multiple output (MIMO) nonlinear relationships can be approximated simultaneously and easily. Owing to their several attractive characteristics, ANNs have been widely used in chemical engineering applications such as steady state and dynamic process modeling, process identification, yield maximization, nonlinear control, and fault detection and diagnosis (e.g., Tambe et. al., 1996; Bulsari 1994, 1995; Huang et al. 2003 and Stephanopoulos and Han, 1996). There exists a number of algorithms—each possessing certain positive characteristics—to train an MLP network, for example, the most popular *error-back-propagation* (EBP) (Rumelhart et. al., 1986), *Quickprop* (Fahlman, 1988) and *Resilient Back-propagation* (RPROP) (Riedmiller and Braun, 1993). Training of an ANN involves minimizing a nonlinear error function (e.g., *root-mean-squared-error*, RMSE) that may possess several local minima. Thus, it becomes necessary to employ a heuristic procedure involving multiple training runs to obtain an optimal ANN model whose parameters (weights) correspond to the global or the deepest local minimum of the error function.

In recent years, support vector regression (SVR) (Vapnik, 1995, 1998; Vapnik et. al., 1996) which is a statistical learning theory based machine learning formalism is gaining popularity due to its many attractive features and promising empirical performance. The salient features of SVR are: (i) like ANNs, SVR is an exclusively data based nonlinear modeling paradigm, (ii) SVR-based models are based on the principle of structural risk minimization, which equips them with greater potential to generalize, (iii) parameters of an SVR model are obtained by solving a quadratic optimization problem, (iv) the objective function in SVR being of quadratic form, it possesses a single minimum thus avoiding the heuristic procedure involved in locating the global or the deepest local minimum on the error surface, and (v) in SVR, the inputs are first nonlinearly mapped into a high dimensional feature space wherein they are correlated linearly with the output. Although the foundation of the SVR paradigm was laid down in mid 1990s, its chemical engineering applications such as fault detection (Agarwal et. al., 2003; Jacj and Nandi, 2002) have emerged only recently.

Once an ANN or SVR-based process model is developed, it can be used for process optimization to obtain the optimal values of the process input variables that maximize or minimize a specified objective function. Thus, it is possible to obtain the

optimal values of process operating variables, which for instance, maximize reactant conversion and selectivity of the desired products, or minimize reactor temperature and the selectivity of undesired by-products. Conventionally, various deterministic gradient-based methods (Edgar and Himmelblau, 1989) are used for optimizing a process model. Most of these methods however require that the objective function should be smooth, continuous, and differentiable. The ANN or SVR models can not be guaranteed to be smooth especially in regions wherein the input-output data (training set) used in model building are located sparsely. Hence, gradient-based methods cannot be used efficiently for optimizing the input space of an ANN or SVR model. In such situations, an efficient optimization formalism known as *genetic algorithms* (GAs), which is lenient towards the form of the objective function, can be used. The GAs are stochastic optimization formalisms originally developed as the genetic engineering models mimicking population evolution in natural systems (Goldberg, 1989; Davis, 1991; Deb, 1995). GAs follow the "survival-of-the-fittest" and "genetic propagation of characteristics" principles of biological evolution for searching the solution space of an optimization problem. The principal advantages of the GAs are: (i) they are zeroth order optimization method requiring only the scalar values – and not the second and / or the first order derivatives – of the objective function, (ii) capability of handling nonlinear and noisy objective functions, (iii) they perform global search and thus are most likely to arrive at or near the globally optimum solution, and (iv) unlike most gradient-based deterministic optimization methods, GAs do not impose preconditions, such as smoothness, differentiability, and continuity, on the form of the objective function. Due to their several attractive properties, GAs have been extensively used in chemical engineering (Ramanathan et.al., 2001; Rodemerck et. al., 2001; Sumanwar et. al., 2002; Imanuel and Doyle III, 2002; Nougues et. al., 2002; Yee et. al., 2003; Lucasius and Kakerman, 1993 and 1994).

In the present study, ANN and SVR formalisms are integrated with genetic algorithms to arrive at two hybrid process modeling and optimization strategies. The strategies (henceforth referred to as "ANN-GA" and "SVR-GA") use an ANN or SVR as the nonlinear process modeling paradigm, and the GA for optimizing the input space of the ANN/SVR model such that an improved process performance is realized. To our knowledge, the hybrid involving SVR and GA is being used for the first time for chemical process modeling and optimization. In this study, the ANN-GA (Nandi et. al., 2002) and SVR-GA hybrid strategies have been used to model and optimize the pilot plant scale process involving benzene isopropylation using the HBeta catalyst. The optimized

operating conditions leading to maximized yield and selectivity of the desired reaction product (cumene) as given by the two strategies have been compared. The best sets of operating conditions obtained thereby when subjected to experimental validation indeed resulted in significant enhancements in cumene yield and selectivity.

The chapter is organized as follows: section 2.2 describes process modeling using ANN and SVR methods; the optimization of the models using GAs is explained in section 2.3. Usage of ANN-GA and SVR-GA strategies for optimizing the benzene isopropylation process along with the results of experimental verification are described in section 2.4. Finally, section 2.5 gives a summary of the study.

2.2 HYBRID PROCESS MODELING AND OPTIMIZATION FORMALISMS

The process optimization objective under consideration is stated as:

Given catalytic process data comprising values of process operating (input) variables and the corresponding values of process output (response) variables, find the optimal values of input variables such that the pre-specified measures of process performance are simultaneously maximized.

This optimization problem can be formulated as:

$$\text{Maximize} \quad y_k = f_k(\mathbf{x}, \mathbf{w}_k); \quad k = 1, 2, ..., K \tag{2.1}$$

where, y_k denotes the kth output variable; $\mathbf{x} = [x_1, x_2, ..., x_N]^T$ is an N-dimensional vector of process operating variables, f_k refers to the function correlating kth output variable with the inputs, and $\mathbf{w}_k$ represents the parameter vector of function, f_k. Equation (2.1) describes a multi-objective (MO) optimization problem since it involves simultaneous maximization of K outputs, $\{y_k\}$, $k = 1,2,..., K$. Using aggregation principle (also known as "weighting objective method"), the MO optimization task can be converted into a single objective (SO) optimization by defining:

$$\text{Maximize} \quad \hat{f} = \sum_{k=1}^{K} \hat{w}_k y_k = \sum_{k=1}^{K} \hat{w}_k f_k(\mathbf{x}, \mathbf{w}_k) \tag{2.2}$$

where $\hat{f}$ denotes the single aggregated objective function and $\hat{w}_k$ represents the weighting coefficient ($0 \le \hat{w}_k \le 1$, $\sum_k \hat{w}_k = 1$). The weighting coefficient, $\hat{w}_k$, signifies the relative importance of kth function in the MO optimization (Eq. 2.1). The hybrid

strategies fulfill the SO optimization task in two steps. In the first step, an ANN or SVR-based process model, $y_k = f_k(\mathbf{x}, \mathbf{w}_k)$, is developed and in the second step, the input space ($\mathbf{x}$) of the process model is optimized using GAs with a view of maximizing the single aggregated objective function defined in Eq. (2.2).

2.2.1 ANN-Based Modeling

For process modeling, the commonly used feed-forward ANN architecture namely multilayer perceptron (MLP) may be employed. The MLP network approximates the nonlinear input-output relationships as defined by, $y_k = f_k(\mathbf{x}, \mathbf{w}_k), k = 1, 2, ..., K$, where $\mathbf{w}_k$ is the vector defining network weights. The MLP network usually consists of three layers. The layers described as *input*, *hidden,* and *output* layers comprise N, L, and K number of processing nodes, respectively. Each node in the input (hidden) layer is linked to all the nodes in the hidden (output) layer using weighted connections. The MLP architecture also houses a bias node (with fixed output of +1) in its input and hidden layers; the bias nodes are also connected to all the nodes in the subsequent layer. Usage of bias nodes helps the MLP-approximated function to be positioned anywhere in the N-dimensional input space; in their absence, the function is forced to pass through the origin of the N-dimensional space. The N number of nodes in the input layer is equal to the number of process operating variables, whereas the number of output nodes (K) equals the number of process outputs. However, the number of hidden nodes (L) is an adjustable parameter whose magnitude is determined by issues such as the desired approximation and generalization performance of the network model. In order that the MLP network accurately approximates the nonlinear relationship existing between the process inputs and the outputs, it needs to be trained in a manner such that a prespecified error function is minimized. In essence, the MLP training procedure aims at obtaining an optimal weight set $\{\mathbf{w}_k\}$ that minimizes a prespecified error function. The commonly employed error function is the *root-mean-squared error* (RMSE) defined as:

$$RMSE = \sqrt{\sum_{i=1}^{P}\sum_{k=1}^{K}\left(\hat{y}_{i,k} - y_{i,k}^{p}\right)^2} \tag{2.3}$$

where, P refers to the number of input-output example patterns used in training; i denotes the index of the example pattern (vector) and, $\hat{y}_{i,k}$ and $y_{i,k}^{p}$ are the desired (target) and MLP predicted values of the kth output node, respectively. The widely used formalism for the RMSE minimization is the *error-back-propagation* (EBP) algorithm (Rumelhart et. al.,

1986) utilizing a gradient-descent technique known as the *generalized delta rule* (GDR) for iterative updation of weights. The details of the heuristic procedure involved in obtaining an optimal network model possessing good prediction and generalization capabilities can be found in e.g., Freeman and Skapura (1991), Bishop (1994), Tambe et al. (1996) and Nandi et al. (2001). The EBP training algorithm makes use of two adjustable (free) parameters namely, the learning rate, η ($0 < \eta \leq 1$) and the momentum coefficient, α_{EBP} ($0 < \alpha_{EBP} \leq 1$). The magnitudes of both these parameters need to be optimized heuristically.

2.2.2 SVR-Based Modeling

Support vector regression is an adaptation of a recent statistical learning theory based classification paradigm, namely *support vector machines* (Vapnik et. al., 1996). The SVR formulation follows structural risk minimization (SRM) principle, as opposed to the empirical risk minimization (ERM) approach which is commonly employed within statistical machine learning methods and also in training ANNs. In SRM, an upper bound on the generalization error is minimized as opposed to the ERM, which minimizes the prediction error on the training data. This equips the SVR with a greater potential to generalize the input-output relationship learnt during its training phase for making good predictions for new input data. The SVR is a linear method in a high dimensional feature space, which is nonlinearly related to the input space. Though the linear algorithm works in the high dimensional feature space, in practice it does not involve any computations in that space, since through the usage of kernels, all necessary computations are performed directly in the input space. In the following, the basic concepts of SVR are introduced. A more detailed description of SVR can be found in Vapnik (1995 and 1998), Burges (1998), Smola et al. (1998) and Schölkoff et al. (2001).

Consider a training data set $\Im = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_P, y_P)\}$, such that $\mathbf{x}_i \in \Re^N$ is a vector of input variables and $y_i \in \Re$, is the corresponding scalar output (target) value. Here, the modeling objective is to find a regression function, $y = f(\mathbf{x})$, such that it accurately predicts the outputs $\{y\}$ corresponding to a new set of input-output examples, $\{(\mathbf{x}, y)\}$, which are drawn from the same underlying joint probability distribution, $P(\mathbf{x}, y)$, as the training set. To fulfill the stated goal, SVR considers following linear estimation function.

$$f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b \tag{2.4}$$

where $\mathbf{w}$ denotes the weight vector; b refers to a constant known as "bias"; $\Phi(\mathbf{x})$ denotes a function termed *feature*, and $\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle$ represents the dot product in the feature space, $\wp$, such that Φ: $\mathbf{x} \rightarrow \wp$, $\mathbf{w} \in \wp$. In SVR, the input data vector, $\mathbf{x}$, is mapped into a high dimensional feature space, $\wp$, via a nonlinear mapping function, Φ, and a linear regression is performed in this space for predicting y. Thus, the problem of nonlinear regression in lower dimensional input space $\Re^N$ is transformed into a linear regression in the high dimensional feature space, $\wp$. Accordingly, the original optimization problem involving nonlinear regression is transformed into finding the flattest function in the feature space $\wp$ and not in the input space, $\mathbf{x}$. The unknown parameters $\mathbf{w}$ and b in Eq. 2.4 are estimated using the training set, $\Im$. To avoid over fitting and thereby improving the generalization capability, following regularized functional involving summation of the empirical risk and a complexity term $\|\mathbf{w}\|^2$, is minimized (Burges, 1998):

$$\begin{aligned} R_{reg}[f] &= R_{emp}[f] + \lambda \|\mathbf{w}\|^2 \\ &= \sum_{i=1}^{P} C\left(f(\mathbf{x}_i) - y_i\right) + \lambda \|\mathbf{w}\|^2 \end{aligned} \tag{2.5}$$

where, R_{reg} and R_{emp} denote the regression and empirical risks, respectively; $\|\mathbf{w}\|^2$ is the Euclidean norm; $C(.)$ is a cost function measuring the empirical risk, and $\lambda > 0$, is a regularization constant. For a given function, f, the regression risk (test set error), $R_{reg}(f)$, is the possible error committed by the function f in predicting the output corresponding to a new (test) example input vector drawn randomly from the same sample probability distribution, $P(\mathbf{x}, y)$, as the training set. The empirical risk $R_{emp}(f)$, represents the error (termed "training set error") committed in predicting the outputs of the training set inputs. Minimization task described in Eq. 2.5 involves: (i) minimization of the empirical loss function $R_{emp}(f)$ and, (ii) obtaining as small a $\mathbf{w}$ as possible, using the training set γ. The commonly used loss function is the "ε-insensitive loss function" given as (Vapnik, 1998):

$$C\left(f(\mathbf{x}) - y\right) = \begin{cases} |f(\mathbf{x}) - y| - \varepsilon \text{ for } |f(\mathbf{x}) - y| \geq \varepsilon \\ 0 \qquad\qquad \text{otherwise} \end{cases} \tag{2.6}$$

where ε is a precision parameter representing the radius of the tube located around the regression function (see Figure 2.1); the region enclosed by the tube is known as "ε-intensive zone". The SVR algorithm attempts to position the tube around the data as shown in Fig. 2.1. The optimization criterion in Eq. 2.6 penalizes those data points whose *y* values lie more than ε distance away from the fitted function, $f(\mathbf{x})$. In Fig. 2.1, the size of the stated excess positive and negative deviations are depicted by ξ and ξ*, which are termed "slack" variables. Outside of the [-ε, ε] region, the slack variables assume non-zero values. The SVR fits $f(\mathbf{x})$ to the data in a manner such that: (i) the training error is minimized by minimizing ξ_i and ξ_i^* and, (ii) $\|\mathbf{w}\|^2$ is minimized to increase the flatness of $f(\mathbf{x})$ or to penalize over complexity of the fitting function. Vapnik (1996, 1998) showed that the following function possessing finite number of parameters can minimize the regularized function in Eq. 2.5.

$$f(\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*) = \sum_{i=1}^{P} (\alpha_i - \alpha_i^*) K(\mathbf{x}, \mathbf{x}_i) + b \tag{2.7}$$

where, α_i and α_i^* (≥ 0) are the coefficients (Lagrange multipliers) satisfying $\alpha_i \alpha_i^* = 0$, $i = 1, 2, \ldots, P$, and $K(\mathbf{x}, \mathbf{x}_i)$ denotes the so called 'kernel' function describing the dot product in the feature space. The kernel function is defined in terms of the dot product of the mapping function as given by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \tag{2.8}$$

The advantage of this formulation (Eqs. 2.7 and 2.8) is that for many choices of the set $\{\Phi_i(\mathbf{x})\}$, including infinite dimensional sets, the form of *K* is analytically known and very simple (Mukherjee et. al., 1997). Accordingly, the dot product in the feature space ι can be computed without actually mapping the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ into that space (i.e., computation of $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$). There exist several choices for the kernel function *K*; the two commonly used kernel functions, namely, radial basis function (RBF) and *n*th degree polynomial are defined below in Eqs. (2.9) and (2.10), respectively.

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right) \tag{2.9}$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \left[1 + (\mathbf{x}_i, \mathbf{x}_j) \right]^n \tag{2.10}$$

In Eq. (2.7), the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$ are obtained by solving following quadratic programming problem.

Maximize:

$$R(\boldsymbol{\alpha}^*,\boldsymbol{\alpha}) = -\frac{1}{2}\sum_{i,j=1}^{P}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)K(\mathbf{x}_i,\mathbf{x}_j) - \varepsilon\sum_{i=1}^{P}(\alpha_i^* + \alpha_i) + \sum_{i=1}^{P} y_i(\alpha_i^* - \alpha_i) \qquad (2.11)$$

subject to constraints: $0 \le \alpha_i$, $\alpha_i^* \le C$, $\forall i$ and $\sum_{i=1}^{P}(\alpha_i^* - \alpha_i) = 0$. Having estimated $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$ and b, using a suitable quadratic programming algorithm, the SVR-based regression function takes the form

$$f(\mathbf{x},\mathbf{w}) = f(\mathbf{x},\boldsymbol{\alpha},\boldsymbol{\alpha}^*) = \sum_{i=1}^{P}(\alpha_i^* - \alpha_i)K(\mathbf{x}_i,\mathbf{x}) + b \qquad (2.12)$$

where, vector $\mathbf{w}$ is described in terms of the Lagrange multipliers $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$. Owing to the specific character of the above-described quadratic programming problem, only some of the coefficients, (α_i^* - α_i), are non-zero and the corresponding input vectors, $\mathbf{x}_i$, are called support vectors (SVs). The SVs can be thought of as the most informative data points that compress the information content of the training set. The coefficients α_i and α_i^* have an intuitive interpretation as forces pushing and pulling the regression estimate $f(\mathbf{x}_i)$ towards the measurements, y_i (Muller et. al., 1997).

In Eq. (2.12), the bias parameter, b, can be computed as follows;

$$b = \begin{cases} y_i - f(\mathbf{x}_i)_{b=0} - \varepsilon & for\ \alpha_i \in (0,C) \\ y_i - f(\mathbf{x}_i)_{b=0} + \varepsilon & for\ \alpha_i^* \in (0,C) \end{cases} \qquad (2.13)$$

where, $\mathbf{x}_i$ and y_i respectively denote the ith support vector and the corresponding target output, respectively. In the SVR formulation, C and ε are two user-specified free parameters; while C represents the trade-off between the model-complexity and the approximation error, ε signifies the width of the ε-insensitive zone used to fit the training data. The stated free parameters together with the specific form of the kernel function control the accuracy and generalization performance of the regression estimate. The procedure of judicious selection of C and ε is explained by Cherkassky and Ma (2002).

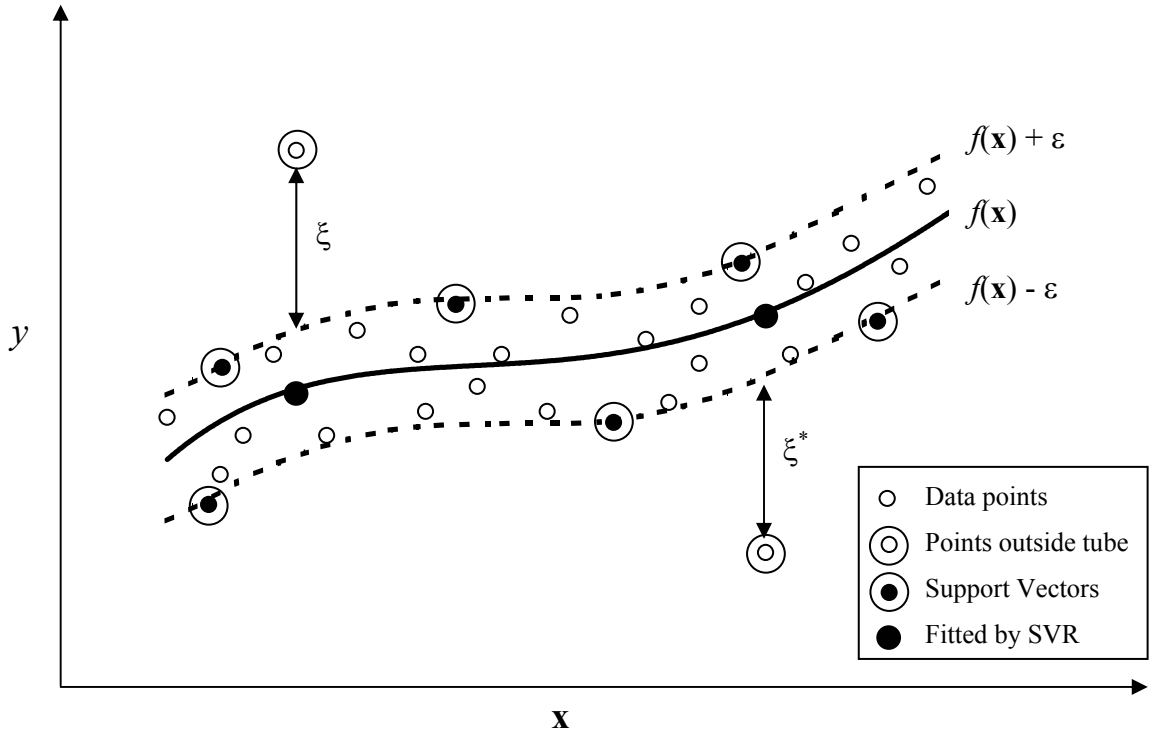


Figure 2.1: A schematic of support vector regression using ε-sensitive loss function

2.3 GA-BASED OPTIMIZATION OF ANN AND SVR MODELS

The optimization objective underlying the GA-based optimization of an ANN or SVR model is defined as: Find the *N*-dimensional optimal decision variable vector, **x*** = $\left[x_1^*, x_2^*, ..., x_N^*\right]^T$, representing optimal process conditions such that it simultaneously maximizes process outputs, y_k ; k = 1, 2, …, K. The corresponding single objective function $\hat{f}$ to be maximized by the GA is defined in Eq. (2.2). In the GA procedure, the search for an optimal solution (decision) vector, **x***, begins from a randomly initialized population of probable (candidate) solutions. The solutions, usually coded in the form of binary strings (*chromosomes*), are then tested to measure their fitness in fulfilling the optimization objective. Subsequently, a main loop comprising following operations is performed: (i) selection of better (fitter) parent chromosomes, (ii) production of an offspring solution population by crossing over the genetic material between pairs of the fitter parent chromosomes, and (iii) mutation (bit-flipping) of the offspring strings.

Implementation of this loop generates a new population of candidate solutions, which as compared to the previous population, usually fares better at fulfilling the optimization objective. The best string that evolves after repeating the above described loop till convergence, forms the solution to the optimization problem (Nandi et. al., 2001 and 2002). The stepwise procedure for the GA-based optimization of an ANN or SVR model is provided in Appendix of this chapter (also see the flowchart in Figure 2.2).

2.4 MODELING AND OPTIMIZATION OF BENZENE ISOPROPYLATION OVER HBETA CATALYTIC PROCESS

Isopropylation of benzene is an important alkylation reaction in the petrochemical industry for the synthesis of cumene, which is the chief starting material in phenol production. In the last decade, several modifications of the zeolite beta were explored as potential catalysts in cumene synthesis (Perego et. al., 1994; Cavani et. al., 1997; Geatti et. al., 1997). More recently, isopropylation of benzene over HBeta (protonic form of Beta catalyst) was investigated by Sridevi et al. (2001). Beta is a crystalline alumino-silicate catalyst with high silica content and its important characteristic is that it is the only large pore zeolite with chiral pore intersections. It consists of 12-membered rings interconnected by cages formed by intersecting channels. The linear channels have pore opening dimensions of 5.7 $\overset{0}{\text{A}} \times 7.5 \overset{0}{\text{A}}$, whereas the tortuous channels with intersections of two linear channels have approximate dimensions of 5.6 $\overset{0}{\text{A}} \times 6.5 \overset{0}{\text{A}}$. The catalyst has pore volume of ≈ 0.2 cm^3/g. In the study by Sridevi et al. (2001), a phenomenological model for benzene isopropylation reaction was developed based on the isopropyl alcohol conversion in a continuous down-flow differential packed bed reactor taking into account the secondary reactions such as the dehydration of alcohol. This model however was restricted to the lower conversion (< 30%) of the limiting reactant, i.e., isopropyl alcohol, wherein heat and mass transfer resistances in the differential bed were assumed to be negligible. For maximizing yield and selectivity of cumene in the vapor phase alkylation of benzene with isopropyl alcohol over Hbeta catalyst, experiments were also conducted in a pilot plant scale reactor. Isopropylation of benzene involves a main reaction producing cumene and multiple side reactions as described below:

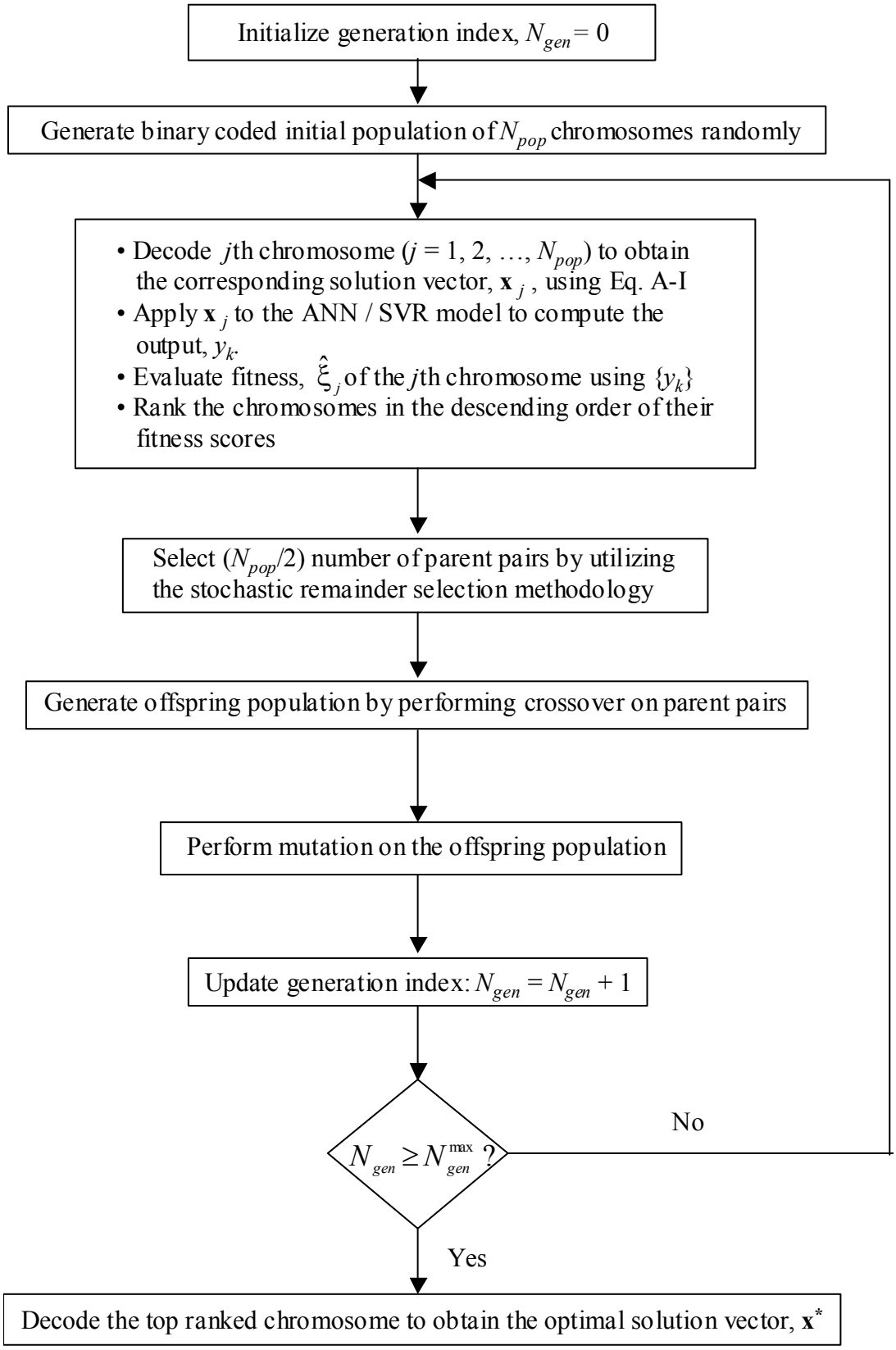


Figure 2.2: Flowchart of GA-based optimization of ANN/SVR model.

Main reaction :

Benzene + Isopropyl Alcohol → Cumene + Water (benzene alkylation)

Secondary reactions :

Cumene + Isopropyl Alcohol → p-Di-isopropyl Benzene + Water (cumene alkylation)

p-Di-isopropyl Benzene → m-Di-isopropyl Benzene (isomerization)

2 Isopropyl alcohol → Di-isopropyl ether + Water (alcohol dehydration)

2.4.1 Materials

Beta catalyst (1.5 mm extrudates with 20 % binder) in its active protonated form with Si to Al ratio of 15 was obtained from M/s UCIL, India, and utilized in the reaction; benzene and isopropyl alcohol (isopropanol) were of "Analytical Reagent" grade.

2.4.2 Experimental set-up

Vapor phase isopropylation of benzene was carried out in a pilot plant scale stainless steel reactor (see Fig. 2.3) with a preheater in its upstream and a condenser in the down-stream. The reactor specifications are as follows –material of construction: SS 316, internal diameter (ID): 25 mm, wall thickness: 6 mm, reactor length: 33 cm and catalyst bed height: 10-15 cm. Heating coils are wound around the reactor to provide proper heating and maintain temperature; the reactor is also jacketed with insulation to minimize the heat loss. During reactor operation, the liquid mixture of benzene and isopropyl alcohol was fed to the reactor by a positive displacement pump; hydrogen was used as the carrier gas. The condensed products collected were analyzed with a Flame Ionization Detector (FID) using a "Xylene Master" capillary column fitted to a Shimadzu 15A Gas Chromatograph (GC).

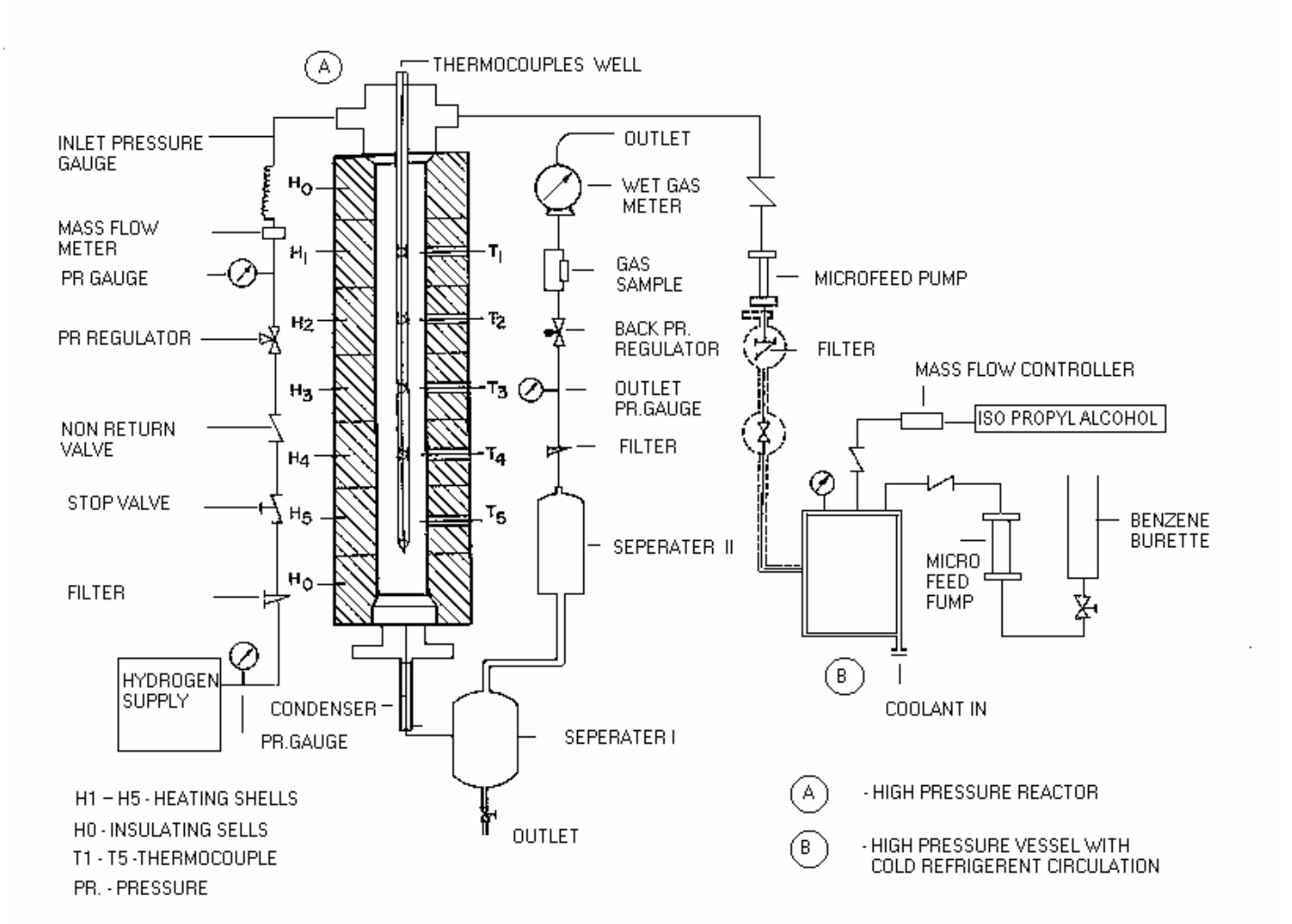


Figure 2.3: Schematic of the reactor set-up

2.4.3 Modeling and Optimization of Isopropylation Process

The objective of the present case study is to model and optimize the isopropylation pilot plant process with a view to simultaneously maximize yield and selectivity of cumene. Though the formation of cumene via isopropylation of benzene is the main reaction, a series of other components are also produced via side reactions. For developing a phenomenological model for the pilot plant scale integral reactor, it is necessary to consider the detailed kinetics of the stated multiple reactions in the conservation equations. Due to the tedious procedure involved in obtaining the requisite kinetic information, the exclusively data-based ANN-GA and SVR-GA methods were chosen for simultaneously maximizing the yield and selectivity of cumene. Accordingly, four reactor operating variables namely, reaction temperature (x_1), pressure (x_2), benzene to isopropyl alcohol mole ratio (x_3) and weight hourly space velocity (WHSV) (x_4), form the input space of the ANN and SVR-based reactor models. Cumene yield and selectivity defined as y_1 and y_2, respectively, are the output variables and they are evaluated as:

$$y_1 = \frac{100 \times \text{weight of cumene formed per unit time}}{\text{weight of isopropyl alcohol fed per unit time}} \tag{2.14}$$

$$y_2 = \frac{100 \times \text{weight of cumene formed per unit time}}{\text{weight of total aromatics produced per unit time}} \tag{2.15}$$

A total of 42 experiments (see Table 2.1) were conducted to seek the effect of varying values of four operating condition variables (x_1 to x_4) on cumene yield and selectivity. The lower (x_n^L) and upper (x_n^U) limits over which the four input variables were varied are: (i) temperature (^{0}C) (x_1) = [100.0, 280.0], (ii) pressure (atm) (x_2) = [1.0, 25.0], (iii) benzene to isopropyl alcohol mole ratio (x_3) = [1.0, 10.0], and (iv) weight hourly space velocity (WHSV) (hr^{-1}) (x_4) = [2.5, 13.0]. It can be easily noticed from the reaction stoichiometry that no change in number of moles occurs and hence pressure will have no significant effect. Low operating pressure should then be favored from a practical consideration. Accordingly, majority of the experiments were conducted at a low pressure (1 atm) with the remaining ones covering the pressure range of 4 to 25 atm. The maximum cumene yield and selectivity values obtained in the 42 experiments were: y_1 = 22.1 wt% (x_1 = 210 ^{0}C, x_2 = 25 atm., x_3 = 6, x_4 = 5 hr^{-1}) and, y_2 = 93.8 wt% (x_1 = 230 ^{0}C, x_2 = 25 atm, x_3 = 6, x_4 = 5 hr^{-1}), respectively. For maximizing cumene yield and selectivity simultaneously, the

single objective optimization problem (Eq. 2) was solved wherein ANN and SVR-based models were built and optimized separately using the GA. This way, it is possible to compare the modeling and optimization performance of the ANN-GA and SVR-GA hybrid formalisms.

2.4.4 ANN-Based Modeling of Benzene Isopropylation Process

An advantage of the ANN-based modeling is that unlike SVR, a comprehensive multiinput-multioutput (MIMO) model can be constructed for both the process outputs y_1 and y_2. To develop such a model, a three-layered MLP architecture was used. For conducting network training, the experimental data set (see Table 1) was partitioned into training (36 patterns) and test (6 patterns) sets. While the training set was utilized for the EBP based iterative updation of the network weights, the test set was used for simultaneously monitoring the generalization ability of the MLP model. The MLP architecture comprised four input ($N = 4$) and two output ($K = 2$) nodes. For developing an optimal MLP model, its structural parameter, namely the number of hidden nodes (L) and the EBP algorithm specific parameters, viz. learning rate (η) and momentum coefficient (α_{EBP}) were varied systematically; the effect of random initialization of the network weights also was examined by changing the seed values of the pseudo-random number generator (Nandi et. al., 2002). For choosing an overall optimal network model, the criterion used was least *RMSE* for the test set. The optimal MLP model that satisfied this criterion has five hidden nodes ($L = 5$), $\eta = 0.7$ and $\alpha_{EBP} = 0.02$. The values of training set RMSE (E_{trn}) and the test set RMSE (E_{tst}) along with the corresponding values of correlation coefficient (CC) are listed in Table 2.2. The low and comparable E_{trn} and E_{tst} values indicate good prediction and generalization ability of the trained network model. Good prediction and generalization performance of the model is also evident from the high and comparable CC values corresponding to both the outputs of training and test sets. Figure 2.4 (panels *a* and *b*) depicts a comparison of the outputs as predicted by the MLP model and their target values.

Table 2.1: Process data used for development of ANN and SVR-based models*

Expt. No.	Temperature (^{0}C)	Pressure (atm.)	Benz/IPA (mole ratio)	WHSV (hr^{-1})	Yield (wt %)	Selectivity (wt %)
1	110	1	8	3.3	0.07	77.03
2[a]	145	1	8	3.3	11.6	58.75
3	180	1	8	3.3	15.78	79.93
4	210	1	8	3.3	17.365	90.72
5	215	1	8	3.3	16.09	91.95
6	150	4	8	3.3	12.2	65.74
7	135	4	8	3.3	12.99	74.58
8[a]	110	4	8	3.3	0.71	80.82
9	100	4	8	3.3	0.19	75.02
10	110	1	10	3.3	0.55	67.74
11	110	1	8	3.3	0.24	54.85
12[a]	110	1	6	3.3	0.37	53.63
13	110	1	3	3.3	0.2	32.13
14	110	1	1	3.3	0.14	21.62
15	110	1	8	6.8	0.24	54.85
16	110	1	8	8	0.15	44.64
17	110	1	8	9.5	0.13	37.38
18	110	1	8	10.5	0.08	39.3
19[a]	110	1	8	12	0.09	39.13
20	110	1	8	13	0.07	39.1
21	105	1	8	6.8	0.3	70.38
22	110	1	8	6.8	0.24	54.85
23	115	1	8	6.8	0.35	48.25
24	130	1	8	6.8	4.61	76.68
25	185	1	8	6.8	9.2	59.23
26	210	1	6.5	3.3	20.04	91.8
27	155	1	6.5	3.3	16.93	77.4
28[a]	180	1	6.5	3.3	20.27	90.9
29	210	1	6.5	3.3	19.86	91.9
30	225	1	6.5	3.3	19.1	89.3
31	250	1	6.5	3.3	17.89	85.2
32	275	1	6.5	3.3	17.29	83.1
33	230	1	6.5	2.5	20.33	91.1
34	215	1	7	5	19.86	91.9
35[a]	215	10	7	5	19.54	92
36	215	18	7	5	18.68	89.1
37	215	25	7	5	17.74	86.8
38	195	25	6	5	18.92	85.6
39	210	25	6	5	22.1	93.7
40	230	25	6	5	22.02	93.8
41	250	25	6	5	21.35	90.7
42	280	25	6	5	20.48	86.2

* The data used as test set are shown using superscript '*a*'.

2.4.5 SVR-Based Modeling of Isopropylation Reaction

Here, two SVR models for cumene yield and selectivity, respectively, were developed using the same training set as used to obtain the ANN-based model. The generalization performance of the SVR models was evaluated using the respective test sets. In the present study, an SVR implementation known as "ε-SVR" in the LIBSVM software library (Chang and Lin, 2001), has been used to develop the two SVR-based models. The LIBSVM package utilizes a fast and efficient method known as sequential minimal optimization (SMO) (Joachims, 1998 and Platt, 1998) for solving large quadratic programming problems and thereby estimating function parameters α, α^* and b (see Eq. 2.12). To obtain an optimal SVR model, it is necessary to examine the effects of kernel function and other algorithm-specific parameters; the three kernel functions that were tested are, polynomial, RBF and sigmoid. Among these, RBF resulted in the least RMSE values for the training and test sets of the outputs, y_1 and y_2. The numbers of support vectors used by the SVR algorithm for fitting the yield and selectivity models were 30 and 33, respectively. The optimal values of the four SVR-specific parameters namely, width of RBF kernel (σ), cost coefficient (C), loss function parameter (ε_{loss}) and tolerance for termination criterion (ε_{tol}) that minimized the E_{trn} and E_{tst} corresponding to yield and selectivity models are listed in Table 2.2; also listed are the values of correlation coefficients for the training and test set predictions along with the corresponding *RMSE* values for both the models. A comparison of the SVR model predicted and the corresponding target values of cumene yield and selectivity are depicted in the panels *c* and *d* of Fig. 2.4.

In the above described modeling simulations, the size of the training set (36 patterns) was closer to its statistical limit (30 patterns) below which any prediction made by the model is considered arbitrary. To avoid the possibility of arbitrary predictions arising out of insufficient training data, a method known as "*k*-fold cross-validation" is commonly used to select an optimal model. In this approach, the training set is first divided randomly into *k* equal sized subsets. Next, *k* numbers of models are constructed by leaving out a different subset each time, with the remaining (*k*-1) subsets collectively representing the training set. An average of the RMSEs corresponding to the left-out subsets, known as "cross-validation error (E_{cv})" gives an estimate of the model performance if a large-sized data set was available for building the model. Accordingly, in a separate exercise numerous SVR models for cumene yield and selectivity were

developed by changing the SVR's algorithm specific parameters and simultaneously using the cross-validation (CV) strategy. For performing CV, the training set was divided into six subsets (k = 6) each comprising six patterns. It was observed from these simulations that the SVR based yield and selectivity models described earlier (see Table 2.2) also minimized the corresponding CV errors. The minimum E_{cv} values for the cross-validated yield and selectivity models were 0.753 and 4.847, respectively. A comparison of the test set RMSEs (0.712 and 4.986) pertaining to the yield and selectivity models (see Table 2.2) with the corresponding CV errors, (0.753 and 4.847) reveals that they are in close agreement. Such a close match between the cross-validation and test set RMSEs indicate that the size of the training data was adequate for building the SVR models.

Table 2.2: Performance indicators of ANN and SVR models

	ANN-based Model				**SVR-based Model**			
	Yield		**Selectivity**		**Yield** *		**Selectivity** #	
	E_{trn}	E_{tst}	E_{trn}	E_{tst}	E_{trn}	E_{tst}	E_{trn}	E_{tst}
RMSE	0.492	0.438	4.641	4.678	0.842	0.712	6.595	4.986
CC	0.998	0.999	0.974	0.976	0.995	0.999	0.955	0.962

*, # ε-SVR parameters: (i) yield model: C = 264, $\gamma = \frac{1}{2\sigma^2}$ = 0.9, ε_{loss} = 0.00001, ε_{tol} = 0.00001 (ii) selectivity model: C = 132, $\gamma = \frac{1}{2\sigma^2}$ = 0.8, ε_{loss} = 0.00001, ε_{tol} = 0.00001.

2.4.6 GA-based optimization of the ANN and SVR models

While performing optimization of the input space of the ANN and SVR models, the best values of following GA-specific parameters were chosen heuristically: population size (N_{pop}) = 25, crossover probability (p_{cross}) = 0.82, mutation probability (p_{mut}) = 0.05, and maximum number of generations (N_{gen}) = 100. In order to obtain the best set of operating conditions, GA runs were replicated several i.e. 50 times, using different random number generator seeds. A different seed value generates a dissimilar population of initial candidate solutions, thus assisting in the exhaustive search of the solution space and thereby locating the globally optimum solution. For computing fitness values ($\hat{\xi}_j$) of the candidate solutions, following fitness function was employed:

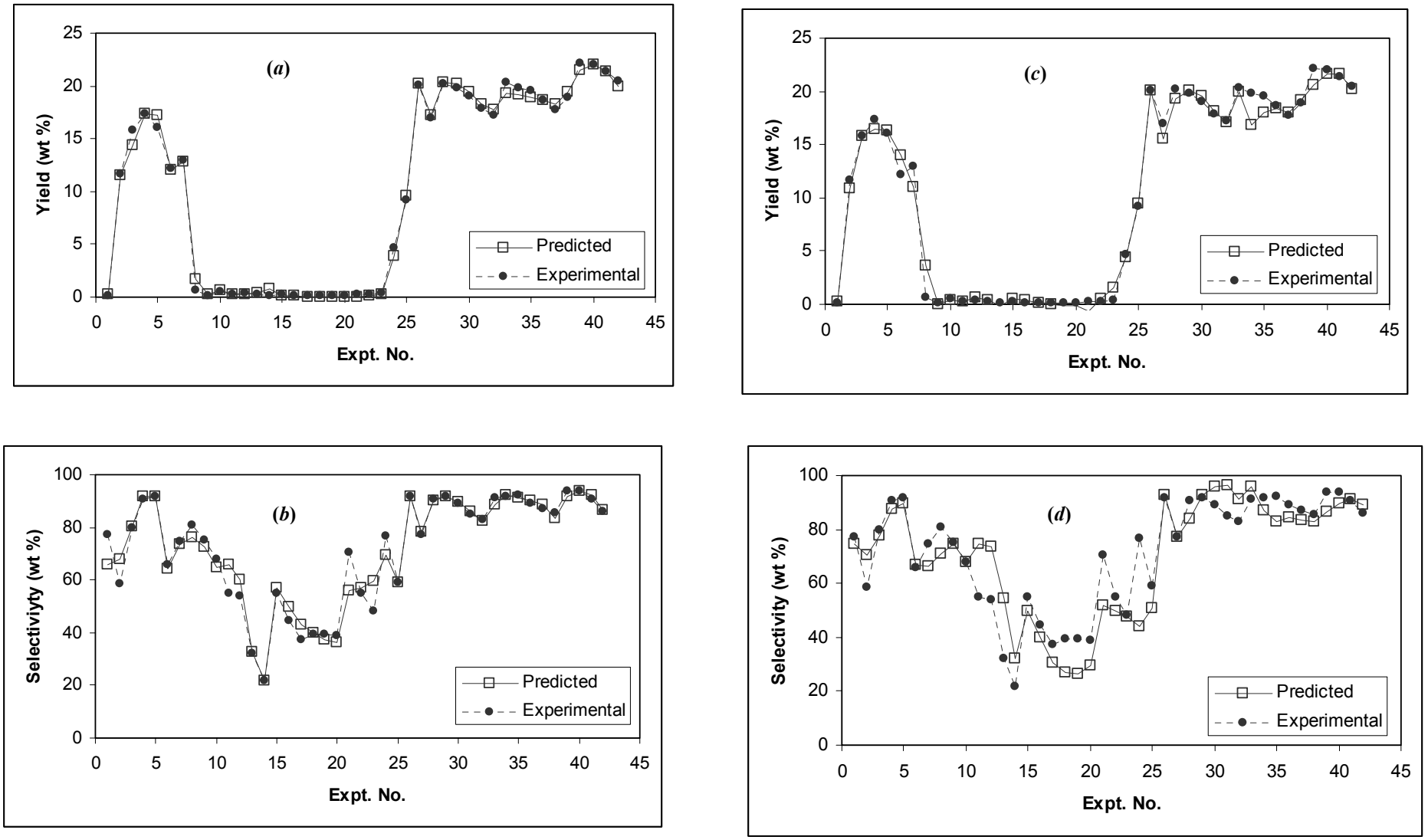


Figure 2.4: Yield and selectivity values predicted by the ANN (panels *a* and *b*) and SVR (panels *c* and *d*) models.

88

$$\hat{\xi}_j = \frac{\hat{w}_1 y_1 + \hat{w}_2 y_2}{100} = \frac{y_1 + y_2}{200};\ \ \hat{w}_1 = \hat{w}_2 = 0.5;\ \ j = 1, 2, ..., N_{pop} \tag{2.16}$$

The three best operating condition sets given by the GA-based optimization of the ANN and SVR models are tabulated in Table 2.3. It is seen from the tabulated optimized values that the ANN-GA method has yielded an overall optimal solution (x_1^* = 271.5, x_2^* = 3.38, x_3^* = 3.69, x_4^* = 12.83) maximizing both the yield (24.88%) and selectivity (99.04%) of cumene. Moreover, the best set of operating conditions given by the SVR-GA method (x_1^* = 270.1, x_2^* = 3.75, x_3^* = 3.55, x_4^* = 13.27) is similar to that obtained using the ANN-GA method. This set though results in approximately same value (24.8%) of cumene yield as given by the ANN-GA method, the corresponding selectivity value (95.76%) is marginally inferior to that from the ANN-GA method (99.04%). Similar observation can also be made from the other two solutions given by the SVR-GA method. The small differences in the selectivity values maximized by the SVR-GA method are possible if: (i) the GA has found a locally—and not globally—optimum solution, and (ii) the function fitted by the SVR differs from that fitted by the ANN. To test first possibility, input space of the SVR-based selectivity model was searched rigorously using GAs, by restricting the search in the neighborhood of the best solution given by the ANN-GA strategy. However, these simulations showed no significant improvement over the best solution given by the SVR-GA method. This result clearly suggests that the GA has indeed searched the global or the tallest local maximum corresponding to the SVR-based model.

