CHAPTER 1

INTRODUCTION

1.1 General

This chapter includes a brief introduction about the CSTR (continuously stirred tank reactor) model, basic theory, adaptive control, parameters for PID controller and finally biological inspired optimization technique i.e. Particle Swarm Optimization. The basic idea behind Adaptive control is that it can adjust its parameter automatically in such a way as to compensate for variations in the characteristics of the process it control and The basic ideas for optimization techniques depend upon their respective foraging behavior and the solution can be made as per the CSTR model.

1.2 Continuously Stirred Tank Reactor (CSTR)

In industries now a day the control of chemical process is important craft. Mostly all the chemical process are highly nonlinear in nature this cause instability of the process. Chemical reactors are ones of the most important plants in chemical industry. Their operation, however, is corrupted with various uncertainties. Some of them arise from varying or not exactly known parameters, as e.g. reaction rate constants, heat transfer coefficients. In other cases, operating points of reactors vary or reactor dynamics is affected by various changes of parameters or even instability of closed loop control systems. The main difficulty in tuning of control is due to the disturbances and parameter uncertainties. Application of robust control approach can be one of way to overcome all these problems.

The design of a nonlinear feedback controller is analyzed for temperature and concentration control of Continuous Stirred Tank Reactors (CSTRs) which have strong nonlinearities. Consequently, we need to introduce a control mechanism that will make the proper changes on the process to cancel the negative impact that such nonlinearities may have on the desired operation of chemical plant [2, 4].

1.3Paradigm and theory of controllers

In control engineering a controlled system is primarily characterized by itsdynamic behavior which also determines the scope and quality required to solve a controltask. The dynamic behavior of the system depends on the system parameters, inputvariables, output variables and operating conditions. The controller maintains the output at desired value by means of a control action. Any deviation of output from the reference input is detected by an error detector. The error thus detected is used as actuating signal for control action through a controller. The various control methods differ from each other depending on the factors based on which action is taken. Various kind of control methods that are availed and employed are discussed at length in this section.

1.3.1 The ‘P’ controller

Proportional or ‘P’ controller is one of the basic controllers in control engineering. In ‘P’ control the actuating signal for the control action is proportional to the error signal. The proportional control is probably the easiest feedback control for implementation, where the error signal being the difference between the reference input signal and theactual signal (feedback) obtained. A constant denoted as ‘Kp’ called as proportional gain is multiplied with error signal, thus the controller actuating signal is obtained, which inturn is fed to the drive. Due to presence of fixed disturbance, a DC offset is observed in‘P’ control. A tradeoff is to be made between the maximum overshoot and steady stateerror. As increase in gain value is desirable to reduce steady state error, the increasedgain also increase the maximum overshoot value. So, there is need for further improvement in terms of eliminating steady state error.

1.3.2 The ‘PI’ controller

The limitation of a DC offset in ‘P’ control can be overcome by adding anintegral term of error signal that provides desired DC stiffness to the system. Theintegral gain represented as ‘K_i’, increased value gives more stiffness at the cost of large overshoot. A sufficient value of ‘K_i’ gain in the controller will eliminate the DC offset, as the presence of even a small value of DC offset will make the integral term large. Although integral gain adds precision to the close loop control but it lacks in the wind up function that is needed to control the gain value during saturation. An improvement in the steady state error is introduced by ‘PI’ controller in the system dynamics.

1.3.3 The ‘PD’ controller

For a derivative control action the actuating signal consist of proportional error signal added with derivative of error signal. The gain of derivative term is represented as‘K_d’ called as derivative gain, which induces a phase lead of 90 degrees in the loop. The derivative term can be represented as a difference term divided by sample time. The difference term is the last value of the position minus the current position value. The difference term divided by time gives a rough estimate of velocity, which helps the future position prediction. The derivative terms allow system's responsiveness to increase but make it more susceptible to noise. A noise is introduced in the system, if the change in position is constant and the sample time varies from sample to sample. The derivative action give high gain at high frequency this improves gain margin but affect the system adversely by adding gain margin at phase crossover frequency, which is typically at high frequency. Hence, the system suffers from noise that is spread evenly across the frequency spectrum and eventually worse in high frequency range. A low pass filter following the controller would eliminate the sample induced irregularities and oscillation in the system especially at high frequency. The control commands and plants outputs are usually of low frequency, so the presence of high pass filter does not affect their functioning.

1.3.4 The 'PID' controller

Conventional proportional integral derivative or 'PID' controllers are most commonly used controller. It can be obtained by combination of 'PI' with 'D', or 'PD'with 'I' control action. The three terms 'P', 'I' and 'D' work on the error value interpreted in terms of time, as 'P' depends on present value of error, 'I' depends on the summation of past values and 'D' depends on the rate of change of error predicting the future error. Each of the control action has different effect on the system and weighted sum of all the three actions is used by the PID controller. The PID control action can be divided in two zone based on the frequency, at lower frequency the 'K_p' gain dominated and at high frequency there is contribution from two gains 'K_p' and 'K_i'. The derivative action helps in setting 'K_p' at higher side then that can be set generally. All this actions work in tandem with in the close loop, depending on the command and the feedback signal of the system. Although the PID is superior to the P, PI, PD controller in term of system dynamics response but it come with an expense of increased sensitivity towards the changes in plant model. The rigorous task of tuning a PID controller is also a matter of great concern [5].