A comparison of the training set RMSE values and the corresponding correlation coefficient magnitudes in respect of ANN and SVR models (see Table 2.2) reveals that there exists a small difference in the values. It can thus be inferred that the functions fitted by the ANN and SVR are indeed different. These differences however affect the optimized solutions given by the two methods only marginally. The differences in the functions fitted by the ANN and SVR can arise owing to very different fitting strategies employed by the two methods — while the ANNs approximate the nonlinear input-output relationships directly, in SVR the inputs are first mapped into a high dimensional feature space and then correlated linearly with the output. It is also possible that ANN and SVR react differently to the noise, which is inherent to the experimental data.

Table 2.3: Optimized process conditions given by ANN-GA and SVR-GA methodologies

Soln. No.	ANN-GA						SVR-GA					
	Optimized Inputs				Maximized Outputs		Optimized Inputs				Maximized Outputs	
	Temp. (^{0} C) (x_1^*)	Press. (atm.) (x_2^*)	Benz/IPA (mol ratio) (x_3^*)	WHSV (hr^{-1}) (x_4^*)	Yield (wt %) (y_1^*)	Selectivity (wt %) (y_2^*)	Temp. (^{0} C) (x_1^*)	Press. (atm.) (x_2^*)	Benz/IPA (mol ratio) (x_3^*)	WHSV (hr^{-1}) (x_4^*)	Yield (wt %) (y_1^*)	Selectivity (wt %) (y_2^*)
1	271.5	3.38	3.69	12.83	24.88	99.04	270.1	3.75	3.55	13.27	24.8	95.76
2	267.2	1.567	4.05	12.83	24.84	98.90	266.1	3.87	3.95	12.93	24.0	93.8
3	270.08	3.6	4.05	11.76	24.82	98.74	266.1	3.63	3.85	12.35	24.63	93.96

2.4.7 Experimental Verification of GA-Optimized Solutions

It is noticed from the optimized reactor conditions listed in Table 2.3 that solutions 1 and 3 given by the ANN-GA method and the three solutions provided by the SVR-GA method fall in a narrow range. However, ANN-GA based solutions result in slightly higher (upto 3.31%) selectivity values when compared to the SVR-GA based best solution. To verify their validity, all the three sets of optimized operating conditions given by the ANN-GA method were subjected to experimental verification and the results obtained thereby are listed in Table 2.4. As can be observed from the tabulated values, the experimental results match the predictions of the ANN-GA method with excellent accuracy. In fact, except the yield value in the second verification experiment, which shows 4.41% error from its GA-maximized value of 24.84%, all other yield and selectivity values validate the corresponding GA-maximized values within 1% accuracy. It is possible to explain the occurrence of 4.41% error on the basis of sparseness in the experimental data. As noted previously, most experiments were conducted at a low pressure (1atm) and therefore experimental data are sparsely located in the higher pressure (>1 atm) range. In regions of sparse data, it is possible that the function fitted by the ANN or SVR shows minor deviations from the true trend line. Consequently, the solution searched by the GA also deviates from its true optimum. In essence, the sparseness of the experimental data in the pressure range of 1 to 4 atm may have resulted in the 4.41% error for the second validation experiment, wherein GA-optimized pressure value was 1.6 atm. From the experimental data used for building the ANN and SVR methods, it is seen that maximum values of the cumene yield and selectivity were 22.1% (expt. no. 39) and 93.8% (expt. no. 40), respectively. A comparison of these values with those from the verification experiments reveal that the GA-based optimized conditions have simultaneously improved the cumene yield and selectivity by 2.59% and 5.1%, respectively.

2.5 CONCLUSION

In this study, two hybrid process modeling and optimization strategies integrating artificial neural networks/support vector regression with the genetic algorithms have been employed for modeling and optimization of benzene isopropylation pilot plant scale catalytic process. The SVR is a novel machine learning based nonlinear modeling paradigm possessing certain unique features such as its formulation involves solution of a

Table 2.4: Results of experimental verification

Expt. No.	Experimental Conditions				Yield (output 1)			Selectivity (output 2)		
	Temp. (^{0}C)	Pressure (atm)	Benz/IPA (mole ratio)	WHSV (hr^{-1})	GA-maximized value (wt %)	Exptl. value (wt %)	Error (%)	GA-maximized value (wt %)	Exptl. value (wt %)	Error (%)
1	271.5	3.4	3.7	12.8	24.88	24.69	0.77	99.04	98.98	0.06
2	267.2	1.6	4.0	12.8	24.84	23.79	4.41	98.90	98.70	0.20
3	270.0	3.6	4.0	11.8	24.82	24.58	0.98	98.74	98.65	0.09

quadratic programming problem possessing a single minimum. Thus, the rigorous heuristic involved in locating the global minimum (as in ANNs) is avoided in SVR-implementation. In the two hybrid strategies, a process model is developed using an ANN or SVR method following which the input space of that model is optimized using GAs such that the process performance is maximized. The hybrid approach involving SVR and GAs is a new method wherein, similar to the ANN-GA formalism, process modeling and optimization can be conducted exclusively using historic process data. Using ANN-GA and SVR-GA strategies separately, a number of optimized sets of operating conditions, which simultaneously maximize the yield and selectivity of cumene (desired product of isopropylation reaction) were obtained. It was observed that the best sets of process operating conditions given by the two strategies were quite similar. Finally, the optimized solutions given by the ANN-GA method when subjected to experimental verification, matched the maximized cumene yield and selectivity values with excellent accuracy.

2.6 APPENDIX: STEPWISE PROCEDURE FOR THE GA-BASED OPTIMIZATION OF AN ANN OR SVR MODEL

Step 1 (*Initialization*): Set generation index (N_{gen}) to zero and generate a population of N_{pop} binary strings (chromosomes) randomly; each string consisting of l_{chr} bits is divided into N segments equal to the number of decision (input) variables to be optimized.

Step 2 (*Fitness computation*): Decode jth binary-coded chromosome (j = 1, 2, …, N_{pop}) to obtain its equivalent decimal-valued solution vector ($\mathbf{x}_j$) using:

$$x_{j,n} = x_n^L + \frac{\left(x_n^U - x_n^L\right) \times S_n}{2^{l_n} - 1} \; ; \quad \sum_{n=1}^{N} l_n = l_{chr} \qquad \text{(A-I)}$$

where, x_n^L and x_n^U respectively refer to the lower and upper bounds on x_n; l_n is the length of nth binary segment and S_n denotes the decimal equivalent of the nth binary segment. Next, depending upon the model to be optimized, vector $\mathbf{x}_j$ is used to compute the output of an ANN or SVR model; this output is subsequently used to calculate the fitness value ($\hat{\xi}_j$) of the jth candidate solution (see Eq. 16). Upon computing the fitness scores of N_{pop} candidate solutions in the current population, the solutions are ranked in the decreasing order of their fitness scores.

Step 3 (*Parent selection*): From the current population, choose N_{pop} number of parent chromosomes to form the mating pool. The members of this pool, which are used to produce offspring population, possess relatively high fitness scores. The commonly used parent selection techniques are the *Roulette-Wheel* (RW) method or its more stable variant known as the *stochastic remainder selection* (SRS) [18].

Step 4 (*Crossover*): Randomly select $N_{pop}/2$ number of parent pairs from the mating pool and perform crossover operation on each pair with probability equal to p_{cross} ($0 < p_{cross} \leq 1$). In crossover, parent strings are cut at the same randomly chosen crossover point to obtain two substrings per parent. The substrings are then mutually exchanged between the parents and combined to form two offspring chromosomes. This crossover operation is performed on all the parent pairs in the mating pool to obtain an offspring population of the size of the mating pool.

Step 5 (*Mutation*): Flip (mutate) the bits of the offspring strings where the probability of a bit getting flipped (zero to one or vice versa) is equal to p_{mut}. The recommended range of p_{mut} is [0.01-0.05].

Step 6 : Increment the generation index: $N_{gen} = N_{gen} + 1$.

Step 7 : Repeat steps 2-6 on the newly generated offspring strings until convergence is achieved. The criteria for the convergence are: N_{gen} exceeds its maximum limit ($N_{gen}^{\max}$), or the fitness score of the best (fittest) string in the offspring population undergoes a very small or no change over successive generations. After convergence, the string possessing highest fitness value is decoded to obtain the optimized decision variable vector, $\mathbf{x}^*$.

2.7 NOMENCLATURE

C : cost function

E_{trn} : RMSE for training set

E_{tst} : RMSE for test set

f_k : function correlating kth output with inputs

$\hat{f}$: single aggregated objective function

K : number of output variables (equal to number of nodes in output layer of network)

k : index for output variable; also, number of folds in cross-validation

l_n : length of nth binary segment

L : number of nodes in hidden layer of neural network model

N : number of input variables (equal to number of nodes in input layer of network)

N_{pop} : population size

$N_{gen}^{\max}$: maximum number of allowable generations for GA

$\wp$: feature space

P : No of input-output example patterns used in training

$P(.)$: Probability density function

p_{cross} : probability of crossover

p_{mut} : probability of mutation

$RMSE$: Root mean squared error

R_{emp} : empirical risk

R_{reg} : regression risk

S_n : decimal equivalent of nth binary segment

$\Im$: training data set

$\mathbf{x}$: N dimensional input vector

$\mathbf{x}^*$: optimal decision vector

x_n^L , x_n^U : lower and upper bounds of nth input variable

$\hat{y}_{i,k}$: desired value of kth output

$y_{i,k}^p$: neural network predicted kth output

$\mathbf{w}$: weight vector

$\mathbf{w}_k$: kth function's parameter vector

$\hat{w}_k$: weighting coefficient

$\|\mathbf{w}\|^2$: Euclidean norm

Greek Symbols

α_{EBP} : momentum coefficient

$\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$: vectors of Lagrange's multiplier

ε : precision parameter

ε_{loss} : loss function parameter

ε_{tol} : tolerance for termination criterion

σ : width of kernel of radial basis function

η : learning rate

λ : regularization constant

λ : regularization constant

ξ , ξ^* : slack variables

$\hat{\xi}_j$: fitness value of *j*th candidate solution

Φ : function termed feature

2.8 REFERENCES

1. Agarwal, M., Jade, A. M., Jayaraman, V. K. and Kulkarni, B. D. "Supprot Vector Machines : A Useful Tool for Process Engineering Applications," ***Chem. Engg. Progr.***, **99**, 57 - 62, (2003).
2. Bishop, C. M. "Neural Networks and their Applications," ***Rev. Sci. Instru.***, **65**, 1803 - 1856 (1994).
3. Bulsari, A. B. "Applications of Artificial Neural Networks in Process Engineering," ***J. Syst. Engng.*, 4,** 131 - 170, (1994)**.**
4. Bulsari, A. B. (Ed.), ***Neural Networks for Chemical Engineers,*** Elsevier, Amsterdam (1995).
5. Burges, C. "A Tutorial on Support Vector Machines for Pattern Recognition," ***Data Mining and Knowledge Discovery***, **2**, 1 – 47, (1998).
6. Cavani, F., Girotti, G., Arrigoni, V. and Terzoni, G. "Alkylation Catalyst for Aromatic Compounds for Lower Olefins", US patent 5650547 A, July 22, (1997).
7. Chang, C. and Lin, C. "LIBSVM: A Library for Support Vector Machines," software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm (2001).
8. Cherkassky V. and Ma Y., "Practical Selection of SVM Parameters and Noise Estimation for SVM Regression," Submitted to ***Neurocomputing*** (special issue on SVM) (2002), available at http://www.ece.umu.edu/users/cherkass/N2002-SI-SVM-13-whole.pdf
9. Geatti, A., Lenarda, M., Storaro, L., Ganzerla, R. and Perissinotto, M. "Solid Acid Catalysts from Clays: Cumene Synthesis by Benzene Alkylation with Propene Catalyzed by Cation Exchanged Aluminium Pillared Clays", ***J. Mol. Catal A: Chem.***, **121**, 111 – 118, (1997).
10. Davis, L., (Ed.), ***Handbook of Genetic Algorithms***, Van Nostrand Reinhold, New York (1991).
11. Deb, K., ***Optimization for Engineering Design: Algorithms and Examples***, Perntice Hall, New Delhi, (1995).
12. Edgar, T. F. and Himmelblau, D. M. ***Optimization of Chemical Processes***, McGraw-Hill, Singapore, (1989).
13. Fahlman, S. E. "An Empirical Study of Learning Speed in Back-propagation Networks," Technical Report CMU-CS-88-162, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA (1988).

14. Freeman, J. A. and Skapura, D. M. ***Neural Networks: Algorithms, Applications and Programming Techniques***, Addision-Wesley, Reading, MA (1991).
15. Goldberg, D. E. ***Genetic Algorithms in search, Optimization, and Machine Learning***, Addison-Wesley, Reading, MA (1989).
16. Huang, K., Zhan, X-L, Chen, F-Q and Lü, D-W "Catalyst Design for Methane Oxidative Coupling by Using Artificial Neural Network and Hybrid Genetic Algorithm," ***Chem. Engg. Sci.***, **58**, 81 - 87, (2003).
17. Immanuel, C. D. and Doyle III, F. J. "Open Loop Control of Particle Size Distribution in Semi-batch Emulsion Copolymerization Using Genetic Algorithms," ***Chem. Engg. Sci.***, **57**, 4415 - 4427, (2002).
18. Jack, L. B. and Nandi, A. K. "Fault Detection Using Support Vector Machines and Artificial Neural Networks Augmented by Genetic Algorithms," ***Mech. Sys. Sig. Proc.***, **16**, 372 - 390, (2002).
19. Joachims, T. "Making Large-scale SVM Learning Practical," In B. Schölkopf, C. J. C. Burges, and A. J. Smola (Eds.) *Advances in Kernel Methods – Support Vector Learning*, MIT Press, Cambridge, MA (1998).
20. Lucasius, C. B. and Kateman, G. "Understanding and Using Genetic Algorithms, Part I: Concepts, Properties and Context", ***Chemo. & Intell. Lab. Syst.***, **19**, 1-33, (1993).
21. Lucasius, C. B. and Kateman, G. "Understanding and Using Genetic Algorithms, Part II: Representation, Configuration and Hybridization", ***Chemo. & Intell. Lab. Syst.***, **25**, 99-145, (1994).
22. Meima, G. R. "Advances in Cumene Production", *CATTECH***,** June, 5-12 (1998)**.**
23. Mukherjee, S., Osuna, E. and Girosi, F. "Nonlinear Prediction of Chaotic Time Series Using Support Vector Machines," In *Proc. IEEE Workshop on Neural Networks for Signal Processing 7* (IEEE NNSP'97), 511 – 519, (1997).
24. Müller, K. R., Smola, A., Ratsch, G., Schölkopf, B. Kohlmorgen, J. and Vapnik, V. "Predicting Time Series with Support Vector Machines," In W. Gerstner, A. Germond, M. Hasler and J-D. Nicoud (Eds.) *Artificial Neural Networks – ICANN'97*, Berlin, Springer Lecture Notes on Computer Science, Vol. 1327, 999 – 1004, (1997).
25. Nandi, S. , Ghosh, S, Tambe, S. S. and Kulkarni, B. D. "Artificial Neural-Network-Assisted Stochastic Process Optimization Strategies," ***AIChE Jour.*****,** **47**, 126 - 141, (2001).

26. Nandi, S., Mukherjee, P., Tambe, S. S., Kumar R., and Kulkarni, B. D. "Reaction Modeling and Optimization Using Neural Networks and Genetic Algorithms: Case Study Involving TS-1 Catalyzed Hydroxylation of Benzene", ***Ind. Engg. Chem. Res***., **41**, 2159 - 2169, (2002).
27. Nougués, J. M., Gran, M. D. and Puigjaner, L. "Parameter Estimation with Genetic Algorithms in Control of Fed-batch Reactor," ***Chem. Engg. & Proces.*** **41**, 303 - 309, (2002).
28. Perego, C. Pazzuconi, G. Girotti, G. and Terzoni, G. "Process for the Preparation of Cumene", *Eur. Pat. Appl. EP629599 A1*, Dec. 21 (1994).
29. Platt, J. C. "Fast Training of Support Vector Machines Using Sequential Minimal Optimization," In B. Schölkopf, C. J. C. Burges, and A. J. Smola (Eds.) *Advances in Kernel Methods – Support Vector Learning*, MIT Press, Cambridge, MA (1998).
30. Ramanathan, S. P., Mukherjee, S., Dahule, R. K., Ghosh, S., Rahman, I., Tambe, S. S. Ravetkar, D. D. and Kulkarni., B. D. "Optimization of Continuous Distillation Columns Using Stochastic Optimization Approaches," ***Trans. Inst. Chem. Eng***., **79**, 310 - 321, (2001).
31. Riedmiller, M. and Braun, H. "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm," Proc. of the IEEE Int. Conf. On Neural Networks, San Fransisco, CA, Mar. 28-Apr. 1, (1993).
32. Rodemerck, U., Wolf, D., Buyevskaya, U. V., Claus, P., Senkan, S. and Baerns, M. "High Throughput Synthesis and Screening of Catalytic Materials Case Study of the Search for a Low Temperature Catalyst for Oxidation of Low Concentration Propane," ***Chem Engg J.***, **82**, 3 - 11, (2001).
33. Rumelhart, D., Hinton, G. and Williams, R. "Learning Representations by Backpropagating Errors***", Nature, 323, 533 (1986).***
34. Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J. and Williamson, R. C. "Estimating Support of a High-dimensional Distribution," ***Neural Comput***. **13**, 1443 – 1471, (2001).
35. Smola, A., Schölkopf, B. and Müller, K. R. "The Connection Between Regularization Operators and Support Vector Kernels," ***Neural Networks***, **11**, 637 - 649, (1998).
36. Sridevi, U., Rao, B. K. B., Pradhan, N. C., Tambe, S. S., Satyanarayana, C. V. and Rao, B. S. "Kinetics of Isopropylation of Benzene Over Hbeta Catalyst", ***Ind. Engg. Chem. Res.***, **40**, 3133 – 3138, (2001).

37. Stephanopoulos, G. and Han, C. “Intelligent Systems in Process Engineering: A Review”, ***Comp. & Chem. Engg.***, **20**, 743 - 791, (1996).
38. Sumanwar, V. S., Jayaraman, V. K., Kulkarni, B. D., Kusumakar, H. S., Gupta, K. and Rajesh, J. “Solution of Constrained Optimization Problems by Multiobjective Genetic Algorithms,” ***Comp. & Chem. Engg.***, **26**, 1481 - 1492, (2002).
39. Tambe, S. S., Kulkarni, B. D. and Deshpande, P. B. ***Elements of Artificial Neural Networks with selected applications in Chemical Engineering, and Chemical & Biological Sciences***, Simulations & Advanced Controls, Louisville, KY (1996).
40. Vapnik, V. ***The Nature of Statistical Learning Theory***, Springer Verlag, New York (1995).
41. Vapnik, V., Golowich, S. and Smola, A. “Support Vector Method for Function Approximation, Regression Estimation and Signal Processing,” ***Adv. in Neural Inform. Proces. Syst.***, **9**, 281 - 287, (1996).
42. Vapnik, V., ***Statistical Learning Theory***, John Wiley, New York (1998).
43. Venkatasubramanian, V. and Sundaram, A. “Genetic Algorithms: Introduction and Applications” in ***Encyclopedia of Computational Chemistry***, John Wiley, Chichester, U.K. (1998).
44. Yee, K., Ray, A. K and Rangaiah, G. P. “Multiobjective Optimization of an industrial Styrene Reactor,” ***Comp. & Chem. Engg.***, **27**, 111 - 130, (2003).

CHAPTER 3

MODELING AND MONITORING OF BATCH PROCESSES USING PRINCIPAL COMPONENT ANALYSIS (PCA) ASSISTED GENERALIZED REGRESSION NEURAL NETWORKS (GRNN)

Published in ***Biochemical Engineering Journal***, Vol. **18**, (No. 5), 193 – 210 (2004).

Abstract

Multivariate statistical methods namely, principal component analysis (PCA) and partial least squares (PLS), which perform dimensionality reduction and regression, respectively, are commonly used in batch process modeling and monitoring. While PCA is used to monitor whether process input variables are behaving normally, the PLS is used for predicting the process output variables. A significant drawback of the PLS is that it is a linear regression formalism and thus makes poor predictions when relationships between process inputs and outputs are nonlinear. In the present chapter, a formalism integrating PCA and generalized regression neural networks (GRNNs) is introduced for batch process modeling and monitoring. The advantages of this PCA-GRNN hybrid methodology are: (i) process outputs can be predicted accurately even when input-output relationship is nonlinear, and (ii) unlike other commonly used artificial neural network paradigms (such as multilayer perceptron), training of GRNN is a one-step procedure, which helps in faster development of nonlinear input-output models. The effectiveness of the PCA-GRNN strategy has been successfully demonstrated by conducting two case studies involving penicillin production and protein synthesis.

3.1 INTRODUCTION

Batch processes exhibit four important characteristics viz. flexible, unsteady, finite-duration operation, and nonlinear dynamical behavior. These processes, which are extensively used in the production of biochemicals as also pharmaceuticals and agrochemicals, are typically operated as follows: (i) a precise sequence is used for charging the vessel with reactants, catalyst, promoter, etc., (ii) a controlled environment is ensured during the progress of a reaction wherein operating variables such as feed-rate, temperature, pressure, and agitation rate are varied according to a specified dynamic trajectory, and (iii) the products are discharged at the end of a finite duration. The product is analyzed (mostly offline) after the batch completion although if feasible and economical, the product quality measurements are also done at finite time intervals as the batch run progresses. To get high quality products consistently, the process operating variables should follow their specified trajectories precisely. However, disturbances arising from the deviations in the specified trajectories, errors in charging the vessel with materials, and variations in impurities, do lead to batch-to-batch variations (Nomikos and MacGregor, 1994a), which affect the product quality adversely. Process variability, which in general is detrimental to maintaining the product quality and safe process operation, can be reduced by employing an efficient monitoring and control system. Such a system while working online must be capable of quickly identifying any abnormal process behavior so that corrective measures could be taken immediately.

The computer-based data logging and control systems used by today's batch processes collect and store process data continuously or at fixed time intervals. These measurements are used for the on-line monitoring of a batch process and thereby timely detecting and diagnosing any abnormal process behavior. There exist mainly four approaches for monitoring and detecting/diagnosing faults in chemical/biochemical processes. The formalisms used by these approaches are: (i) dynamic phenomenological (first principles) model (Willsky, 1976; Iserman, 1984; Kozub and McGregor, 1992; Basseville, 1998), (ii) a knowledge-based expert system (Cheung and Stephanopoulos, 1989; Janusz and Venkatasubramaniun, 1991; Bonastre et. al., 2001) (iii) artificial neural networks (ANNs) (Hoskins and Himmelblau, 1988; Watanabe et. al., 1989; Venkatasubramanium et. al., 1990; Vaidyanathan and Venkatasubramanian, 1992; Vora et. al., 1997), and (iv) multivariate

statistical (MVS) methods namely principal component analysis (PCA) and partial least squares (PLS) (Geladi and Kowalski, 1986; Wold et. al., 1987). Among these, the fourth approach utilizing MVS methods is by far most popular.

The PCA is a dimensionality reduction technique for compressing noisy and correlated process input measurements into a smaller, informative latent variable space. It forms latent variables (*principal components* or *scores*) that are expressed as the linear combinations of the input variables and the corresponding weights (*loadings*). When measurements are highly correlated, first few principal components (PCs) capture maximum amount of variance in the measurements. Thus, measurements can be represented using fewer PCs without significant loss of the information. A PCA-based technique known as multiway principal component analysis (MPCA) has been developed by Wold et al. (1987) whereby dimensionality reduction can be realized for multivariate dynamical data emanating from several batches.

The PLS is a PCA-related technique that simultaneously finds latent variables for both the input and output measurement sets following which the latent variables are correlated linearly. The principal advantage of the PLS method is that outputs can be predicted using only a few PCs that capture the maximum variance in the input-output measurements. Multiway version of the PLS (MPLS) has also been proposed (Nomikos and MacGregor, 1994) to deal with the batch systems for which product quality data are available along with the measurements of the input variables. In recent years, several variants of PCA, PLS, MPCA and MPLS have been introduced (see e.g., Nomikos and MacGregor, 1994b, 1995; Kosanovich et. al., 1996; Zheng et. al., 2001). The principal advantage of these formalisms is that the information contained in the original database is extracted in a lower dimensional vectors and matrices; the data reduction achieved thereby is significant, which simplifies the monitoring task substantially. The PLS and MPLS though are fast and efficient methods, they suffer from a significant drawback that being linear methods they capture only the linear relationships between the principal components of the predictor (input) and response (output) variables. Consequently, they perform poorly in predicting response variables of nonlinearly behaving batch processes, which are abundant in chemical/biochemical industry.

To overcome the limitation imposed by the linearity of PLS/MPLS-based models, the neural network PLS (NNPLS) method was proposed (Qin and McAvoy, 1992; Holcomb and

Morari, 1992). In NNPLS, an outer mapping is performed first wherein measurement matrices of the input and output variables are decomposed into their respective scores and loadings matrices. Next, an inner model as defined by

$$\mathbf{u}_k = N(\mathbf{t}_k) + \mathbf{r}_k \tag{3.1}$$

is constructed where $\mathbf{u}_k$ and $\mathbf{t}_k$ are the kth score vectors of the input and output matrices respectively, $\mathbf{r}_k$ is the residual, and $N(.)$ denotes a feed-forward neural network (FFNN) model such as the multilayered perceptron (MLP) and the radial basis function network (RBFN). The NNPLS method differs from the direct ANN approach in that the input-output data are not directly used to train the FFNN but are preprocessed by the PLS outer transform (Qin, 1997). This transform decomposes a multivariate regression problem into a number of univariate regressors wherein each regressor is implemented separately using a simple single input – single output (SISO) FFNN. This neural network can be easily trained using various algorithms, such as error-back-propagation (Rumelhart et. al., 1986), conjugate gradient (Reifman and Vitela, 1994), Quickprop (Fahlman, 1988) and Rprop (Reidmiller and Braun, 1993). The NNPLS is a powerful technique for monitoring nonlinear batch processes although usage of even a simple SISO neural network does involve some heuristic in choosing an optimal value of the network's structural parameter namely the number of nodes in the hidden layer. Depending upon the choice of a training algorithm, the values of algorithm-specific parameters (e.g., learning rate and momentum coefficient in the EBP algorithm) also need to be fixed heuristically. A heuristic search of the network's weight (parameter) space is also required to obtain a globally optimum solution, since none of the above-stated training algorithms guarantees convergence to the globally optimum solution in a single training run. Such a computation intensive heuristic is especially undesirable when the network is trained online and the time duration to take a corrective action in the event of an abnormal process behavior is short. Thus, in the present chapter we propose a process modeling and monitoring formalism integrating PCA and a numerically efficient ANN paradigm namely *Generalized Regression Neural Network* (GRNN) for nonlinearly correlating the process output variables with the operating variables. The principal advantages of the GRNN-based models are: (i) unlike PLS they can efficiently and simultaneously approximate nonlinear multiinput-multioutput (MIMO) relationships and, (ii) they can be developed in a significantly shorter

time as compared to the MLP or RBFN-based process models since their training which is a one step procedure, involves fixing of only a single free parameter.

This chapter is organized as follows. In section 3.2, an overview of the PCA/MPCA, PLS and GRNNs is presented. Next in section 3.3, the details of the proposed hybrid formalism integrating PCA and GRNN (hereafter termed "PCA-GRNN") are provided, and section 3.4 presents the results of the two simulated case studies wherein PCA-GRNN formalism has been utilized for modeling and monitoring of two bioprocesses.

3.2 MULTIVARIATE STATISTICAL METHODS AND GRNNS

3.2.1 PCA / MPCA / PLS

Experimental input variable data from a batch process can be represented as a three-way array (see Fig. 3.1*a*) forming a matrix $\hat{\mathbf{X}}$ ($I \times J \times K$) where I represents the number of batches, J refers to the number of measured variables, and K denotes the number of fixed-length time intervals over which measurements are made. A method to analyze this data is MPCA (Wold et. al., 1987) that creates an empirical reference model from the past measurements involving normal (in control) and abnormal (out-of-control) batches for detecting any unusual behavior of a test batch (Nomikos and MacGregor, 1994; Janusz, andVenkatasubramanian, 1991; Kourti and MacGregor, 1996). In MPCA procedure, PCA is performed on the large two-dimensional matrix $\mathbf{X}$ formed by unfolding the three-way array, $\hat{\mathbf{X}}$. The unfolding is done by taking vertical slices along the time axis of matrix $\hat{\mathbf{X}}$ and ordering them side by side to the right to generate a 2-D matrix, $\mathbf{X}$, of dimensions ($I \times JK$) (see Fig. 3.1*b*). Thus, the first J columns of $\mathbf{X}$ describe all the input variables sampled at time $k = 1$ from I batches. Similarly, the next set of J columns refer to the same set of input variables sampled at the end of the next time interval ($k = 2$), and so on till $k = K$. The MPCA decomposes the $\mathbf{X}$ array into the summation of the product of the score vectors, $\mathbf{t}_r$, and loading vectors, $\mathbf{p}_r$, as given by:

$$\mathbf{X} = \sum_{r=1}^{R} \mathbf{t}_r \mathbf{p}_r^{'} + \mathbf{E} \tag{3.2}$$

where R denotes the number of extracted PCs; $\mathbf{p}_r^{'}$ represents transpose of the loading vector $\mathbf{p}_r$, and $\mathbf{E}$ represents the residual matrix. The score vectors express the relationship among batches while the loadings are related to the measured variables and their time variation.

The PLS utilizes measurements of both the input variables ($\mathbf{X}$) and output variables ($\mathbf{Y}$), where $\mathbf{Y}$ refers to a 2-D array of dimensions ($I \times M$) and M denotes the number of output variables. In PLS, while the $\mathbf{X}$ matrix is decomposed as described in Eq. 3.2, matrix $\mathbf{Y}$ is decomposed as:

$$\mathbf{Y} = \sum_{r=1}^{R} \mathbf{u}_r \mathbf{q}_r^T + \mathbf{F} \tag{3.3}$$

where vectors $\mathbf{u}_r$ and $\mathbf{q}_r$, and matrix $\mathbf{F}$ refer to the scores, loadings and residual, respectively. Next, a linear model of the following form is constructed to correlate $\mathbf{X}$ and $\mathbf{Y}$ (Geladi and Kowalski, 1986);

$$\mathbf{Y} = \mathbf{X}\beta + \Gamma \tag{3.4}$$

where β and Γ respectively represent the matrix of regression coefficients and regression residual.

3.2.2 Generalized regression neural networks (GRNNs)

GRNNs are memory based feedforward networks and these were introduced (Specht, 1991) as a generalization of both the radial basis function networks (RBFNs) and probabilistic neural networks (PNNs). With increasing number of training samples, the GRNN asymptotically converges to the optimal regression surface. In addition to having a sound statistical basis, the GRNNs possess a special property that they do not require iterative training.

Consider a J-dimensional vector, $\mathbf{x} = [x_1, x_2, \ldots, x_J]^T$, describing process input variables and the corresponding scalar output, y, representing the quality (output) variable. GRNN performs the regression by computing the conditional expectation of y given $\mathbf{x}$. Specifically, the GRNN estimates the joint probability density function (PDF) of $\mathbf{x}$ and y, i.e. $f(\mathbf{x}, y)$, to create a probabilistic model for predicting y.

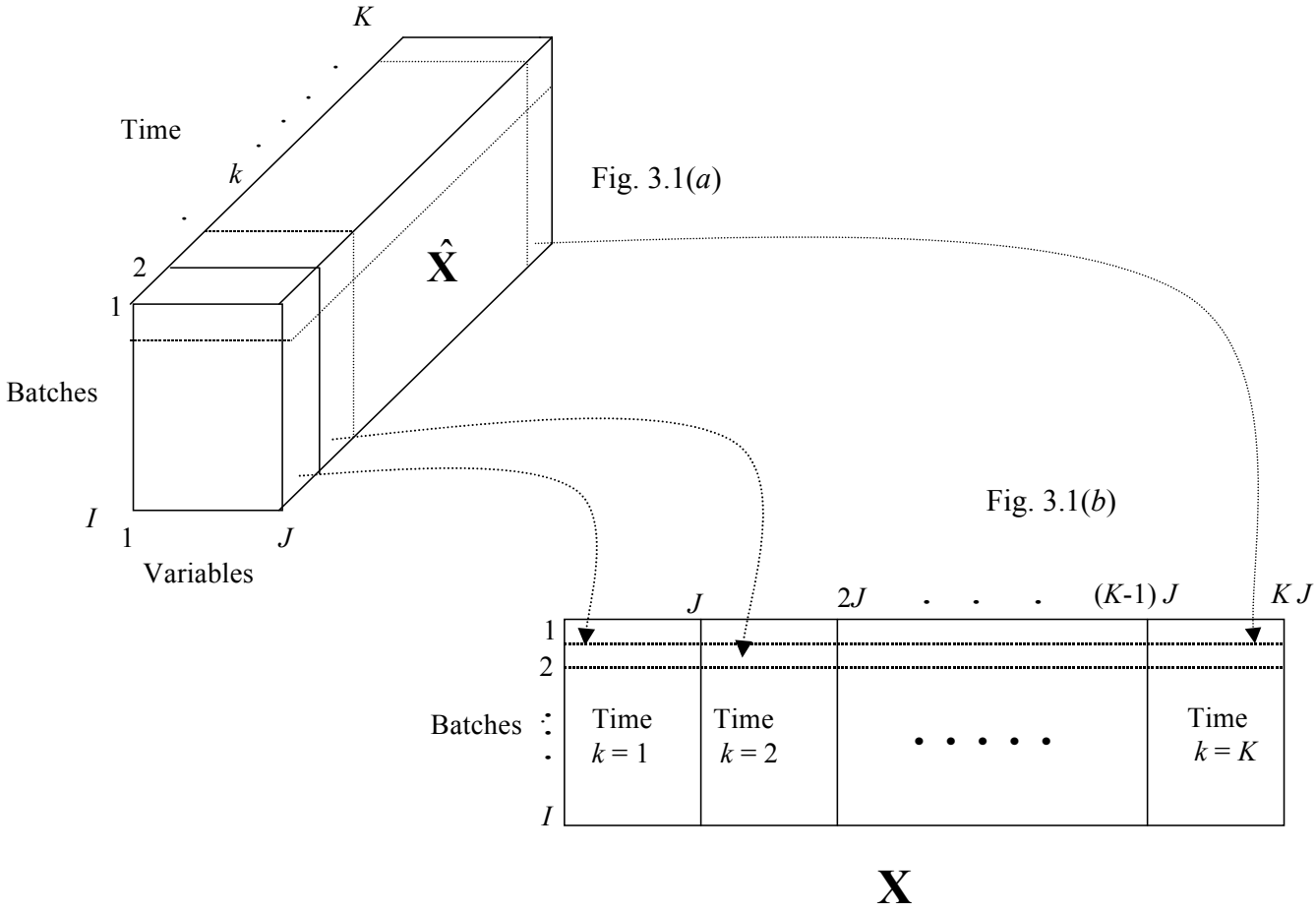


Figure 3.1: (*a*) Structure of a three-way data array ($\hat{\mathbf{X}}$) describing input (predictor) variable measurements from a batch process,

(b) Unfolding of $\hat{\mathbf{X}}$ array into a large 2-dimensional matrix **X**.

The PDF estimator model is constructed from the training input-output data set $\{\mathbf{x}_i, y_i\}$; $i = 1, 2, \ldots, I$, via nonparametric density estimation (NDE). Given $\mathbf{x}$ and assuming that the function being approxiamted is continuous and smooth, the expected value of y, ($E[y \mid \mathbf{x}]$) can be estimated as:

$$E\left[y \mid \mathbf{x}\right] = \frac{\int_{-\infty}^{\infty} y\, f(\mathbf{x}, y)\, dy}{\int_{-\infty}^{\infty} f(\mathbf{x}, y)\, dy} \tag{3.5}$$

Using the training set and assuming Gaussian PDF, the function $f(\mathbf{x}, y)$ can be defined as:

$$f(\mathbf{x}, y) = \frac{1}{(2\pi)^{\frac{(J+1)}{2}} \sigma^{J+1}} \times \frac{1}{I} \sum_{i=1}^{I} \left[\exp\left(\frac{-(\mathbf{x} - \mathbf{x}_i)^T (\mathbf{x} - \mathbf{x}_i)}{2\sigma^2} \right) \right] \exp\left[\left(-\frac{(y - y_i)^2}{2\sigma^2} \right) \right] \tag{3.6}$$

where $\mathbf{x}_i$ and y repectively denote the ith training input vector and the corresponding output, and σ denotes the width (*smoothing parameter* or *spreading factor*) of the Gaussian PDF. Given $\mathbf{x}$, the corresponding regression estimate, $\hat{y}(\mathbf{x})$, can be determined as a conditional mean by substituing Eq. 3.6 in Eq. 3.5:

$$\hat{y}(\mathbf{x}) = E\left[y \mid \mathbf{x}\right] = \frac{\sum_{i=1}^{I} y_i h_i}{\sum_{i=1}^{I} h_i} \;;\; h_i = \exp\left[-d_i^2 \Big/ 2\sigma^2 \right] \tag{3.7}$$

where h_i denotes the Gaussian radial basis function and d_i^2 represents the squared Euclidean distance between vectors $\mathbf{x}$ and $\mathbf{x}_i$ defined as:

$$d_i^2 = (\mathbf{x} - \mathbf{x}_i)^T (\mathbf{x} - \mathbf{x}_i) \tag{3.8}$$

In Fig. 3.2, GRNN's multiinput-multioutput (MIMO) architecture comprising of four layers, namely, the *input*, *hidden*, *summation* and *output* layers is depicted. Unlike the most popular error–back-propagation (EBP) algorithm (Rumelhart et. al., 1986) that trains multilayer feedforward networks iteratively, the GRNN training is a single pass procedure. Consequently, the GRNN trains itself in a significantly shorter time as compared to the EBP-based training.

Given a test input vector, **x**, the GRNN procedure for predicting the value of an mth output variable y_m (m = 1, 2, …, M), can be viewed as computing the weighted average of all the target values of that variable wherein weights are taken proportional to the Euclidean distances between the training input vectors and the test input vector. GRNN's input layer houses J nodes to serve as 'fan-out' units. The hidden layer contains I number of nodes such that each hidden node represents a different training input vector. When an input vector is applied to the input nodes, its distance (d_i) from each of the I training vectors stored in the hidden nodes is computed, which is then transformed using the Gaussian RBF to compute the hidden unit output, h_i (i = 1, 2, …, I). GRNN's third layer consists of two types (I and II) of summation units. An mth type-I unit (indexed as N_m and shown by dark border), computes the summation ($\sum_{i=1}^{I} y_i h_i$) defined in the numerator of Eq. 3.7, by utilizing the hidden unit outputs, h_i, and the mth elements of all the M-dimensional target output vectors, $\mathbf{y}_i$. The single type-2 unit in the third layer performs summation of all hidden node outputs (see denominator of Eq. 3.7). Finally, the mth output layer node performs the normalization step defined in Eq. 3.7 to compute the GRNN-predicted value of the mth output variable, $\hat{y}_m(\mathbf{x})$. GRNN's training algorithm uses only one adjustable (free) parameter namely the width (σ) of the Gaussian RBF. Significance of the width parameter is that as its value becomes smaller (larger) the regression performed by the GRNN becomes more local (global). Hence, the magnitude of σ needs to be choosen judiciously as it significantly affects the accuracy of GRNN's predictions. The commonly employed technique for automatically selecting the optimal σ value is the 'leave–one–out' cross-validation method. In this technique, one input-output vector is removed from the training set of I vectors and a GRNN model is built using the remaining I-1 vectors for predicting the outputs corresponding to the removed pattern. This procedure is repeated I times by setting aside each time a different training pattern and the mean-squared-error (MSE) is evaluated by comparing the GRNN-predicted and the corresponding target output values. This procedure is repeated by varying the σ value systematically and the value that minimizes the MSE is chosen to be optimal.

3. 3 BATCH PROCESS MONITORING USING GRNNS

The GRNN – likewise most ANNs – performs poorly if its input space contains irrelevant inputs, which unnecessarily increase the dimensionality of the input space and thus the size of the training set. This difficulty is easily overcome by performing PCA on the predictor variable data thereby achieving the dimensionality reduction of the input space. To implement the PCA-GRNN strategy for batch process modeling and monitoring, first the three way process data $\hat{\mathbf{X}}$ ($I \times J \times K$) is unfolded – as in MPCA – to obtain a 2-D matrix, **X**. The MPCA formalism for on-line monitoring requires complete history of the batch process under scrutiny thus making it necessary to fill up the future unmeasured values from the current time to the end of the batch. Various methods for filling up unmeasured data though are available (Specht, 1991), they make certain assumptions, which may not be always valid. Filling up of future observations might cause false detection since they may distort data information without considering the proper dynamic relationship. Hence, MPCA (or MPLS) performs poorly when a large number of future measurements are unknown especially in the initial stage of a new batch run. This difficulty can be overcome by developing a PCA-based submodel by considering data measured only at the current time instant. Specifically, for a process being monitored over a total of K equally spaced time instances, a separate submodel is built at each time instance $k = 1, 2, \ldots, K$, by considering the input measurements made at that instant. Since a sub-model corresponding to the kth instant requires data measured only at that instant, values corresponding to $k+1$ to K time intervals are not required thus altogether avoiding the necessity of filling up future measurements. This approach reduces the numerical effort involved in computing the future unmeasured data and the inaccuracies associated therewith.

The task of batch process monitoring considered here in this study is defined as: *Given historic process input-output data in the form of a 3-way array* $\hat{\mathbf{X}}$*, analyze the process behavior at equally spaced time instances, k = 1, 2, ..., K, with a view to detect any abnormality in the current (test) process inputs, and more importantly to predict any abnormal behavior of the process outputs at the end of the batch run.* The stepwise procedure to address this task using the PCA-GRNN strategy is described in the Appendix 3.1.

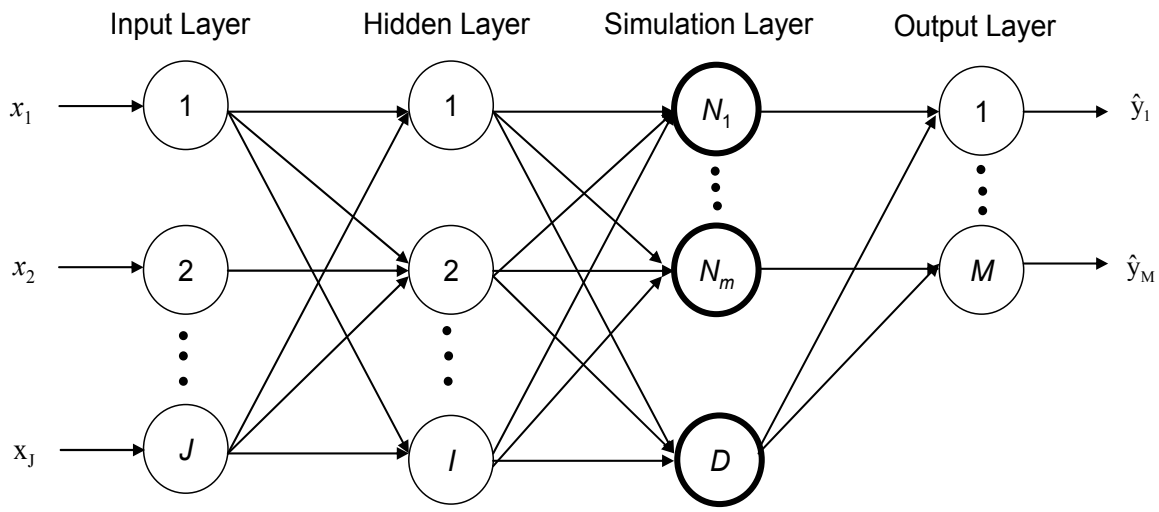


Figure 3.2: The schematic of a multiinput – multioutpur (MIMO) GRNN

3.4 CASE STUDIES

Applications of the PCA-GRNN strategy for batch process modeling and monitoring are illustrated by conducting two case studies involving biosynthesis of penicillin and protein, respectively. The dynamic process input-output data used in the case studies were obtained by simulating the phenomenological models of the two processes. It may however be noted that the PCA-GRNN strategy does not require the phenomenological process model for its implementation. In here, the phenomenological models are used only to generate process input-output data for demonstrating the efficacy of the proposed strategy. In real practice, historic input-output data from the batch process are used for conducting process modeling and monitoring. Using typical variations in the initial values of the process operating variables, input-output data for a number of batches were generated. Batches with their input variables within the acceptable operating range (normal behavior) and also outside the range (abnormal behavior) were simulated to generate the training set. Also, a number of input-output vectors to be used as the test set, were generated by assuming the input process conditions both in the acceptable and unacceptable ranges. The effectiveness of the PCA-GRNN formalism has been studied in the presence of clean as well as noisy inputs.

3.4.1 Fed-Batch Fermenter for Biosynthesis of Penicillin

The fermenter system for penicillin production (Lim et. al., 1986; Bajpai and Reuss, 1981) is characterized in terms of four input-output variables namely, biomass concentration (x_1), substrate concentration (x_2), reactor volume (x_3) and product concentration (y). Process data pertaining to these input-output variables were generated using the set of four ordinary differential equations (ODE's) given in Appendix 3.2. Specifically, input-output data for a total of 54 batches were generated using the fermenter's phenomenological model; the data set comprises values of three predictor variables (x_1, x_2, x_3) measured at one-hour time intervals and the value of the single output variable, y, measured at the end of the batch run (16 hr). The set to be used as the training set consisted noise-free data from 48 batches with their initial conditions ($x_1(0)$, $x_2(0)$, $x3(0)$) in the normal ranges ($1.45 < x_1(0) < 1.6$, $x_2(0) = 0$ and $6.5 < x_3(0) < 7.5$) and six batches (5, 11, 20, 32, 38 and 41), with their initial conditions outside the normal ranges (abnormal batches). Additionally, an input-output test set comprising two batches (55 and 56) with abnormal initial conditions and two batches (57 and 58) with normal initial conditions, was generated in a manner similar to the training set. The task of PCA-GRNN strategy is to analyze at each one-hour interval whether: (i) the predictor variables are behaving normally, and (ii) the process output at the end of the batch will be in the acceptable range. While inference about the normal behavior of the predictor variable space can be drawn from the principal component scores (monitoring), the normal behavior of the process output can be ascertained by training the GRNN model and using it for predicting the outputs. Accordingly, the training input set ($\hat{\mathbf{X}}$) of dimensions ($54 \times 3 \times 16$) was unfolded into a 2-D matrix ($\mathbf{X}$) and following mean-centering and autoscaling individual matrix columns, PCA was conducted on the input data pertaining to each one-hour time interval. The loading vectors obtained thereby were then used to obtain the PC scores pertaining to the four test batches. In Fig. 3.3, scores of PC-1 are plotted against those of PC-2 corresponding to a representative case involving eighth hour predictor variable data. In this case, PC-1 and PC-2 scores explained 99% variance in the original data. As seen in the figure, test batches 55 and 56 lie outside the 95% control limit and thus have been correctly identified as those with abnormality in the predictor variables. The PCA has also identified the six training set batches with abnormal initial conditions correctly. It is moreover seen in Fig. 3 that the test batches 57 and 58 lie well within the

95% confidence region thus rightly indicating their normal behavior. Similar results were obtained using input measurements at all other time intervals.

The real benefit of PCA-GRNN strategy lies in predicting the process outputs as the batch run progresses. Accordingly, scores of first two PCs were used to build GRNN-based models. Here, a separate GRNN model was developed using the scores of the predictor variable data pertaining to each of the sixteen time intervals. The training input data for developning the GRNN models contained scores of 54 normal and abnormal batches; the training output data comprised values of the product concentration (y) at the end of 54 batches. Optimality of the GRNN models resulting in accurate output predictions was ensured by optimizing the smoothing parameter, σ, using the leave-one-out cross-validation method. Computation of PCs and the development of an optimal GRNN model took 46 seconds of CPU time on a Pentium-IV (1.2GHz) personal computer. The CPU time consumed suggests that the PCA-GRNN implementation is computationally inexpensive and thus model building can be done on-line. Each of the 16 GRNN models were then used for predicting the output variable values at the end of the four test batches; for comparison purposes the output values were also computed using the PLS method. For illustration, the results of the GRNN and PLS-based prediction for 54 batches using fifth hour noise-free data are shown in Fig. 3.4(*a*). As can be observed, both the GRNN and PLS have predicted the values of the output at the end of the batch accurately. It is seen from the 3-D surface plotted in Fig. 3.5 that the relationship between the PC scores and the output is almost linear and thus the PLS could predict the outputs with high accuracy.

These results point to GRNN's ability to make accurate predictions also for linear systems. Real-life process data are always noisy and therefore it is necessary to test the performance of PCA-GRNN method using noisy measurements. To this end, three data sets comprising 1%, 3% and 5% Gaussian noise in each input variable were created and those were used to build the PCA and GRNN models. It was consistently observed that the GRNN based output predictions are more accurate as compared to the PLS method (see Table 3.1). From the table it is seen that while the GRNN makes prediction with almost same accuracy even as noise level increases, the PLS predictions become progressively poorer. As a representative case, the values of the output variable at the end of the 54 batches predicted by GRNN and PLS methods using fifth hour noisy input data are plotted

in Fig. 3.4(*b*). It is clearly noticed in this figure that GRNN predictions (CC = 0.999) are very accurate as compared to those obtained using the PLS (CC = 0.98).

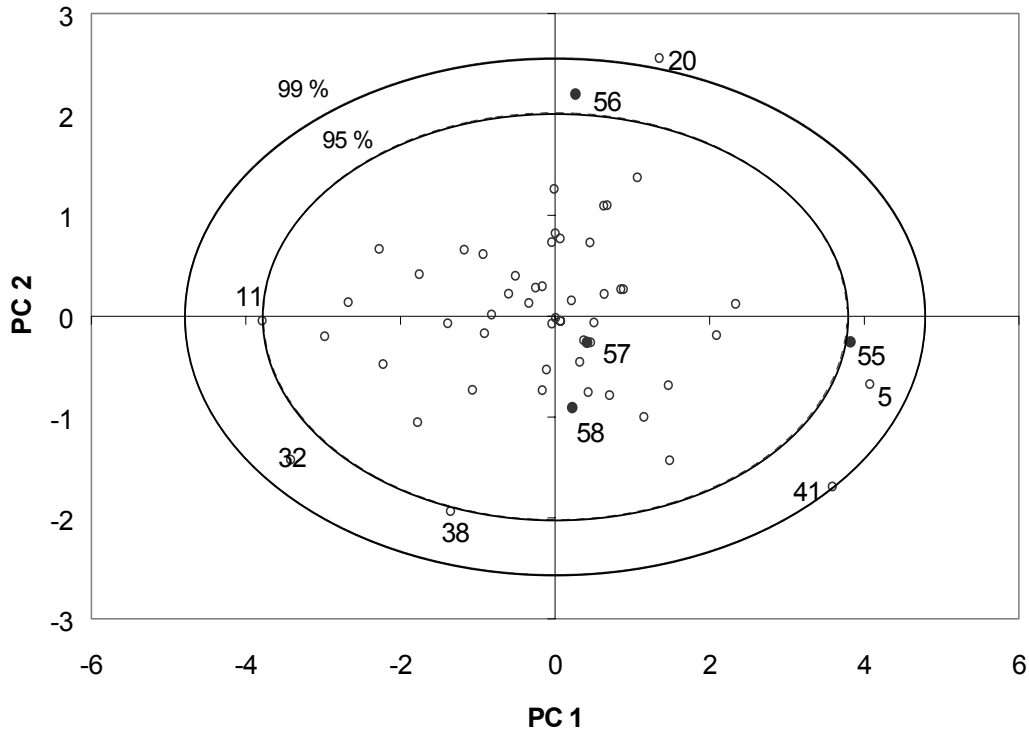


Figure 3.3: Plots of PC-1 versus PC-2 scores pertaining to eighth hour predictor variable data for case study-1.

Table 3.1: Comparison of GRNN and PLS prediction of product concentration using predictor variable data containing different levels of Gaussian noise

Noise level	**CC** #		**Error (%)***	
	GRNN	PLS	GRNN	PLS
Clean data	0.999	0.999	0.028	0.057
1 % noise	0.999	0.998	0.027	0.101
3 % noise	0.999	0.991	0.027	0.304
5 % noise	0.999	0.980	0.025	0.450

Correlation Coefficient

* Average absolute prediction error

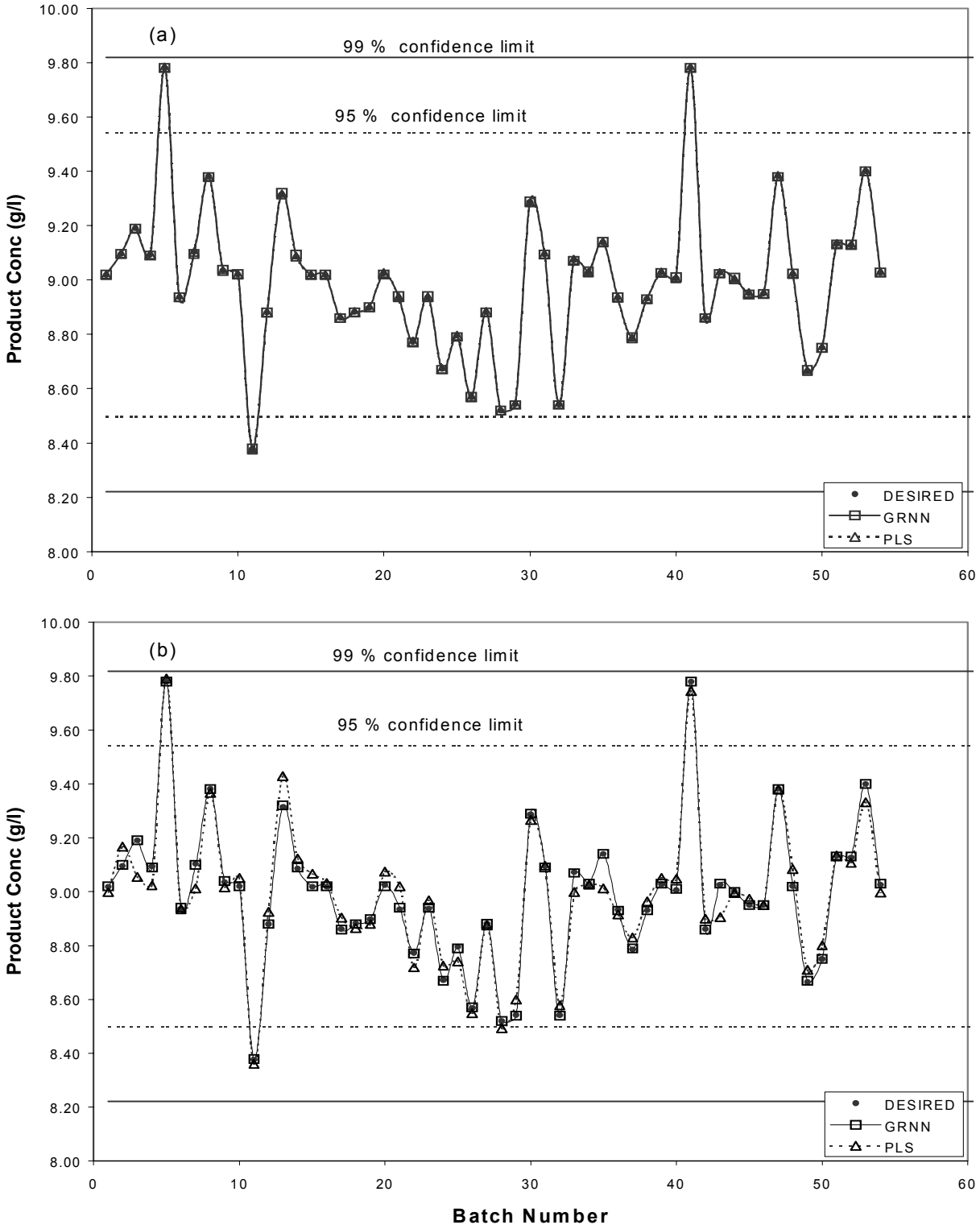


Figure 3.4: (*a*) Comparison of GRNN and PLS predicted product concentration using fifth hour clean data, (*b*) same as in panel (a) but using fifth hour data containing 5% Gaussian noise.

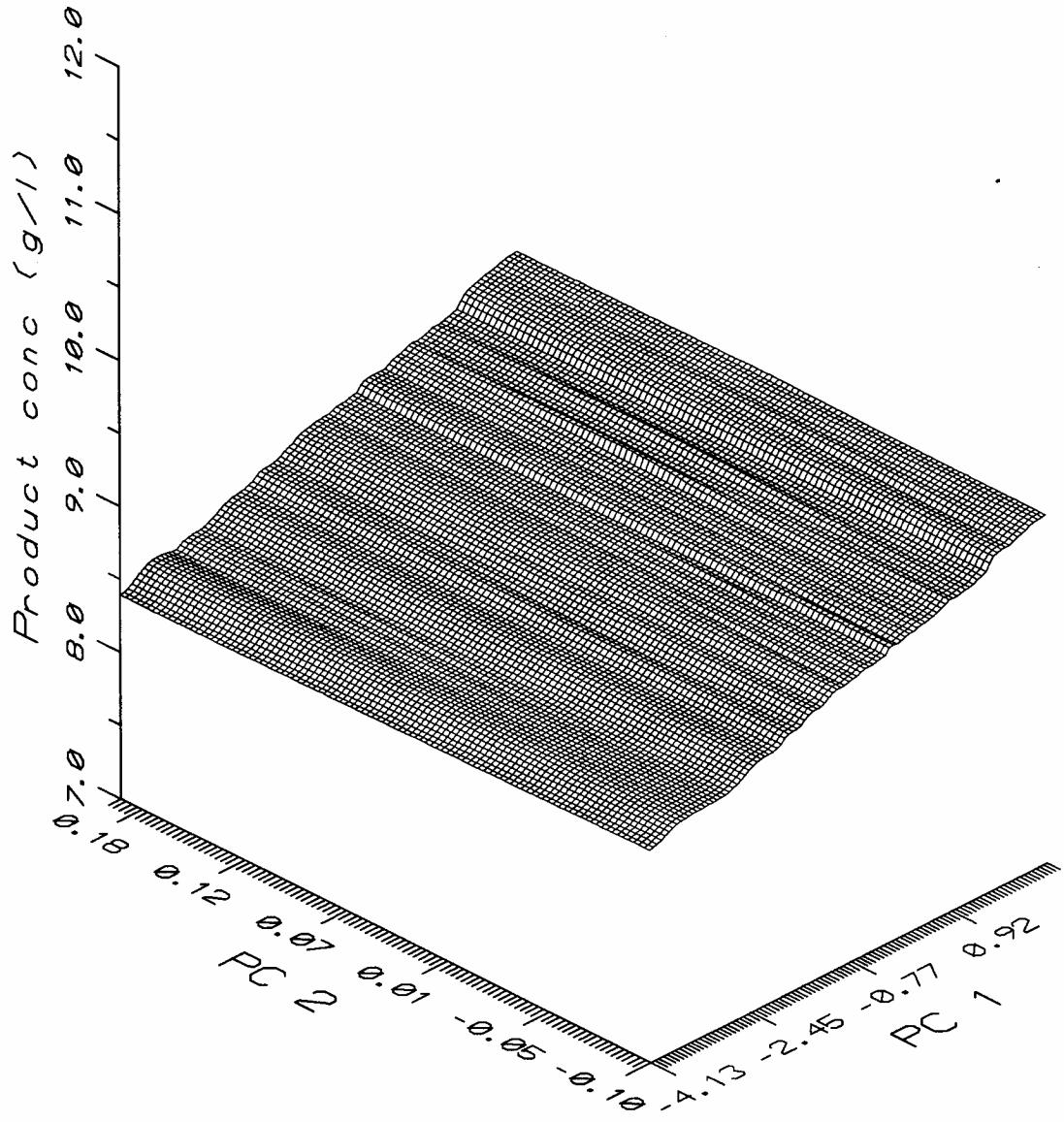


Figure 3.5: Plot showing linear behavior of product concentration as a function of PC-1 and PC-2 scores for bio-synthesis of penicillin

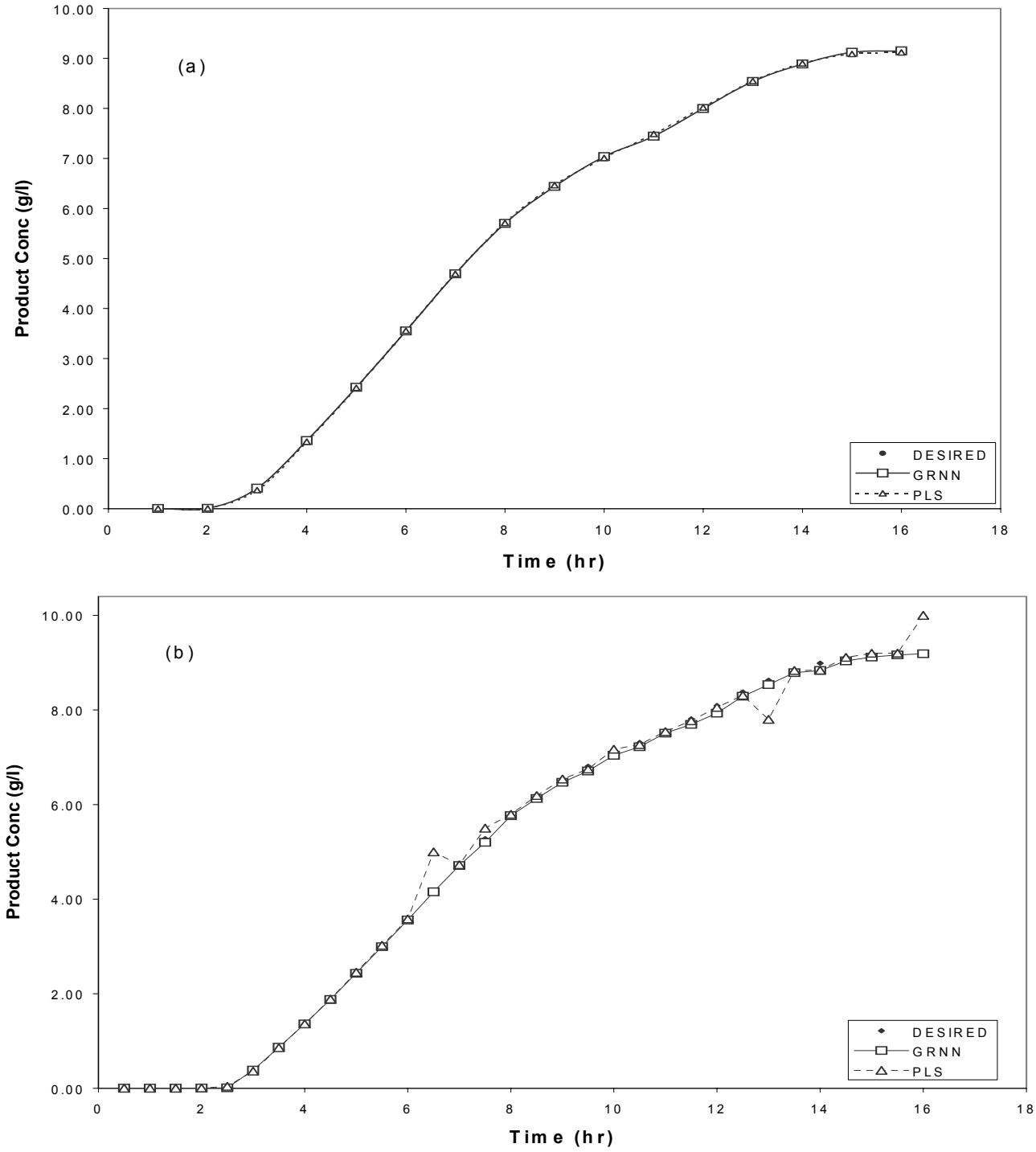


Figure 3.6: (*a*) GRNN and PLS based prediction of product concentration from hourly sampled noise-free input-output data for a test batch, (*b*) same as in panel (*a*) but using hourly sampled input variable data containing 5% Gaussian noise

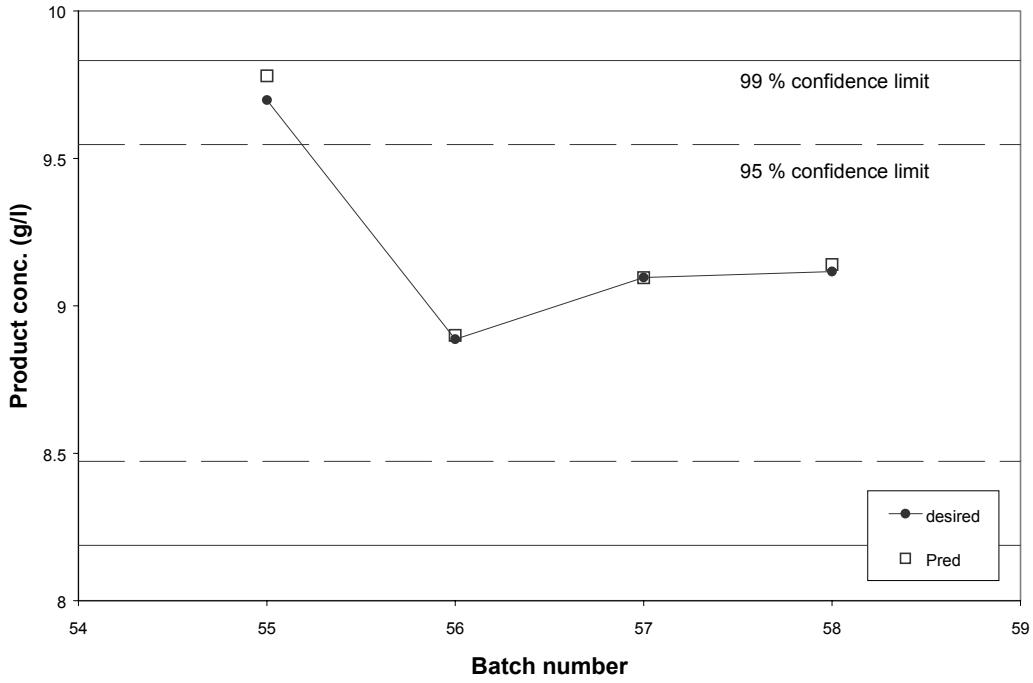


Figure 3.7: GRNN prediction of the output for four test batches based on eighth hour data.

Although in most batch processes the product quality variable is measured only at the end of the batch, occasionally (i.e. wherever feasible and economical) the product is sampled and analyzed intermittently. In such situations, the GRNN-based models can serve as 'soft-sensors' whereby the model predicted values of the quality variable become available almost instantaneously upon sampling the product. To test the efficacy of the GRNNs to act as a soft-sensor, sixteen models were constructed using the training set comprising scores of the hourly sampled input data and the corresponding values of the output variable. Next, these models were used to predict the value of the quality variable sampled at one-hour time intervals corresponding to the test batches. The results of such a GRNN-based prediction using hourly sampled noise-free and noisy inputs for the 57th normal test batch are portrayed in Figs. 3.6(*a*) and 3.6(*b*), respectively. It can be seen in these plots that in the case of clean data, both GRNN and PLS predict the hourly sampled output value accurately. However, when inputs are noisy, GRNN predictions are more accurate as compared to the PLS predictions. These results are indicative of the effectiveness of GRNN to act as a soft sensor in real-life processes.

During the progress of a batch, it is important to know whether the batch will end as successful or otherwise. Since PCA gives an insight into the normal or abnormal behavior of only process inputs, it is risky to conclude from the PCA results that the batch will end as successfully or not. It is therefore essential to consider the behavior of both the process inputs as well as the outputs to draw a conclusion about the success of a batch. Consider for instance Fig. 3.7 wherein product concentration values at the end of the batch as predicted by the GRNN model based on the eighth hour data, are depicted. In the figure it is seen that the test batches 57 and 58 with normal initial conditions end as successful batches since their output values fall within the 95% control limit. However, among the two test batches (55 and 56) with abnormal initial conditions, the 56th batch is a successful batch, whereas 55th batch is an unsuccessful one since the corresponding output value lies outside the 95% control limit. It is thus seen that the 56th batch though began with abnormal initial conditions, still ended as a successful batch. A probe into the possible reasons behind this observation reveals that the product concentration (y) is influenced more by the deviation in the initial value of the reactor volume ($x_3(0)$), than that of the biomass concentration ($x_1(0)$). In the case of 55th batch, the initial value for biomass concentration ($x_1(0) = 1.52$) was in the normal range [1.45-1.6] while that of the reactor volume ($x_3(0) = 6.1$) deviated from the normal range [6.5-7.5]. Since $x_3(0)$ has more influence on y than $x_1(0)$, 55th batch with deviated $x_3(0)$ value ended as an unsuccessful batch when compared to the 56th batch that began with deviated $x_1(0)$ value (= 1.63) and normal $x_3(0)$ value (= 7.1). It is thus clear that the PCA-GRNN methodology helps in making accurate forecast about the success of a batch as compared to the one using only the PCA approach.

In the simulations described above, GRNN-based models were developed to predict the output variable value at the current time interval and at the end of a batch (last time interval). Similar strategy can be employed to predict the output variable value at intermittent time intervals. It would however require output variable measurements at those intermittent intervals for all historic batches. The principal advantage of this approach is that the output variable values at future time intervals can be predicted in advance, which greatly assists in making a decision about when to end the batch. In here, depending upon the current time interval (k), GRNN based models can be developed to predict the output variable value at the lth time interval where $l = k, k+1, k+2, \ldots, K$. The

results described above for: (i) $k = 8$, $l = K = 16$ (see Fig. 7), and (ii) $l = k$, $k = 1, 2, \ldots, K$ (see soft-sensor predictions in Fig. 3.6), clearly indicate that the PCA-GRNN approach can be successfully extended to predict the output variable values at future time intervals.

3.4.2 Fed-Batch Fermentation for Protein Synthesis

The fed-batch fermenter system for protein synthesis is characterized in terms of three

predictor variables namely culture cell density (x_1), substrate concentration (x_2) and hold-up volume (x_3), and two output variables, viz. secreted protein concentration (y_1) and total protein concentration (y_2). The input-output training set for a total of 52 batches comprising values of the three predictor variables (x_1, x_2, x_3) measured at one hour time intervals and the values of two output variables (y_1, y_2) measured at the end (15 hr) of the batch was generated using the phenomenological model (Lim et. al., 1977; Balsa-Canto, et. al., 2001) given in Appendix 3.3. This training set consisted data from 48 batches with their initial conditions in the normal ranges ($0.95 < x_1(0) < 1.5$, $4.5 < x_2(0) < 5.5$, and $0.95 < x_3(0) < 1.5$) and two abnormal batches (17 and 38) with their initial conditions outside the stated normal ranges. Additionally, a test set consisting of two normal batches (53 and 54) and two abnormal batches (55 and 56), was generated in a similar manner to the training set to test the efficacy of the GRNN models.

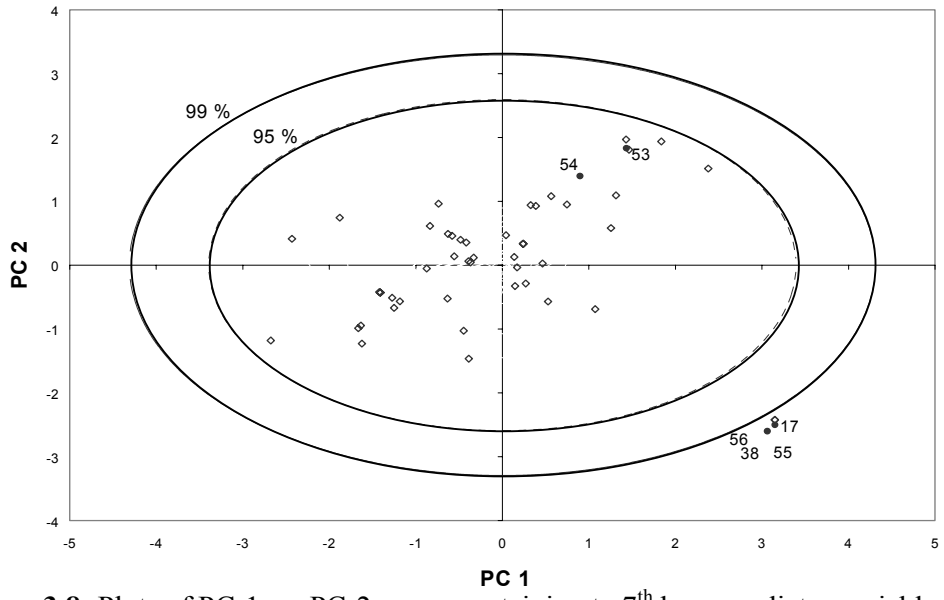


Figure 3.8: Plots of PC-1 vs. PC-2 scores pertaining to 7th hour predictor variable data for case study 2

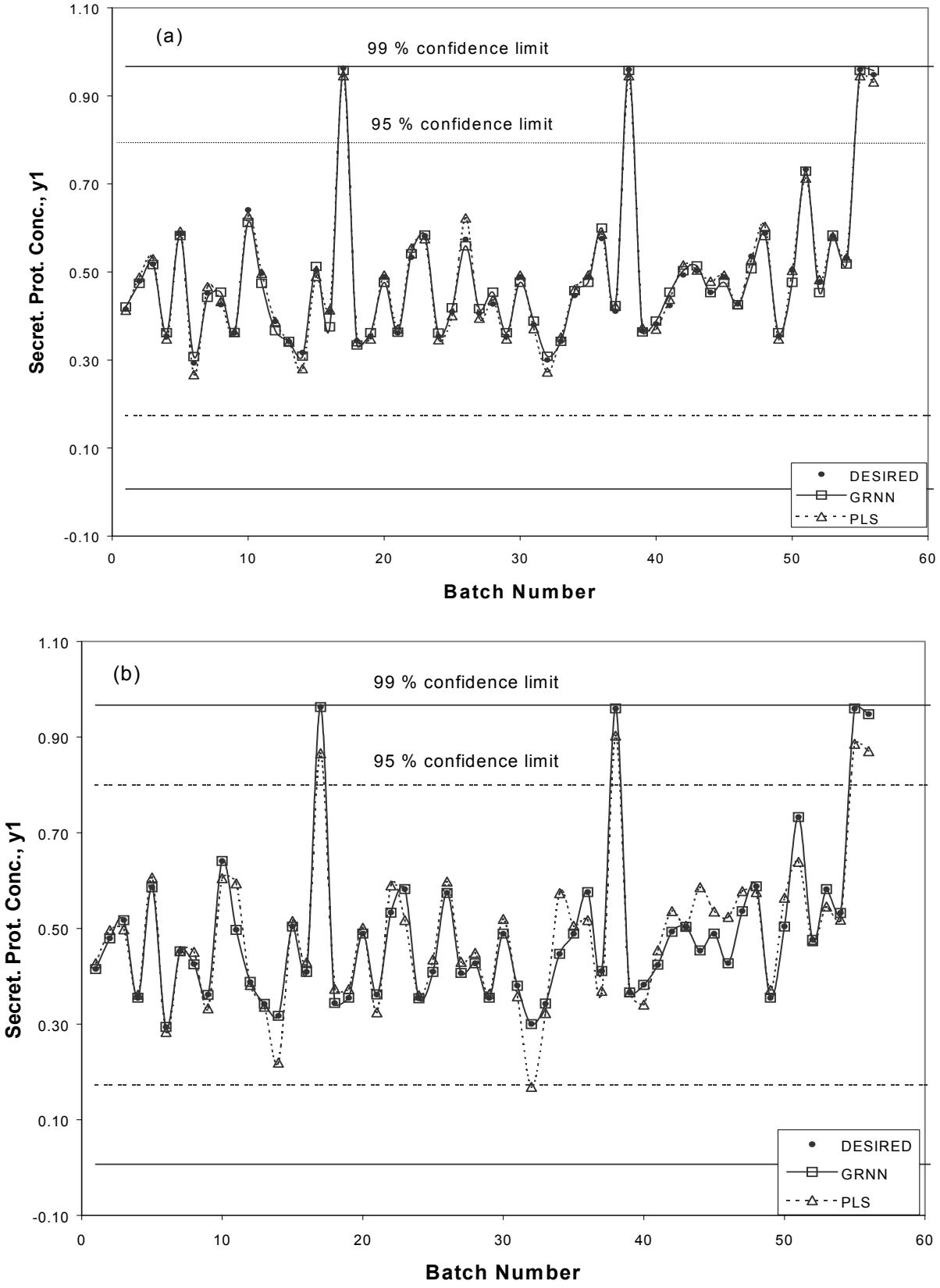


Figure 3.9: (*a*) Comparison of GRNN and PLS based prediction of secreted protein concentration using third hour clean data, (*b*) same as in panel (*a*) but using input data containing 5% Gaussian noise

The 3-D input array ($\hat{\mathbf{X}}$) of dimensions ($52 \times 3 \times 15$) was unfolded, preprocessed (mean and variance scaling) and subjected to PCA as described in the stepwise procedure (Appendix-3.1). Specifically, PCA was conducted separately on hourly sampled input variable data and the loadings vectors obtained thereby were used to compute the scores of the four test batches. A plot of PC-1 versus PC-2 for all the batches showed that the test batches 55 and 56 lie consistently outside the 99% control limit and thus have been correctly identified as abnormal batches; the PCA also identified the two abnormal training batches 17 and 38 correctly. Figure 3.8 shows the plots of the scores of PC-1 versus those of PC-2 for a representative case using seventh hour predictor variable data. In this case, PC-1 and PC-2 explained 99% variance in the original data. Next, the scores of PC-1 and PC-2 corresponding to 52 batches were used to build GRNN-based models. The training output data for this modeling comprised values of y_1 and y_2 at the end of 52 batches. Here, fifteen GRNN models were developed for simultaneously predicting y_1 and y_2 from the scores of hourly readings of the predictor variables. The CPU time required by the Pentium-IV personal computer (1.2GHz) for conducting the PCA and developing an optimal GRNN model was 65 seconds. Next, the models were used for predicting y_1 and y_2 values corresponding to the four test batches. To illustrate, the results of GRNN and PLS–based prediction using third hour data are depicted in Fig. 3.9(*a*), where it is observed that the GRNN has predicted the output values at the end of the batch with higher accuracy (CC = 0.999) when compared to the predictions made by the PLS method (CC = 0.982). Simulations similar to the above-stated ones were conducted using 1%, 3% and 5% noise in the predictor variables. A statistical comparison of the GRNN and PLS-based output predictions for noisy data is provided in Table 3.2 (also see Fig. 3.9(*b*)). It is seen in this case study also that the GRNN outperforms PLS when data contain noise. Moreover, it is witnessed that with increasing noise level, the PLS makes increasingly inaccurate predictions. Figure 3.10 helps in understanding the factor behind higher accuracy of GRNN predictions vis-a-vis those made by the PLS. It is seen from the 3-D surface plotted in Fig. 3.10 that the relationship between the scores and the output (y_1) is nonlinear. This strong nonlinearity modelled accurately by the GRNN, arises from the exponential dependence of the output variables, y_1 and y_2, on the substrate concentration, x_2

(refer model equations in Appendix 3.3). It is thus seen that the GRNN possesses a significant ability to make accurate predictions for nonlinear batch systems even in presence of noisy data.

Table 3.2: Comparison of GRNN and PLS prediction of process outputs using predictor variable data containing different levels of Gaussian noise

Noise level	**Prediction of y_1**				**Prediction of y_2**			
	CC		**Error (%)**		**CC**		**Error (%)**	
	GRNN	PLS	GRNN	PLS	GRNN	PLS	GRNN	PLS
Clean data	0.999	0.982	0.061	4.02	0.999	0.981	0.049	4.48
1 % noise	0.999	0.946	0.016	7.88	0.999	0.945	0.014	7.60
3 % noise	0.999	0.947	0.061	7.89	0.999	0.947	0.001	7.64
5 % noise	0.999	0.940	0.061	8.16	0.999	0.945	0.001	7.85

The results of GRNN-based predictions of y_1 and y_2 can be used to assess whether a test batch will be successful or not. Consider for instance predictions made using the last hour data and portrayed in Fig. 3.11, as can be seen, the predictions correctly show that the test batches 53 and 54 (55 and 56) will be successful (unsuccessful) as the corresponding y_1 and y_2 values lie well within (outside) the 95% control limit. It is also seen in the figure that training batches 17 and 38 with abnormal initial values of the predictor variables, have resulted in unsuccessful batches. Results similar to that shown in Fig. 3.11 have been obtained using predictor variable data pertaining to all the 15 one-hour time intervals thereby proving the ability of the GRNNs in accurately predicting whether a given batch will be successful or not.

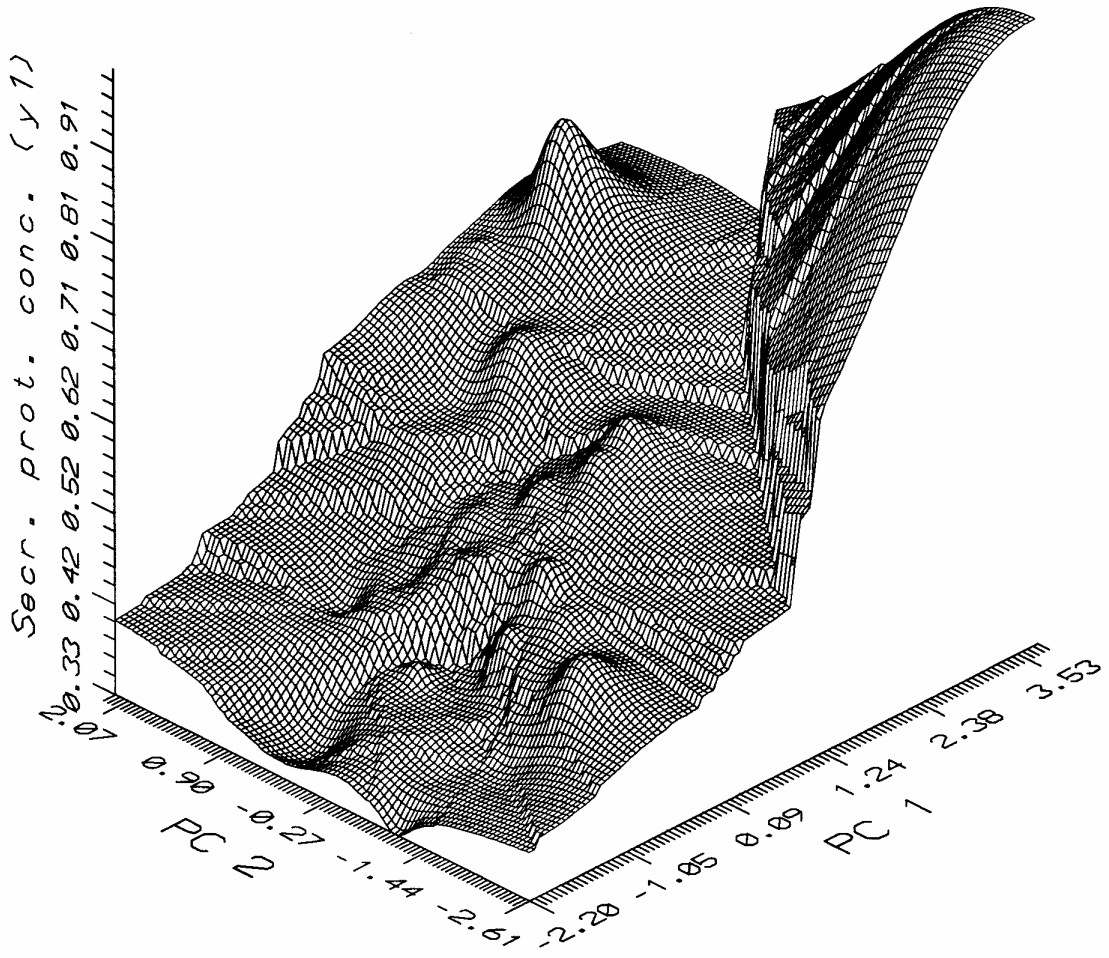


Figure 3.10: Plot showing the nonlinear behavior of secreted protein concentration (y_1) as a function of PC-1 and PC-2 scores

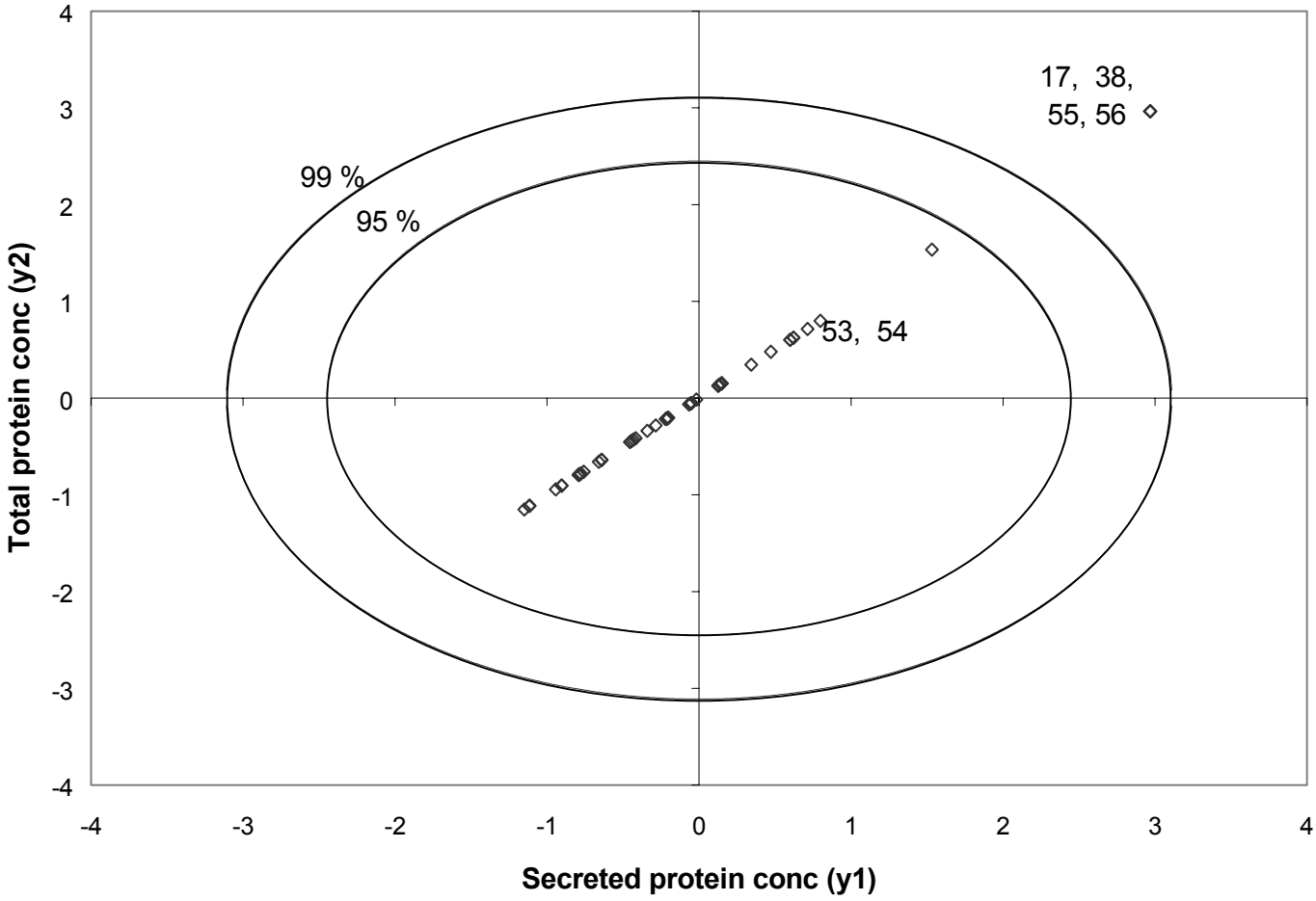


Figure 3.11: Plot of GRNN-predicted values of y_1 and y_2 pertaining to the training and test batches using 15th hour data for protein synthesis

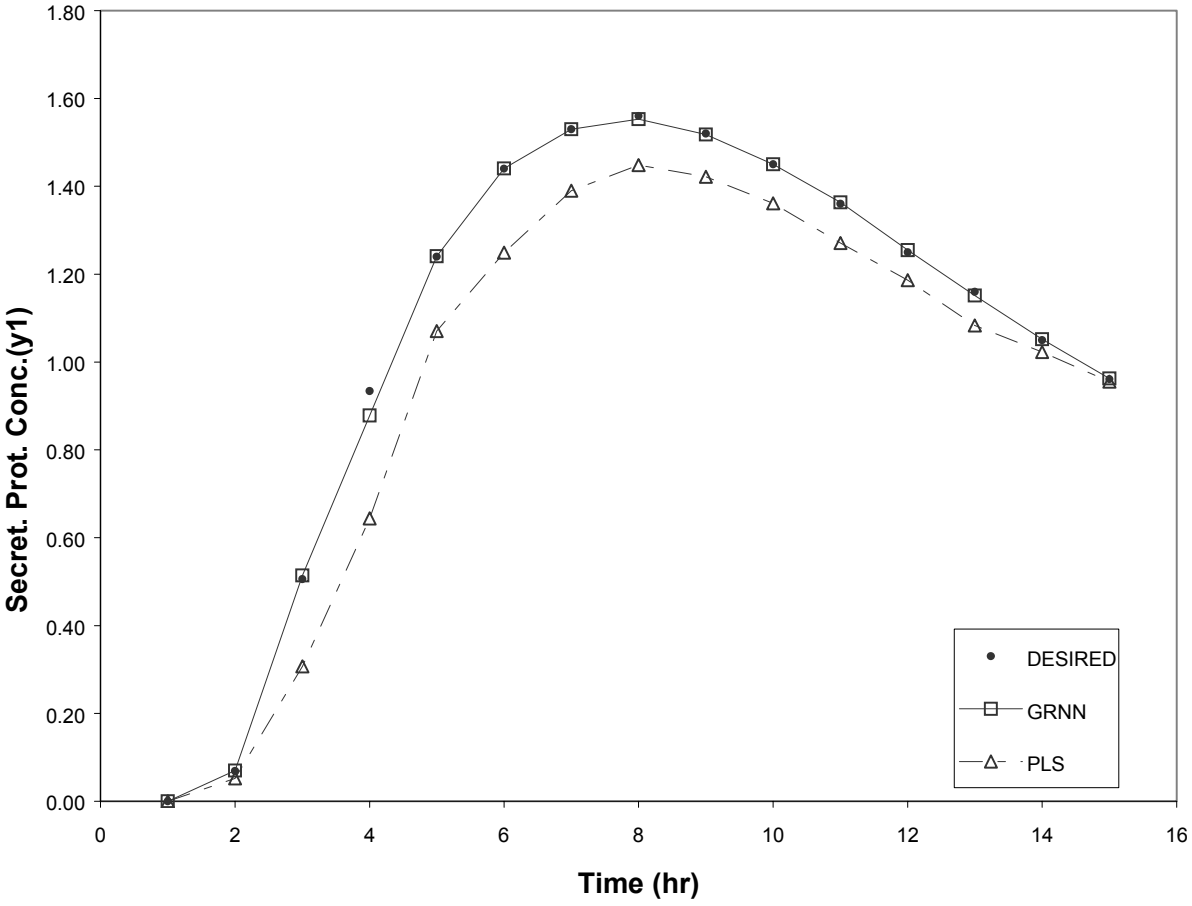


Figure 3.12: Predictions of GRNN and PLS for a test batch using input data containing 5% Gaussian noise for protein synthesis problem

Ability of GRNNs to act as a soft-sensor and thereby predict the hourly monitored values of y_1 and y_2 was also tested. The results of GRNN-based predictions using hourly sampled inputs and outputs for a normal test batch (53rd) are portrayed in Figure 3.12. Here, 5% Gaussian noise was added to the input variables. As can be seen in these plots, while the GRNN has predicted the output values accurately (CC = 0.999), the PLS based predictions are inaccurate (CC = 0.965). As stated before, the relationship between the scores and the output variables is highly nonlinear (see Fig. 3.10) and the GRNN being capable of representing efficiently the nonlinear input-output relationships thus outperforms the linear PLS approach. Results similar to these were obtained also in the case of noise-free inputs.

3.5 CONCLUSION

In this chapter, a hybrid strategy integrating PCA and generalized regression neural networks has been presented for modeling and monitoring of batch processes. The proposed PCA-GRNN strategy uses PCA for reducing the dimensionality of the process input space and the first few principal component scores that explain a large amount of variance in the input data are used to develop a GRNN model correlating inputs and outputs. Principal advantages of GRNN-based models are: (i) ability to approximate nonlinear input-output relationships efficiently, (ii) a model can be constructed exclusively from the historic process input-output data, (iii) since there exists only one adjustable parameter in its formulation, and its training being a one-step procedure, the development of an optimal GRNN model is computationally inexpensive when compared with other neural network paradigms such as the error-back-propagation and radial basis function networks, and (iv) potential to undergo on-line training. The effectiveness of the PCA-GRNN strategy for modeling and monitoring of batch processes has been successfully demonstrated by conducting two case studies involving penicillin production and protein synthesis. The results obtained thereby, using both clean and noisy data, demonstrate that the PCA-GRNN methodology is an attractive formalism for modeling and monitoring of nonlinearly behaving batch processes.

3.6 APPENDIX OF CHAPTER THREE

Appendix 3.1

The stepwise procedure to the implement PCA-GRNN strategy is given below:

Step 1 (training data preparation): Given the 3-D input array, $\hat{\mathbf{X}}(I \times J \times K)$, and the corresponding two-dimensional array $\mathbf{Y}(I \times M)$ describing values of M output variables measured at the end of I batches, unfold the $\hat{\mathbf{X}}$ array into a 2-D matrix $\mathbf{X}$ $(I \times JK)$ as depicted in Fig. 1. Preprocess $\mathbf{X}$ and $\mathbf{Y}$ matrices (training data) such that elements of each matrix column are zero mean-centered with unit variance.

Step 2 (principal component analysis): To analyze process behavior at kth time interval (k = 1, 2, …, K), the segment (defined as sub-matrix, $\mathbf{X}^k$) of the $\mathbf{X}$ matrix comprising values of J input variables measured at the kth time interval from I batches, is extracted and subjected to the PCA using NIPALS (Geladi and Kowalski, 1986) algorithm. The PCA decomposes $\mathbf{X}^k$ as follows:

$$\mathbf{X}^k = \sum_{r=1}^{R} \mathbf{t}_r^k \left(\mathbf{p}_r^k\right)' + \mathbf{E}^k \qquad \text{(A-3.1)}$$

where r denotes the principal component index; R is the number of extracted PCs; $\mathbf{t}_r^k$ refers to the I-dimensional rth score (column) vector; $(\mathbf{p}_r^k)'$ represents the transpose of the J-dimensional loading vector and $\mathbf{E}^k$ is the residual matrix.

Step 3 (analyzing the test batch): The test batch behavior at the kth time interval is analyzed as follows: (i) compute scores corresponding to the test batch inputs using the set of loading vectors $\{\mathbf{p}_r^k\}$ obtained in Step 2 and, (ii) use the scores of the first few (R) principal components that explain a large amount of variance in the $\mathbf{X}^k$, to determine whether the scores pertaining to the test batch fall within the prespecified (e.g. 95%) control limit.

Step 4 (GRNN-based input-output modeling): Develop a GRNN model using the set of scores $\{\mathbf{t}_r^k\}$; r = 1, 2, …, R, as the input and the corresponding elements in the matrix $\mathbf{Y}$ as the target outputs. The GRNN architecture to be considered has R input nodes, I pattern nodes, M type-I summation nodes, a single type-II summation node, and M

output nodes. An optimal GRNN model is constructed by optimizing the smoothing parameter, σ, using the 'leave-one-out' cross-validation procedure.

Step 5 (output prediction for the test batch): Use the optimal GRNN model to predict the values of the output variables at the end of the test batch. The scores pertaining to the test batch obtained in the step 3(i) are utilized as the GRNN input for making the prediction.

Appendix 3.2

The simple phenomenological model of a fed-batch fermentor producing penicillin is given as (Bajpai and Reuss, 1981):

$$\frac{dx_1}{dt} = \mu(x_1, x_2)x_1 - \left[\frac{x_1}{(s_f x_3)}\right] u \tag{A-3.2}$$

$$\frac{dx_2}{dt} = -\mu(x_1, x_2)\left(\frac{x_1}{a_{x1/x2}}\right) - \rho(x_2)\left[\frac{x_1}{b_{y/x2}}\right] - \gamma(x_2)x_1 + \left[1 - \left(\frac{x_2}{s_f}\right)\right]\left(\frac{u}{x_3}\right) \tag{A-3.3}$$

$$\frac{dx_3}{dt} = \frac{u}{s_f} \tag{A-3.4}$$

$$\frac{dy}{dt} = \rho(x_2)x_1 - k_d y - \left[\frac{y}{(s_f x_3)}\right] u \tag{A-3.5}$$

$$\mu(x_1, x_2) = \frac{(\mu_m x_2)}{(k_x x_1 + x_2)} \tag{A-3.6}$$

$$\gamma(x_2) = \frac{(m_{x_2} x_2)}{(k_m + x_2)} \tag{A-3.7}$$

$$\rho(x_2) = \frac{(\rho_m x_2)}{\left[k_P + x_2\left(1 + \left\{\frac{x_2}{k_I}\right\}\right)\right]} \tag{A-3.8}$$

where x_1, x_2, and x_3 represent, respectively, the biomass concentration (g l^{-1}), substrate concentration (g l^{-1}) and reactor volume (l); t, u and s_f denote time (min), feed flow rate (manipulated variable, l h^{-1}), and feed substrate concentration, respectively, and y represents the product concentration. Values of the various parameters and the initial conditions used in the simulations are (Lim et. al., 1986):

$\mu_m = 0.11$ h^{-1} ; $\rho_m = 0.0055$ g g^{-1} h^{-1} ; $k_x = 0.006$ g g^{-1} h^{-1} ; $k_p = 0.0001$ g l^{-1};

$k_I = 0.1$ g l^{-1} ; $k_d = 0.01$ h^{-1} ; $k_m = 0.0001$ g l^{-1} ; $m_{x_2} = 0.029$ g g^{-1} h^{-1};

$a_{x_1/x_2} = 0.47$ g g^{-1} ; $b_{y/x_2} = 1.2$ g g^{-1} ; $s_f = 500$ g l^{-1}

Initial conditions: $x_1(0) = 1.5$ g l^{-1} ; $y(0) = 0.0$ g l^{-1}

$x_2(0) = 0.0$ g l^{-1} ; $x_3(0) = 7.0$ l

The feed-rate profile is given by:

$$u = \left(4 \times 10^{-5}\right) t^3 - 0.0075 t^2 + 0.303 t + 12.267 \qquad \text{(A-3.9)}$$

Appendix 3.3

The phenomenological model for protein synthesis in a fed-batch reactor is given as (Lim et. al., 1977; Balsa-Canto, et. al., 2001):

$$\frac{dy_1}{dt} = g_1(y_2 - y_1) - \frac{u}{x_3} y_1 \tag{A-3.10}$$

$$\frac{dy_2}{dt} = g_2 x_1 - \frac{u}{x_3}(y_2) \tag{A-3.11}$$

$$\frac{dx_1}{dt} = g_3 x_1 - \frac{u}{x_3} x_1 \tag{A-3.12}$$

$$\frac{dx_2}{dt} = -7.3 g_3 x_1 + \frac{u}{x_3}(20 - x_2) \tag{A-3.13}$$

$$\frac{dx_3}{dt} = u \tag{A-3.14}$$

$$g_1 = \frac{4.75 g_3}{0.12 + g_3}, \tag{A-3.15}$$

$$g_2 = \frac{x_2}{0.1 + x_2} \exp(-5.0 x_2), \tag{A-3.16}$$

$$g_3 = \frac{21.87 x_2}{(x_2 + 0.4)(x_2 + 62.5)} \tag{A-3.17}$$

where t, x_1, x_2 and x_3 refer to time (min), culture cell density (g l^{-1}), substrate concentration (g l^{-1}), and hold up volume (l), respectively; y_1 and y_2 are the concentrations (g l^{-1}) of the secreted protein and the total protein, respectively, and u refers to the nutrient (glucose) feedrate (l h^{-1}). The fed-batch culture is fed at an exponentially increasing nutrient feed rate, $u = u_o e^{0.219t}$ (Balsa-Canto, et. al., 2001), with the constraint, $0 \le u \le 2.5$, where u_0 is a constant (= 0.0926 l h^{-1}).

3.7 NOMENCLATURE

d : Eucledian distance between vectors $\boldsymbol{x}$ and $\boldsymbol{x_i}$

E: residual of **X** matrix after subtraction of r number of components

F : residual of **Y** matrix after subtraction of r number of components

h : Gaussian residual basis function

$\mathbf{p_r}$: row vector of loadings for input data patterns

P : matrix of loadings for input data patterns

$\mathbf{u_r}$: column vector of scores for output data patterns

U : matrix scores for the output data patterns

X : unfolded matrix of predictor variable measurement of size ($n \times m$)

$\underset{\sim}{\mathbf{X}}$: three dimensional array of predictor variable measurements

y : a column vector of the process output variables

Y : matrix of the process output variables of size ($n \times p$)

Greek Symbols

β : regression coefficient for a single PLS component

Γ : regression residuals in PLS

σ : smoothing parameter for GRNN

3.8 REFERENCES

1. Bajpai, R. K. and Reuss, M. "Evaluation of Feeding Strategies in Carbon Regulated Secondary Metabolic Production Through Mathematical Modeling," ***Biotech. & Bioengg.***, **23**, 714 – 738, (1981).
2. Balsa-Canto, E., Banga, J. R., Alonso, A. A. and Vassiliadis, V. S. "Dynamic Optimization of Chemical and Biochemical Processes Using Restricted Second Order Information," ***Comp. & Chem Engg.***, **25**, 539 – 546, (2001).
3. Basseville, M. "Detecting Changes in Signals and Systems: A Survey," ***Automatica***, **24**, 309 – 348, (1998).
4. Bonastre, A., Ors, R. and Peris, M. "Distributed Expert Systems as a New Tool in Analytical Chemistry," ***Trends in Anal. Chem.***, **20**, 5 – 17, (2001).
5. Cheung, J. T. and Stephanopoulos, G. "Representation of Process Trends: 1. A Formal Representation Framework," ***Comp. & Chem. Engg.***, **14**, 495 – 502, (1989).
6. Fahlman, S. E. "An Empirical Study of Learning Speed in Back-Propagation Networks," Technical Report CMU-CS-88-162, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA (1988).
7. Geladi, P. and Kowalski, B. R. "Partial Least-Squares Regression: A Tutorial," ***Anal. Chim. Acta***, **185**, 1 – 17, (1986).
8. Holcomb, T. R. and Morari, M. "PLS / Neural Networks," ***Comp. & Chem. Engg.***, **16**, 392 – 411, (1992).
9. Hoskins, J. and Himmelblau, D. "Artificial Neural Network Models of Knowledge Representation in Chemical Engineering," ***Comp. & Chem. Engg.***, **12**, 881 – 890, (1988).
10. Iserman, R. "Process Fault Detection Based on Modeling and Estimation Methods. A Survey," *Automatica*, **20**, 387 – 404, (1984).
11. Janusz, M. and Venkatasubramanian, V. "Automatic Generation of Qualitative Description of Process Trends for Fault Detection and Diagnosis," ***Engg. Appl. in Artif. Intell.***, **4**, 329 – 342, (1991),.
12. Kosanovich, K. A., Dahl, K. S. and Piovoso, M. J. "Improved Process Understanding Using Multi-way Principal Component Analysis," ***Ind. Engg. Chem. Res.***, **35**, 138 – 146,

(1996).

13. Kourti, T. and MacGregor, J. F. "Multivariate SPC Methods for Process and Product Monitoring," ***J. of Qual. Tech***., **28**, 409 – 428, (1996).
14. Kozub, D. J. and MacGregor, J. F. "State Estimation for Semi-batch Polymerization Reactors," ***Chem. Engg. Sci***., **47**, 1047 – 1062, (1992).
15. Lennox, B., Hiden, H. G., Montague, G. A., Kornfeld, G. and Goulding, P. R. "Application of Multivariate Statistical Process Control to Batch Operations," ***Comp. & Chem. Engg.***, **24**, 291 – 296, (2000).
16. Lim, H. C., Chen, B. J. and Creagan, C. C. "An Analysis of Extended and Exponentially-Fed-batch Cultures," ***Biotech. & Bioengg.***, **14**, 425 – 433, (1977).
17. Lim, H. C., Tayeb, Y. J, Modak,, J. M. and Bonte, P. "Computational Algorithm for a Class of Fed-batch Fermentation," ***Biotech. & Bioengg***., **28**, 1408 – 1420, (1986).
18. Nomikos, P. and MacGregor, J. F. "Monitoring Batch Processes Using Multi-way Principal Component Analysis," ***AIChE Jour.***, **40**, 1361 – 1375, (1994a).
19. Nomikos, P. and MacGregor, J. F. "Multiway Partial Least Squares in Monitoring Batch Processes," 1st *International Chemometrics InterNet Conference*, InCINC. (1994b). URL: http://www.emsl.pnl.gov: 2080/docs/incinc/n_way_anal/PNdoc.html
20. Nomikos, P. and MacGregor, J. F. "Multivariate SPC Charts for Monitoring Batch Processes," ***Technometrics***, **37**, 41 – 59, (1995) .
21. Qin, S. J. and McAvoy, T. J. "Nonlinear PLS Modeling Using Neural Networks," ***Comp. & Chem. Engg.***, **16**, 379 – 391, (1992),
22. Qin, S. J. "Neural Networks for Intelligent Sensors and Control – Practical Issues and Some Solutions," In D. Elliot, ***Neural Networks for Control***, Chap. 8, (pp. 215 – 236) Academic Press. (1997).
23. Reifman, J. and Vitela, J. "Accelerating Learning of Neural Networks with Conjugate Gradients for Nuclear Power Plant Applications," ***Nucl. Tech.***, **106**, 225 – 241, (1994).
24. Riedmiller, M. and Braun, H. "A Direct Adaptive Method for Faster Backpropagation Learning: the RPROP Algorithm," ***Proc. of the IEEE Int. Conf. On Neural Networks***, San Fransisco, CA, Mar. 28-Apr. 1, (1993).
25. Rumelhart, D., Hinton, G. and Williams, R. "Learning Representations by

Backpropagating Errors," ***Nature***, **323**, 533 – 536, (1986).

26. Specht, D. F. "A General Regression Neural Network," ***IEEE Trans. on Neural Net.***, **2**, 568 – 576, (1991).
27. Vaidyanathan, R. and Venkatasubramanian, V. "Representing and Diagnosing Dynamic Process Data Using Neural Networks," ***Engg. Appl in Artif. Intell.***, **5,** , 11 – 23 (1992).
28. Venkatasubramanian, V., Vaidyanathan, R. and Yamamoto, Y. "Process Fault Detection and Diagnosis Using Neural Networks: 1. Steady State Processes," ***Comp. & Chem. Engg***., **14**, 699 – 712, (1990).
29. Vora, N., Tambe, S. S. and Kulkarni, B. D. "Counterpropagation Neural Networks for Fault Detection and Diagnosis," ***Comp. & Chem. Engg.***, **21**, 177 – 185, (1997).
30. Watanabe, K., Abe, M., Kubota, M. and Himmelblau, D. M. "Incipient Fault Diagnosis of Chemical Processes via Artificial Neural Networks," ***AIChE Jour.***, **35**, 1803 – 1812, (1989).
31. Willsky, A. S. "A Survey of Design Methods for Failure Detection in Dynamic Systems," ***Automatica***, **12**, 601 – 611, (1976).
32. Wold, S., Esbensen, K. and Geladi, P. "Principal Component Analysis," ***Chemo. & Intell. Lab. Syst***., **2**, 37 – 52, (1987).
33. Zheng, L. L., McAvoy, T. J., Huang, Y. and Chen, G. "Applications of Multivariate Statistical Analysis in Batch Processes," ***Ind. Engg. Chem. Res***., **40**, 1641 – 1649, (2001).

CHAPTER 4

FORECASTING BATCH PROCESS DYNAMICS USING GENERALIZED REGRESSION NEURAL NETWORKS

Proceedings of ***National Seminar of Instrumentation and Control in Chemical Industries*** (***ICCI 2K2***), at LIT Nagpur (2002)

Abstract

Process monitoring plays an important role in quality control and ensuring process safety. A Generalized Regression Neural Network (GRNN) based strategy has been proposed in this chapter for monitoring and modeling of batch processes. The methodology has been successfully utilized to forecast the dynamics of a fed batch bio-reactor for ethanol production.

4.1 INTRODUCTION

Owing to their flexibility in producing low-volume, high-value and high-quality products, batch and fed-batch processes are extensively used in the production of pharmaceuticals, polymers, biochemicals and agrochemicals Constant monitoring of batch processes is critical from the viewpoint of safe operation and maintaining the product quality. To get high quality products consistently, the batch process operating variables should follow their specified trajectories precisely. However, disturbances arising from the deviations in the specified trajectories, errors in charging the vessel with materials, and variations in impurities, do lead to batch-to-batch variations (Nomikos and MacGregor, 1994a), which adversely affect the product quality. Process variability can be reduced by employing an efficient monitoring and control system. Such a system while working online must be capable of identifying any abnormal process behavior so that corrective measures could be taken well before the batch completes its duration.

The process history-based approach, which is especially suitable for process monitoring purposes, requires a large amount of data in order to capture and model the features of the process. The history-based models can be subdivided into qualitative and quantitative models. The basis of qualitative models consists of rule-based and trend modeling methodologies, whereas the quantitative methods are divided into statistical and non-statistical, neural networks based pattern recognition models (Venkatasubramanian, et. al., 2003). Common features of the statistical methods used are their ability to reduce correlations between variables, compress data, and reduce the dimensionality of the data. These characteristics enable efficient extraction of the relevant information and analysis of the data. The most important statistical monitoring methods are based on principal component analysis (Wold et. al., 1987; Chen and Liu, 2002) and partial least squares regression (Geladi and Kowalski, 1986).

Artificial neural networks (ANNs) find applications in process monitoring and forecasting whereby an ANN-based model capable of identifying and diagnosing various faults (that can occur in a process operation) is constructed (trained) from the process data-base describing faulty and normal process operation. Such a database can be created from the measurements of past fault occurrences. The multi-layered perceptron is the most widely used ANN paradigm for the stated tasks. Usually, training of an MLP-based model is

conducted off-line, following which the trained model can detect and diagnose faults while being on-line with the process. The principal advantages of the ANN-based monitoring are: (i) the fault diagnosis / detection system can be developed solely from the historic process data without the knowledge of detailed process phenomenology and, (ii) the system can represent nonlinear relationships existing between the abnormally behaving causal process variables and the corresponding faults. Several studies dealing with the ANN-based fault detection and diagnosis have been performed mainly for continuous processes operating at a steady-state (Hoskins and Himmelblau, 1988; Vora et. al., 1997). Notwithstanding its attaractiveness, the principal drawbacks of the ANN-based approach to process monitoring are: (i) though ANNs model process nonlinearities effeciently, they are essentially empirical models and, therefore, their parameters (weights) in most cases, cannot be interpreted using sound statistical basis (Nomikos and MacGregor, 1994b), (ii) development of an optimal ANN-based model is a heuristic procedure and, therefore, it may not be feasible to build the model in short time intervals as may be necessary for batch processes exhibiting fast dynamics, and (iii) the ANN-based models need upgradation whenever the process exhibits a new type of behavior unrepresented in the historic database used for developing the models. Development of an ANN model possessing good prediction and generalization accuracy is essentially a heuristic procedure wherein effects of network's structural parameters (number of hidden layers, number of nodes in each hidden layer), and training algorithm specific parameters (learning rate, momemtum coefficient, etc.) need to be studied rigorously. Such heuristic makes the MLP-based method computationally expensive and therefore unsuitable for batch processes wherein time duration to take corrective action in the event of an abnormal process behavior is short. Thus in the present work, we propose a process modeling and monitoring formalism utilizing a numerically efficient ANN paradigm called generalized regression neural network (GRNN). Specifically, the proposed formalism makes use of the GRNN-based models for nonlinearly correlating the process output variables with the operating variables. The principal advantages of the GRNN-based models are: (i) they can efficiently approximate multiinput-multioutput (MIMO) nonlinear relationships and, (ii) the stated relationships can be developed in a significantly shorter time as compared to the MLP or RBFN-based process models, (iii) GRNN can approximate linear as well as nonlinear relationships existing between process inputs and outputs.

A brief overview of GRNN is presented in the following section. Next section presents the results and discussion on the GRNN-based modeling and monitoring of ethanol production process. The chapter concludes with the findings and concluding remarks.

4.2 GENERALIZED REGRESSION NEURAL NETWORKS

GRNNs were introduced (Specht, 1991) as a generalization of both the radial basis function networks (RBFNs) and probabilistic neural networks (PNNs). They are memory based feedforward networks meaning that all the training samples are stored in the network. With increasing number of training samples, the GRNN asymptotically converges to the optimal regression surface. GRNNs possess a special property that they do not require iterative training. They are based on the estimation of probabilistic functions and hence are founded on a sound statistical basis (Seng et al., 2001). A detail mathematical description of GRNN is already resented in chapter 3 (refer to section 3.2.2). Unlike the most popular error–back-propagation (EBP) algorithm that trains multilayer feedforward networks iteratively, the GRNN training is a single pass procedure. Consequently, the GRNN trains itself in a significantly shorter time as compared to the EBP-based training. For simplicity, a GRNN with multiple input-single output (MISO) architecture has been shown in Fig. 4.1, consisting of four layers, namely, the *input*, *hidden*, *summation* and *output* layers.

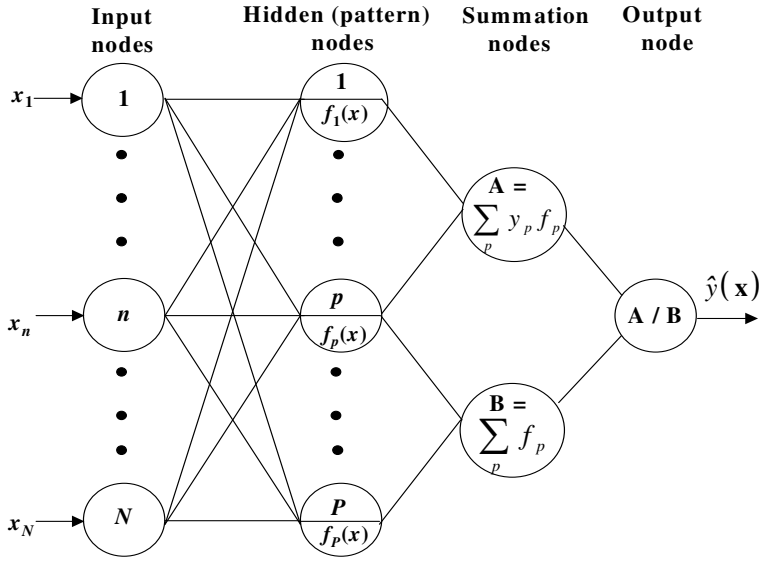


Figure 4.1: Schematic representation of a MISO Generalized Neural Network architecture

To train an MIMO type GRNN, the training data comprises a set of P number of N–dimensional input vectors, $\{\mathbf{x}_p\}$, and the corresponding M-dimensional output (target) vectors, $\{\mathbf{y}_p\}$. Given a test input vector, $\mathbf{x}$, the GRNN procedure for predicting the value of an mth response (output) variable ($m = 1, 2, \ldots, M$) can be viewed as computing the weighted average of all the target values of that variable wherein weights are taken proportional to the Euclidean distances between the training input vectors and the test input vector (nearest-neighbor criterion). The input layer of GRNN houses as many nodes as the dimension (N) of the input vectors, $\mathbf{x}_p$. They serve as 'fan-out' units and simply pass on the input vector information to the hidden units (also called 'pattern' units). GRNN's hidden layer houses a number (P) of nodes that equals the number of training vectors (patterns); that is, each hidden node represents a different training input pattern. When an input pattern is applied to the input nodes, its distance from each of the training patterns stored in the hidden nodes is computed. Next, the distance is transformed using the Gaussian RBF thus forming the output, h_p, of the pth hidden unit. The third, i.e. the summation layer, comprises two types of summation units as shown in Fig. 4.1. An mth type-I unit (indexed as N_m), computes the summation ($\sum_{p=1}^{P} y_p\, h_p$) defined in the numerator of Eq. 3.7, by utilizing the hidden unit outputs, $\{h_p\}$, and the mth elements of all the M-dimensional target output vectors,$\{\mathbf{y}_p\}$. There exists a single type-2 unit in the GRNN's third layer that performs summation of the outputs of all the hidden nodes ($\sum_{p=1}^{P} h_p$). Finally, the mth output layer node performs the normalization step defined in Eq. 3.7 to compute the GRNN-predicted value of the mth output variable, $\hat{y}_m(\mathbf{x})$. As can be observed in the above procedure that GRNN's training algorithm uses a single adjustable parameter, namely, the width or smoothing parameter (σ) of the Gaussian PDF. It is also possible to use varying smoothing parameter values for the Gaussian RBFs associated with the pattern nodes. The significance of the smoothing parameter is that as its value becomes smaller (larger) the regression performed by the GRNN becomes more local (global). That is, as σ decreases, the GRNN's output predictions for an input vector $\mathbf{x}$, go closer to the target output vector of that training input vector, which is nearest to $\mathbf{x}$ (more accurate approximation). Conversely, as σ increases (thereby applying greater smoothing for noisy $\mathbf{x}$ vectors), the target output vectors pertaining to multiple

nearest neighbors of **x** influence GRNN's prediction. In the extreme case of $\sigma = +\infty$, the GRNN returns the mean value of all the training output vectors regardless of the input. It can thus be noticed that the magnitude of σ needs to be choosen judiciously as it significantly affects the accuracy of GRNN's predictions. The commonly employed technique for automatically selecting the optimal σ value is the 'leave–one-out' cross-validation method. In this technique, one input-output vector is removed from the training set of *N* vectors and a GRNN model is built using the remaining *N*-1 vectors for predicting the outputs corresponding to the removed pattern. This procedure is repeated *N* times by setting aside each time a different training pattern and the mean-squared-error (MSE) is evaluated by comparing the GRNN-predicted and the corresponding target output values. This entire procedure is repeated by varying the σ value systematically and the value that minimizes the MSE is chosen to be optimal.

4.3 FORECASTING FED-BATCH BIOREACTOR FOR PRODUCTION OF ETHANOL

An application of the GRNN-based strategy for batch process modeling and monitoring is illustrated by conducting a case study comprising biosynthesis of ethanol. In the absence of real life plant data, dynamic process input-output data were obtained by simulating the phenomenological models of the process (Banga et al., 1997; Jayaraman et al., 2001). It may however be noted that the GRNN-based strategy does not require the phenomenological process model for its implementation. In real practice, historic input-output data from the batch process should be used for conducting process modeling and monitoring. Using typical variations in the initial values of the process operating variables, input-output data for a number of batches were generated. Batches with their input variables within the acceptable operating range (*normal* behavior) and also outside the range (*abnormal* behavior) were simulated to generate the training input-output data set for developing the PCA-based GRNN models. Also, a number of input-output vectors to be taken as the "test" set, were generated by assuming the input process conditions both in the acceptable and unacceptable ranges.

The data were generated assuming exponential feed policy as given by

$$u(t) = 0.2 \times \exp(0.0662 \times t) \tag{4.1}$$

The overall phenomenological equations of the fed-batch fermentation process are as follows (Banga et al., 1997; Jayaraman et al., 2001):

$$\frac{dx_1}{dt} = g_1 x_1 - \frac{x_1}{x_3} u(t) \tag{4.2}$$

$$\frac{dx_2}{dt} = -10\, g_1 x_1 + \frac{150 - x_2}{x_3} u(t) \tag{4.3}$$

$$\frac{dx_3}{dt} = u(t) \tag{4.4}$$

$$\frac{dy_1}{dt} = g_2 x_1 - \frac{y_1}{x_3} u(t) \tag{4.5}$$

$$y_2 = y_1\, x_3 \tag{4.6}$$

$$g_1 = \left\{ \frac{0.408}{\left(1 + \frac{y_1}{16}\right)} \right\} \left\{ \frac{x_2}{0.22 + x_2} \right\} \tag{4.7}$$

$$g_2 = \left\{ \frac{1}{\left(1 + \frac{y_1}{71.5}\right)} \right\} \left\{ \frac{x_2}{0.44 + x_2} \right\} \tag{4.8}$$

$$0 \le x_3 \le 200; \quad 0 \le u(t) \le 12 \tag{4.9}$$

The initial conditions are :

$x_1(0)$ = 1 g/L; $x_2(0)$ = 150 g/L; $x_3(0)$ = 10 L; $y_1(0)$ = 0 g/L; $y_2(0)$ = 0 g

The bioreactor is characterized in terms of three process variables, namely, cell mass concentration (x_1) (g/L), substrate concentration (x_2) (g/L), reactor volume (x_3) (L) and two outputs product concentration (y_1) (g/L) and total ethanol production (y_2) (g). Specifically, the process input-output data for a total of 56 batches were generated using the above described phenomenological model; the data set comprises values of three predictor variables (x_1, x_2, x_3) measured at the end of one-hour time intervals and the value of the output variables y_1 and y_2 measured at the end of the batch run (54 hr). The set to be used as the

training set consisted data from 50 batches with their initial conditions ($x_1(0)$, $x_2(0)$, $x_3(0)$) in the normal range ($0.5 \leq x_1(0) \leq 1.5$, $9.0 \leq x_2(0) \leq 11.0$ and $148.0 \leq x_3(0) \leq 152.0$) and six batches, numbered 11, 31, 43, 44, 48 and 49, with their initial conditions outside the normal range (abnormal batches) as evident from Fig. 4.2 and 4.3. Additionally, an input-output test set comprising the batch number 54 with abnormal initial conditions and five batches (51, 52, 53, 55 and 56) with normal initial conditions, were generated in a manner similar to the training set. The task of the GRNN strategy is to analyze, at the end of each one-hour interval, whether: (i) the predictor variables are behaving normally (monitoring), and (ii) the process output at the end of the batch will be in the acceptable range (input-output modeling). The real benefit of the GRNN-based process monitoring strategy lies in predicting during the progression of the batch the process outputs, which are nonlinearly dependent on the inputs. Here, a separate GRNN model was developed using the predictor variable data pertaining to each of the fifty four one-hour time intervals. The training input data for developning the GRNN models contained batches with normal as well as abnormal initial conditions. Consideration of all the available data for the GRNN training was done primarily to increase the training data size, which helps in improving model's prediction accuracy. The training output data comprised values of the ethanol concentration (y_1) and total ethanol produced (y_2) at the end of the batch. Optimality of the GRNN models resulting in accurate output predictions was ensured by optimizing the smoothing parameter, σ, using the leave-one-out cross-validation method described earlier.The GRNN–predicted values of y_1 and y_2 at the end of training branches using 54 hour predictor data are shown in Figs. 4.2 and 4.3, respectively. It can be seen that the predictions exhibit good accuracy. The optimal GRNN models were subsequently used in the prediction mode for forecasting the magnitude of the output variable for the test batches. The values of the output variable at the end of all six test batches were predicted in this manner and compared with the corresponding desired values. For illustration, the results of the GRNN-based output prediction with respect to all the test batches (51 to 56) using 30th hour input variable data are shown in Figs. 4.4 and 4.5, where the desired values of the output are also indicated. As can be observed, the GRNN has predicted the values of the output at the end of each batch accurately. It has predicted correctly that batch number 54 is an abnormal batch (see Figs. 4.4 and 4.5) since its output lies outside 95 % and 99 % confidence limit.

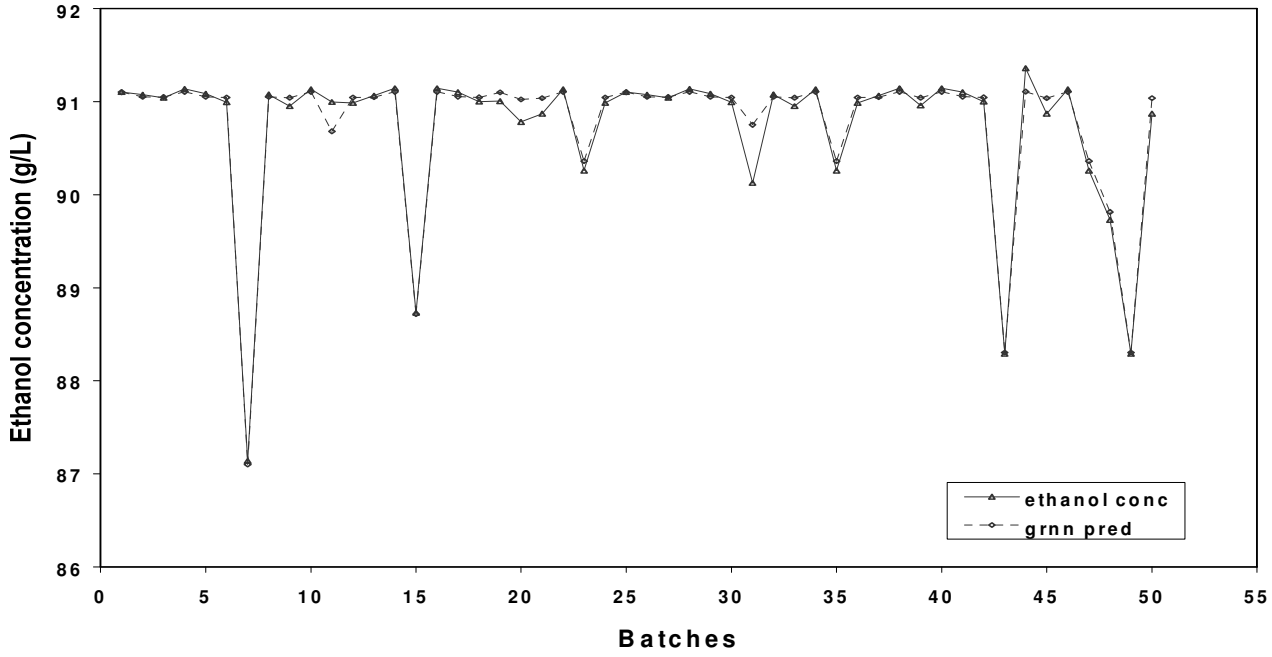


Figure 4.2: GRNN-based model to predict ethanol concentration at end of the batch

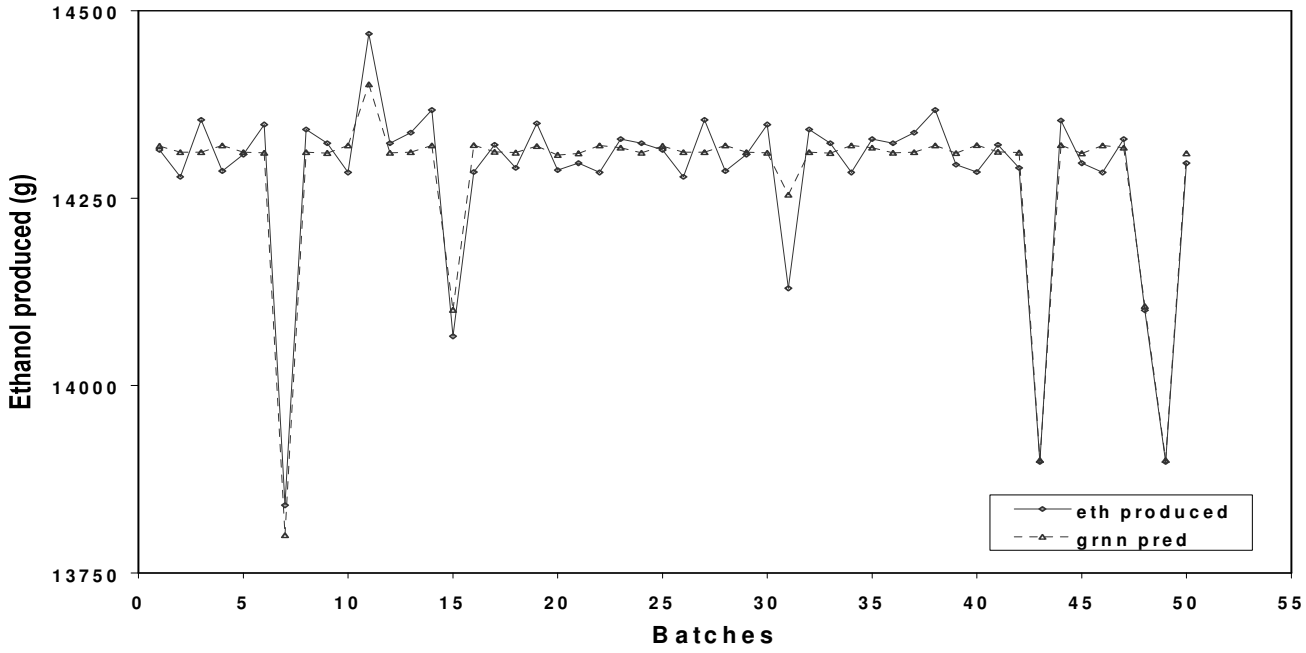


Figure 4.3: GRNN-based model to predict total ethanol produced at end of the batch

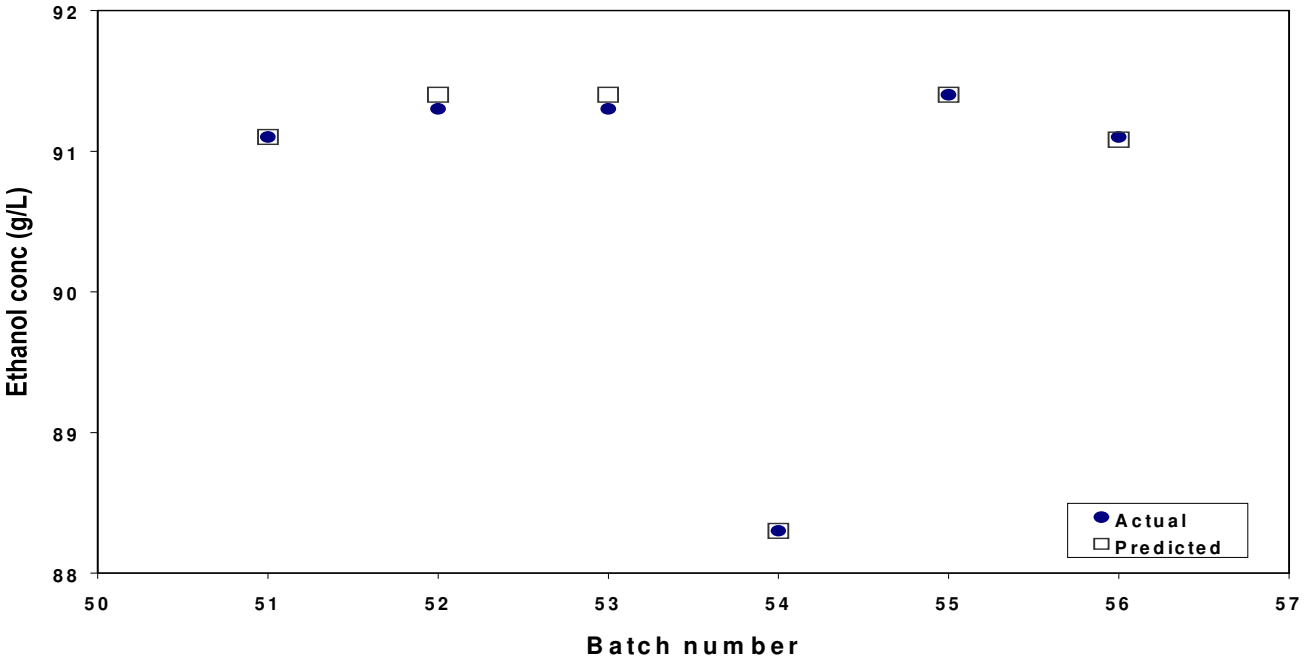


Figure 4.4: GRNN-based model to predict ethanol concentration at end of the batch based on 30th hour input – output data

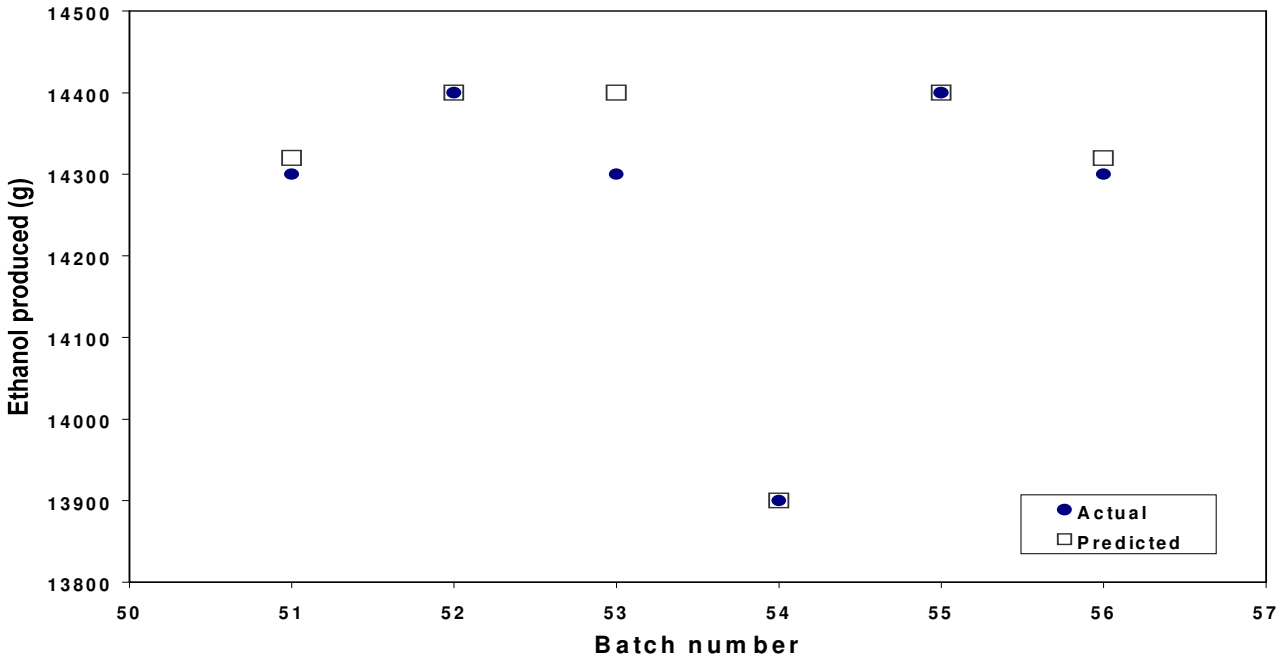


Figure 4.5: GRNN-based model to predict total amount of ethanol produced at the end of the batch based on 30th hour input – output data

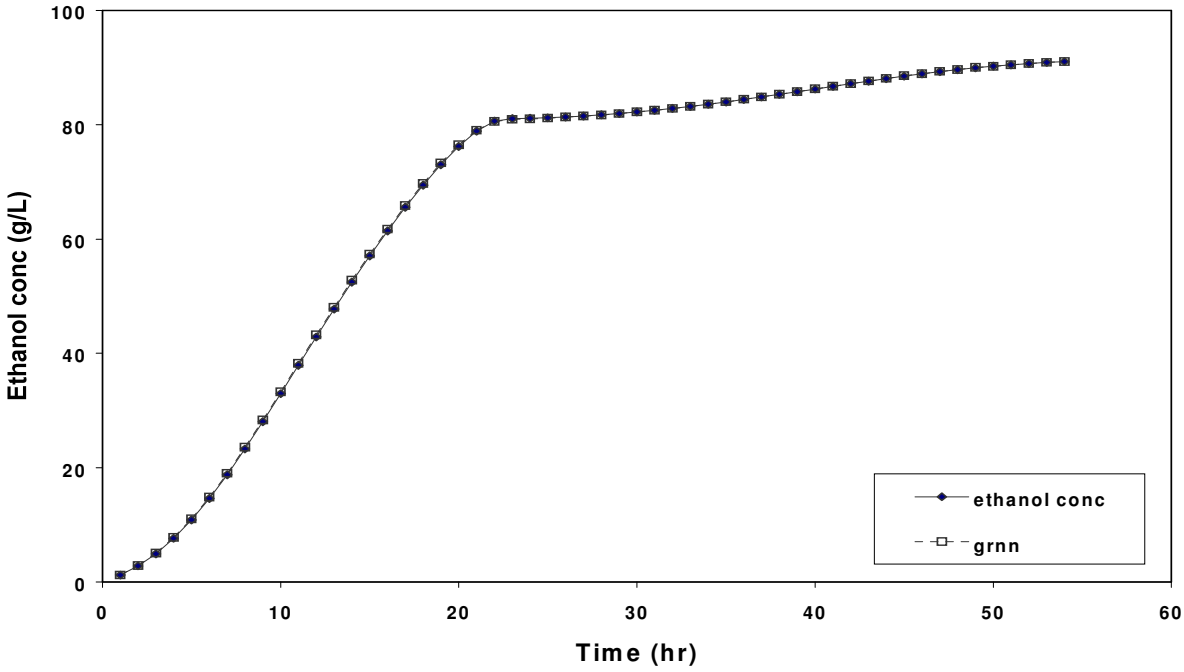


Figure 4.6: GRNN-based prediction of ethanol concentration for batch no. 56 for all the time intervals compared with the actual one

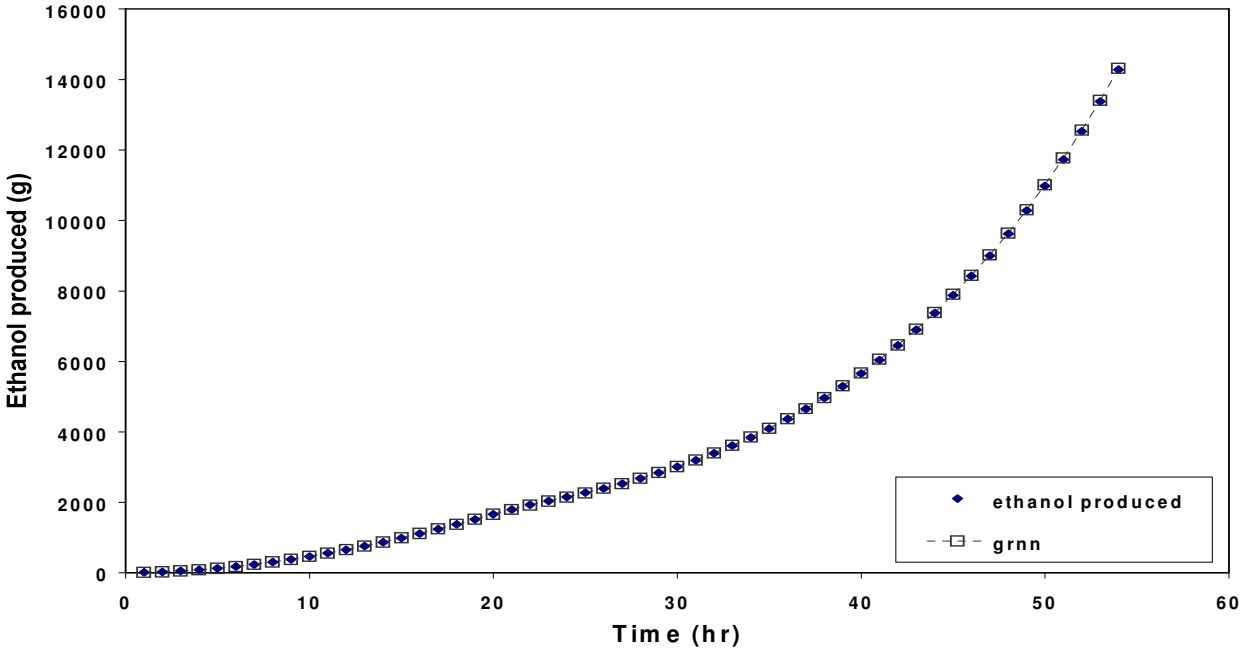


Figure 4.7: GRNN-based prediction of total ethanol produced for batch no. 56 for all the time intervals compared with the actual one

Although in most batch processes the product quality variable is measured only at the end of the batch, occasionally (i.e. wherever feasible and economical) the product samples are drawn intermittently and analyzed in the quality control laboratory. In such situations, the GRNN-based models can serve as a 'soft-sensor' whereby the model predicted values of the quality variable become available almost instantaneously upon sampling the product. To test the efficacy of the GRNNs to act as a soft-sensor, fifty four models were constructed using the training set comprising scores of the hourly sampled input data and the corresponding values of the output variable. Next, these models were used to predict the magnitude of the quality variable sampled at one-hour time intervals corresponding to the test batches. The results of such a GRNN-based prediction using hourly sampled inputs for a test batch (batch number 56) with normal initial conditions are portrayed in Figs. 4.6 and 4.7. As can be seen in these plots that the GRNN has predicted the hourly sampled output values accurately thereby proving its effectiveness as a soft sensor.

4.4 CONCLUSION

We have presented a GRNN-based process monitoring and modeling strategy to forecast dynamics of a process. The strategy has been shown to provide excellent results when applied to a fed batch bio-reactor for the production of ethanol. The advantage of the strategy is that: (i) it can make accurate forecast even when the process behavior is nonlinear and (ii) the network model training is much faster compared to the classical error back propagation strategy, which can be gainfully utilized for real-time forecasting.

4.5 NOMENCLATURE

$f(\mathbf{x}, y)$: Gausian probability density function

h_p : a distance – based Gaussian redial basis function

x_1 : cell mass concentration (g/L)

x_2 : substrate concentration (g/L)

x_3 : reactor volume (L)

y_1 : product concentration (g/L)

y_2 : total ethanol production (g)

$\hat{y}(\mathbf{x})$: response variable y i.e., regression variable

$u(t)$: feed flow rate as a function of time

4.6 REFERENCES

1. Banga, J. R., Alonso, A. A. and Singh R. P. "Stochastic Dynamic Optimization of Batch and Semicontinuous Bioprocesses", ***Biotech. Progress***, **13**, 326 – 335, (1997).
2. Chen, J. and Liu, K.C. "On-line Batch Process Monitoring using Dynamic PCA and Dynamic PLS Models", ***Chem. Engg. Sci.***, **57**, 63 – 75, (2002).
3. Geladi, P. and Kowalski, B. R. "Partial Least-Squares Regression: A Tutorial", ***Analyt. Chimica Acta***, **185**, 1 – 17, (1986).
4. Hoskins, J. and Himmelblau, D. "Artificial Neural Network Models of Knowledge Representation in Chemical Engineering", ***Comp. & Chem. Engg.***, **12**, 881 – 890, (1988).
5. Jayaraman, V. K., Kulkarni B. D., Gupta, K., Rajesh, J. and Kusumaker, H. S. "Dynamic Optimization of Fed-batch Bioreactors using the Ant Algorithm", ***Biotech. Progress***, **17**, 81 – 88, (2001).
6. Nomikos, P. and MacGregor, J. F. "Monitoring Batch Processes using Multi-way Principal Component Analysis", ***AIChE Jour.***, **40**, 1361 – 1375, (1994a).
7. Nomikos, P. and MacGregor, J. F. "Multiway Partial Least Squares in Monitoring Batch Processes", *First International Chemometrics InterNet Conference*, InCINC., Available at: http://www.emsl.pnl.gov:2080/docs/incinc/n_way_anal/PNdoc.html (1994b).
8. Seng, T.L., Khalid, M. and Yusof, R. "Adaptive GRNN Modeling of Dynamic Plants, http://www.cairo.utm.my/publications/lsteo_agrnn.pdf (2001)
9. Specht, D.F. "A General Regression Neural Network", ***IEEE Trans. on Neural Networks***, **2**, 568 – 576, (1991).
10. Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N. and Yin, K. "A Review of Process Fault Detection and Diagnosis Part III: Process History Based Methods", ***Comp. & Chem. Engg.***, **27**, 327 – 346 (2003).
11. Vora, N., Tambe, S. S. and Kulkarni, B. D. "Counterpropagation Neural Networks for Fault Detection and Diagnosis", ***Comp. & Chem. Engg.***, **21**, 177 – 185, (1997).
12. Wold, S., Esbensen, K. and Geladi. P., "Principal Component Analysis", *Chemo. & Intell. Lab. Syst.*, **2**, 37 – 52, (1987).

CHAPTER 5

HYBRID PCA-FUZZY NEURAL NETWORK FORMALISM FOR BATCH PROCESS MONITORING

Proceedings of ***International Conference of Chemical and Bioprocess Engineering***, at University Malaysia Sabah, Vol. 2, pp. 773 - 779 (2003)

Abstract

Batch processes due to their complex nonlinear nature, are difficult to model phenomenologically and, therefore, multivariate statistical methods namely, principal component analysis (PCA) and partial least squares (PLS) are commonly used to model and monitor batch processes. In particular, the PCA is used to monitor whether process operating (predictor) variables are behaving normally and PLS is used for predicting the process output (response) variables. A significant drawback of the PLS is that it is a linear regression formalism and thus it makes poor predictions when relationships between process predictor and response are nonlinear. To overcome this difficulty, a formalism integrating PCA and fuzzy neural networks (FNNs) has been developed for modeling and monitoring of batch processes. The proposed methodology is generic in nature and has been successfully validated for modeling and monitoring of the protein synthesis process.

5.1 INTRODUCTION

As discussed in chapter 3, the PCA is a dimensionality reduction technique for compressing noisy and correlated process operating (input) variable measurements into a smaller, informative latent variable (*principal components* or *scores*) space. For highly correlated inputs, first few principal components (PCs) capture maximum amount of variance in the measurements. A PCA-based technique "multiway principal component analysis (MPCA)" has been developed by Wold et al. (1987), whereby dimensionality reduction can be realized for multivariate dynamic batch processes data. The PLS is a PCA-related technique that simultaneously finds latent variables for both the input and output measurement sets following which the latent variables are correlated linearly. The major advantage of the PLS method is that outputs can be predicted using only a few PCs that capture maximum variance in the input-output measurements; multiway version of the PLS (MPLS) has also been proposed (Nomikos and MacGregor, 1994a; 1994b). The PLS and MPLS though are fast and efficient, they suffer from a significant drawback that they capture only linear relationships between the principal components of the predictor and response variables. Hence, the methods perform poorly in predicting response variables of nonlinearly behaving batch processes, which are abundant in chemical industry. To overcome this difficulty, a hybrid formalism integrating PCA and fuzzy neural networks (FNN) has been developed for modeling and monitoring of batch processes. In PCA-FNN hybrid formalism, the nonlinear relationship between operating and output variables is represented using an FNN. Since an FNN may perform poorly if operating variables are correlated linearly, the PCA is performed on the operating variable space for removing the linear correlations therein and reducing its dimensionality. The major advantage of the proposed strategy is that modeling and monitoring can be performed exclusively on the basis of historic process data without invoking process phenomenology. The chapter is organized as follows. In section 5.2, an overview of the PCA / MPCA and FNN system is presented. Next in section 5.3, the proposed hybrid formalism integrating PCA and FNN (hereafter termed PCA-FNN) is described along with the results of the study performed for modeling and monitoring of protein synthesis batch process.

5.2 PROCESS MONITORING TOOLS

5.2.1 PCA / MPCA

Experimental input data from a batch process can be represented as a three-dimensional array (see Fig. 5.1*a*) forming a matrix $\hat{\mathbf{X}}(I \times J \times K)$ where I represents batch index, J refers to the number of measured variables, and K is the number of fixed-length time intervals over which measurements are made. A method to analyze this data is MPCA (Nomikos and MacGregor, 1994a; 1994b) that creates an empirical reference model from the past measurements involving "normal (in control)" and "abnormal (out-of-control)" batches for detecting unusual behavior of a test batch (Nomikos and MacGregor, 1994a; Nomikos and MacGregor, 1995; Kourti and MacGregor, 1996). In MPCA procedure, PCA is performed on the large two-dimensional matrix $\mathbf{X}$ formed by unfolding the three-way array, $\hat{\mathbf{X}}$. The unfolding is done by taking vertical slices along the time axis of matrix $\hat{\mathbf{X}}$ and ordering them side by side to the right to generate a 2-D matrix, **X,** of dimensions $(I \times JK)$ (see Fig. 5.1*b*). Accordingly, the first J columns of $\mathbf{X}$ describe all the input variables from I batches sampled at time $k = 1$. Similarly, the next set of J columns refer to the same set of input variables sampled at the end of the next time interval ($k = 2$), and so on until $k = K$. The MPCA decomposes the $\mathbf{X}$ array into the summation of the product of the score vectors, $\mathbf{t}_r$, and loading vectors, $\mathbf{p}_r$, as given by:

$$\mathbf{X} = \sum_{r=1}^{R} \mathbf{t}_r \mathbf{p}_r' + \mathbf{E} \tag{5.1}$$

where R denotes the number of extracted PCs; $\mathbf{p}_r'$ refers to the transpose of the loading vector $\mathbf{p}_r$, and $\mathbf{E}$ is the residual matrix. In this formulation, the score vectors express the relationship among batches while the loadings are related to the measured variables and their time variation.

5.2.2 Fuzzy Neural Networks (FNNs)

Recently an alternative approach to modeling has emerged under the realm of AI – based techniques for imperfect and / or incomplete data. One of the constituents of this approach is the 'artificial neural network' which maps multivariable space of information (input variables) to another (output variables). A neural network model has many remarkable

features: (i) it is adaptive and has learning ability, (ii) it can generalize and handle noisy even imperfect data, (iii) it can capture nonlinear and complex interaction among the variables of the system. Another very common method is 'fuzzy logic', which is an approximate reasoning method to cope with the uncertainties (Sugeno, 1985). This provides a systematic way of dealing with the imprecise and vague information on input data, their effects on the system, and the response of the system in form of process output (Whitnell, 1993; Rao et. al., 1999; Ghasem, 2006).

Fuzzy logic and neural networks are complementary to each other, in order to utilize strength of both; they can be combined to an integrated system. The integrated system will then have advantages of both neural networks (namely, learning abilities and connectionist structure) and the fuzzy systems (humanlike 'IF – THEN' rules and ease of incorporating expert knowledge and judgment available in linguistic terms). The complexity and nonlinearity of the underlying system are handled through a neural network while the imprecision associated with the system parameters are incorporated through fuzzy logic (Ronen et. al., 1998; Hanai et. al., 2003).

An approach to integrate fuzzy logic and neural network is to simply fuzzify some of the neural network system parameters and retain the basic properties and architectures of the neural network model. In such models, a crisp neuron becomes fuzzy and response of the neuron to its next layer activation signal is of a fuzzy relation (Huang and Wang, 1999; Pai et. al., 2009). The learning mechanisms and interpretation capability of the neural network system is enhanced by the fuzzy representations of the knowledge domain. The neural network architecture remains unchanged while the only change is in the weights connecting neurons of different layers. In order for a neural network to handle the fuzzy inputs, the network parameters, i.e., the weights that connected each neuron and the bias in each layer are fuzzified based on a two stage training mechanism (Ni et. al, 1996; Rahman et. al., 2002).

A schematic of the fuzzy neural network (FNN) used in this study is shown in Fig. 5.2 with two term nodes for each of the three input variables, one output variable, and eight rule nodes. The FNN system consists of four layers. Nodes in the first layer are input nodes which represent "linguistic variables" nodes; in layer-II are membership nodes denoting the membership functions; each membership node is responsible for mapping an input linguistic variable into a corresponding probability distribution. The rule nodes reside in layer-III which

form a fuzzy rule base. Finally, layer-IV contains output variable node (Chen and Teng, 1995).

The links between the membership and rule nodes are the antecedent links and those between the rule nodes and output node are the consequent links. For each rule node, there is at most one antecedent link from a membership node of a linguistic variable. Hence there are $\prod_i |T(x_i)|$ rule nodes in the proposed FNN structure. Here, $|T(x_i)|$ denotes the number of fuzzy partitions of the input linguistic variable, x_i. Moreover, all consequences links are fully connected to the output nodes and interpreted directly as the strength of the output variable. In this way the consequence of a rule is simply the product of the rule node output, which is the firing strength of the fuzzy rule and the consequence link. Thus, the overall net output is treated as a linear combination of the consequences of all rules instead of the complex composition of a rule of inference and defuzzification process. The fuzzification strategy, decision making logic and defuzzification strategies are formulated as the functions of the nodes in network and the fuzzy rules are represented by the learnable weights of the input links on one of the layers in the network.

5.2.2.1 Reasoning Method:

For an *n*-input – single output system, let x_i be the *i*th input linguistic variable and we define α_k as the firing strength of rule *k*, which is obtained from the product of the grades of the membership functions $\mu_{A_i^k}(x_i)$ in the antecedent. If w_k represents the *k*th consequence link weight, the inferred value, *y**, is obtained from the weighted sum of its inputs, i.e., $\sum_k w_k \alpha_k$. The FNN realizes the inference as follows:

R^k : IF x_1 is $A_1^k(x_1)$, …, and x_n is $A_n^k(x_n)$

THEN $y = w_k$, $k = 1, 2, \ldots, m$ (5.2)

$$y^* = \sum_{k=1}^{m} \alpha_k w_k \ , \ \alpha_k = \prod_{i=1}^{n} \mu_{A_i^k}(x_i) \tag{5.3}$$

This reasoning method is a variation of the method introduced by Sugeno (1985), where the consequence of a rule is a function of the input variables. For the proposed FNN, this function is replaced by a constant value and a different defuzzification process is used.

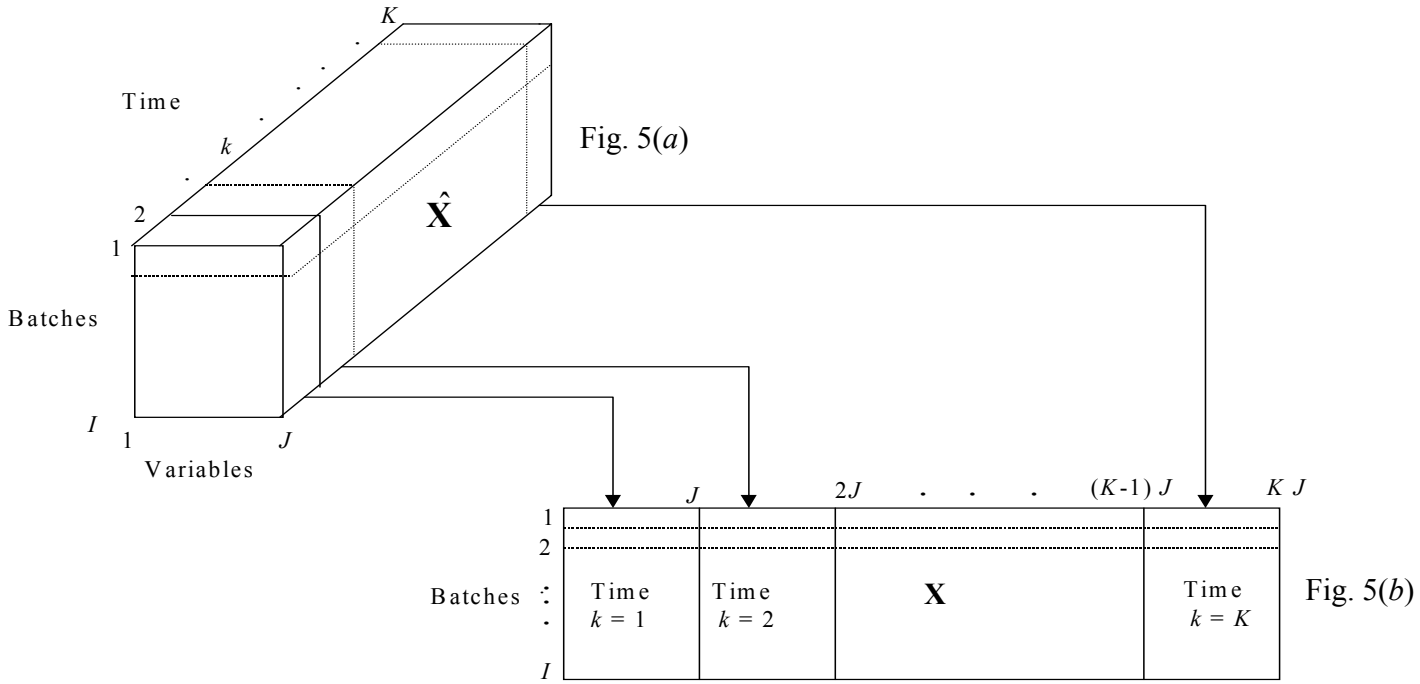


Figure 5.1: (*a*) Schematic of a three-way data array structure ($\hat{\mathbf{X}}$) describing process input variables for a batch process (*b*) Unfolding the 3-dimensional $\hat{\mathbf{X}}$ array to obtain a large 2-dimensional matrix **X**.

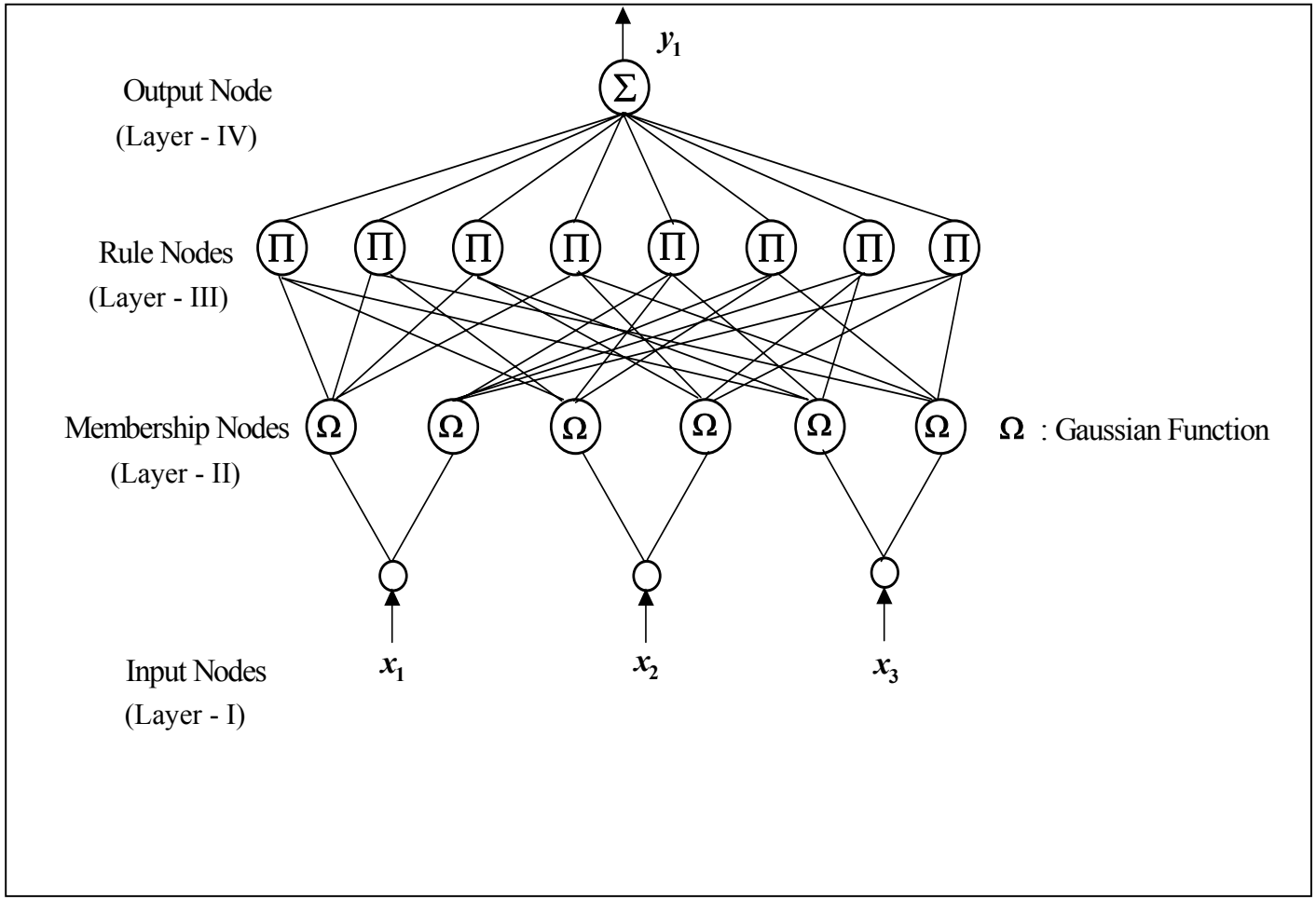


Figure 5.2: Schematic of fuzzy neural network (FNN)

$$net_j^4 = \sum_i^m w_{ij}^4 x_i^4, \qquad y_j^4 = f_j^4\left(net_j^4\right) = net_j^4 \tag{5.7}$$

where the link weight w_{ij}^4 is the output action strength of the *j*th output associated with the *i*th rule.

Note that net_j and f_j are the summed net input (or activation level) and activation function of node *j*, respectively, and the superscript denotes the layer number. The membership functions can be fine tuned and all the consequence strengths of fuzzy rules could be identified by modifying the centres and widths of layers II and the link weights of layer IV.

5.2.2.3 Supervised Gradient Descent Learning:

The adjustment of parameters in the proposed FNN can be divided into two tasks, corresponding to the IF (premise) part and THEN (consequence) part of the fuzzy logical rules. In the premise part, we need to initialize the center and width of the the Gaussian functions. Since the final performance will mainly depend on supervised learning, we have chosen normal fuzzy sets to fully cover the input space. In the consequence part, parameters are output connections. They are initialized with small random values, as in the standard neural network. A supervised learning rule based on gradient descent is used to train the proposed model.

5.3 BATCH PROCESS MONITORING

An FNN may perform poorly if its input space contains irrelevant inputs, which unnecessarily increase the dimensionality of the input space and hence requires a large number of membership functions. This difficulty is overcome by performing PCA on the predictor variable data thereby achieving the dimensionality reduction of the input space. To implement the PCA-FNN strategy for batch process modeling and monitoring, first the three way process data $\hat{\mathbf{X}}$ ($I \times J \times K$) is unfolded – as in MPCA – to obtain a 2-D matrix $\mathbf{X}$. The MPCA formalism for on-line monitoring requires the complete history of the batch process under scrutiny thus making it necessary to fill up the future unmeasured data from the current time to the end of the batch. Various methods for filling up unmeasured data though are

$$net_j^4 = \sum_i^m w_{ij}^4 x_i^4, \qquad y_j^4 = f_j^4\left(net_j^4\right) = net_j^4 \tag{5.7}$$

where the link weight w_{ij}^4 is the output action strength of the *j*th output associated with the *i*th rule.

Note that net_j and f_j are the summed net input (or activation level) and activation function of node j, respectively, and the superscript denotes the layer number. The membership functions can be fine tuned and all the consequence strengths of fuzzy rules could be identified by modifying the centres and widths of layers II and the link weights of layer IV.

5.2.2.3 Supervised Gradient Descent Learning:

The adjustment of parameters in the proposed FNN can be divided into two tasks, corresponding to the IF (premise) part and THEN (consequence) part of the fuzzy logical rules. In the premise part, we need to initialize the center and width of the the Gaussian functions. Since the final performance will mainly depend on supervised learning, we have chosen normal fuzzy sets to fully cover the input space. In the consequence part, parameters are output connections. They are initialized with small random values, as in the standard neural network. A supervised learning rule based on gradient descent is used to train the proposed model.

5.3 BATCH PROCESS MONITORING

An FNN may perform poorly if its input space contains irrelevant inputs, which unnecessarily increase the dimensionality of the input space and hence requires a large number of membership functions. This difficulty is overcome by performing PCA on the predictor variable data thereby achieving the dimensionality reduction of the input space. To implement the PCA-FNN strategy for batch process modeling and monitoring, first the three way process data $\hat{\mathbf{X}}$ ($I \times J \times K$) is unfolded – as in MPCA – to obtain a 2-D matrix $\mathbf{X}$. The MPCA formalism for on-line monitoring requires the complete history of the batch process under scrutiny thus making it necessary to fill up the future unmeasured data from the current time to the end of the batch. Various methods for filling up unmeasured data though are

available (Lennox et. al., 2000), they make certain assumptions, which may not be always valid. Filling up of future observations might cause false detection since they may distort data information without considering the proper dynamic relationship. Hence, MPCA performs poorly when a large number of future measurements are unknown especially in the initial stage of a new batch run. This difficulty can be overcome by developing a PCA-based submodel by considering data measured only at the current time instant. Specifically, for a process being monitored over a total of K time instances, a separate submodel is built at each time instance $k = 1, 2, \ldots, K$ by considering input measurements only at that instant. Since a sub-model corresponding to the kth instant requires data measured only at that instant, values corresponding to $k+1$ to K time intervals are not required thus altogether avoiding the necessity of filling up future measurements. This approach reduces the numerical effort involved in computing the future unmeasured data and also in extracting principal components since the PCA is now performed on a matrix of smaller size.

The task of monitoring considered here is defined as: *Given historic process input-output data in the form of a 3-way array* $\hat{\mathbf{X}}$*, analyze the process behavior at equally spaced time instances, k = 1, 2, ..., K, with a view to detect any abnormality in the current (test) process inputs, and, more importantly, to predict any abnormal behavior of the process outputs at the end of the batch run.*

The PCA-FNN strategy for batch process modeling and monitoring is illustrated by conducting a case study involving biosynthesis of protein. The dynamic process input-output data were obtained by simulating the phenomenological models of the process (Balsa-Canto et. al., 2001; Lim et. al., 1977). It may however be noted that the hybrid strategy does not require the phenomenological process model for its implementation. In real practice, historic input-output data from the batch processes should be used for process modeling and monitoring. Using typical variations in the initial values of the process operating variables, input-output data for a number of batches were generated. Batches with their input variables within the acceptable operating range (normal behavior) and also outside the range (abnormal behavior) were simulated to generate the training set. Also, a number of input-output vectors to be used as the test set, were generated assuming the input process conditions both in the acceptable and unacceptable ranges.

The fed-batch fermenter system for protein synthesis is characterized in terms of three predictor (causal) variables namely culture cell density (x_1), substrate concentration (x_2) and hold-up volume (x_3), and two output (response) variables, viz. secreted protein concentration (y_1) and total protein concentration (y_2). The input-output training set for a total of 52 batches comprising values of the three predictor variables (x_1, x_2, x_3) measured at one hour time intervals and the values of two output variables (y_1, y_2) measured at the end (15 hr) of the batch was generated using the phenomenological model available in the open literature (Balso-Canto et al., 2001):

$$\frac{dy_1}{dt} = g_1(y_2 - y_1) - \frac{u}{x_3} y_1 \tag{5.8}$$

$$\frac{dy_2}{dt} = g_2 x_1 - \frac{u}{x_3}(y_2) \tag{5.9}$$

$$\frac{dx_1}{dt} = g_3 x_1 - \frac{u}{x_3} x_1 \tag{5.10}$$

$$\frac{dx_2}{dt} = -7.3 g_3 x_1 + \frac{u}{x_3}(20 - x_2) \tag{5.11}$$

$$\frac{dx_3}{dt} = u \tag{5.12}$$

$$g_1 = \frac{4.75 g_3}{0.12 + g_3}, \tag{5.13}$$

$$g_2 = \frac{x_2}{0.1 + x_2} \exp(-5.0 x_2), \tag{5.14}$$

$$g_3 = \frac{21.87 x_2}{(x_2 + 0.4)(x_2 + 62.5)} \tag{5.15}$$

where t, x_1, x_2 and x_3 refer to time (min), culture cell density (g l^{-1}), substrate concentration (g l^{-1}), and hold up volume (l), respectively; y_1 and y_2 are the concentrations (g l^{-1}) of the secreted protein and the total protein, respectively, and u refers to the nutrient (glucose) feedrate (l h^{-1}). The fed-batch culture is fed at an exponentially increasing nutrient feed rate, $u = u_o e^{0.219t}$ (Lim et al., 1977), with the constraint, $0 \le u \le 2.5$, where u_0 is a constant (= 0.0926 l h^{-1}).

This training set consisted data from 50 batches with their initial conditions in the normal ranges ($0.95 < x_1(0) < 1.5$, $4.5 < x_2(0) < 5.5$, and $0.95 < x_3(0) < 1.5$) and two abnormal batches (17 and 38) with their initial conditions outside the stated normal ranges. Additionally, a test set consisting of two normal batches (53 and 54) and two abnormal batches (55 and 56), was generated in a similar manner to the training set to test the efficacy of the hybrid PCA-FNN models.

The 3-D input array ($\hat{\mathbf{X}}$) of dimensions ($52 \times 3 \times 15$) was unfolded, preprocessed (mean and variance scaling) and subjected to the PCA. Specifically, PCA was conducted separately on each three hourly sampled input variable data and the loading vectors obtained thereby were used to compute the scores of the four test batches. A plot of PC-1 versus PC-2 for all the batches showed that the test batches 55 and 56 lie consistently outside the 99% control limit and thus have been correctly identified as abnormal batches (see Figure 5.3*a* and 5.3*b*); the PCA has also identified the two abnormal training batches 17 and 38 correctly. Figure 5.3 shows the plots of the scores of PC-1 versus those of PC-2 for a representative case using seventh hour predictor variable data and at the end of the batch. In this case, PC-1 and PC-2 explained 99% variance in the original data. Next, the scores of PC-1 and PC-2 corresponding to 52 batches were used to build multi input – single output (MISO) FNN models. Here separate models were constructed for predicting y_1 and y_2 at the end of 52 batches. A total of twelve FNN models were developed for predicting y_1 and y_2 from the scores of three hourly readings of the predictor variables. Number of inputs to FNN is two (PC–1 and PC–2), number of membership function for each variable is considered to be 3 (Ω of Fig. 5.2), number of rule nodes in layer 3 (Π of Fig. 5.2) is taken as 10 and output is one (either y_1 or y_2 as the case may be). Next, the models were used for predicting y_1 and y_2 values corresponding to the four test batches.

It is seen in Fig. 5.4 that training batches 17 and 38 with abnormal initial values of the operating variables, have resulted in unsuccessful batches. The results of FNN-based predictions of y_1 and y_2 can be used to assess whether a test batch will be successful or not. Consider, for instance, predictions made using the seventh hour data and portrayed in Fig. 5.5. As can be seen, the predictions correctly show that the test batches 53 and 54 (55 and 56) will be successful (unsuccessful) as the corresponding y_1 and y_2 values lie well within (outside) the 95% control limit. Results similar to that shown in Fig. 5.5 have been obtained using predictor

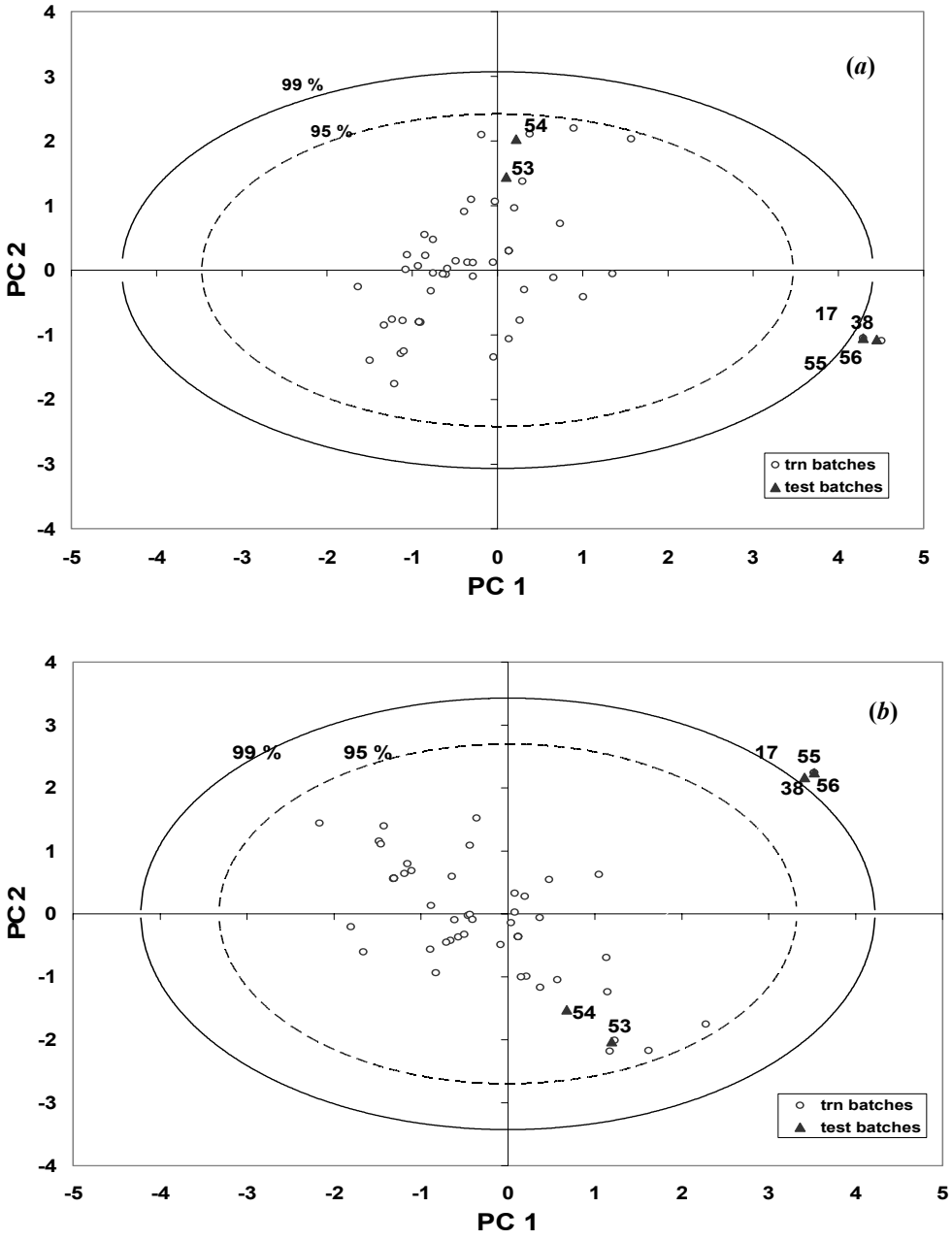
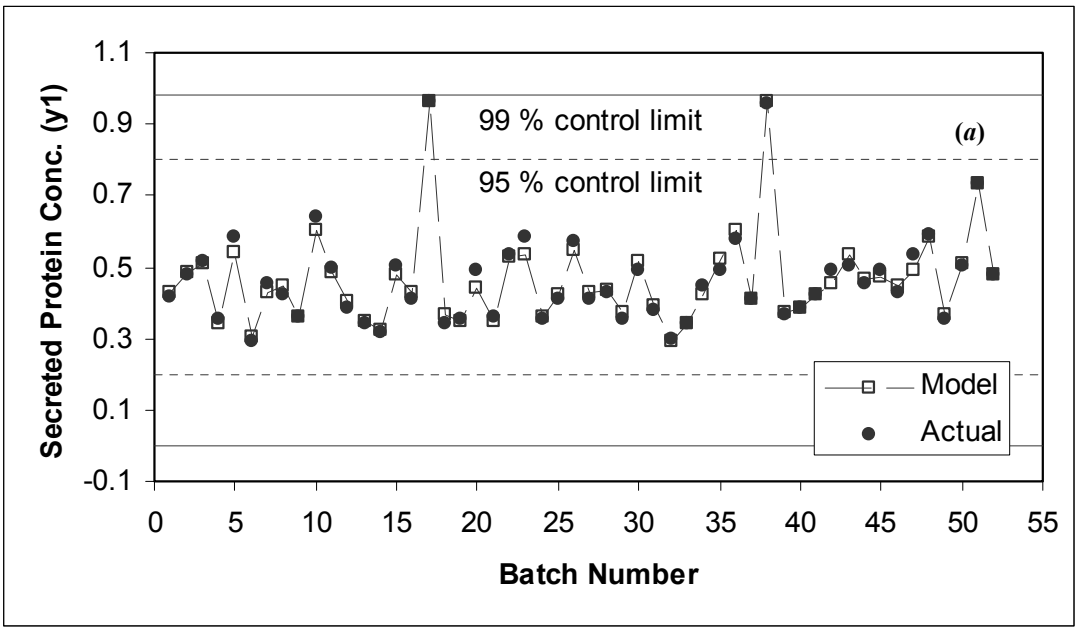


Figure 5.3: (*a*) Plots of PC-1 vs. PC-2 scores pertaining to seventh hour predictor variable data indicating training batches 17, 38, and test batches 55 and 56 lying outside 99 % confidence interval.

(*b*) Plots of PC-1 vs. PC-2 scores at end of the batch indicating batch numbers 17, 38, 55 and 56 leading to product outside the acceptable domain.



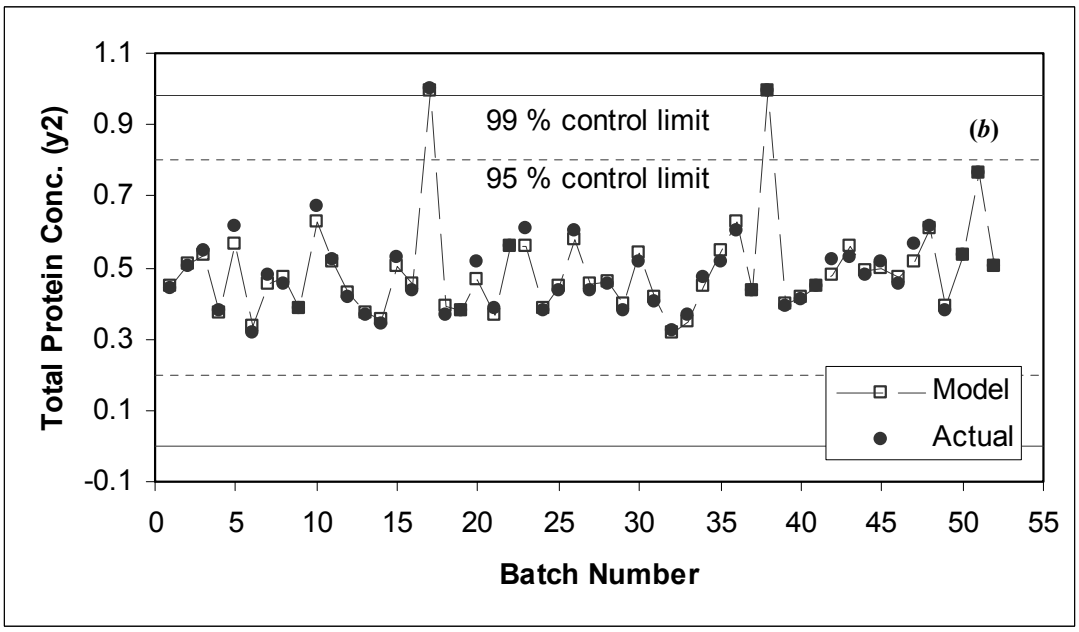


Figure 5.4: (*a*) Comparison of actual vs. PCA-FNN model based prediction of secreted protein concentration with seventh hour data

(*b*) Comparison of actual vs. PCA-FNN model based prediction of total protein concentration using seventh hour data.

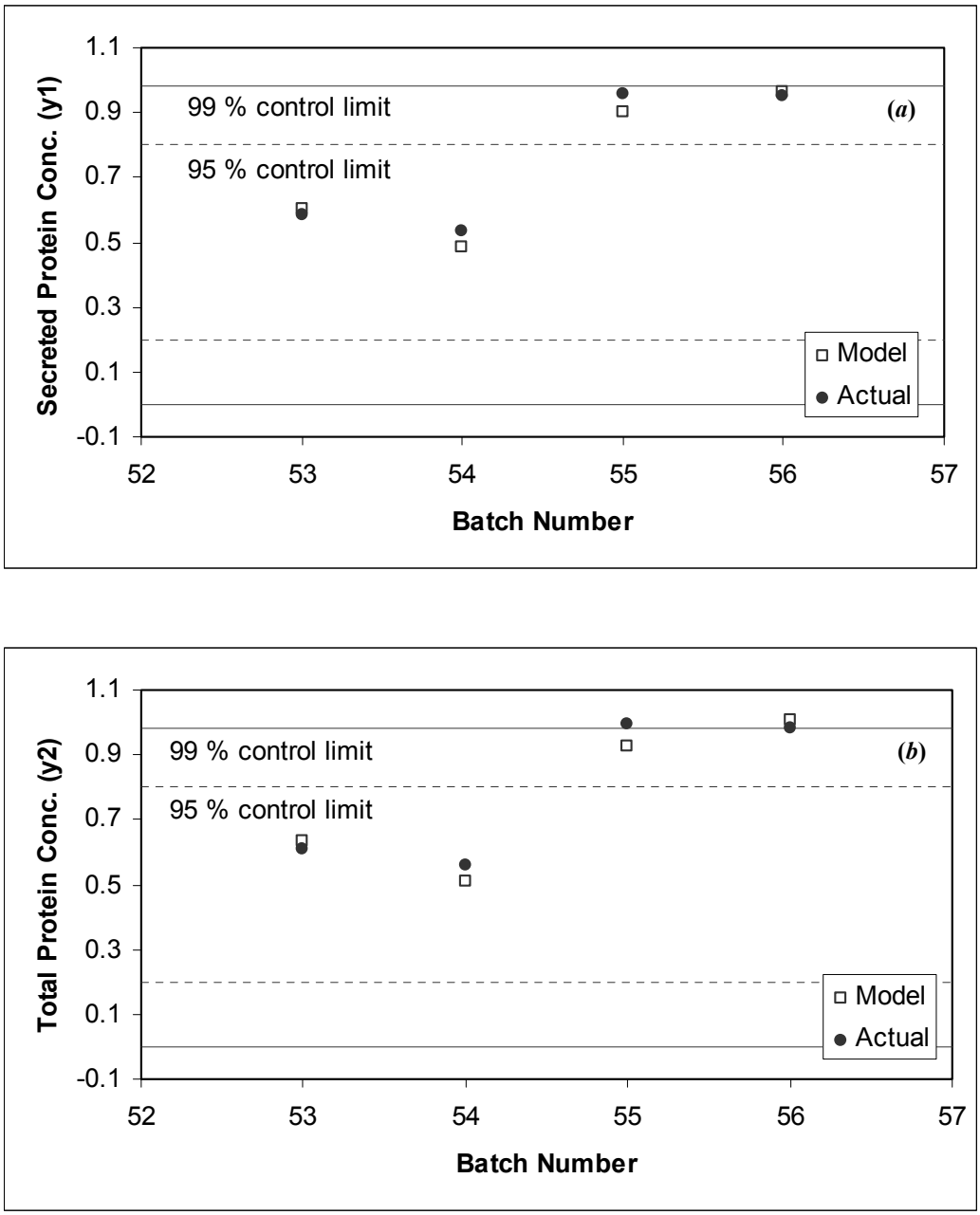


Figure 5.5: (*a*) PCA-FNN model based prediction of the secreted protein conc. for four test batches based on seventh hour data.

(*b*) PCA-FNN model based prediction of the total protein concentration for four test batches based on seventh hour data.

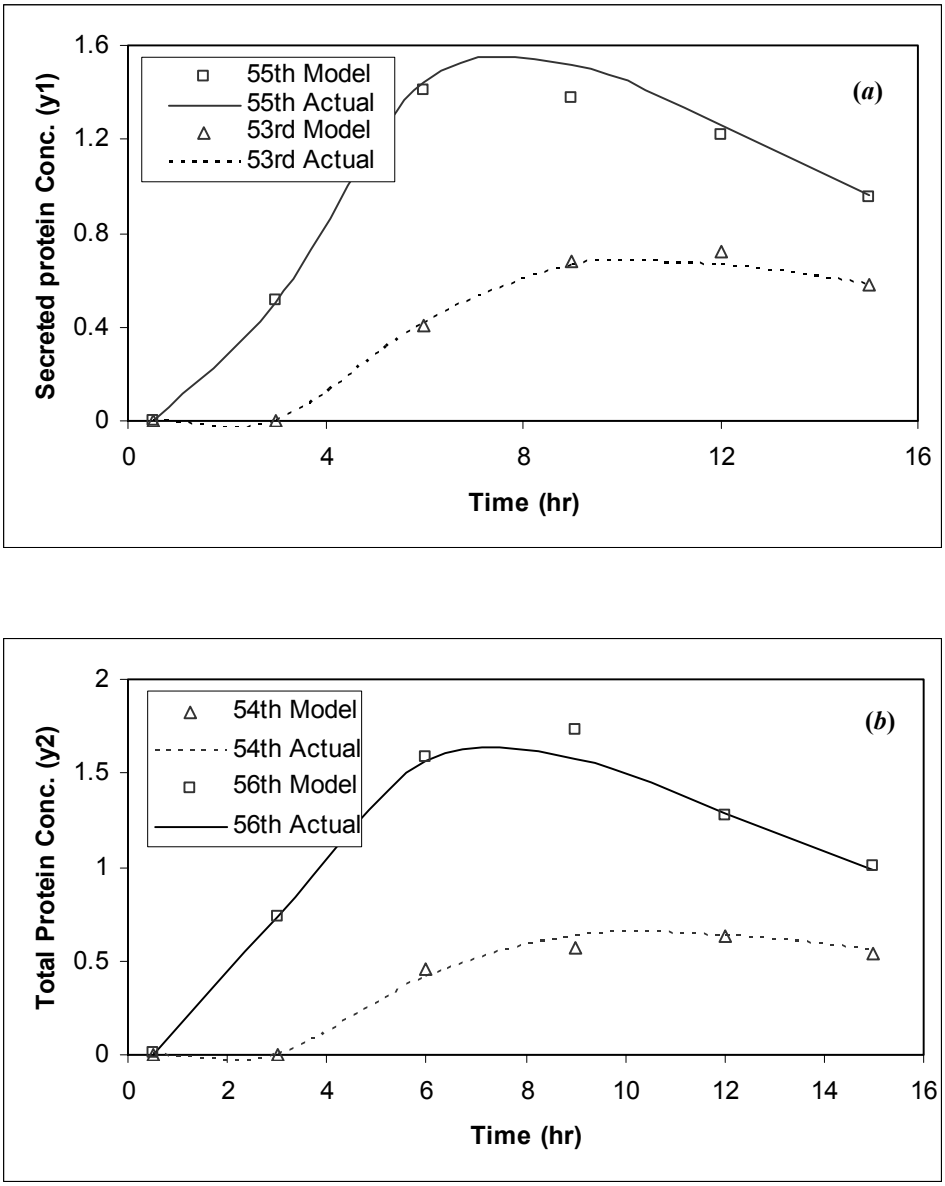


Figure 5.6: (*a*) Predictions of secreted protein concentration for one normal (53^{rd}) and one abnormal (55^{th}) test batches indicating soft-sensor ability of the PCA-FNN strategy.

(*b*) Predictions of total protein concentration for one normal (54^{th}) and one abnormal (56^{th}) test batches confirming soft-sensor ability of the PCA-FNN strategy.

variable data pertaining to all the 15 one hour time intervals thereby proving the ability of FNNs in accurately predicting whether a given batch will be successful or not.

Ability of PCA-FNN to act as a soft-sensor and thereby predicting the hourly monitored values of y_1 and y_2 was also tested. The results of predictions using three hourly sampled inputs and outputs for normal test batches (53 and 54) (shown by dotted line) as well as abnormal batches (55 and 56) (shown as continuous line) are portrayed in Figures 5.6*a* and 5.6*b*. It is clearly seen that the PCA-FNN strategy is capable of excellently representing the trends of output variables y_1 and y_2. In the present case study, the relationship between the scores and the output variables is highly nonlinear. This strong nonlinearity arises from the exponential dependence of the output variables, y_1 and y_2, on the substrate concentration, x_2 (refer model equations Eqn. 5.8 to 5.15). The PCA-FNN, being capable of efficiently representing nonlinear input-output relationships, thus is an attractive alternative to the linear PLS approach.

5.4 CONCLUSION

In this study, a hybrid strategy integrating principal component analysis and fuzzy neural network has been presented for modeling and monitoring of a batch process. The proposed PCA-FNN strategy uses PCA for reducing the dimensionality of the process input space and the first few principal component scores that explain a large amount of variance in the input data are used to develop a fuzzy neural network model correlating process inputs and outputs. Principal advantages of the hybrid model are: (i) it can approximate nonlinear input-output relationships efficiently, and (ii) the model can be constructed exclusively from the historic process input-output data. The effectiveness of the hybrid PCA-FNN strategy for modeling and monitoring of batch processes has been successfully demonstrated on the bio-process of protein synthesis. The results obtained thereby, show that the hybrid methodology is an attractive formalism for modeling and monitoring of nonlinearly behaving batch processes.

5.5 NOMENCLATURE

E : the residual matrix

$\mathbf{p}_r$: loading vectors

$\mathbf{p}_r^{'}$: transpose of the loading vector $\mathbf{p}_r$

$\mathbf{t}_r$: score vectors

$|T(x_i)|$: denotes the number of fuzzy partitions of the input linguistic variable, x_i

u : the nutrient (glucose) feedrate (l/h)

w_k : represents the kth consequence link weight

X : two-dimensional matrix obtained after unfolding the three-way array, $\hat{\mathbf{X}}$

x_1 : culture cell density (g/l)

x_2 : substrate concentration (g/l)

x_3 : hold-up volume (l)

y_1 : secreted protein concentration (g/l)

y_2 : total protein concentration (g/l)

Greek Symbols:

$\prod_i |T(x_i)|$: rule nodes in the proposed FNN structure

$\mu_{A_i^k}(x_i)$: membership functions

5.6 REFERENCES

1. Balsa-Canto, E., Banga, J. R., Alonso A. A. and Vassiliadis, V. S., "Dynamic Optimization of Chemical and Biochemical Processes Using Restricted Second Order Information", ***Comp. & Chem. Engg.***, **25**, 539 – 546 (2001).
2. Chen, Y-C. and Teng, C-C. "A Model Reference Control Structure Using Fuzzy Neural Network", ***Fuzzy Sets and Syst.***, **73**, 291 – 312 (1995).
3. Ghasem, N.M. "Design of a Fuzzy Logic Controller for Regulating the Temperature in Industrial Polyethylene Fluidized Bed Reactor", ***Chem. Engg. Res. & Des.***, **84**, 97 – 106 (2006).
4. Hanai, T., Ohki, T., Honda, H. and Kobayashi, T. "Analysis of Initial Conditions for Polymerization Reaction Using Fuzzy Neural Network and Genetic Algorithm", ***Comp. & Chem. Engg.***, **27**, 1011 – 1019 (2003).
5. Huang, Y. C. and Wang, X. Z. "Application of Fuzzy Causal Networks to Waste Water Treatment Plants", ***Chem. Engg. Sci.***, **54**, 2731 – 2738 (1999).
6. Kourti, T. and MacGregor, J. F. "Multivariate SPC Methods for Process and Product Monitoring", ***Jour. of Quality Tech.*****, 28**, 409 – 428 (1996).
7. Lim, H.C.. Chen, B. J. and Creagan, C. C., "An Analysis of Extended and Exponentially-Fed Batch Cultures", ***Biotech. & Bioengg.*****, 14**, 425 – 433 (1977).
8. Ni, S. H., Lu, P. C. and Juang, C. H. "A Fuzzy Neural Network Approach to Evolution of Slope Failure Potential", ***Microcomput. Civil Engg.***, **11**, 59 – 66 (1996).
9. Nomikos, P. and MacGregor, J. F. "Multiway Partial Least Squares in Monitoring BatchProcesses", In Proceedings of First International Chemometrics InterNet Conference, InCINC., available online at: http://www.emsl.pnl.gov: 2080/docs/incinc/n_way_anal/PNdoc.html (1994a).
10. Nomikos, P. and MacGregor, J. F. "Monitoring Batch Processes Using Multi-way Principal Component Analysis", ***AIChE Jour.***, **40**, 1361 – 1375 (1994b).
11. Nomikos, P. and MacGregor, J. F. "Multivariate SPC Charts for Monitoring Batch Processes", ***Technometrics,*** **37**, pp. 41 – 59 (1995).
12. Pai, T.Y. , Wan, T.J., Hsu, S.T., Chang, T.C., Tsai, Y.P., Lin, C.Y., Su, H.C. and Yu, L.F. "Using Fuzzy Inference System to Improve Neural Network for Predicting

Hospital Wastewater Treatment Plant Effluent", ***Comp. & Chem. Engg.***, **33**, 1272 – 1278 (2009).

13. Rahman, M. S. and Wnag, J., "Fuzzy Neural Network Model for Liquefaction Prediction", ***Soil Dyn. & Earthquak. Engg.***, **22**, 685 – 694 (2002).
14. Rao, A., Jayaraman, V.K., Kulkarni, B.D., Japanwala, S. and Shevgaonkar, P. "Improved Controller Performance with Simple Fuzzy Rules", ***Hydrocarb. Proces.***, 97-100, May (1999).
15. Ronen, M., Shabtai, Y. and Guterman, H. "Rapid Process Modeling — Model Building Methodology Combining Unsupervised Fuzzy-Clustering and Supervised Neural Networks", ***Comp. & Chem. Engg.***, **22**, S1005 - S1008 (1998).
16. Sugeno, M. "An Introductory Survey of Fuzzy Control", ***Inform. and Sci.***, **36**, pp. 59 – 83 (1985).
17. Wold, S., Esbensen, K. and Geladi. P. "Principal Component Analysis", ***Chemom. & Intell. Lab. Syst.***, **2**, pp. 37 – 52 (1987).
18. Whitnell, G.P., Davidson, V.J., Brown, R.B. and Hayward, G.L. "Fuzzy Predictor for Fermentation Time in a Commercial Brewery", ***Comp. & Chem. Engg.***, **17**, 1025 – 1029 (1993).

CHAPTER 6

ARTIFICIAL NEURAL NETWORK ASSISTED GENETIC ALGORITHM BASED FORMALISM FOR OPTIMIZATION OF BATCH POLYMERIZATION REACTOR

Presented at ***Indian Chemical Engineering Congress*** **(*CHEMCON 2004*)** in Mumbai

Abstract

Polymerization reactions are often carried out in a batch or semi-batch mode. Optimization of these processes becomes necessary for cost minimization as well as environmental and safety consideration. A novel methodology integrating artificial neural networks (ANNs) and genetic algorithms (GAs) is utilized for optimizing a free radical batch solution polymerization of methyl methacrylate (MMA) to produce poly-MMA. This study is concerned with the searching the optimal feed temperature and the initial initiator concentration for the stated batch polymerization system so as to reduce the residual monomer concentration to the desired level and simultaneously produce the polymer with the desired number average molecular weight.

6.1 INTRODUCTION

Significant economic advantages can be obtained by operating polymerization reactors in an optimal way such that a consistent product quality is maintained and polymerization time minimized. The molecular weight distribution and its averages namely number average and weight average molecular weights determine many of the physical and processing properties of the polymer formed (Odian, 1970). Thus, a stringent control of molecular weight distribution is essential. The residual monomer concentration needs to be low since presence of monomer in the final polymer product degrades its property. Also, some amount of unreacted monomer in the product is a net loss of reactant which could have been used for the on other way be used for production of more polymer. Hence, it is essential to reduce the residual monomer concentration to low levels. At times it is also important to control the molecular weight of the polymer formed. Clearly these objectives fall under the domain of multi-objective process optimization.

The optimization of batch polymer reactor using temperature variations or semi-continuous operations involving addition of different combinations of initiator, monomer and chain transfer agents or the addition of inhibitor is well studied (Ponnuswamy et. al., 1987; Schmidt and Ray, 1981). This study deals with the development of optimal temperature and initial initiator concentration policies to reduce the residual monomer concentration to the desired level to produce a polymer having desired number average molecular weight. The effect of the relative cost of the initiator on the optimal policy is also analyzed.

Owing to the gel effect and inherent nonlinear behavior makes the development of phenomenological models for polymerization reactions difficult and time consuming. To overcome this difficulty, this study employs artificial neural networks (ANN) for the development of a sufficiently generalized process model of the polymer system which can be developed solely from the process input-output data without considering the details of the process intricacies (such as kinetics and heat and mass transfer). The problem considered here requires multi-objective optimization since it involves multiple objectives namely, reduction of residual monomer concentration to the desired level, while simultaneously producing the polymer with desired average number molecular weight. Accordingly we have used a suitable meta-heuristic nonlinear search and optimization method called genetic algorithms (GA), which is more efficient than conventional gradient based optimization methodologies. The

objective of this study is to present an efficient hybrid ANN-GA methodology for optimization of polymer reactor.

6.2 PROCESS MODELING USING ANN METHODOLOGY

A phenomenological process model is constructed from the detailed knowledge of mass, momentum and energy balances along with other chemical engineering principles. For the development of phenomenological polymerization models understanding of the underlying detailed physico-chemical phenomena is cumbersome, costly and time consuming. Empirical (linear and nonlinear) models such as Volterra models, polynomial auto-regressive moving average models (ARMAX) etc., can be obtained solely from the process input-output data however difficulty associated with this approach is that the model structure needs to be specified *a-priori*. The specification of a nonlinear model structure is a cumbersome task as it requires several trial and errors.

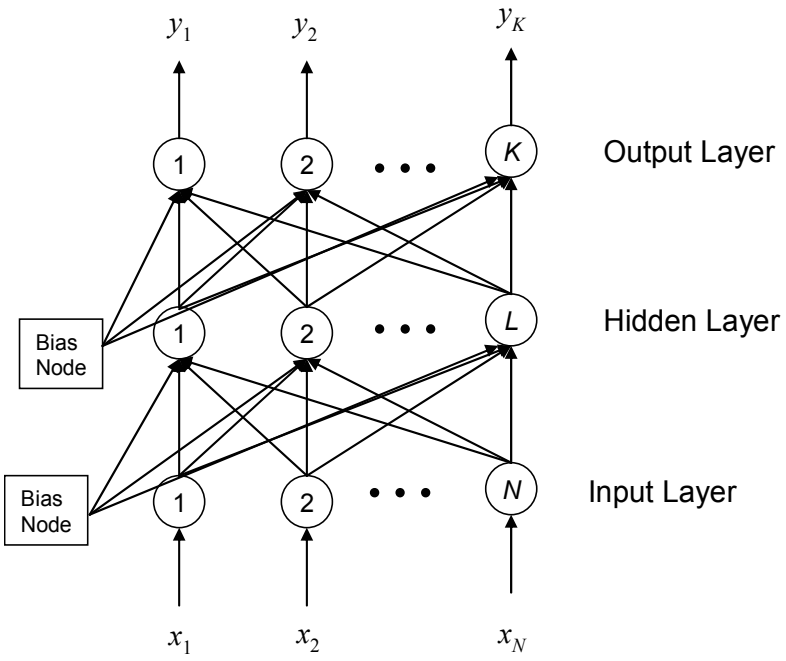


Figure 6.1: Schematic of a multi-layered perceptron

In the last two decades Artificial Neural Networks (ANNs) are used as an efficient formalism in the modeling of steady-state and dynamic processes (Tambe et. al., 1996; Nandi

et al., 2001). ANNs are based on the concept that a highly interconnected system of simple processing elements (nodes) can approximate the complex nonlinear relationships between the independent and dependent variables with a reasonable degree of accuracy. They can be derived solely from the historic process input-output data. Multi-input multi-output (MIMO) systems can be modeled simultaneously. The developed model is sufficiently generalized. In this study, multilayered feedforward networks (MFFN) has been used for polymerization process modeling.

An MFFN is a nonlinear mapping device between an input set (I) and outout set (O). It represents a function f that maps I into O, $y = f(x)$ where $y \in O$ and $x \in I$. Most generally used MFFN paradigm is *multi-layered perceptron* (MLP) comprising three layers of processing inputs (neurons / nodes). The layers are described as *input*, *hidden* and *output* layers comprising of N, L and K number of processing elements respectively (refer Fig, 6.1). Each node of the input layer are connected to all the nodes of hidden layer; similarly, each node of the hidden layer is connected to all the nodes of the output layer MLP can have two hidden layers as well.. The MLP architecture also houses a bias node in its input and hidden layers to provide additional adjustable parameters (weights) for the model fitting. The number of nodes in the MLP network's input layer (N) is equal to the number of inputs to the process whereas number of output nodes (K) equals to the number of process outputs. The number of nodes in hidden layer (L) is adjustable parameter whose magnitude is determined by desired approximation and generalization capabilities of the network model

For training the MLP network commonly *Error-back-propagation* (EBP) algorithm is employed. The algorithm utilizes a gradient descent technique named *generalized-delta-rule* (GDR). Here, an input pattern is applied to the input nodes and processed through connection weights and then transformed utilizing nonlinear activation function to produce outputs. The computed output is compared with the desired one and the prediction error is evaluated in terms of *Root mean squared error* defined as:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{k=1}^{P}\sum_{i=1}^{N_{out}}\left(t_k^i - o_k^i\right)^2}{P \times N_{out}}} \tag{6.1}$$

where, P refers to the number of training patterns, N_{out} represents the number of output nodes, t_k^l denotes the actual (desired) output and o_k^l denotes model predicted value for lth output of pattern k. To prevent *overfitting*, the entire input-output data set is divided into two sets; one for the training of the network and the other set called "test-set" is utilized for testing how well the network model generalizes. Two RMSE values are compared after each training iteration. These are E_{trn} and E_{tst} representing training and test set RMSEs respectively. The E_{trn} indicates data-fitting ability of MLP and the E_{tst} measures the generalization ability of the network. The optimal weight set is considered to be that set for which smallest E_{tst} is reached after systematic variation of hidden nodes and other EBP parameters. A comprehensive discussion on obtaining a generalization capable MLP model is provided in chapter 2.

6.3 GA-BASED OPTIMIZATION OF THE DEVELOPED MODEL

Once a generalization capable MLP network model is trained, its input space is optimized in a manner such that the optimized inputs lead to the desired (maximized / minimized) output. Commonly used gradient based optimization techniques require the objective function to be smooth, continuous and differentiable. For the ANN-based process model which serves as objective functions, simultaneous fulfillment of those characteristics can not be guaranteed. Genetic algorithms (GA) is a general purpose stochastic nonlinear search and optimization formalism which can be used effectively in these situations (Goldberg, 1989; Deb, 1995). GA differs from traditional optimization methods in several ways. First, the GA works with a population of strings searching many peaks (candidate solutions) simultaneously. Information between the peaks is exchanged to search new peaks and hence possibility to converging at local optima is minimized. Secondly, GA works with coding (e.g., binary coding) of the candidate solutions instead of the solution themselves. Thirdly, GA does not need the derivative or other auxiliary information about the system, but requires measurement of the objective function only.

GA begins its search from a randomly initialized population of probable (candidate) solutions. The solutions usually coded in the form of binary strings are tested to measure their fitness in fulfilling the desired objective. The important steps involved in the process are: (i) reproduction (based on fitness function), (ii) generation of offspring population utilizing the process of cross-over of genetic material between pairs of fitter parent chromosomes and (iii)

mutation (bit flipping) of the elements of the offspring strings. Through reproduction, strings with high fitness receive multiple copies in the next generation while strings with low fitness receive fewer copies or even none at all. The crossover operation produces two offspring (new solutions) by recombination of information of two parent solutions. In simplest form, crossover with a single crossing site refers to taking string, splitting it into two parts at a randomly generated crossover point and recombining it with another string which has also been split at the same crossover point. This procedure serves to promote change in the best strings which could give them even higher fitness. Mutation is the random alteration of a bit in the string, assisting in keeping diversity in the population. These steps generate a new population of candidate solutions, which as compared to the previous population.

6.4 IMPLEMENTATION OF ANN-GA STRATEGY TO POLYMERIZATION

It is assumed that the monomer undergoes solution polymerization in a batch reactor, resulting in a high solids content and a relatively low monomer concentration (M_0). At this point, a certain amount of initiator is added to bring the initiator concentration to I_0. It is desired to reduce the monomer concentration M_0 (around 5 vol %) to a final concentration of M_f (around 0.5 vol %) in the minimum possible time and at the same time produce a final desired number average molecular weight M_{nw} by a proper choice of temperature T and initial initiator concentration I_0. Here we have neglected the influence of diffusion control on the termination or other rate processes as those effects are small when solution is sufficiently dilute or when the polymer is of low molecular weight.

The optimization problem can be formulated as follows: *for a given initial monomer concentration M_0, determine an optimal isothermal temperature (T) and initial initiator concentration (I_0) that will produce a polymer with the desired conversion (x_f) and number average molecular weight (M_{nw}) in the minimum time*. Minimizing the cost of initiator is also an important objective to be considered along with the minimization of time. Thus, the objective function is written as:

$$J = \delta_1 t_f + \delta_2 I_0 \tag{6.2}$$

where δ_1 and δ_2 are the cost factors associated with time and initial initiator concentration, respectively. For simplicity it has been assumed that the reaction is carried in a reactor having

volume of one liter. Accordingly, J will be the cost associated with the one liter reactor. Equation (6.2) can be modified as follows to take into account the relative cost of the batch time and initiator concentration.

$$J_1 = \left(\frac{J}{\delta_1} \right) = t_f + \delta I_0 \tag{6.3}$$

where, δ refers to the relative cost of the initiator with respect to the cost associated with the time of reaction and is given in the units of s/mol.

6.5 RESULTS AND DISCUSSION

For convenience, the data required for the MLP model development is generated from the kinetic model for batch polymerization (O'Driscoll and Ponnuswamy, 1990). In the actual practice, the model can be developed from the data generated by performing actual experiments. The MMA polymerization reaction is monitored based on the three inputs namely: (i) initial initiator concentration (I_0), (ii) temperature of reactor (T) in K and (iii) time required to form the desired polymer (t_f) in min. The two monitored outputs are, (i) desired conversion (x_f) and (ii) desired number average molecular weight (M_{nw}).

The MLP network has three input nodes equal to the number of process inputs and two output nodes representing the outputs. Fifty data points were simulated for training of the network model. These data points are sub divided into two sets: 40 points consisting of the training set and 10 points comprising the test set. The MLP model is developed based on the points in the training set while the test set is utilized to check the generalization capability of the model. The effect of variation of number of nodes in hidden layers on the root-mean-squared-error (RMSE) pertaining to the training and test sets studied was studied rigorously. The results of this study are shown in Fig. 6.2 where it is seen that the MLP architecture with three hidden nodes in the hidden layer resulted in the minimum RMSE value for the test set (E_{tst} = 0.0288); the corresponding training set RMSE (E_{trn}) was 0.0326. The small and comparable E_{trn} and E_{tst} magnitudes suggests good generalization capability of the developed MLP model. The values of the learning coefficient and momentum coefficient determined heuristically were 0.6 and 0.02, respectively.

The three-dimensional space of the MLP model was optimized utilizing the GA formalism (Nandi et. al., 2002). The values of the GA specific parameters used in the optimization simulations were chromosome length l_{chr} = 60, population size N_{pop} = 30, crossover probability p_{cr} = 0.95, mutation probability p_{mut} = 0.01, and maximum number of generations $N_m^{\max}$ = 250.

Our objective in this work was to get the desired polymer in minimum time and using minimum amount of the initiator. The desired outputs considered are 90 % conversion of monomer and a pre-specified number average molecular weight. In this case study we have optimized the process conditions corresponding to the number average molecular weights of 1000, 2000, 3000, 4000 and 5000 respectively. The optimized values that were obtained utilizing the ANN-GA hybrid modeling-optimization strategy were compared with those reported earlier by O'Driscoll and Ponnuswamy (1990) (refer to Table 6.1).

Most of the optimization procedure consider the polymerization process with the sole objective of minimization of the batch time. However, cost of the initiator is also an important factor in the batch process economics. Thus the optimization consisted of minimization of the cost of initiator as well. The objective function considered is:

$$f' = \sigma_1 t_f + \sigma_2 I_0 \tag{6.4}$$

where, σ_1 and σ_2 are the cost factors associated with the batch time and initial initiator concentration, respectively. Since the said optimization is carried in a one liter reactor, the f' is modified to take into account the relative cost of batch time and initiator concentration as follows:

$$f' = \left(\frac{f'}{\sigma_1} \right) = t_f + \left(\frac{\sigma_2}{\sigma_1} \right) I_0 = t_f + \sigma I_0 \tag{6.5}$$

where, σ represents the relative cost of the initiator with respect to the cost associated with the time of reaction and expressed in units of s/mol. The modified function is used during optimization.

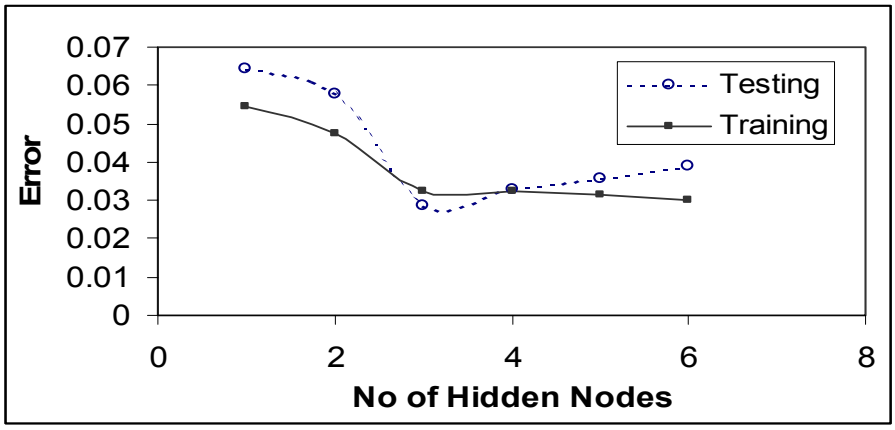


Figure 6.2: Effect of hidden nodes on neural network performance

A close inspection of Table 6.1 reveals that the ANN-GA formalism is able to search for better optimal conditions as compared to that of O'Driscoll and Ponnuswamy (1990). The number average molecular weight (M_{nw}) values for both the methods are almost identical (refer to columns 1 and 2). The initial initiator concentration (I_0) requirement is consistently lower for all the ANN-GA simulations (refer to columns 3 and 4). Temperature requirement of the polymerization as predicted by both the methodologies are very close (see column 5 and 6). The significant improvement on process operating conditions can be visualized from time requirement (t_f) of the process for ANN-GA vis-à-vis optimization methodologies followed by O'Driscoll and Ponnuswamy (1990) (refer to column 7 and 8). As batch time for individual operation is becoming lower, more number of batches can be conducted for given time (in a month, say). Initial initiator concentration (I_0) and batch time (t_f) are affecting the polymerization process in their own ways and the proposed ANN-GA formalism is able to search the best combination to produce the desired quality polymer (having standard M_{nw}).

6.6 CONCLUSIONS

The major advantage of the ANN-GA hybrid modeling-optimization strategy proposed in this chapter for optimizing batch polymerization is that the process optimization can be done solely from historic process data without the detailed knowledge of process

phenomenology. A sufficiently generalized ANN model is first developed for the isothermal batch polymerization of MMA. It is subsequently utilized for optimizing the operating condition variables using GAs. The objective function minimized in the optimization represents the process operating cost, which in turn depends on the batch time and initiator cost. The task of optimization was to reduce the residual monomer concentration to the desired level while simultaneously producing the polymer with desired average number molecular weight. The results given by the ANN-GA scheme have been compared with those available in the literature and they reveal significant improvement over the published results. Specifically, the ANN-GA hybrid formalism could bring about upto 4.8 % reduction in the process operating cost (refer to Table 6.1).

Table 6.1: Comparison of the obtained optimized results with that of the previously published data

M_{nw}		I_0 (mol/lit)		T (^{0}C)		t_f (min)	
O P Method	**ANN-GA**	**O P Method**	**ANN-GA**	**O P Method**	**ANN-GA**	**O P Method**	**ANN-GA**
1000	1020	0.2325	0.2250	75.2	75.47	152	138.56
2000	2022	0.0620	0.0582	75.0	74.92	304	302
3000	3016	0.0325	0.0315	74.4	74.5	458	437.7
4000	4018	0.0250	0.0195	74.3	73.8	614	586
5000	5023	0.0146	0.0097	73.8	73.7	773	736

O P Method: Results from O'Driscoll and Ponnuswamy, *J. Appl. Polym. Sci.*, **39**, (1990), p. 1299

ANN-GA : Results obtained using ANN based modeling followed by GA-based optimization (i.e., proposed method)

6.7 NOMENCLATURE

E_{trn} : root-mean-squared-error for training data set

E_{tst} : root-mean-squared-error for test data set

f : nonlinear function which maps input into the output

I_0 : initial initiator concentration

J : objective function for optimization

K : number of nodes in output layer

L : number of nodes in the hidden layer

M_0 : initial monomer concentration

M_{nw} : number average molecular weight

N : number of nodes in input layer

N_{out} : represents number of output nodes

o_k^l : denotes model predicted value for lth output of pattern k

P : refers to the number of training patterns

t_f : time required to form the desired polymer in min

t_k^l : denotes actual value for lth output of pattern k

T : optimal isothermal temperature

x_f : desired conversion

Greek Symbols:

δ_1 and δ_2 : the cost factors associated with time and initial initiator concentration

δ : relative cost of the initiator with respect to the cost associated with the time of reaction and is given in the units of s/mol

6.8 REFERENCES

1. Deb, K. ***Optimization for Engineering Design: Algorithms and Examples***, Prentice-Hall, New Delhi (1995).
2. Goldberg, D. E. ***Genetic Algorithms in search, Optimization, and Machine Learning***, Addison-Wesley, Reading, MA (1989).
3. Nandi, S., Ghosh, S, Tambe, S. S. and Kulkarni, B. D. "Artificial Neural-Network-Assisted Stochastic Process Optimization Strategies," ***AIChE Jour.*****, 47**, 126 - 141, (2001).
4. Nandi, S., Mukherjee, P., Tambe, S. S., Kumar R., and Kulkarni, B. D. "Reaction modeling and optimization using neural networks and genetic algorithms: case study involving TS-1 catalyzed hydroxylation of benzene", ***Ind. Engg. Chem. Res***., **41**, 2159 - 2169, (2002).
5. Odian, G. ***Principles of Polymerization***, McGraw-Hill, New York (1970).
6. O'Driscoll, K. F. and Ponnuswamy, S. R., "Optimization of a Batch Polymerization Reactor at the Final Stage of Conversion: II Molecular Weight Constraint", ***J. Appl. Polym. Sci***., **39**, 1299 – 1308, (1990).
7. Ponnuswamy, S. R., Shah, S. L. and Kiparissides, C. "Computer Optimal Control of Batch Polymerization Reactors", ***Ind. Eng. Chem. Res.***, **26**, 2229 – 2236 (1987).
8. Schmidt, A. D. and Ray, W. H. "The Dynamic Behavior of Continuous Polymerization Reactors I: Isothermal Solution Polymerization in a CSTR", ***Chem. Engg. Sci.***, **36**, 1401 - 1412 (1981).
9. Tambe, S. S., Kulkarni, B. D. and Deshpande, P. B. ***Elements of Artificial Neural Networks with selected applications in Chemical Engineering, and Chemical & Biological Sciences***, Simulations & Advanced Controls, Louisville, KY (1996).

CHAPTER 7

CONTROLLING NONLINEAR DYNAMICS OF A NON-ISOTHERMAL FLUIDIZED BED REACTOR

Presented in ***Indian Chemical Engineering Congress*** **(*CHEMCON 2002*)** held at Hyderabad

Abstract

A non-isothermal fluidized bed catalytic reactor in certain parameter regions can exhibit complex nonlinear dynamical behavior such as multiplicity of steady states, sustained oscillations and even chaos. For the consecutive exothermic reaction $A \rightarrow B \rightarrow C$ *with B as the desired component, yield maximization can be secured if the reactor is operated at the middle unstable steady state (USS). A novel gain scheduling based nonlinear control methodology is demonstrated in this chapter to control the fluidized – bed reactor exactly at the USS for achieving yield maximization. The artificial intelligence based control strategy proposed here has only one tunable parameter, optimal value of which is obtained with help of genetic algorithm.*

7.1 INTRODUCTION

Fluidized bed reactors (FBR) are widely used in petroleum refining and petrochemicals industries. Some of the well-known fluidized catalytic processes are cracking, production of phtahlic anhydride from o-xylene, styrene from ethylbenzene, and butadiene from butane. The dynamics of the above-stated processes are nonlinear and, depending upon operating parameter region these systems can exhibit a variety of complex behavior such as multiplicity of steady-states, instability, sustained and quasi-periodic oscillations, and even chaos. In general, it is easy to control a system at a stable steady-state. However, from the economical and operational view-point it may be advantageous to operate the plant at an unstable steady state since it may lead to conversion/yield maximization, lower operating temperatures, etc. In petroleum refining industry, fluidized catalytic cracking (FCC) units are generally operated at an unstable steady state that provides maximum gasoline yield. Production of phthalic anhydride by partial oxidation of o-xylene and oxidative dehydrogenation of ethylbenzene to produce styrene (Ajbar and Elnashaie 1996) are similar examples from the petrochemicals field. In general, it is not easy to stabilize a process at an unstable steady state (USS) since the system, under the influence of even a minor perturbation, moves away from that steady state.

In recent years, a large number of strategies have emerged to control unstable, oscillatory or chaotic dynamics of nonlinear systems. The strategies aim at either of the following two control tasks : (i) to stabilize the unstable periodic orbits (UPO) of a chaotic system by application of small perturbations in the neighborhood of the desired UPO (Ott et. al., 1990 and its several modifications e.g., Shinbrot et. al., 1993) and, (ii) to stabilize the unstable dynamical trajectories exactly at an USS. Since an USS repels trajectories in its neighborhood, deriving a robust control strategy to stabilize the unstable dynamics is in general a difficult task. Novel studies by Singer et al. (1991) have successfully demonstrated the stabilization of unstable trajectories at the corresponding USS. In the present study, a gain scheduling based control strategy, which does not require a phenomenological model for implementing the control action is presented for regulating the unstable dynamics of a fluidized bed reactor exactly at an unstable steady state. The method can also be used to

implement servo control and for controlling oscillatory or quasi-periodic or chaotic dynamical behavior.

7.2 CONTROL STRATEGY

In feed-back control, the control law utilizes the set-point error (difference between the set point and actual value of the control variable) as a measure to force the control variable towards the set point. In case of nonlinear systems following expression is found to be very useful for control :

$$\frac{du_t}{dt} = \varepsilon\left(x^{set} - x\right) \tag{7.1}$$

where, u_t denotes the controller output, ε is controller gain (tuning parameter) and t is time. The set point of the control variable x is denoted by x^{set} and thus (x^{set} – x) defines the set point error (e). Equation 1 with constant ε has been used as an adaptive controller by Sinha et al. (1990). In their control law, the process model parameters appear explicitly. Thus the strategy needs phenomenological model for adaptive control. In many instances phenomenological model is not available. Accordingly, based on an heuristic reasoning Bandyopadhyay et. al. (1997) suggested a nonlinear control strategy, which does not require a phenomenological model to implement control. Their strategy begins by defining :

$$u_t = \varepsilon\, e \tag{7.2}$$

Upon differentiating w.r.t. time Eq. 7.2 becomes,

$$\frac{du_t}{dt} = \frac{d\left(\varepsilon\, e\right)}{dt} \tag{7.3}$$

which can also be written as

$$\frac{du_t}{dt} = e\frac{d\varepsilon}{dt} + \varepsilon(t)\left(\frac{de}{dt}\right) \tag{7.4}$$

where, ε is a time dependent proportionality constant. For separable systems,

$$\varepsilon = \varepsilon_0\, f(t) \tag{7.5}$$

and hence, Eq. 7.4 becomes

$$\frac{du_t}{dt} = \varepsilon_0\, f'(t)e + \varepsilon_0\, f(t)\left(\frac{de}{dt}\right) \tag{7.6}$$

where, $f'(t)$ is the time derivative of $f(t)$. As can be seen, controller law given by Eq. 7.6 contains terms proportional to the set point error, e, as well as its time derivative. The gain terms $\varepsilon_0 f'(t)$ and $\varepsilon_0 f(t)$ are continuously adapted in time and thus Eq. 7.6 is non-autonomous in character. For simplicity it is possible to assume that $f(t) = t$ and hence $f'(t) = 1$. This way first gain term for the proportional control action, $\varepsilon_0 f'(t)$ becomes independent of time but the second gain term for derivative action, $\varepsilon_0 f(t)$, retains its time dependence. Accordingly, Eq. 7.6 can be viewed as a variant of proportional-derivative controller with the time adaptive gain terms. It is also seen that ε_0 is the only tunable parameter. Determination and tuning of this parameter (ε_0) is of immense importance as it has a great influence on the stability and performance of the proposed control system.

Genetic algorithm (GA) is an AI-based search and optimization technique and has been widely used for many control engineering applications (see, Kim and Park, 2005; Sadashivrao and Chidambaram, 2006; Shin et. al., 2007). In recent times the GA is being efficiently used for PID type controller tuning purposes (Gundogdu, 2005; Kumar et. al., 2008). In this chapter we propose to apply GA to obtain the optimal setting of the only tunable parameter (ε_0) of our proposed controller for stabilizing a fluidized bed reactor at its unstable steady state.

7.3 CONTROL OF FLUIDIZED BED REACTOR

The dynamics a non-isothermal fluidized bed reactor (FBR) with catalytic exothermic reaction $A \rightarrow B \rightarrow C$ has been investigated by Ajbar and Elnashaie (1996). Figure 7.1 provides a schematic of the non-isothermal FBR. The dynamics of the FBR is described in terms of concentrations of the reactant A and desired product B along with the temperature of the reactor dense phase, T. For effecting control, a parameter region wherein system possess following three steady states has been chosen : (i) a low-temperature (quenched) steady state ($C_A = 1.0$, $C_B = 0.0$, $T = 0.9$), (ii) a high temperature (burn-out) steady state ($C_A = 0.0$, $C_B = 0.0$, $T = 1.55$) and (iii) a saddle type unstable steady state (USS) ($C_A = 0.2437$, $C_B = 0.6336$, $T = 0.92955$), which is the desired operating point of the reactor as it provides maximum yield of the desired component B. Here, it may be noted that only for convenience the steady states

have been obtained using the model equations. In actual control application, the phenomenological model is not required.

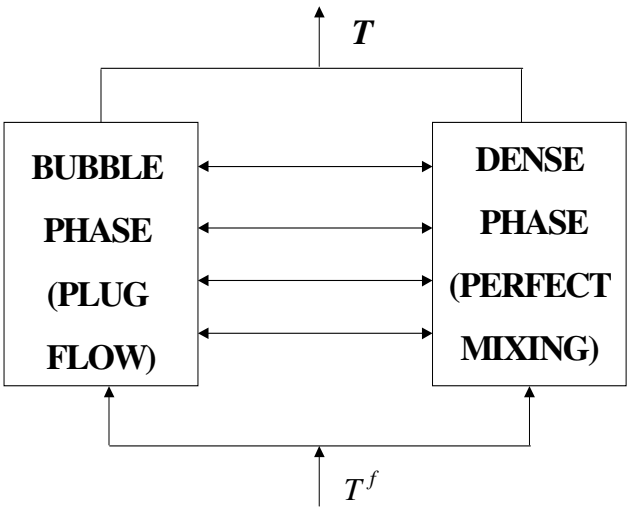


Figure 7.1: Schematic of a two phase fluidized bed reactor

The phenomenological equations for the dense phase material and energy balances for the FBR can be represented in the dimensionless form as (Ajbar and Elnashaie, 1996):

$$\frac{1}{Le_A}\frac{dC_A}{dt} = \psi\left(C_A^0 - C_A\right) - \alpha_1 \exp\left(-\frac{\gamma_1}{T}\right) C_A \tag{7.7}$$

$$\frac{1}{Le_B}\frac{dC_B}{dt} = \psi\left(C_B^0 - C_B\right) + \alpha_1 \exp\left(-\frac{\gamma_1}{T}\right) C_A - \alpha_2 \exp\left(-\frac{\gamma_2}{T}\right) C_B \tag{7.8}$$

$$\frac{dT}{dt} = \psi\left(T^f - T\right) + \alpha_1 \beta_1 \exp\left(-\frac{\gamma_1}{T}\right) C_A + \alpha_2 \beta_2 \exp\left(-\frac{\gamma_2}{T}\right) C_B + \psi\, u_t + d \tag{7.9}$$

where, C_A and C_B are the dimensionless concentrations of components A (reactant) and B (product), T is the dimensionless reactor dense temperature, T^f refers to the feed temperature,

u_t describes the manipulated variable (deviation from reference value of T^f), t is the dimensionless time, and d denotes load disturbances. Other symbols appearing in Eqs. 7-9 are defined in the nomenclature section.

The bubble phase mass and energy balances are assumed to be at pseudo-steady state owing to negligible mass and heat capacities. The objective of the controller is to stabilize the system at its USS (C_A = 0.2437, C_B = 0.6336, and T = 0.92955), which leads to maximization of the desired component B. For control purposes, we have considered the dense phase reactor temperature T as the manipulated variable. The reactor behavior was simulated using Eqs. 7.7–7.9 in conjunction with Eq. 7.6 where the set point error is calculated by:

$$e = \left(T^{set} - T\right) \tag{7.10}$$

7.4 RESULTS AND DISCUSSION

7.4.1 Controlling the Reactor at the USS

Here the control objective was to bring the process operating with initial conditions [C_A^0 = 1.0, C_B^0 = 0.0 and T^0 = 0.55342 and d = 0 (no load disturbance)] and stabilize it to the USS (C_A = 0.2437, C_B = 0.6336, and $T = T^{set}$) to achieve maximum yield of B. To begin ε_0 was chosen arbitrarily equal to 8.43 and the results of the controller action are shown in Figs. 7.2 *a*, *b*, *c* respectively. It can be clearly seen that the controller unable to stabilize the trajectory at the USS instead the trajectories reach to the burn – out steady state (C_A = 0.0, C_B = 0.0, T = 1.55). Thus, ε_0 was optimized using GA formalism.

7.4.1.1 GA Based Tuning of the Controller

Genetic algorithm is an efficient stochastic nonlinear optimization technique that performs an exploration of the search space that evolves in analogy to the evolution in nature (Goldberg, 1985; Deb, 1998). Details of the GA and its implementation strategies are provided in chapters 1 and 2 respectively. Here, only the implementation scheme of GA to obtain the optimal value of ε_0 (only tunable parameter of the proposed controller), is briefly discussed.

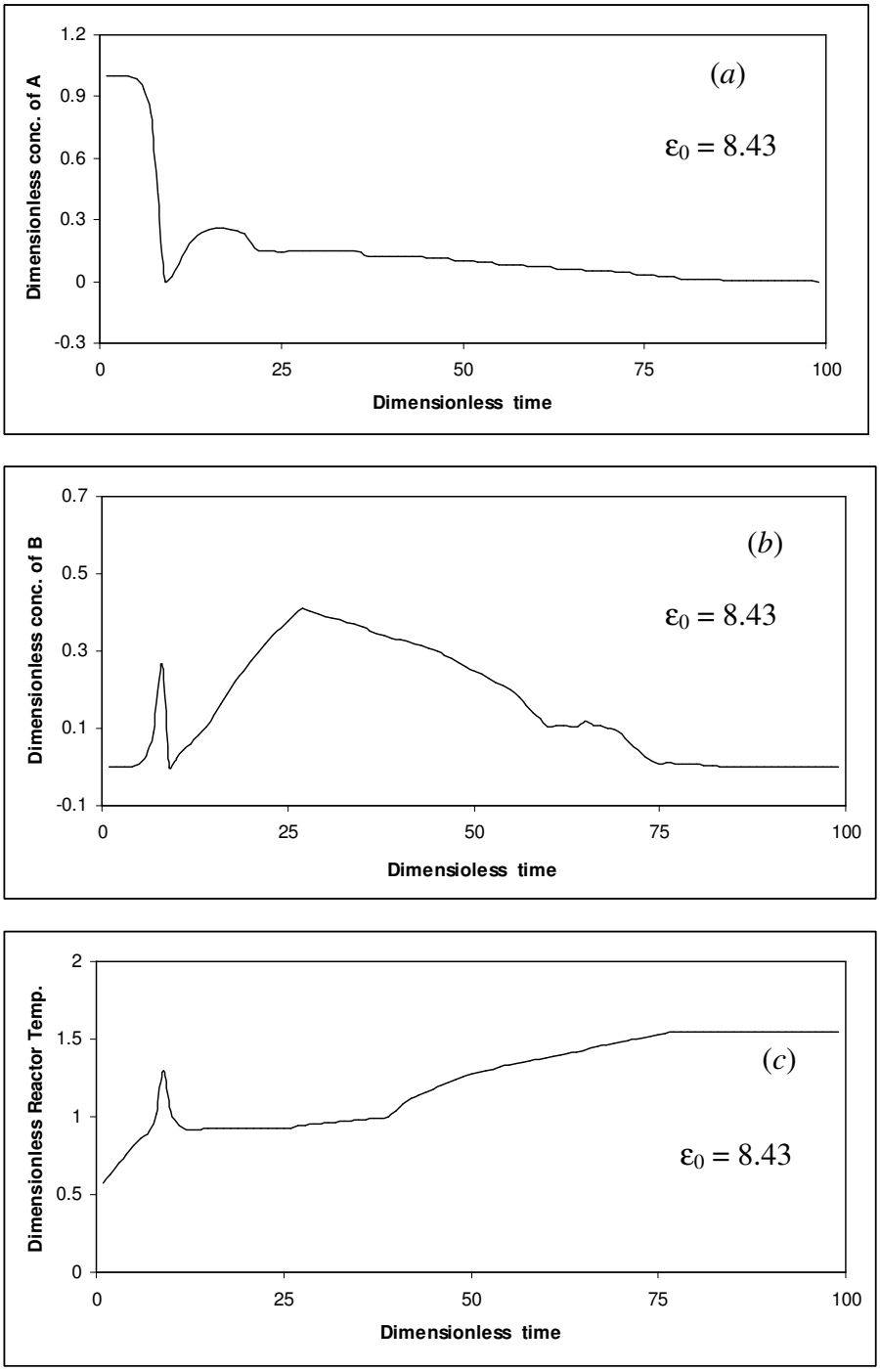


Figure 7.2: Performance of controller with arbitrary ε_0 value (= 8.43) to stabilize fluidized bed reactor at Unstable Steady State (USS), the process reaches burn-out steady state (C_A = 0.0, C_B = 0.0 and T = 1.55) as evident from the above panels *a*, *b*, *c* respectively.

The tuning parameter (ε_0) is considered to be bounded between 0 and 10. Initially a binary coded population (N_{pop}) of 40 candidate solutions is generated randomly within the lower and upper bounds. Chromosome length (l_{chr}) of all the binary coded decision variables is taken to be six. A single point crossover with probability (p_{cross}) of 0.75 and uniform mutation with low probability (p_{mut}) of 0.08 wer considered in this study. The maximum number of generations ($N_{\max}^{gen}$) to be evolved by the GA was fixed at 200.

The objective function to be minimized by the GA defined in terms of a variant of the set point error criterion. Several such criteria are available. Here, controller's performance is evaluated based on integral time absolute error (ITAE). Specifically The error criterion is defined as the integral of time multiplied by absolute error as given below:

$$I_{ITAE} = \int_0^{t_{ss}} t \times \left| e(t) \right| dt \tag{7.11}$$

The limits of Eq. 7.11 are from time $t = 0$ to $t = t_{ss}$ where, t_{ss} is the time when error value (e) becomes negligibly small ($\leq 10^{-6}$).

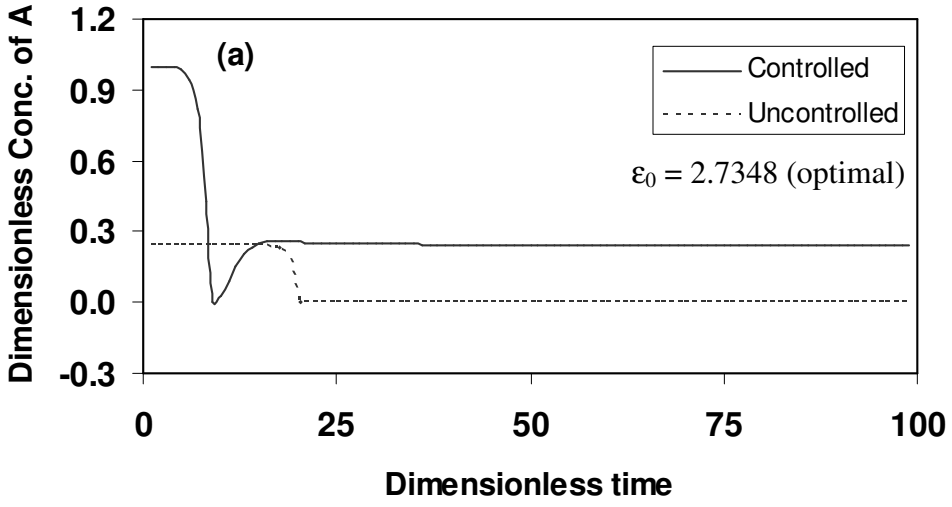
Termination of the optimization can be done two ways: (i) when maximum number of iterations are over, or (ii) on attainment of satisfactory fitness score. Fitness score (ξ) is reciprocal of the magnitude of the objective function.

$$\xi = \frac{1}{I_{ITAE}} \tag{7.12}$$

The gain scheduling based controller is to be designed based upon the best value of its tunable parameter ε_0 over all 200 GA iterations (generations) of 40 populations. The GA took 10 – 12 runs to arrive at an overall optimal solution while each time starting with a different set of randomly generated binary coded population. The best solution predicted by the GA – based search is $\varepsilon_0 = 2.7348$. This value of tuning parameter (ε_0) is believed to be the optimal one and is considered for all the subsequent simulations.

Figures 7.3*a*, *b*, *c* respectively portray the time evolution of the concentrations of *A*, *B*, and reactor dense phase temperature *T* in presence and absence ($u_t = 0$) of the control action. The corresponding u_t versus time profile is shown in Fig. 7.3*d*. We can clearly see in Figs. 7.3*a*-7.3*c* that the controller has imparted excellent control action in taking the process to the USS and stabilizing it there without allowing any offset. It is seen from Figs. 7.3*a*-7.3*c* that in

the absence of control the system diverges from the USS and reaches the nearest stable steady state (C_A = 0, C_B = 0, T = 1.55) (burn-out steady state) leading to insignificant production of B. Note that here all the control simulations were performed with the optimized ε_0 (= 2.7348).



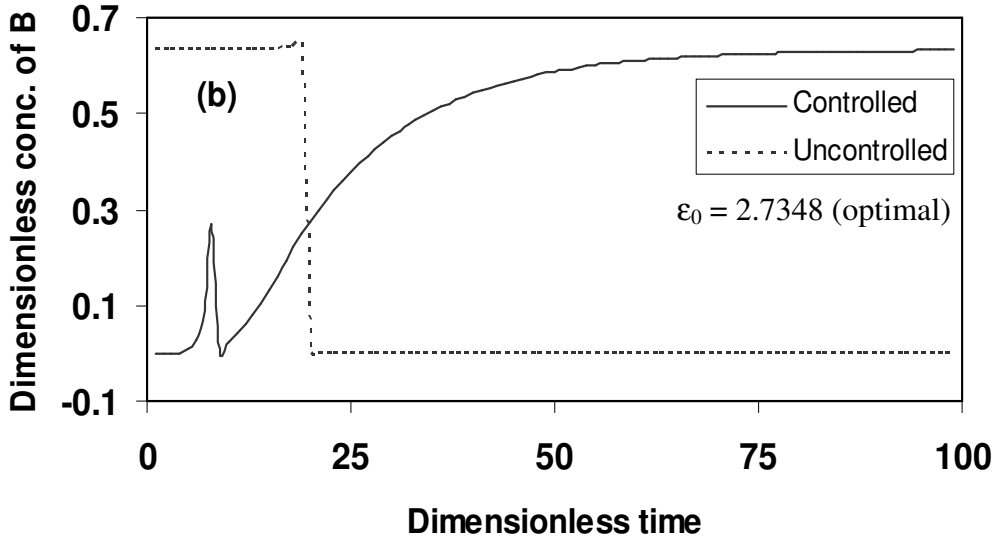
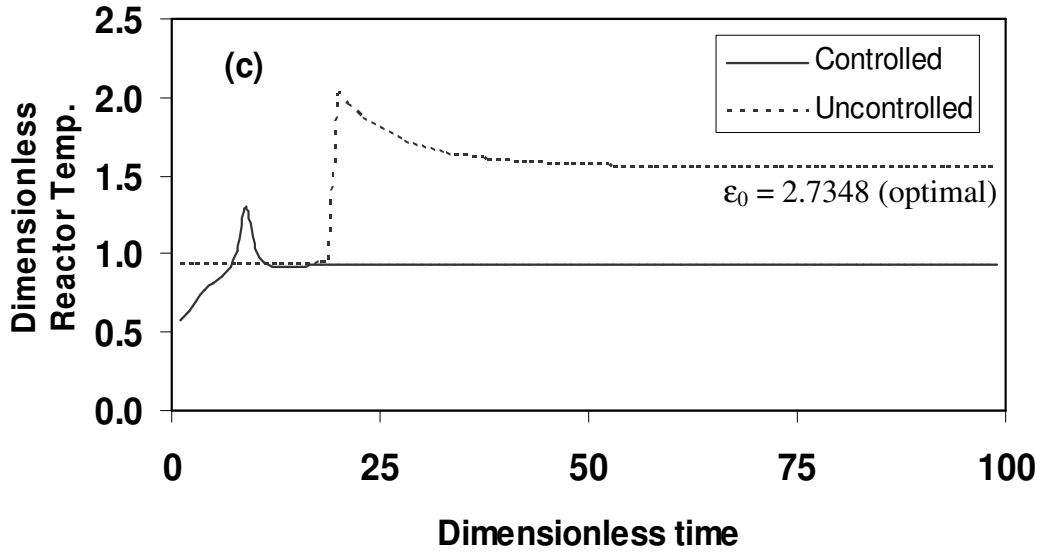


Figure 7.3 (panels *a*, *b*, *c*, *d*): Performance of Controller to Stabilize Fluidized Bed Reactor at Unstable Steady State (USS) without any load disturbances (i.e., d = 0) (continued in next page)



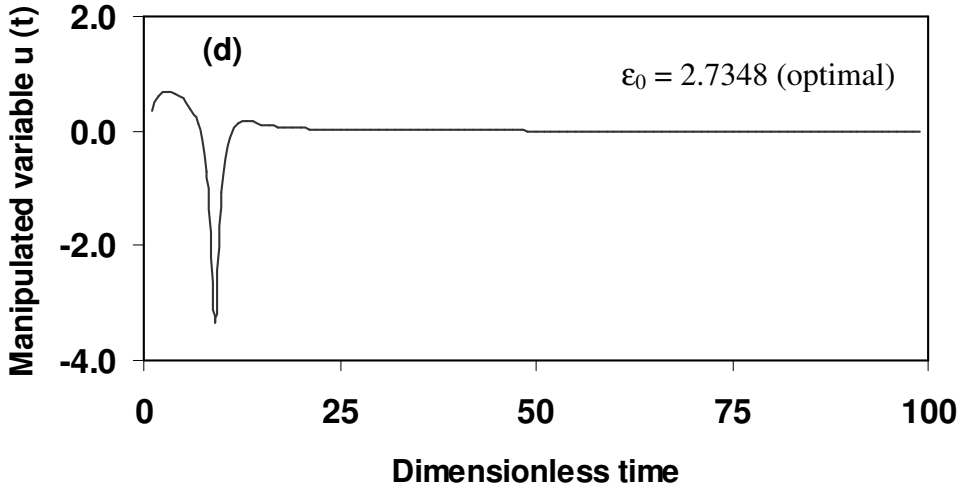
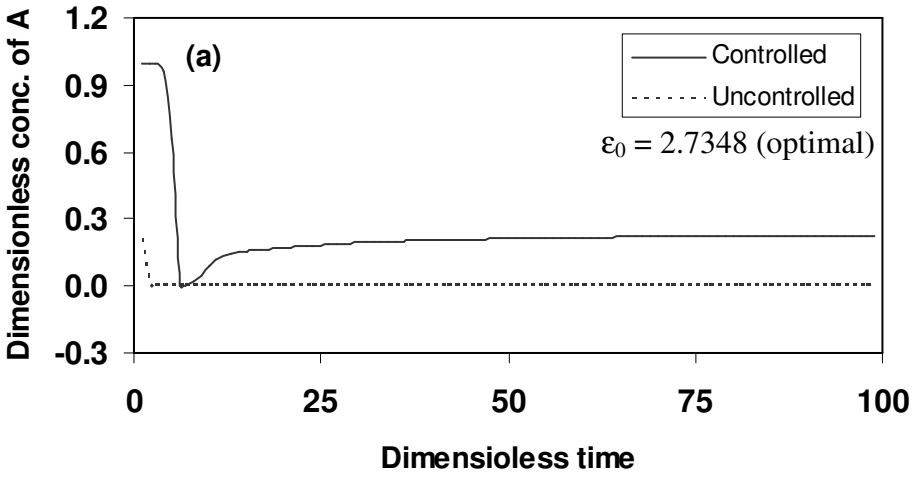


Figure 7.3 (panels *a*, *b*, *c*, *d*): Performance of Controller to Stabilize Fluidized Bed Reactor at Unstable Steady State (USS) without any load disturbances (i.e., $d = 0$)

7.4.2 Controlling the Process at USS in Presence of Deterministic Load Disturbance

The controller performance in stabilizing the process at the USS in presence of deterministic load disturbance was evaluated by applying a constant deterministic load of magnitude $d = 0.05$ to the dense phase reactor temperature T. The controlled (continuous line) and uncontrolled (dotted line) time profiles of C_A, C_B and T in presence of deterministic load disturbance are portrayed in Figs. 7.4*a*-7.4*c*, respectively. The corresponding u_t vs. t plot is shown in Fig. 7.4*d*. It is seen in this case that the controller could impart excellent control action and thereby stabilize the process at the USS even in the presence of deterministic load disturbance.



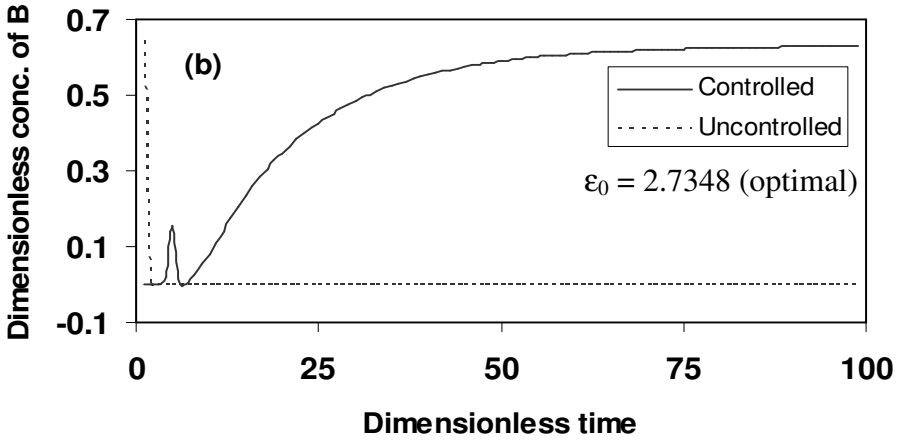
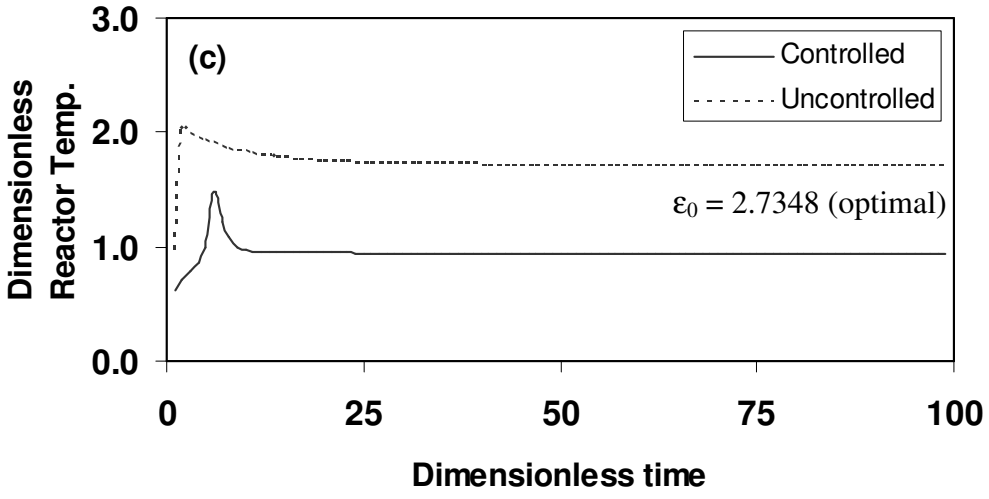


Figure 7.4 (panels *a*, *b*, *c*, *d*): Stabilization of Fluidized Bed Reactor at Unstable Steady State (USS) in presence of deterministic load disturbance $d = 0.05$

(continued in next page)



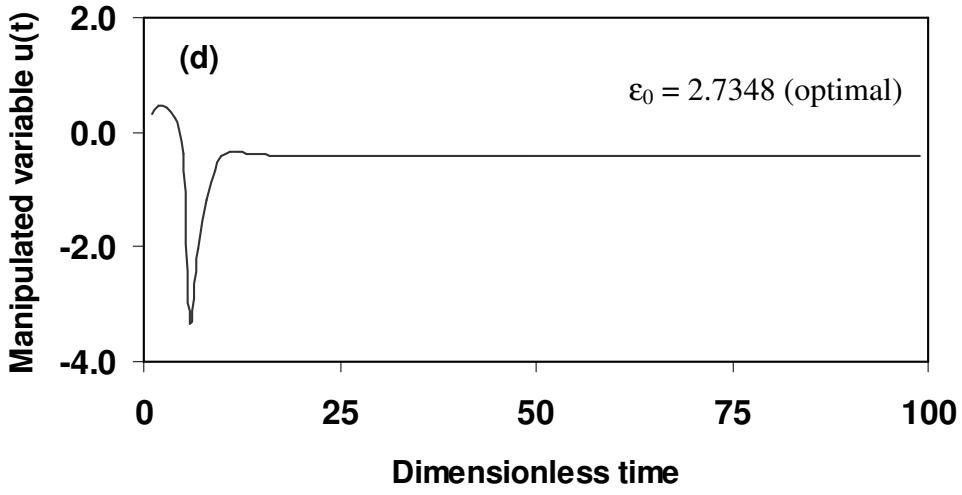
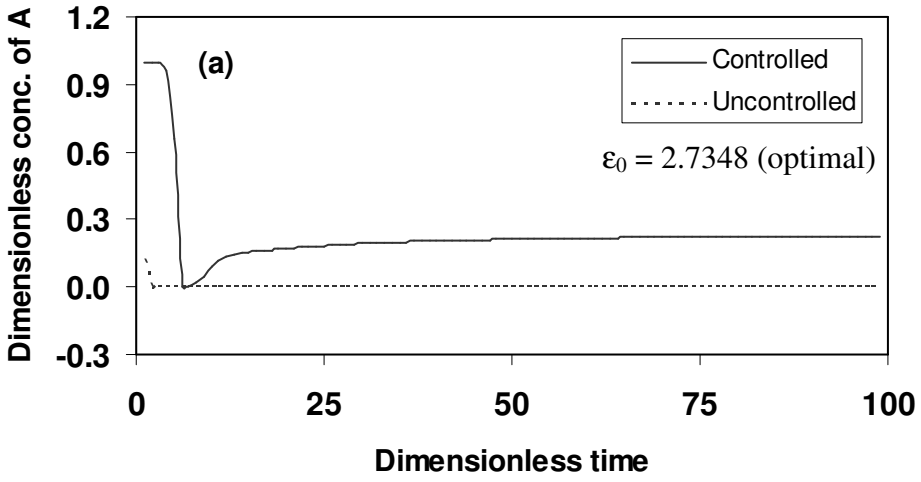


Figure 7.4 (panels *a*, *b*, *c*, *d*): Stabilization of Fluidized Bed Reactor at Unstable Steady State (USS) in presence of deterministic load disturbance $d = 0.05$

7.4.3 Controlling the Process at USS in Presence of Stochastic Load Disturbance

Control simulations similar to the control objective addressed in section 7.4.2 but in presence of the stochastic load disturbance have also been conducted. For simulating stochastic load disturbance, the *d* term in the governing equations was replaced by a random noise. Specifically, random noise obeying normal distribution (mean = 0 and standard deviation = 0.25) was added at each integrating step in Eq. 7.9. The controlled (continuous line) and uncontrolled (dotted line) time profiles of C_A, C_B and T in the presence of stochastic load disturbances are shown in Figs. 7.5*a*-7.5*c*, respectively. The corresponding u_t vs. t plot is shown in Fig. 7.5*d*. In this case too controller could impart excellent control action and thereby stabilize the process exactly at USS even in presence of stochastic load disturbance.



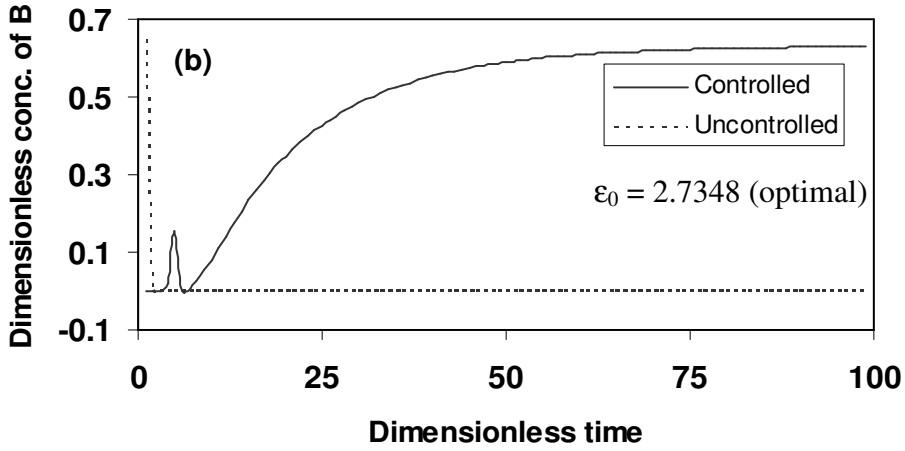
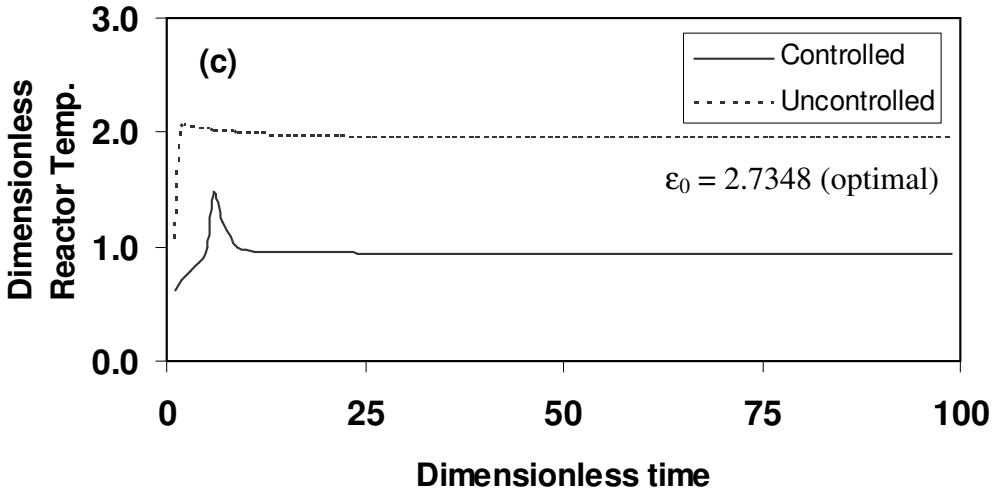


Figure 7.5 (panels *a*, *b*, *c*, *d*): Stabilization of Fluidized Bed Reactor at Unstable Steady State (USS) in presence of stochastic load disturbance (continued in next page)



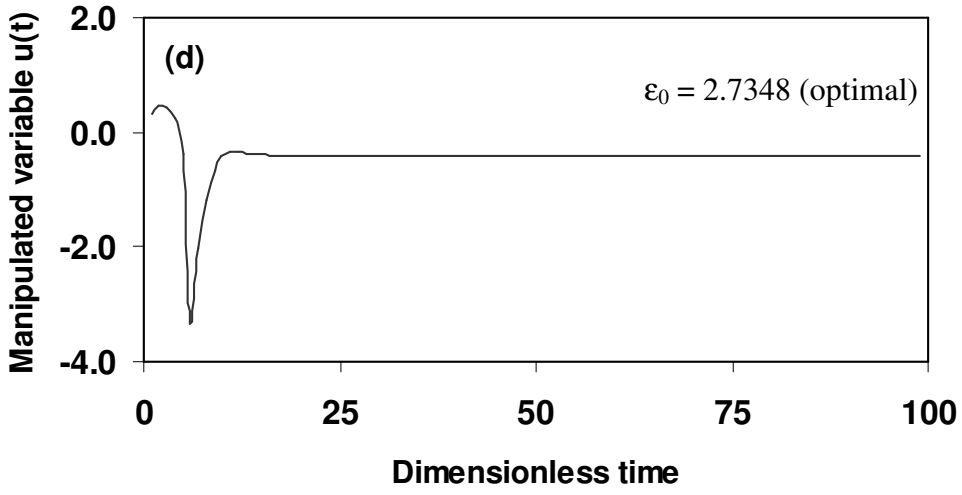


Figure 7.5 (panels *a*, *b*, *c*, *d*): Stabilization of Fluidized Bed Reactor at Unstable Steady State (USS) in presence of stochastic load disturbance

7.5 CONCLUSION

We have utilized a gain scheduling based control strategy, which does not require a phenomenological model for implementing the control action. The optimal value of the only tunable parameter ε_0, is obtained using the genetic algorithm optimization strategy. The control strategy is presented for regulating the dynamics of the fluidized bed reactor exactly at an unstable steady state. We have also studied the performance of the controller in presence of deterministic and stochastic load disturbances. The presented methodology can also be extended to control oscillatory, quasi-periodic and even chaotic dynamic behavior. It can also be used to implement servo control. We have utilized a simple time dependent linear variation of the controller gain, which can also be extended to other gain-adaptive techniques. The manipulated variable moves rapidly under the influences of error term *e*, and its time-derivative (*de*/*dt*) as soon as the control action is switched on. As time progresses, the set point error *e* and hence its derivative (*de*/*dt*) approaches zero. With increasing time, the numerical value of *t* eventually becomes much larger compared to *e* or (*de*/*dt*), which in turn results in faster system movement towards the desired set point.

7.6 NOMENCLATURE

C_A : dimensionless concentration of component A (reactant)

C_B : dimensionless concentration of component B (product)

Le_A : Lewis number of component A (= 1.0)

Le_B : Lewis number of component B (= 0.454545)

T : dimensionless reactor dense phase temperature

T^f : base value of dimensionless feed temperature to the reactor (= 0.55342)

T^{set} : set point for the controller (= 0.92955)

u_t : manipulated variable (deviation from reference value of T^f)

Greek Symbols:

ψ : reciprocal of the effective residence time of the bed (= 0.12543)

α_1 : normalized pre-exponent factor for the reaction $A \rightarrow B$ (= 10^8)

α_2 : normalized pre-exponent factor for the reaction $B \rightarrow C$ (= 10^{11})

β_1 : dimensionless overall exothermicity factor for the reaction $A \rightarrow B$ (= 0.4)

β_2 : dimensionless overall exothermicity factor for the reaction $B \rightarrow C$ (= 0.6)

γ_1 : dimensionless activation energy for the reaction $A \rightarrow B$ (= 18.0)

γ_2 : dimensionless activation energy for the reaction $B \rightarrow C$ (= 27.0)

ε_0 : tunable parameter of the proposed controller

7.7 REFERENCES

1. Ajbar, A. and Elnashaie, S. S. "Controlling Chaos by Periodic Perturbations in Nonisothermal Fluidized-Bed Reactor", ***AIChE Jour.***, **42**, 3008 - 3019 (1996).
2. Bandyopadhyay, J. K., Tambe, S. S., Jayaraman, V. K., Deshpande, P. B. and Kulkarni, B. D. "On Control of Nonlinear System Dynamics at Unstable Steady State", ***Chem. Engg. J.***, **67**, 103 - 114 (1997).
3. Deb, K., ***Optimization for Engineering Design: Algorithms and Examples***, Prentice Hall, New Delhi (1995).
4. Goldberg, D. E., ***Genetic Algorithms in Search, Optimization and machine Learning***, Addison – Wesley, Reading MA (1989).
5. Ott, E., Grebogi, C. and Yorke, J. A. "Controlling Chaos" ***Phys. Rev. Lett.***, **64**, 1196 - 1199 (1990).
6. Shinbrot, T., Grebogi, C., Ott, E. and Yorke, J. A. "Using Small Perturbations to Control Chaos", ***Nature***, **363**, 411 - 417 (1993).
7. Singer, J., Wang, Y-Z. and Bau, H. H. "Controlling a Chaotic System" ***Phys. Rev. Lett***., **66**, 1123 - 1125 (1991).
8. Sinha, S., Ramaswamy, R. and Subba Rao, J. "Adaptive Control in Nonlinear Dynamics" ***Physica D***, **43**, 118 - 123 (1990).

CHAPTER 8

ARTIFICIAL NEURAL NETWORK ASSISTED NOVEL OPTIMIZATION STRATEGY FOR PROCESS PARAMETERS AND TOLERANCES

Manuscript under preparation to be communicated to ***Chemical Engineering Journal***

Abstract

This chapter presents an AI-based robust process optimization strategy integrating artificial neural networks (ANN) with a novel stochastic optimization formalism namely, differential evolution (DE). Firstly an ANN based process model was developed from the process input-output data and subsequently design and operating variables were optimized using the DE formalism. The efficacy of the ANN-DE hybrid modeling – optimization methodology was demonstrated using a case study involving a non-isothermal continuously stirred tank reactor (CSTR). The optimization task addressed in this study is complex since it involves the issue of optimal tolerance design while performing the conventional parameter design. The results obtained using ANN-DE strategy are compared with those obtained earlier using (i) the ANN – GA strategy (Nandi et. al., 2001) and (ii) phenomenological model optimized by a gradient descent based strategy (Bernardo and Saraiva, 1998). These results show that the ANN-DE is an attractive alternative for conducting process optimization.

8.1 INTRODUCTION

In this chapter we propose a new AI-based hybrid process modeling and optimization strategy integrating an artificial neural network (ANN) and *differential evolution* (DE) (Storn and Price, 1995; 1997) strategy. The efficacy of the proposed formalism, namely ANN-DE, has been successfully demonstrated for optimizing a process wherein a non-trivial objective involving simultaneous optimization of both the process operating parameters as also their tolerances was addressed. The process considered is a non-isothermal continuously stirred tank reactor (CSTR). Additionally the results obtained using the ANN-DE strategy are compared with the results published earlier using (i) the ANN-GA strategy (Nandi et. al., 2001) and (ii) a phenomenological model and its optimization using a standard gradient descent-based nonlinear optimization strategy (Bernardo and Saraiva, 1998).

The stochastic optimization such as differential evolution (DE) and genetic algorithms (GAs) among others, are not heavily constrained by the properties of the objective function and thus they are potential candidates for employment in the optimization of an exclusively data driven ANN model. An important characteristic of the DE and GA methodologies is that they need measurements of the objective function only, and not the measurements (or direct calculation) of the gradient (or higher order derivatives) of the objective function. This characteristic of the DE and GA methods can be gainfully exploited for optimizing the ANN-based models whose functional forms do not assuredly satisfy the requirements (i.e., smoothness, continuity and differentiability) of the commonly used gradient-based optimization methods. Additional benefit of the DE and GA methods is that they can be used in situations where input information into the optimization method (e.g., objective function evaluations) may be noisy. The objective of this chapter, therefore, is to present a hybrid "modeling-optimization" technique namely, ANN-DE, for the purpose of robust chemical process design and optimization. In this approach, an ANN-based process model is first developed and its input space is optimized next using DE formalism. The principal advantage of the ANN-DE hybrid method is that process design and optimization can be conducted solely from the steady-state process input-output data.

For validating the ANN-DE method, we have considered a non-trivial process optimization objective, which not only aims at obtaining the optimal values of the operating process variables, but also their optimal tolerance values (operating windows) Fixing the

values of tolerances becomes important owing to the fact that chemical processes involve a number of variables and/or parameters that are always subjected to some degree of uncertainty (stochastic variability). For instance, irrespective of how good a control system is, process variables such as concentration, temperature and pressure, do vary randomly, albeit within a narrowly bounded window. Depending upon their origin, uncertainties can be classified into following four categories (Pistikopoulos, 1995; Pistikopoulos and Ierapetritou, 1995; also see Diwekar and Kalagnanam, 1996; 1997a,b).

- *Process-inherent uncertainty*: Due to random variations in process parameters/variables, such as flow rate, temperature and pressure.
- *Model-inherent uncertainty*: Accounts for variations in the phenomenological model parameters representing, for instance, kinetic constants, heat / mass transfer coefficients, and physical properties.
- *External uncertainty*: Considers variations in parameters that are external to the process, but influencing the process cost (feed stream availability, product demand, pollution/economic indices, etc.).
- *Discrete uncertainty*: Accounts for the equipment availability and other random discrete events.

The conventional deterministic process optimization approach ignores uncertainties, thereby resulting in sub-optimal solutions. Uncertainties are capable of influencing, for instance, the product quality and control cost and, therefore, they need to be considered during the process design and optimization activity. Accounting for uncertainties lead to the *tolerance design*, which aims at obtaining the optimal size of the operating window for each uncertainty-affected process variable / parameter. Best average process performance can be achieved consequent to the optimal tolerance design so long as the process operates within the optimized operating zones.

Bernardo and Saraiva (1998) have introduced a novel robust optimization (RO) framework that deals with the optimization objective alluded to above. An advantage of the optimal solution given by the RO framework is that it provides best operating regions for designing a control system. One of the optimization problems considered by Bernardo and Saraiva was minimization of the continuous stirred tank reactor's (CSTR's) annual plant cost that comprises four components namely, *equipment*, *operating*, *control* and *quality* costs. The

RO formalism, which accounts for the control costs right at the process design stage, was shown to yield qualitatively improved results as compared to those obtained using a fully deterministic optimization approach. Specifically, it was shown for the case of CSTR that simultaneous optimization of process operating variables and respective tolerances could reduce the total annual plant cost by nearly one order of magnitude, i.e., from 101872 $/yr to 14716 $/yr. For the sake of affording a direct comparison, we adopt the RO framework with necessary modifications to account for the ANN-based process model. The principal differences between the RO methodology and the hybrid formalisms presented here are:

- While the RO framework assumes the knowledge of a phenomenological process model, the ANN-DE formalism utilizes an artificial neural network based process model.
- In the RO approach, the phenomenological process model is optimized using a deterministic successive quadratic programming algorithm (NPSOL package, Gill et. al., 1986), whereas the hybrid method optimizes the ANN-based model using an inherently stochastic optimization technique namely, DE. It may however be noted that evaluation of the objective function accounting for the stochastic behavior of the uncertainty-affected process variables, remains same in the RO and ANN-based hybrid formalisms.
- It is shown in the present study that the hybrid optimization method is capable of yielding comparable solutions when noise-free as also noisy process input – output data are utilized for constructing the ANN-based process model.

This chapter is structured as follows. First, the mathematical formulation of the ANN model based robust optimization is briefly discussed. A detailed discussion on the development of ANN-based process model and the step-wise implementation of the ANN-DE hybrid optimization methodology is provided next. Finally, results pertaining to the CSTR optimization case study are presented and compared with those obtained using the ANN-GA and RO approaches.

8.2 ANN-ASSISTED ROBUST OPTIMIZATION FRAMEWORK

While developing the framework, it is not necessary to take into account the model-inherent uncertainty, since ANN-DE strategy does not utilize a phenomenological model. Among the remaining three uncertainty categories, only process-inherent uncertainty has been considered although the optimization framework presented below is sufficiently general towards inclusion of the remaining two (external and discrete) uncertainties. The origin of the commonly encountered process-inherent uncertainty lies in the small but significant random (uncontrolled) fluctuations present in the process operating variables.

We define the scope of the optimization problem under consideration as: *given process input data comprising steady-state values of equipment (design) and operating variables, and the corresponding values of process output variables, obtain in a unified manner, the optimal values of (i) design variables, (ii) operating variables, and (iii) tolerances defining bounds on the operating variables. The optimal solutions so obtained should ensure minimization of the annual plant cost while maintaining the desired product quality.*

The ANN-assisted RO framework assumes that a steady-state process model defined as:

$$y = f\left(\Psi, \Phi, W\right) \tag{8.1}$$

is available where y represents the process output variable that also determines product quality; Ψ and Φ respectively refer to the M- and N-dimensional vectors of design and operating process variables ($\Psi = \left[\psi_1, \psi_2, ..., \psi_m, ..., \psi_M\right]^T$; $\Phi = \left[\phi_1, \phi_2, ..., \phi_n, ..., \phi_N\right]^T$); W denotes the weight matrix of the ANN model, and f represents the ANN-approximated nonlinear function.

The total annual plant cost (C_{yr}) to be minimized, is assumed to consist of four components, namely, equipment cost (C_{eqp}), operating cost (C_{op}), control cost (C_c), and quality cost (C_q):

$$C_{yr} = C_{eqp} + C_{op} + C_c + C_q \tag{8.2}$$

Among the four cost components, the operating, control and quality costs have associated with them uncertainties emanating from random fluctuations in the process operating variables. However, the equipment cost usually being a deterministic quantity, has no

uncertainty attached to it. The extent of uncertainty in an operating variable can be characterized in terms of a probability density function (PDF) (Diwekar and Rubin, 1991; 1994) where mean value of the PDF represents the nominal value of that operating variable. Accordingly, defining $J(\Phi)$ to be the set of PDFs associated with the operating variables, Φ, the corresponding set $\left(\hat{\Phi}\right)$ describing the operating space regions can be represented as:

$$\hat{\Phi} = \left\{\Phi : \Phi \in J\left(\Phi\right)\right\} \tag{8.3}$$

The physical operating regions, $\hat{\Phi}$, comprise a set of operating windows $\left\{\hat{\Phi}_{\phi_n}\right\}$, where Φ_{ϕ_n} denoting the window for nth operating variable, is defined as:

$$\hat{\Phi}_{\phi_n} = \left[\phi_n^l, \phi_n^u\right] \quad ; \; n = 1, 2, \ldots, N \tag{8.4}$$

Here, ϕ_n^l and ϕ_n^u, respectively representing the lower and upper bounds on the nth operating variable, are expressed as:

$$\phi_n^l = \mu_n\left(1 - \varepsilon_n\right) \quad ; \qquad \phi_n^u = \mu_n\left(1 + \varepsilon_n\right) \tag{8.5}$$

where, μ_n refers to the mean value of nth operating variable and ε_n is the associated tolerance. Commonly, variations in ϕ_n obey Gaussian (normal) probability distribution and, therefore, the respective tolerance $\left(\varepsilon_n\right)$ can be approximated as:

$$\varepsilon_n = 3.09\left(\sigma_n / \mu_n\right) \tag{8.6}$$

where σ_n refers to the standard deviation of the Gaussian PDF.

Owing to the random fluctuations in process operating variables, the steady-state value of the process output (quality) variable, y, also deviates from its desired (nominal) set point. Thus, it becomes essential to define a PDF, $L(y)$, pertaining to the quality variable y as well. Accordingly, an expression similar to Eq. 7.3, but involving $L(y)$ can be written for defining the corresponding space region $\left(\hat{y}\right)$:

$$\hat{y} = \left\{y : y \in L\left(y\right)\right\} \tag{8.7}$$

Using Eqs. 8.2-8.7, it is possible to write the complete mathematical formulation of the robust optimization problem under consideration as:

$$\min_{\Psi, \hat{\Phi}} C_{yr}\left(\Psi, \hat{\Phi}, \hat{y}\right) = C_{eqp}\left(\Psi\right) + C_{op}\left(\hat{\Phi}\right) + C_c\left(\hat{\Phi}\right) + C_q\left(\hat{y}\right) \tag{8.8}$$

subject to,

$$\text{(I)}\ \ f\left(\Psi,\Phi,W\right)=y \quad \text{for all} \quad \Phi\in\hat{\Phi}\ ;\ \Phi=\left[\phi_1,\phi_2,...,\phi_N\right]^T\ ;\ \hat{\Phi}=\left\{\Phi:\Phi\in J\left(\Phi\right)\right\} \tag{8.9}$$

$$\text{(II)}\ \hat{y}=\left\{y:\ y\in L\left(y\right)\right\} \tag{8.10}$$

This formulation makes use of an ANN model (Eq. 8.1) possessing following properties: (i) the input space of the ANN model comprises design variables, Ψ, and uncertainty-involving process operating variables, Φ, (ii) the output space of the ANN model represents of the quality variable y, (iii) the weight matrix (W) of the ANN model is available, and (iv) the model is valid over operating variable regions $\hat{\Phi}$, and the output region, $\hat{y}$; consequently, the equality constraint defined in Eq. 8.9 always holds.

In the objective function defined by Eq. 8.8, the elements of vectors Ψ and $\hat{\Phi}$ signify the decision variables; $\hat{y}$ representing the space region of the output variable depends on Ψ and $\hat{\Phi}$, and therefore is not a decision variable. However, the optimization objective involving simultaneous determination of operating variables (nominal operating points) and associated tolerances, necessitates: (i) optimization of the mean values $\left(\mu_n;n=1,2,...,N\right)$ of the Gaussian PDFs characterizing uncertainty-involving operating variables Φ, and (ii) optimization of the tolerances, $\varepsilon_n\ \left(n=1,2,...,N\right)$. Simultaneous determination of mean $\{\mu_n\}$ and tolerance $\{\varepsilon_n\}$ values also fixes the corresponding standard deviations, $\{\sigma_n\}$ (see Eq. 8.6), which can be used to characterize the PDF set, $J\left(\Phi\right)$. It is thus clear that optimization of mean and associated tolerance values in turn leads to the optimization of the PDF set, $J\left(\Phi\right)$.

Following the prescription of Bernardo and Saraiva (1998), the objective function defined in Eq. 8.8 can be evaluated as:

$$C_{yr}=E\left(C_q\right)+E\left(C_{op}\right)+C_{eqp}\left(\Psi\right)+C_c\left(\hat{\Phi}\right) \tag{8.11}$$

where $E(C_q)$ and $E(C_{op})$ refer to the expected values of the quality cost and the operating cost, respectively. For computing these expected costs, an efficient sampling technique known as *Hammersley sequence sampling* (HSS) (Diwekar and Kalgnanam, 1996; 1997a,b) could be utilized. In this technique, statistically adequate number (N_{obs}) of observations are sampled from the Gaussian PDFs associated with the uncertainty-involving operating variables. Next,

each of the N_{obs} sampled sets, $\Phi_j \left(j = 1, 2, ..., N_{obs} \right)$, along with the design variable vector, Ψ, is applied to the ANN model for computing the magnitude of the process output, y. The estimate of the quality cost can then be computed using the Taguchi loss function (Taguchi, 1986) as given below:

$$E\left(C_q\right) = k_l \left[\left(\mu_y - y^* \right)^2 + \sigma_y^2 \right] \tag{8.12}$$

where, k_l refers to the *quality loss coefficient*; y^* is the desired value of y, and σ_y denotes the standard deviation of N_{obs} number of y values. The mean value of the quality variable, μ_y, is calculated as:

$$\mu_y = \frac{\sum_{j=1}^{N_{obs}} f\left(\Psi, \Phi_j, W\right)}{N_{obs}} \tag{8.13}$$

For computing the estimate of the operating cost, $E\left(C_{op}\right)$, following expression is used:

$$E\left(C_{op}\right) = \frac{\sum_{j=1}^{N_{obs}} C_{op}\left(\Phi_j\right)}{N_{obs}} \tag{8.14}$$

Since design variables are not associated with any uncertainty, the Ψ-dependent equipment cost, $C_{eqp}\left(\Psi\right)$, can be calculated deterministically. The last of the four cost components representing the control cost, C_c, can be determined using mean $\left(\mu_n\right)$ and standard deviation $\left(\sigma_n\right)$ of the PDFs describing operating variables:

$$C_c = \sum_{n=1}^{N} \left(a + b \frac{\mu_n}{\sigma_n} \right) \tag{8.15}$$

where a, b are constants and n refers to the operating variable index.

8.3 CONSTRUCTION OF ANN-BASED PROCESS MODEL

First a class of ANNs known as *multilayered feedforward networks* (MFFNs) (see Figure 8.1) is to be developed. An MFFN is a nonlinear mapping device between an input set (I) and an output set (O). It represents a function f that maps I into O i.e., $f : I \rightarrow O$, or $y = f\left(\mathbf{x}\right)$, where $y \in O$ and $\mathbf{x} \in I$. The widely used MFFN paradigm is *multi-layered perceptron* (MLP), mostly comprising three sequentially arranged layers of processing

units. The three successive layers, namely, *input*, *hidden*, and *output* layers house N_I, N_H, and N_O number of nodes, respectively.

The problem of neural network modeling is to obtain a set of weights such that the prediction error measured in terms of a suitable error function, for instance, the *root-mean-squared error* (RMSE), is minimized. The RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{\sum_{l=1}^{N_{pat}} 2E_l}{N_{pat} \times N_o}} \tag{8.16}$$

where l refers to the input pattern index (l = 1, 2, …, N_{pat}); N_o denotes the number of output layer nodes, and E_l is a measure of the *sum-of-squares* error (SSE), defined as:

$$E_l = \frac{1}{2}\sum_{i=1}^{N_o} \left(y_l^i - o_l^i \right)^2 \tag{8.17}$$

where, y_l^i denotes the desired output of the ith output node when lth input pattern is presented to the network, and o_l^i refers to the corresponding actual output. The task of RMSE minimization is accomplished by "training" the network wherein a gradient descent technique, such as the *generalized delta rule* (GDR) (Rumelhart et al., 1986; Tambe et. al., 1996), is utilized for the updating the connection weights.

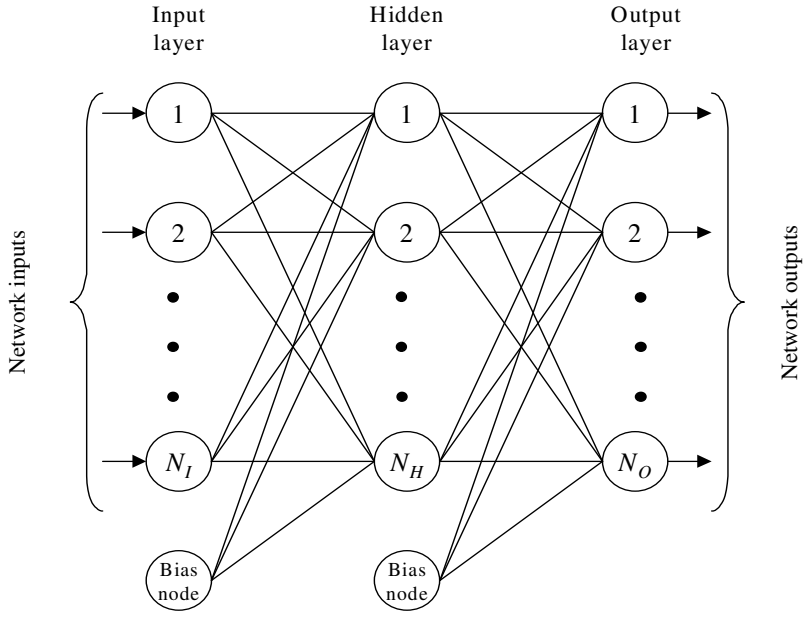


Figure 8.1: A schematic of three-layered feed-forward neural network

8.4 ANN MODEL ASSISTED STOCHASTIC PROCESS OPTIMIZATION METHODOLOGIES

The principal difference between the widely used deterministic gradient-based optimization schemes and the stochastic ones such as DE and GA is that the latter class of methodologies involves a random component at some stage in their implementation. For instance, GAs manipulate a set of candidate solutions at random with the objective of sampling the search (solution) space as widely as possible while at the same time trying to locate promising regions for further exploration (Venkatasubramanian and Sundaram, 1998). In the present work, the DE method has been used, in conjunction with an ANN-based process model, for optimizing: (i) process design variables, Ψ, (ii) process operating variables, Φ, and (iii) tolerances, $E = [\varepsilon_1, \varepsilon_2, ..., \varepsilon_n, ..., \varepsilon_N]^T$, associated with the process operating variables. In what follows, the salient features and implementation details of the ANN-DE methodology is provided. For the sake of brevity a brief description of the ANN-GA method has also been furnished.

8.4.1 ANN-DE Optimization Methodology

The Differential Evolution (DE) paradigm was proposed by Storn and Price (1995; 1999). It soon became a popular tool for solving global optimization problems because of its several attractive features such as having fewer control parameters, ease in programming, efficiency, etc. DE is similar to genetic algorithms in the sense that it uses same evolutionary operators like *mutation*, *crossover* and *selection* for guiding the population towards the optimum solution. DE has been successfully applied to solve a wide range of real life application problems (Storn 1995; Onwubolu and Babu, 2004) and has reportedly outperformed other optimization techniques for applications such as estimation of heat transfer parameter for multiphase reactors, optimization of adiabatic styrene reactor, global optimization of MINLP problems, shell and tube heat exchange design, phase equilibrium and phase stability problems, etc. (Babu and Sastry, 1999; Babu et. al., 2005; Babu and Angira, 2006; Babu and Munawar, 2007; Srinivas and Rangaiah, 2007). Among the DE's advantages are its simple structure, ease of use, speed and robustness. Price and Storn suggested ten different strategies of DE (Price, 1999). A strategy that works out to be the best for a given

problem may not work well when applied for a different problem. Also, the strategy and key parameters to be adopted for a problem are to be determined by trial and error.

The schematic diagram in Figure 8.2 provides a way to visualize the working principle of DE. The crucial idea behind DE is a scheme for generating trial parameter vectors. Basically, DE adds the weighted difference between two population vectors to a third vector. The key parameters of control in DE are: *NP* - the population size, *Cr* - the crossover constant, and F_{sc} - the weight applied to random differential (scaling factor). Price & Storn (1995, 1999) have given some simple rules for choosing key parameters of DE for any given application. Normally, *NP* should be about 5 to 10 times the dimension (number of parameters in a vector) of the problem. As for F_{sc}, it lies in the range 0.4 to 1.0. Initially F = 0.5 can be tried then F_{sc} and / or *NP* is increased if the population converges prematurely. As for *Cr*, it lies in the range 0.1 to 1.0.

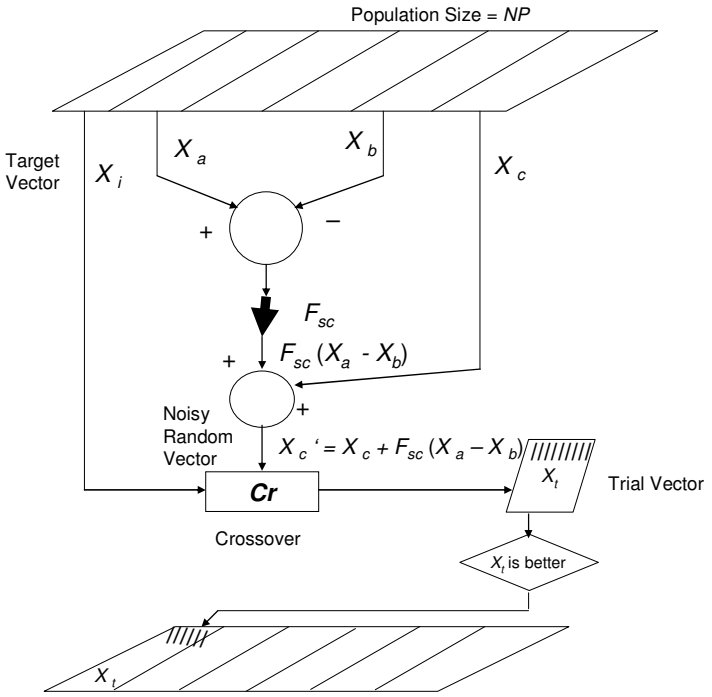


Figure 8.2: Schematic Diagram of Differential Evolution Strategy

DE starts with a population of *NP* candidate solutions which may be represented as $X_{i,G}$, i = 1, 2, . . . , *NP*, where index i denotes the population and G denotes the generation to which the population belongs. The working of DE depends on the manipulation and efficiency of three main operators; mutation, reproduction and selection.

Mutation:. The mutation operation of DE applies the vector differentials between the existing population members for determining both the degree and direction of perturbation applied to the individual subject of the mutation operation. The mutation process at each generation begins by randomly selecting three individuals in the population. The *i*th perturbed individual, $V_{i,G+1}$, is then generated based on the three chosen individuals as follows:

$$V_{t,G+1} = X_{r_3,G} + F_{sc} \times \left(X_{r_1,G} - X_{r_2,G} \right)$$

(8.18)

where, $i = 1, 2, \ldots, NP$; $r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$ are randomly selected such that $r_1 \neq r_2 \neq r_3 \neq i$, $F_{sc} \in [0, 1+]$, F is the control parameter.

Crossover: The perturbed individual, $V_{i,G+1} = (v_{1,i,G+1}, v_{2,i,G+1}, \ldots, v_{n,i,G+1})$, and the current population member, $X_{i,G} = (x_{1,i,G}, x_{2,i,G}, \ldots, x_{n,i,G})$, are then subject to the crossover operation, that generates the population of candidates, or "trial" vectors,$U_{i,G+1} = (u_{1,i,G+1}, u_{2,i,G+1}, \ldots, u_{n,i,G+1})$, as follows:

$$u_{j,i,G+1} = \begin{cases} v_{j,i,G+1} & \text{if } rand_j \leq Cr \vee j = k \\ x_{j,i,G} & \text{otherwise} \end{cases} \qquad (8.19)$$

j = 1. . . *n*, $k \in \{1, \ldots, n\}$ is a random parameter's index, chosen once for each *i*, and the crossover rate, $Cr \in [0, 1]$, is set by the user.

Selection: The population for the next generation is selected from the individual in current population and its corresponding trial vector according to the following rule

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & \text{if } f\left(U_{i,G+1}\right) \leq f\left(X_{i,G}\right) \\ X_{i,G} & \text{otherwise} \end{cases} \qquad (8.20)$$

Each individual of the temporary (trial) population is compared with its counterpart in the current population. The one with the lower objective function value will survive from the tournament selection to the population of the next generation. As a result, all the individuals

of the next generation are as good as or better than their counterparts in the current generation. In DE trial vector is not compared against all the individuals in the current generation, but only against one individual, its counterpart, in the current generation.

For illustrating the working principles of DE, we recast the cost minimization problem (Eq. 8.8) as:

$$\text{Minimize} \quad C_{yr}(\mathbf{x}) ; \quad x_k^L \le x_k \le x_k^U \ ; \ k = 1, 2, ..., K \ ; \quad \mathbf{x} = \Psi \cup \Phi \cup \mathrm{E} \qquad (8.21)$$

where K-dimensional vector, $\mathbf{x} = [\, x_1, x_2, \ldots, x_k, \ldots, x_K]^T$, represents the set of decision variables and x_k^L and x_k^U are the lower and upper bounds on x_k. The implementation procedure of ANN-DE formalism for overall cost minimization problem as defined by function defined by Eqn. 8.21 can be given as (also see flowchart in Figure 8.3):

Step 1: Set the iteration index, t, to zero, and initialize control parameters scaling factor F, crossover frequency Cr and population size NP and maximum number of generation t_{max}.

Step 2: Create and initialize population P (0) of NP individuals. Here real value points are generated randomly, not the binary coded ones.

Step 3: Using the HSS technique on all the operating variable sets from the Gaussian PDFs associated with the operating variables, Φ.

Step 4: Apply jth (j = 1, 2, …, N_{obj}) sampled set $\left(\Phi_j\right)$ along with the corresponding decision variable vector $\left(\Psi_i\right)$ to the ANN model and obtain the model output, y_j. Next, compute mean $\left(\mu_y\right)$ and standard deviation $\left(\sigma_y\right)$ of the ANN output set, $\{y_j\}$.

Step 5: Evaluate objective function and fitness score $f(\mathbf{x}_i(t))$ for each individual $\mathbf{x}_i(t)$ in P (t)

Step 6: Create the trial vector $\mathbf{u}_i(t)$ by applying mutation operator.

Step 7: Create an offspring $\mathbf{x}_i'(t)$ by applying crossover operator.

Step 8: Check the fitness using Eq. 8.18, if $f(\mathbf{x}_i'(t))$ is better than $f(\mathbf{x}_i(t))$ then add $\mathbf{x}_i'(t)$ to P (t + 1) else add $\mathbf{x}_i(t)$ to P (t + 1) .

Step 9: Update generation index by one: $t = t + 1$.

Step 10: Repeat steps 3-9 on the new generation strings until a convergence criterion, such as (i) t exceeds the pre-specified maximum generations limit t_{max}, or (ii) the fitness score of the best string in a population no longer increases, is satisfied.

Step 11: Return the individual with the best fitness as the solution.

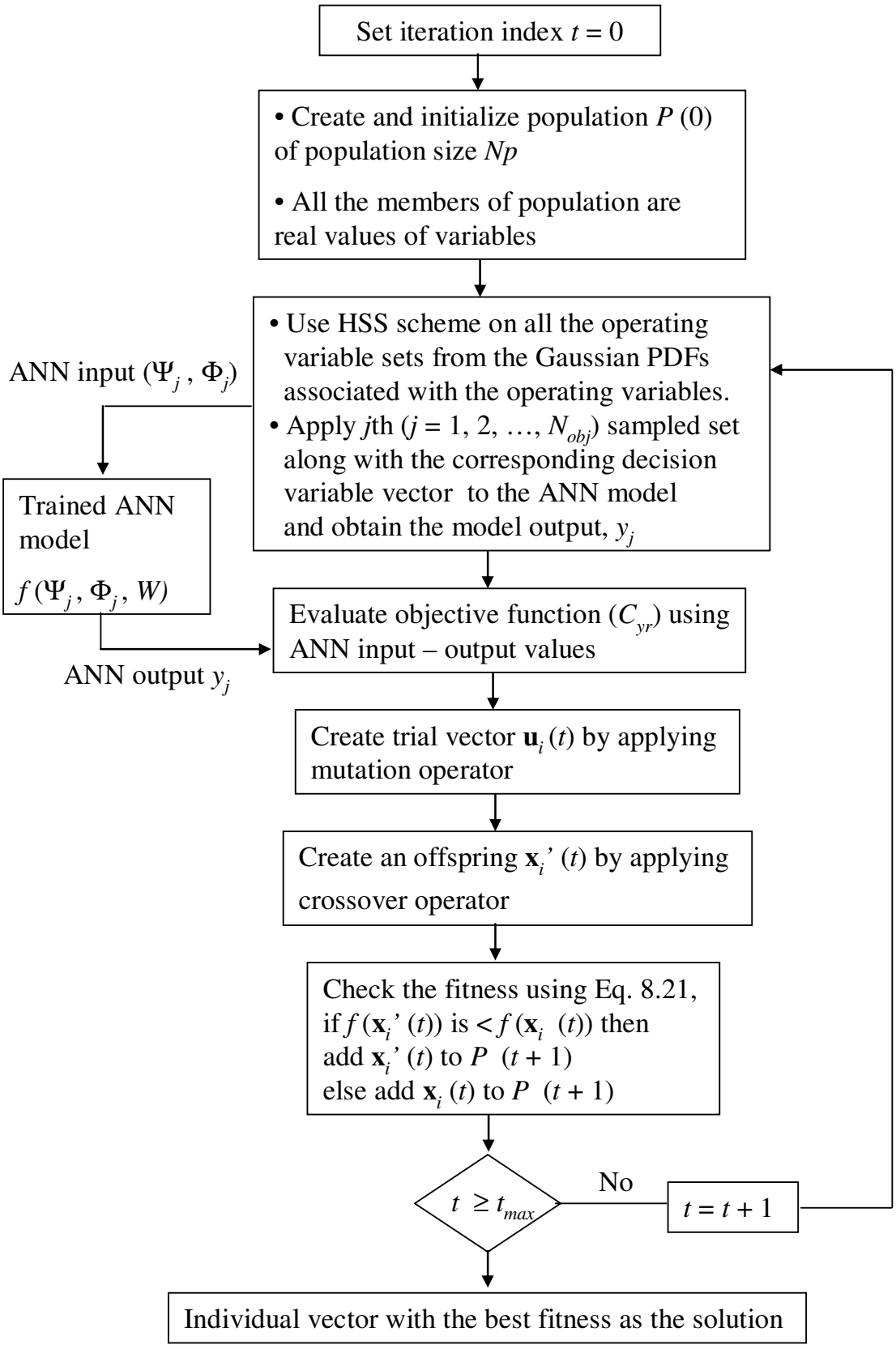


Figure 8.3: Flowchart for implementation of ANN-DE hybrid methodology

8.5 OPTIMIZATION OF CSTR USING HYBRID STRATEGIES

Consider a steady-state process involving a jacketed non-isothermal CSTR (see Figure 8.4) wherein two first order reactions in series, $A \rightarrow B \rightarrow C$, take place. For implementing the ANN-DE scheme, an MLP-based model approximating the functional relationship between CSTR's design and operating variables, and the corresponding steady-state values of the output variable was first developed. For convenience, the steady-state CSTR data required for developing the MLP model were generated using CSTR's phenomenological model. In actual practice, the MLP model can be developed from a representative steady-state database that is already at hand or generated by conducting specially designed experiments. The phenomenological equations of CSTR used for simulating its steady-state behavior are given in Appendix 8.1, where volume (V, m^3) represents the design variable and, flow rate (F, m^3/min), heat removal rate (Q, kJ/min), inlet concentration of reactant A (C_A^0, mol/m^3), inlet concentration of B (C_B^0, mol/m^3), and inlet temperature (T^0, K), collectively denote the five operating variables. The CSTR output variable, namely, the rate of production (mol/min) of B has been chosen as the quality variable and its steady-state value (y) has been obtained as:

$$y = \left(\hat{C}_B - C_B^0\right) \times F \tag{8.22}$$

where, $\hat{C}_B$ represents the steady-state concentration of B. The desired value of y defined as y^* is 600 mol/min.

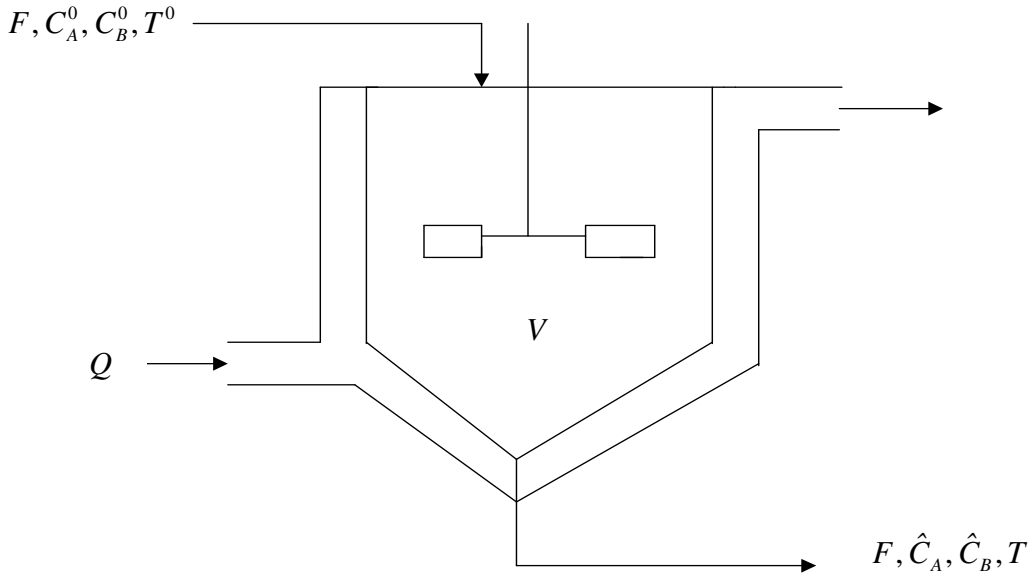


Figure 8.4: Schematic of a continuous stirred tank reactor (CSTR)

For operating a process, ranges of design and operating variables are usually specified. Accordingly, following ranges were considered for the steady-state CSTR simulation: V = [0.3-0.4] m^3, F = [0.5-1.1] m^3/min, Q = [100-1100] kJ/min, $C_A^0 = [3000 - 4000]$ mol/m^3, $C_B^0 = [30 - 600]$ mol/m^3 and $T^0 = [300 - 320]$ K. Using these ranges, fifty random combinations of CSTR's design and operating variables were generated, and using each combination, the corresponding steady-state value of the quality variable, y, was computed. The data set comprising design and operating variables forms network's input space and the corresponding y values represent network's desired (target) output space. After normalizing and partitioning this data into the training set (40 patterns) and the test set (10 patterns), an optimal MLP network model was developed in accordance with the standard network training procedure described earlier. In the MLP training simulations, use of the sigmoid transfer function was made for computing the outputs of the hidden and output layer nodes. The optimal MLP architecture obtained thereby has six input nodes, two hidden nodes and one output node (N_I = 6, N_H = 2, N_O = 1); the corresponding values of the learning rate (η_l) and momentum coefficient (α_m) were 0.7 and 0.01, respectively. An MLP network with a good function approximation and generalization abilities results in small but comparable *RMSE* values for both the training set (E_{trn}) and the test set (E_{tst}). In the case of MLP-based CSTR model, the E_{trn} and E_{tst} magnitudes were 0.0061 and 0.0063, respectively. Additionally, values of the coefficient of correlation (CC) between the MLP-predicted and the corresponding desired y values were calculated. The CC values for the training and test sets were 0.999 and 0.998, respectively. It can be inferred from the very small magnitudes of the training and test set RMSEs and the corresponding high (≈1) CC magnitudes, that the MLP network has excellently approximated and generalized the nonlinear relationship existing between its six inputs and the single output.

The generalized framework of the ANN-based robust optimization aims at not only obtaining the optimal values of the design and operating variables, but also the optimal values of tolerances (E) associated with the process operating variables. Thus, for the CSTR case study, the overall decision space denoted by vector **x** (see Eq. 8.21) becomes eleven-dimensional (one design variable + five operating variables + five tolerances); for clarity, the correspondence between **x**-vector elements and the CSTR variables has been tabulated in

Table 8.1. Upon appropriate substitution from the notation given in Table 8.1 and the definition of Taguchi loss function (Eq. 8.12), the CSTR-specific form of the objective function representing the total annual cost (Eq. 8.8) can be written as:

$$C_{yr} = C_{eqp}(V) + C_{op}(\mu, \mathrm{E}) + C_c(\mu, \mathrm{E}) + C_q(\mu_y, y^*, \sigma_y) \qquad (8.23)$$

where, μ_y and σ_y respectively represent the mean and standard deviation of the quality variable, y, and y^* denotes the desired value of y (= 600 mol/min). The five-dimensional vectors μ and E, respectively representing the mean and tolerance values of the operating variables are defined as: $\mu = \left[\mu_F, \mu_Q, \mu_{C_A^0}, \mu_{C_B^0}, \mu_{T^0}\right]^T$ and $\mathrm{E} = \left[\varepsilon_F, \varepsilon_Q, \varepsilon_{C_A^0}, \varepsilon_{C_B^0}, \varepsilon_{T^0}\right]^T$.

Table 8.1: Equivalence between the decision vector elements and CSTR variables

Decision variable index (k)	**Decision vector variables**	**Decision variable type***	**Corresponding CSTR variable**
1	x_1	DES	volume, V (m^3)
2	$x_2\ (=\mu_F)$	OPR	flowrate, F (m^3/min)
3	$x_3(=\mu_Q)$	OPR	heat removal rate Q (kJ/min)
4	$x_4(=\mu_{C_A^0})$	OPR	inlet concentration of A, C_A^0 (mol/m^3)
5	$x_5(=\mu_{C_B^0})$	OPR	inlet concentration of B, C_B^0 (mol/m^3)
6	$x_6(=\mu_{T^0})$	OPR	inlet temperature, T^0 (K)
7	$x_7(=\varepsilon_F)$	TOL	tolerance for F (m^3/min)
8	$x_8(=\varepsilon_Q)$	TOL	tolerance for Q (kJ/min)
9	$x_9(=\varepsilon_{C_A^0})$	TOL	tolerance for C_A^0 (mol/min)
10	$x_{10}(=\varepsilon_{C_B^0})$	TOL	tolerance for C_B^0 (mol/min)
11	$x_{11}(=\varepsilon_{T^0})$	TOL	tolerance for T^0 (K)

* DES : design variable; OPR : operating variable; TOL : tolerance of an operating variable

8.5.1 DE-Based Optimization of CSTR

Implementation of the ANN-DE formalism was performed using following DE-specific parameter values: scaling factor, F_{sc} = 0.6, crossover constant, Cr = 0.8, population size, NP = 55 and maximum number of iterations, t_{max} = 250. It was observed during implementation of DE methodology that proper choice of the governing parameters, namely F, Cr, and NP, is a prerequisite to successful convergence. For a judicious selection of the stated parameters, please refer to several guidelines provided by Price and Storn (1999).

For sampling values of the five operating variables from their respective PDFs, a statistically adequate sample size of 400 measurements (Bernardo and Saraiva, 1998) was utilized (i.e., N_{obs} = 400). The function used for computing the fitness score of a candidate solution (ξ_i) was:

$$\xi_i = \frac{15000}{15000 + C_{yr}^i} \quad ; \; i = 1,2,\ldots,N_{pop} \tag{8.24}$$

where, C_{yr}^i refers to the overall annual plant cost corresponding to the ith candidate solution string. The functions used for computing various components of the annual plant cost are described in Appendix 8.2.

In a typical plant, the controller action constraints an operating variable from deviating beyond a specific limit. Accordingly, the tolerances $\left(\varepsilon_F, \varepsilon_Q, \varepsilon_{C_A^0}, \varepsilon_{C_B^0}, \varepsilon_{T^0}\right)$ for five operating variables were made to satisfy following constraints: (i) $0.0001 \le \varepsilon_F \le 0.15$, (ii) $0.0001 \le \varepsilon_Q \le 0.15$, (iii) $0.0001 \le \varepsilon_{C_A^0} \le 0.1$, (iv) $0.0001 \le \varepsilon_{C_B^0} \le 0.1$, and (v) $0.0001 \le \varepsilon_{T^0} \le 0.02$; these constraints can also be expressed as inequality constraints. Any candidate solution violating the constraints was penalized during fitness evaluation by resetting its fitness score to zero magnitude (Onwubolu and Babu, 2004).

The results of DE-based CSTR optimization are presented in column 2 of Table 8.2 The table also lists results obtained using the ANN-GA strategy (Nandi et. al., 2001). It is seen from the table that the DE-minimized annual total cost (13023.27 $/yr) is even lower than that given by the GA-based optimization (13853.46 $/yr). In the results of GA-based optimization, it is noticed that the σ_y value of 11.48 signifying standard deviation of product quality values for 400 sample points is almost identical to DE-based value of 11.43. As a

result, the product quality will exhibit nearly identical variability as evident from the quality cost values of 866.01 $/yr and 977.57 $/yr. In the case of DE-based solution, it is observed that the control cost has lower magnitude as compared to the GA-based solution. By definition, the control cost is inversely proportional to the tolerance magnitudes (refer Appendix 8.2, Eq. A-8.7) since smaller tolerances necessitate stricter process control, thereby increasing the cost of control. This can be verified from the tolerance values corresponding to the DE-based solution. It is observed that the optimized tolerances, 0.1047, 0.0651, 0.0666, 0.0692 and 0.0121 in respect of the process variables F, Q, C_A^0, C_B^0 and T^0 are larger as compared to those optimized by the GA (0.0936, 0.059, 0.031, 0.0505 and 0.005). Consequently, the control cost has assumed a smaller value (1451.0 $/yr as against 2201.02 $/yr).

8.5.2 GA-Based Optimization of CSTR

We have optimized the process with help of binary coded genetic algorithm as well (please refer to Nandi et. al., 2001 for details). For performing the GA-based optimization of the CSTR model, following values of the GA-specific parameters were used: number of variables K = 11, population size N_{pop} = 30, chromosome length l_{chr} = 55, crossover probability P_{cross} = 0.95, mutation probability, P_{mut} = 0.01 and maximum number of generations N_{gen}^{max} = 250. The same objective function as given by Eq. 8.24 was used for minimization of overall annual plant cost (C_{yr}). The search domain and constraints are identical to that discussed in DE-based optimization.

8.5.3 CSTR Optimization in Presence of Noisy Process Data

Sensors monitoring process variables and parameters often generate noisy measurements. Consequently, the mean recorded value of the noise-corrupted steady-state measurements may show a positive or negative deviation from its desired mean (nominal set point). The deviation magnitude, which is variable / parameter-specific, is likely to vary from one run to another. This situation is different from the process uncertainties that are caused by the random physical variations in the process variables / parameters.

Table 8.2: Comparison of solutions obtained using RO framework and ANN-GA and ANN-DE hybrid methodologies

	ANN model based on Noise-free process data		ANN model based on noisy process data		RO framework Solution [a]
	GA–based Solution	DE–based Solution	GA–based Solution	DE–based Solution	
	1	2	3	4	5
V (m^3)	0.3428	0.3209	0.3726	0.3407	0.3463
F (m^3/min)	0.5437(1±0.0936)	0.53299(1±0.1047)	0.5270(1±0.0649)	0.5807(1±0.0664)	0.5023(1±0.099)
Q (kJ/min)	262.253(1±0.0585)	1048.169(1±0.0651)	684.137(1±0.0617)	1046.65(1±0.1257)	146.7(1±0.080)
C_A^0 (mol/m^3)	3312.74(1±0.0314)	3192.87(1±0.0666)	3801.78(1±0.0219)	3663.29(1±0.0150)	3140(1±0.050)
C_B^0 (mol/m^3)	422.062(1±0.0505)	482.235(1±0.0692)	542.44(1±0.0543)	275.275(1±0.0480)	510.7(1±0.050)
T^0 (K)	310.444(1±0.0051)	313.196(1±0.0121)	304.42(1±0.0051)	306.938(1±0.0185)	313.8(1±0.005)
μ_y (mol/min)[b]	599.12	600.347	599.85	605.52	600
σ_y (mol/min)	11.48	11.43	11.26	10.67	16.8
C_{eqp} ($/yr)	1057.03	1014.5	1113.31	1052.92	1064
C_{op} ($/yr)	9729.4	9580.2	9623.98	9633.61	9712
C_c ($/yr)	2201.02	1451.0	2288.2	1598.03	2105
C_q ($/yr)	866.02	977.57	828.32	944.07	1835
C_{yr} ($/yr)	13853.46	13023.27	13853.81	13228.57	14716

[a] Obtained by Bernardo and Saraiva (1998). Solutions with respect to the operating variables are listed using $\mu_n(1\pm\varepsilon_n)$ format.

[b] Desired μ_y value $(=y^*)$: 600 mol/min

For the present case study, we consider a scenario wherein steady-state values of all the monitored process variables are corrupted with noise obeying the Gaussian PDF. Accordingly, the steady-state values of CSTR's design, operating and quality variables obtained earlier by solving the phenomenological model, were corrupted using the Gaussian noise. The extent of measurement noise in each variable was assumed to lie within ± 5% tolerance limit. Letting μ_l to be the true steady-state value (nominal set point) of *l*th process input/output variable, the corresponding standard deviation σ_l, required for generating the noisy measurements was computed as:

$$\sigma_l = \frac{0.05 \times \mu_l}{3.09} \tag{8.25}$$

All the seven elements of the 50 patterns representing CSTR's noise-free steady-state input-output data set were randomly corrupted using variable-specific Gaussian mean (μ_l) and standard deviation (σ_l) values. Specifically, a time series sequence comprising one thousand noisy measurements was generated for each pattern element. The sequence obtained thereby was denoised using a nonlinear noise-reduction algorithm (Kantz and Schreiber, 1997), and the resulting sequence was averaged out. The data-base obtained thereby consists of fifty patterns representing noise-filtered steady-state values of CSTR's seven input-output variables. It is worth pointing out here that even after noise-filtration, the resultant steady-state values do contain a small amount of residual noise. For creating training and test sets the noise-filtered steady-state database was normalized and partitioned in 4:1 ratio. Using these sets, an MLP-based optimal steady-state model was developed following the three-step training procedure elaborated earlier. The optimal network model comprising 6, 2 and 1 nodes in its input, hidden and output layers, respectively, was trained using η_l and λ_m values of 0.7 and 0.01 respectively. For this network model, the RMSE magnitude for the training set was 0.0134, and for the test set, the magnitude was 0.0123. The corresponding CC magnitudes were 0.998 (training set) and 0.996 (test set). The RMSE (CC) values are sufficiently low (high) to infer that the MLP-network has captured well the inherent relationship between CSTR's noise-filtered input and output variables. A comparison of the E_{trn} and E_{tst} values (0.0134 and 0.0123) pertaining to the noise-reduced steady-state data with the corresponding ones (0.0061 and 0.0063) when noise-free data were used for constructing an ANN model,

reveals that the former set of RMSE values are marginally higher. This indicates that the network model has fitted the noise-filtered data with marginally lower accuracy. It may be noted that the MLP-training procedure utilized in this study ensures that the network is not an overfitted one. Due to avoidance of overfitting, the network has not fitted the small amount of residual noise contained in the noise-filtered data but instead has approximated the underlying trends (physico-chemical phenomenon) therein. This in turn has resulted in marginally higher RMSE values in respect of the ANN model trained on the noise-filtered steady-state data. Similar inference can also be drawn from the lower CC values in respect of the predictions made by the network trained on the noise-filtered data.

The above described MLP-network model was optimized using DE formalism; values of the various DE parameters used in the respective optimization simulations were:

- $F_{sc} = 0.65$, $Cr = 0.72$, $NP = 55$ and $t_{max} = 250$.

The optimal solutions searched by the DE methods are presented in columns 4 of Table 8.2.

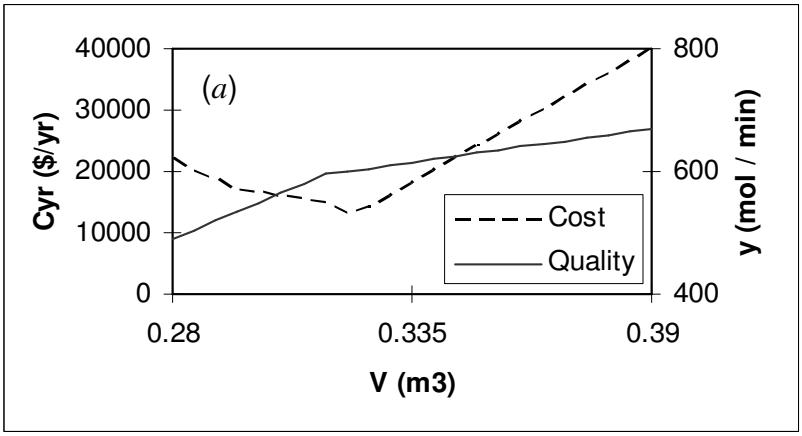
We have provided the results obtained by ANN-GA also in column 3 of Table 8.2 for comparison of the optimal results. The details of the ANN-GA technique applied on noisy data are available in Nandi et. al., (2001). The GA parameters used there were: population size, $N_{pop} = 30$, length of chromosome, $l_{chr} = 55$, crossover frequency, $P_{cross} = 0.95$, mutation probability, $P_{mut} = 0.01$ and maximum number of generations, $N_{gen}^{max} = 250$.

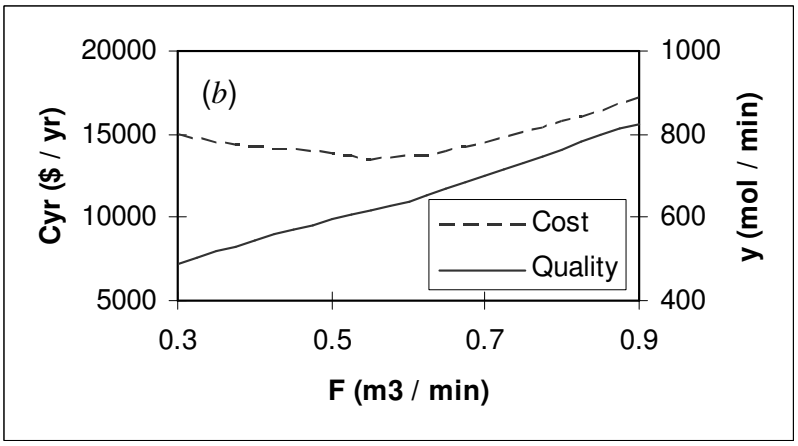
In the case of nonlinear objective functions, the decision surface may comprise several local minima with varying shapes and sizes. Thus, for a problem involving function minimization, it becomes important to obtain a solution that corresponds to the deepest local or the global minimum on the objective function surface. Stochasticity in the implementation procedures of the DE method to some extent helps in achieving the stated goal. Notwithstanding this fact, it was ensured during DE - implementation that the search space is thoroughly explored. This was done by using different pseudo-random number sequences (generated by changing the random number generator seed) for initializing real coded guess solution vector for DE – based optimization. Usage of different random initializations in essence helps in exploring different sub-sets of the decision space thereby locating the deepest local minimum on the decision surface. By mapping the decision surface, it is possible to verify whether the optimization algorithm has indeed captured a solution corresponding to the deepest local minimum. In the present case study, it is not possible to view the surface formed

by the objective function since the decision space is eleven-dimensional. We therefore resort to mapping the objective function in a single dimension only. For such mapping, the DE-optimized solution listed in column 2 of Table 8.2 has been considered. Accordingly, values of the objective function defined in Eq. 8.24 were evaluated by systematically varying the magnitude of a design or an operating variable while maintaining values of the remaining ten decision variables at their optimum. The six panels in Figure 8.5 depict the effect of variations in V, F, Q, C_A^0, C_B^0 and T^0 on the values of C_{yr} and μ_y. It is witnessed in all the six C_{yr} profiles that a single minimum exists and the DE-searched solution always lies at the valley bottom. In view of the efforts made towards locating a deepest local or the global minimum, it can thus be inferred that the DE has succeeded in fulfilling the objective.

8.6 DISCUSSION

An examination of the solutions (listed in Table 8.2) given by the hybrid ANN-DE and ANN-GA methodologies, it is observed that in all the cases the mean values of the quality variable (599.12, 604.35, 599.85 and 605.52) are very close to their desired magnitude (600 mol/min). Upon comparing with the RO solution (Table 8.2, column 5) obtained by Bernardo and Saraiva (1998), it is seen that the annual plant costs ($/yr) in respect of the DE-based solutions (13023.27, 13228.57) are a few percent lower than the corresponding RO solution value (14716 $/yr). Such a lowering of C_{yr} has been brought about either by the reduction in the control cost, C_c (see column 1 of Table 8.2), or by the reduction in the quality cost, C_q (see columns 2, 3, 4 of Table 8.2). It is noticed from the standard deviations (σ_y) of the quality variable that their magnitudes 11.43 and 10.67, pertaining to the solutions given by the DE formalism are smaller than the corresponding RO solution value of 16.8, although the respective μ_y values are marginally deviated from their desired magnitude of 600 mol/min. These results suggest that in respect of the DE solution, although a minor deviation from the designed mean value of the quality variable may occur, the variability around the mean value will be lower than that envisaged by the RO solution.





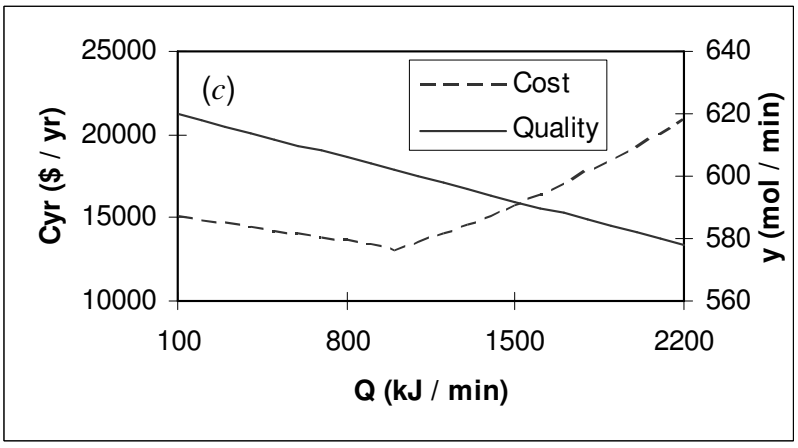
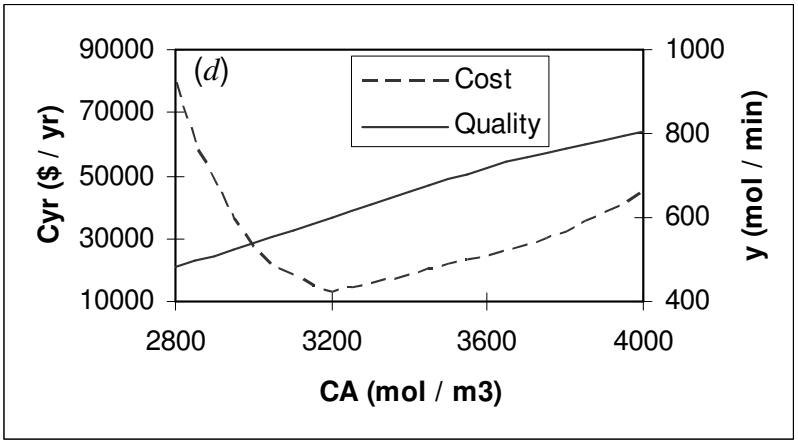
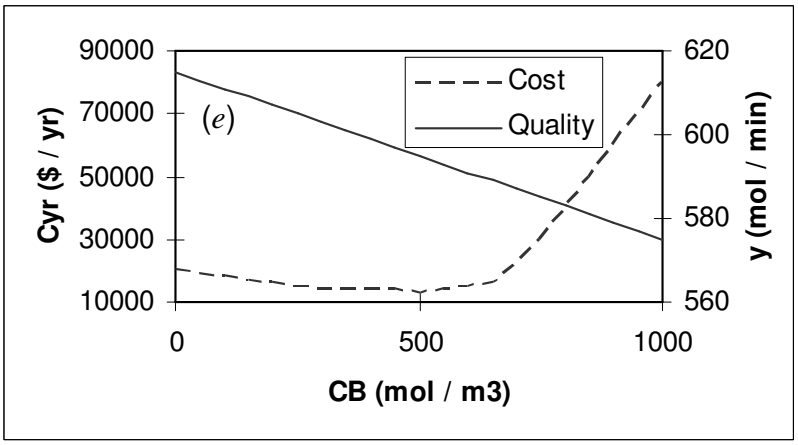


Figure 8.5: Plots showing the effect of variation in a process variable on (i) mean value of quality variable (y mol/min) and (ii) annual plant cost (C_{yr}, \$/yr). Panels ($a - f$) depict results corresponding to variations in V, F, Q, C_A^0, C_B^0 and T^0 respectively (continued in next page)





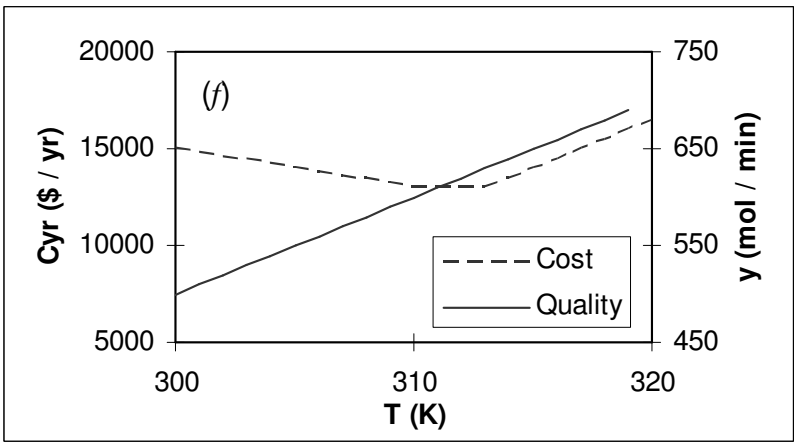


Figure 8.5: Plots showing the effect of variation in a process variable on (i) mean value of quality variable (y mol/min) and (ii) annual plant cost (C_{yr}, \$/yr). Panels (a – f) depict results corresponding to variations in V, F, Q, C_A^0, C_B^0 and T^0 respectively

A peculiar feature of the DE technique – which is shared by most stochastic optimization methods including GA - is that the obtained solution is influenced by the random number sequence used during their implementation. As a result, multiple optimization runs taking each time a different random number sequence (by changing random number generator seed) were performed for obtaining an overall optimal solution. In the case of DE-based optimization, it took 15 – 20 runs to arrive at the overall optimal solutions reported in Table 8.2. Here, it was noticed that the converged solution in each case were different ones – so we have to keep a track of them to obtain the overall global optimal solution.

Table 8.3: Process performances evaluated using phenomenological models with help of optimal conditions searched by differential evolution method

Quality variable and different components of overall cost	Noise-free data		Noisy data		RO Framework Solution (Ref: Bernardo and Saraiva, 1998)
	ANN-DE searched solution	DE searched optimal conditions applied to phenomenolo-gical model	ANN-DE searched solution	DE searched optimal conditions applied to phenomeno-logical model	
μ_y (mol/min)	600.347	597.49	605.52	608.17	600.0
σ_y (mol/min)	11.43	10.87	10.67	11.31	16.8
C_{eqp} ($/yr)	1014.5	1014.5	1052.92	1052.92	1064.00
C_{op} ($/yr)	9580.2	9580.2	9633.61	9633.61	9712.00
C_c ($/yr)	1451.0	1451.0	1598.03	1598.03	2105.00
C_q ($/yr)	977.57	1058.84	944.07	1077.33	1835.00
C_{yr} ($/yr)	13023.27	13104.54	13228.57	13361.89	14716.00

In this study, the CSTR process is again simulated with help of its phenomenological model (Eqs. A-8.1 to A-8.3) utilizing the optimized process conditions searched by DE technique (refer to Table 8.2). It is therefore possible to examine the actual process performance when the said process is operating under optimized operating conditions as predicted by the ANN-DE methodology. The results of such simulations are tabulated in Table 8.3. For comparison purposes we have provided ANN-DE searched optimized cost values along with those values obtained when optimized process operating conditions were applied to actual phenomenological model equations. Here phenomenological models are used in absence of real life plant data, ideally these are to be replaced with new sets of experimental runs at the optimized process operating conditions (please refer to chapter 2, section 2.4.7). It can be noticed from Table 8.3 that the solutions provided by the proposed search technique leads to lowering of annual plant cost appreciably [$/yr 13104.54 (clean data) or 13361.89 (noisy data) as compared to 14716.00] when compared to that of RO strategy (Bernardo and Saraiva, 1998). The phenomenological model predicts a different quality cost (C_q) as compared to ANN-DE as it depends on quality of the product (μ_y) and its standard deviation (σ_y) values. The magnitude of μ_y and σ_y as predicted by ANN and phenonomenological models are close but not exactly equal. Hence, there remains a small difference on overall annual plant cost (C_{yr}) provided by ANN-DE searched technique and the phenomenological models working on optimal operating conditions.

8.7 CONCLUSIONS

To summarize, this chapter presents a novel AI-based process modeling – optimization strategy combining an ANN-based process model with stochastic optimization formalism, namely, differential evolution (DE). The principal advantage of using neural networks for process modeling is that the model can be developed exclusively from process input-output data without invoking process phenomenology. Having built an ANN model, its input space comprising process input variables, is optimized using the DE technique. The DE optimization paradigm possess positive characteristics, such as: (i) only objective function measurements (and not the measurements of objective function derivatives) are needed in their optimization procedures, and (ii) the paradigm can tolerate noisy objective functions. It

is necessary to point out at this juncture that the magnitudes of various algorithmic parameters utilized in the development of the ANN models and implementation of the DE method, are problem-specific and barring a few (for instance, F_{sc} and Cr and NP of the DE algorithm), must be selected heuristically. Notwithstanding this fact, development of ANN-based process models is still an easier and cost-effective task as compared to the development of phenomenological models. The efficacy of the ANN-DE hybrid formalism has been demonstrated by considering a non-trivial optimization objective involving a CSTR, which in addition to the parameter design, also addresses the issue of tolerance design. The specific optimization objective considered was minimization of CSTR's total annual cost. In this case study, two ANN models were developed using noise-free and noisy steady-state process data. The input space of the ANN model consisting of CSTR's design and operating variables was then optimized using the DE method; simultaneously tolerances associated with the operating variables were also optimized. The solutions obtained thereby have been found to compare excellently with that given by a robust deterministic optimization formalism. The solutions obtained using the ANN-DE hybrid formalism were also compared with those obtained earlier using ANN-GA method and it was found that the ANN-DE have produced better solutions, as evident from lower annual plant costs (refer to Table 8.2). The ANN-DE approach presented here is sufficiently general and, therefore, can be employed for all types of process design and optimization problems. These strategies become considerably simple to implement when optimization objective involves only parameter design.

8.8 APPENDIX OF CHAPTER EIGHT

Appendix 8.1: CSTR model equations and parameter values

The steady-state model for the non-isothermal CSTR wherein two first order reactions, $A \rightarrow B \rightarrow C$ take place is given below. While equation (A1) refers to the energy balance, the remaining two account for the material balances of components *A* (Eq. A2) and *B* (Eq. A3), respectively.

$$\rho c_p F\left(T^0 - T\right) + k_{A_0} \exp\left(-E_A / RT\right)\hat{C}_A\left(-H_1\right)V + k_{B_0} \exp\left(-E_B / RT\right)\hat{C}_B\left(-H_2\right)V - Q = 0 \qquad \text{(A-8.1)}$$

$$F\left(C_A^0 - \hat{C}_A\right) - k_{A_0} \exp\left(-E_A / RT\right)\hat{C}_A V = 0 \qquad \text{(A-8.2)}$$

$$F\left(C_B^0 - \hat{C}_B\right) + k_{A_0} \exp\left(-E_A / RT\right)\hat{C}_A V - k_{B_0} \exp\left(-E_A / RT\right)\hat{C}_B V = 0 \qquad \text{(A-8.3)}$$

For generating representative steady-state data comprising 50 input-output patterns, following parameter values were considered: E_A = 3.64 × 10^4 J/mol, E_B = 3.46 × 10^4 J/mol, k_{A_0} = 8.4×10^5 min^{-1}, k_{B_0} = 7.6×10^4 min^{-1}, H_1 = − 2.12 × 10^4 J/mol, H_2 = − 6.36 × 10^4 J/mol,

ρ = 1180 kg/m^3, c_p = 3.2 × 10^3 J/(kg . K), and R = 8.314 J/(mol . K). Values of other model parameters, namely, *V*, *F*, *Q*, C_A^0, C_B^0 and T^0, describing design and operating variables were randomly chosen within the ranges specified in the main text.

Appendix 8.2: **Evaluation of the annual plant cost (C_{yr})**

The final expressions used for computing various CSTR costs have been given below. For further details please refer to Bernardo and Saraiva (1998).

A. The equipment cost C_{eqp} is computed using volume (V, m^3), as given by:

$$C_{eqp}\,(\$/\text{yr}) = 4199.55\left(\frac{V}{\pi}\right)^{0.6227} \tag{A-8.4}$$

B. The operating cost (C_{op}) includes utility cost (C_{util}) and pumping cost (C_{pump}). The C_{util} value is calculated using heat recovery rate, Q (J/min), and Q_N (= 2.54×10^7 J/min) according to:

$$C_{util}\,(\$/\text{yr}) = 1.145\begin{pmatrix} 7896 - 6327\,(Q/Q_N) + 4.764\times10^4\,(Q/Q_N)^2 \\ -1.022\times10^4\,(Q/Q_N)^4 \end{pmatrix} \tag{A-8.5}$$

and C_{pump} is evaluated using flow rate (F, m^3/min) as given by;

$$C_{pump}\,(\$/\text{yr}) = 13.8831\,(264.2F)^{0.8050} \tag{A-8.6}$$

C. The overall control cost (C_c) is obtained by summing control cost contributions of five operating variables (see Eq. 15). Using $a = 143.2$ and $b = 1.736$, C_c has been evaluated as:

$$\begin{aligned} C_c\,(\$/\text{yr}) &= 5a + b\left(\frac{\mu_F}{\sigma_F} + \frac{\mu_Q}{\sigma_F} + \frac{\mu_{C_A^0}}{\sigma_{C_A^0}} + \frac{\mu_{C_B^0}}{\sigma_{C_B^0}} + \frac{\mu_{T^0}}{\sigma_{T^0}}\right) \\ &= 716 + \left[1.736\times3.09\left(\frac{x_2}{x_5} + \frac{x_3}{x_8} + \frac{x_4}{x_9} + \frac{x_5}{x_{10}} + \frac{x_6}{x_{11}}\right)\right] \end{aligned} \tag{A-8.7}$$

where, μ_n and σ_n refer to the mean and standard deviation of the PDF pertaining to the nth operating variable.

D. The quality cost has been computed using Taguchi loss function (Taguchi, 1986) given as:

$$C_q\,(\$/\text{yr}) = k_l\left[(\mu_y - y^*)^2 + \sigma_y^2\right] \tag{A-8.8}$$

where μ_y and σ_y respectively denote the mean and standard deviation of N_{obs} number of quality variable values, $\{y\}$, obtained using the ANN-based CSTR model. y^* refers to the desired value (600 mol/min) of the quality variable, y, and k_l (=6.536) is the loss coefficient.

8.9 NOMENCLATURE

a : constants used in Eq. 8.15

b : constant used in Eq. 8.15

C_c : control cost, \$/yr.

C_{eqp} : equipment cost, \$/yr.

C_{op} : operating cost, \$/yr.

C_q : quality cost, \$/yr.

Cr : crossover constant

C_{yr} : annual plant cost in \$/yr

C_A^0: inlet concentration of reactant A in mol/m^3

C_B^0: inlet concentration of component B in mol/m^3

$\hat{C}_B$: steady-state concentration of B

E : tolerances associated with the process operating variables $[\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_N]^T$

$E(C_q)$: expected value of the quality cost

$E(C_{op})$: expected value of the operating cost

E_l : sum-of-squares error

E_{trn} : RMSE for the training set

E_{tst} : RMSE for the test set

F : flowrate in m^3 / min

F_{sc} : the weight applied to random differential scaling factor

f : nonlinear function approximated by the ANN model

$f(\mathbf{x})$: objective function

$J(\Phi)$: set of PDFs associated with operating variables Φ

k_l : quality loss coefficient

$L(y)$: PDF associated with quality variable y

N_H : number of nodes in hidden layer

N_I : number of nodes in input layer

N_O : number of nodes in output layer

N_{obs} : number of observations taken for sampling from Gaussian PDFs associated with uncertainty-involving operating variables

NP : the population size

o_l^i : actual output from the ith output node

Q : heat removal rate, kJ/min

RMSE : root-mean-squared-error

T^0 : inlet temperature in K

t : iteration index

V : volume of CSTR in m^3

$\mathbf{V}$: perturbed vector in differential evolution searching

W : weight matrix of ANN model

$\mathbf{X}$: K-dimensional vector representing set of decision variables

x_k^L : lower bound of x_k

x_k^U : upper bound of x_k

y : process output variable determining product quality

y^* : desired value of quality variable

Greek Symbols

α_m : momentum coefficient

β : constant, 0.101

ε_n : tolerance associated with nth operating variable

Φ : N-dimensional operating process variable vector, $\Phi = [\phi_1, \phi_2, \ldots, \phi_n, \ldots, \phi_N]^T$

$\hat{\Phi}$: operating space region

$\hat{\Phi}_{\phi_n}$: window for the nth operating variable

ϕ_n^l : lower bound on the nth operating variable

ϕ_n^u : upper bound on the nth operating variable

η : constant, 0.602

η_l : learning rate

μ_n : mean value of the nth operating variable

μ_y : mean of the quality variable

σ_n : standard deviation of the Gaussian PDF

σ_y : standard deviation of N_{obs} number of y values

ξ_i : fitness score for the ith string

Ψ : M-dimensional design vector, $\Psi = [\psi_1, \psi_2, \ldots, \psi_m, \ldots, \psi_M]^T$

8.10 REFERENCES

1. Babu, B. V. and Sastry, K. K. N. "Estimation of Heat Transfer Parameters in a Trickle-Bed Reactor Using Differential Evolution and Orthogonal Collocation", ***Comp.& Chem. Engg.***, **23**, 327 – 339 (1999).
2. Babu, B. V., Pallavi, G., Chakole, J. H. and Syed M. "Multiobjective Differential Evolution (MODE) for Optimization of Adiabatic Styrene Reactor", ***Chem. Engg. Sci***, **60**, 4822 – 4837 (2005).
3. Babu, B. V. and Angira, R. "Modified Differential Evolution (MDE) for Optimization of Non-linear Chemical Processes", ***Comp. & Chem. Engg.***, **30**, 989 – 1002 (2006).
4. Babu, B. V. and Munawar, S. A. "Differential Evolution Strategies for Optimal Design of Shell-and-Tube Heat Exchangers", ***Chem. Engg. Sci***, **62**, 3720 – 3739 (2007).
5. Bernardo, F. P. and Saraiva, P. M. "Robust Optimization Framework for Process Parameter and Tolerence Design", ***AIChE Jour.***, **44**, 2007 - 2017 (1998).
6. Diwekar, U. M. and Kalagnanam, J. R. "Robust Design Using an Efficient Sampling Technique", ***Comp. & Chem. Engg.***, **20**, S389 – S394 (1996).
7. Diwekar, U. M. and Kalagnanam, J. R. "An Efficient Sampling Technique for Optimization Under Uncertainty" ***AIChE Jour.***, **43**, 440 - 4449 (1997a).
8. Diwekar, U. M. and Kalagnanam, J. R., "An Efficient Sampling Technique for Off-line Quality Control", ***Technometrics***, **39**, 308 – 317 (1997b).
9. Diwekar, U. M. and Rubin, E. S. "Stochastic Modeling of Chemical Processes", ***Comp. & Chem. Engg.,*** **15**, 105 - 116 (1991).
10. Diwekar, U. M. and Rubin, E. S. "Parameter Design Methodology for Chemical Processes Using a Simulater", ***Ind. & Eng. Chem. Res.***, **33**, 292 - 304 (1994).
11. Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H."User's Guide for NPSOL: A Fortran Package for Nonlinear Programming," Technical Report SOL 86-2, Systems Optimization Laboratory, Stanford University, Stanford, California (1986).
12. Nandi, S., Ghosh, S., Tambe, S. S. and Kulkarni, B. D. "Artificial Neural-Network-Assisted Stochastic Process Optimization Strategies, ***AIChE Jour.***, **47**, 126 – 141 (2001).
13. Onwubolu, G. C. and Babu, B. V., ***New Optimization Techniques in Engineering***, in Series of Studies in Fuzziness and Soft Computing, Vol. 141, Springer Verlag, Brelin Heidelberg, Germany (2004).

14. Pistikopoulos, E. N. “Uncertainty in Process Design and Operations”, ***Comp. & Chem. Engg.***, **19**, S553 – S5559 (1995).
15. Pistikopoulos, E. N. and Ierapetritou, M. G. “Novel Approach for Optimal Process Design and Operations”, ***Comp. & Chem. Engg.,*** **19**, 1089 - 1097 (1995).
16. Price K. V. “An Introduction to Differential Evolution”, in ***New Ideas in Optimization***, Eds. Corne, D., Dorigo, M. and Glover, F., McGraw Hill, London (UK), pp. 79 – 108 (1999).
17. Rumelhart, D., Hinton, G. and Williams, R. “Learning Representations by Backpropagating Errors”, ***Nature***, **323**, 533 - 536 (1986).
18. Srinivas, M. and Rangaiah, G. P. “A Study of Differential Evolution and Tabu Search for Benchmerk, Phase Equilibrium and Phase Stability Problems”, ***Comp. & Chem. Engg.,*** **31**, 760 - 772 (2007).
19. Storn, R. and Price, K. V., “Differential Evolution – a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces” Technical Report TR–95- 012, ICSI, March 1995, Available via ftp://ftp.icsipberkeley.edu/pub/techreports/1995/tr-95-012.ps.z (1995)
20. Storn, R. and Price K. V. “Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces”, ***Jour. of Global Optim.***, **11**, 341 – 359 (1997).
21. Tambe, S. S., Kulkarni, B. D. and Deshpande, P. B. ***Elements of Artificial Neural Networks with Selected Applications in Chemical Engineering, and Chemical & Biological Sciences***, Simulation & Advanced Controls Inc., Louisville, KY (1996).
22. Taguchi, G., ***Introduction to Quality Engineering***, Amer. Supplier Inst., Dearborn, MI (1986).
23. Venkatasubramanian, V. and Sundaram, A., “Genetic Algorithms: Introduction and Applications”, ***Encyclopedia of Computational Chemistry***, Wiley, Chichestar, UK (1998).

CHAPTER 9

CONCLUSIONS AND FUTURE SCOPE

9.1 OVERALL CONCLUSIONS

Most chemical processes are inherently nonlinear. Associated factors such as heat and mass transfer make them more difficult and challenging to analyze using conventional phenomenological methodologies. Development of a truly generalized phenomenological model is a time-consuming, tedious and costly (due to the involved experimentation) activity and often requires art and skills of a domain expert to understand and mathematically represent the physico-chemical phenomenon underlying a chemical process. Another difficulty often faced by the phenomenological models is that as the underlying phenomena get more complex, the corresponding models also require increased complexity which necessitates computation-intensive procedures for their solution. As a result, it becomes difficult to use complex phenomenological models in real time, online process applications. Accordingly this thesis presents a number of strategies wherein a wide variety of AI paradigms such as Artificial Neural Networks (ANN), Generalized Regression Neural Networks (GRNN), Support Vector Regression (SVR), Fuzzy Neural Networks (FNN), Genetic Algorithms (GA), Differential Evolution (DE) along with statistical tools such as Principal Component Analysis (PCA) and Partial Least Squares (PLS) and their multiway counterparts have been employed for chemical process modeling, optimization, monitoring and control purposes. The stated strategies have been successfully validated on a number of chemical, biochemical, petrochemical and polymer processes.

In Chapter 2, two hybrid process modeling and optimization strategies integrating artificial neural networks / support vector regression with the genetic algorithms have been employed for modeling and optimization of benzene isopropylation pilot plant scale catalytic process. First a process model is developed either using an ANN or SVR methodology. Subsequently the input space of the model is optimized using GAs in order to maximize the process performance (i.e., yield and selectivities of desired product cumene). In both the hybrid approaches process modeling and optimization is done exclusively using historic process data. Using ANN-GA and SVR-GA strategies separately, a number of optimized sets of operating conditions, which simultaneously maximize the yield and selectivity of cumene were obtained. It was observed that the best sets of process operating conditions given by the two strategies were similar. Finally, the optimized solutions given by the ANN-GA method

when subjected to experimental verification, matched the maximized cumene yield and selectivity values with excellent accuracy (mostly within 99 %).

In Chapter 3, a hybrid strategy integrating PCA and generalized regression neural networks has been presented for modeling and monitoring of batch processes. The proposed PCA-GRNN strategy uses PCA for reducing the dimensionality of the process input space and the first few principal component scores that explain a large amount of variance in the input data are used to develop a GRNN model correlating inputs and outputs. Principal advantages of GRNN-based models are: (i) ability to approximate nonlinear input-output relationships efficiently, (ii) a model can be constructed exclusively from the historic process input-output data, (iii) since there exists only one adjustable parameter in its formulation, and its training being a one-step procedure, the development of an optimal GRNN model is computationally inexpensive when compared with other neural network paradigms such as the error-back-propagation and radial basis function networks, and (iv) potential to undergo on-line training. The effectiveness of the PCA-GRNN strategy for modeling and monitoring of batch processes has been successfully demonstrated by conducting two case studies involving penicillin production and protein synthesis. The results obtained thereby, using both clean and noisy data (with upto Gaussian 5% noise), demonstrate that the PCA-GRNN methodology is an attractive formalism for modeling and monitoring of nonlinearly behaving batch processes.

A GRNN-based process modeling and monitoring strategy to forecast dynamics of a bio-process in the Chapter 4. The strategy has been shown to provide excellent results when applied to a fed batch bio-reactor for the production of ethanol. The advantage of the strategy is that : (i) it can make accurate forecast even when the process behavior is nonlinear and (ii) the network training is much faster compared to the classical error-back-propagation strategy for ANN-training. These advantages enable the proposed strategy as an attactive alternative for implementation in the on-line modeling and monitoring of chemical and biochemical processes.

In Chapter 5, a hybrid strategy integrating the PCA and fuzzy neural network has been presented for modeling and monitoring of a batch process. The proposed PCA-FNN strategy uses PCA for dimensionality reduction of the process input space (operating variables) and the first few principal component scores that explain a large amount of variance in the input data are used to develop a fuzzy neural network model correlating process inputs and outputs.

Principal advantages of the hybrid model are: (i) it can approximate nonlinear input-output relationships efficiently, and (ii) the model can be constructed exclusively from the historic process input-output data. The effectiveness of the hybrid PCA-FNN strategy for modeling and monitoring of batch processes has been successfully demonstrated in a case study comprising the bio-process of protein synthesis. The results obtained indicate that the hybrid methodology is an attractive formalism for modeling and monitoring of nonlinearly behaving batch processes.

The ANN-GA hybrid modeling-optimization strategy is employed for optimizing a batch polymerization process in Chapter 6. A sufficiently generalized ANN model is first developed for the isothermal polymerization of methyl methacrylate (MMA) operating in the batch mode. This model is subsequently utilized for optimizing the operating condition variables using GAs. The objective function to be minimized in the optimization represents process operating cost, which in turn depends on the batch time and initiator cost. The task of optimization was to reduce the residual monomer concentration to the desired level while simultaneously producing the polymer with the desired average number molecular weight. The results given by the ANN-GA scheme have been compared with those available in the open literature and they reveal a significant improvement over the published results. Specifically, the ANN-GA hybrid formalism could bring about upto 4.8 % reduction in the process operating cost.

In Chapter 7, a gain scheduling based control strategy, which does not require a phenomenological model for implementing the control action has been demonstrated for controlling an unstable process. The sole tunable parameter ε_0, involved in the strategy is optimized using the GA formalism. Efficacy of the control strategy has been successfully validated for regulating the dynamics of a fluidized bed reactor (FBR) exactly at an unstable steady-state (USS). The performance of the controller in presence of deterministic and stochastic load disturbances has also been studied. The presented methodology has application potential in controlling oscillatory, quasi-periodic and even chaotic dynamic behavior.

Finally, Chapter 8 introduces a novel AI-based hybrid process modeling – optimization strategy combining an ANN-based process model with the stochastic optimization formalism, namely, *differential evolution* (DE). Having built an ANN model

based on the historic process data, the input space (i.e., process operating variables) is optimized using the DE technique. The DE optimization paradigm possesses positive characteristics, such as: (i) only objective function measurements (and not the measurements of the objective function derivatives) are needed in their optimization procedures, and (ii) the paradigm can tolerate noisy objective functions. The efficacy of the ANN-DE hybrid formalism has been demonstrated by considering a non-trivial optimization objective involving a CSTR, which in addition to the parameter design, also addresses the issue of tolerance design. The specific optimization objective considered was minimization of CSTR's total annual cost. In this case study, two ANN models were developed using noise-free and noisy (5% Gaussian noise) steady-state process data. The input space of the ANN model consisting of CSTR's design and operating variables was then optimized using the DE method; simultaneously tolerances associated with the operating variables were also optimized. The solutions obtained thereby have been found to compare favorably with those given by a robust deterministic optimization formalism. The solutions obtained using the ANN-DE hybrid formalism were also compared with those obtained earlier using ANN-GA method and it was found that the ANN-DE method has produced better solutions, as evident from the lower annual plant cost. The ANN-DE approach presented here is sufficiently general and, therefore, can be employed for a variety of process design and optimization problems.

9.2 SUGGESTIONS FOR FUTURE RESEARCH

This thesis presents a wide variety of artificial intelligence (AI) based methodologies for process modeling, monitoring, optimization and control purposes. All of the methodologies are generic in character and have additional potential applications in process engineering tasks such as scheduling, parameter estimation, data reconciliation, process identification, soft-sensor development, fault detection and diagnosis and model based control. These chemical engineering applications could be explored using strategies provided in the thesis. New AI-based optimization paradigms such as particle swarm and artificial immune system are continuously being developed. It is necessary to employ these paradigms individually and in combination with the AI-based modeling formalisms for chemical engineering tasks and evaluate their performance with the existing approaches.

List of Publications

A. Publications Received from the Work Presented in the Thesis:

1. **Nandi, S.**, Cheema, J. S., Tambe, S. S. and Kulakrni, B. D. "Predicting Batch Process Behavior Using Generalized Regression Neural Networks", ***Proceedings National Seminar on Instrumentation and Control in the Chemical Industry* (*ICCI 2K2*)**, pp. 7 – 12, Laxminarayan Institute of Technology (LIT), Nagpur, India (2002).
2. **Nandi, S.**, Tambe S. S. and Kulkarni, B. D. "Controlling Nonlinear Dynamics of a Fluidized Bed Reactor", Presented at Indian Chemical Engineering Congress **(*CHEMCON 2002*)** in Hyderabad, India. Available in CD-ROM
3. **Nandi, S.**, Rahman, I., Kulkarni, S. G., Tambe, S. S. and Kulkarni, B. D. "Hybrid PCA-Fuzzy Neural Network Formalism for Batch Process Monitoring", ***Proceedings of the International Conference of Chemical and Bioprocess Engineering***, University Malaysia Sabah, Vol. 2, pp. 773 – 779 (2003).
4. **Nandi, S.**, Badhe, Y., Lonari, J., Sridevi, U., Rao, B. S., Tambe,S. S. and Kulkarni, B. D. "Hybrid Process Modeling and Optimization Strategies Integrating Neural Networks / Support Vector Regression and Genetic Algorithms: Study of Benzene Isopropylation on HBeta Catalyst", ***Chem. Engg. Jour.***, **97**, 115 – 129 (2004).
5. Kulkarni, S. G., Chaudhary, A. K., **Nandi**, **S.** Tambe, S. S. and Kulkarni, B. D. "Modeling and Monitoring of Batch Processes Using Principal Component Assisted Generalized Neural Networks", ***Biochem. Engg. Jour.* 18**, 193 - 210 (2004).
6. **Nandi**, **S.**, Tambe, S. S. and Kulkarni, B. D. "Artificial Neural Networks Assisted Genetic Algorithms Formalism for Optimization of Batch Polymerization Reactor", Presented at Indian Chemical Engineering Congress **(*CHEMCON 2004*)** in Mumbai, India. Available in CD-ROM.
7. **Nandi, S.**, Tambe, S. S. and Kulkarni, B. D. "Novel Optimization Strategy Integrating Artificial Neural Network and Differential Evolution for Process Parameters and Tolerances", Manuscript under preparation, to be communicated to ***Chemical Engineering Journal*** shortly.

B. Other Research Publications:

1. **Nandi, S.**, Rahman, I., Tambe, S. S. and Kulkarni, B. D. "Process Identification Using Genetic Programming: A Case Study Involving Fluidized Catalytic Cracking (FCC) Unit", pp. 195-201 in ***Petroleum Refining and Petrochemical Based Industries in Eastern India*** Edited by R. K. Saha, B. R. Maity, D. Bhattacharyya, S. Ray, S. Ganguly, S. L. Chakraborty, Allied Publishers Ltd., New Delhi (2000).

2. Rahman, I., **Nandi**, **S.**, Tambe, S. S., and Kulkarni, B. D. "Optimization of Fuzzy Logic Controller Using Genetic Algorithms", Vol. I, pp. PIC 33-36, Technical Sessions Transcription of Indian Institute of Chemical Engineering Congress (***CHEMCON 2000***), Calcutta.

3. **Nandi, S.**, Ghosh, S., Tambe, S. S. and Kulkarni, B. D. "Artificial Neural-Network-Assisted Stochastic Process Optimization Strategies, ***AIChE Jour.***, **47**, 126 – 141 (2001).

4. Sonawane, P. D., **Nandi**, **S.**, Tambe, S. S., Kumar, A. and Kulkarni, B. D. "Solvent Activity Prediction of Electrolyte Solutions Using Neural Networks", TSM –182, Proceedings (CDROM) of Indian Chemical Engineering Congress (***CHEMCON 2001***) Central Leather Research Institute, Chennai.

5. **Nandi, S.**, Mukherjee, P., Tambe, S. S. Kumar, R. and Kulkarni, B. D. "Reaction Modeling and Optimization Using Neural Networks and Genetic Algorithms: Case Study Involving TS-1 Catalyzed Hydroxylation of Benzene", ***Ind. Engg. Chem. Res***., **41**, 2159 - 2169, (2002).

6. Kalgaonkar, R., **Nandi**, **S.** Tambe, S. S. and Jog J. "Analysis of Viscoelastic Behavior and Dynamic Mechanical Relaxation of Copolyester based Layered Silicate Nanocomposites using Havriliak-Negami Model", ***Jour. of Polym. Sci. Part B: Polym. Phys.*** **42**, 2657 - 2666 (2004).