1.4 Adaptive Control

An adaptive controller is a controller that can modify its behavior in response to changes in the dynamics of the process and the disturbances. Adaptive control can be considered as a special type of nonlinear feedback control in which the stages of the process can be separated in two categories, which can be change at different rates. The slowly changing states are viewed as parameters with a fast time scales for the ordinary feedback and a slower one for updating regulator parameters. One of the goal of adaptive control is to compensate for parameter variations, which may occur due to nonlinear actuators, changes in the operating conditions of the process and non-stationary disturbances acting on the process.

In common sense, ‘to adapt’ means to change a behavior to conform to new circumstances.Intuitively,an adaptive controller is thus a controller that can modify its behavior in response to the changing dynamics of the process and the character of the disturbances. The core element of all the approaches is that they have the ability to adapt the controller to accommodate changes in the process. This permits the controller to maintain a required level of performance in spite of any noise or fluctuation in the process .An adaptive system has maximum application when the plant undergoes transitions or exhibits non-linear behavior and when the structure of the plant is not known. Adaptive is called a control system, which can adjust its parameter automatically in such a way as to compensate for variations in the characteristics of the process it control.

An adaptive control system can be thought of as having two loops.one loops is normal feedback loop with the process and controller. The other loop is a parameter adjustment loop. The parameter adjustment loop is slower than the normal feedback loop [33].

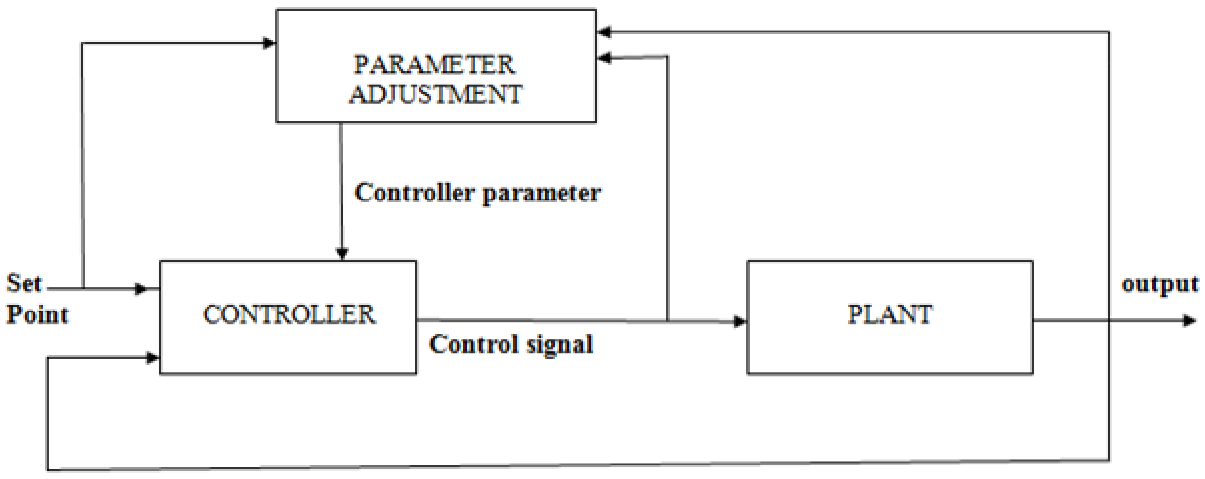

Fig. 1.1 Block diagram of an Adaptive Controller

As compared to fixed gain PID controllers adaptive controllers are very effective to handle the situation where the parameter variations and environmental changes are frequent. The controller parameters are adjusted to give desired closed loop performance.

The adaptive controller maintains constant dynamic performance in the presence of unpredictable and immeasurable variations. Adaptive control changes the control algorithm coefficients in real time to compensate for variations in the environment or in the system itself. It also varies the system transfer function according to situation.

Intuitively an adaptive system has maximum application when the plant undergoes transitions or exhibits non-linear behavior and when the structure of the plant is not known. This permits the Controller to maintain a required level of performance in spite of any noise or fluctuation in the process. The basic PID controllers have difficulty in dealing with problems that appear in complex non-linear processes. This work presents a practical non-linear adaptive and PID controller that deals with these non-linear difficulties

Now-a-days the adaptive control schemes are making their place where the conventional control system is not able to cope-up with the situation, like

1) Loads, inertias and other forces acting on system change drastically.
2) Possibility of unpredictable and sudden faults.
3) Possibility of frequent or unanticipated disturbances.

1.4.1 Adaptive Schemes

In the adaptive control there are 4 types of schemes:

1) Gain scheduling
2) Model reference adaptive control
3) Self tuning regulator
4) Dual control

1.4.1.1 Gain Scheduling:

In many cases it is possible to find measurable variables that correlate well with changes inprocess dynamics. These variables can then be used to change the controller parameters. This approach is called gain scheduling because the scheme was originally used to measure the gain and then change, thatis, schedule, thecontroller to compensate for changes in the process gain. A block diagram of system with gain scheduling is shown in fig 1.2.the system can be viewed as having two loops. There is an inner loop composed of the process and the controller and an outer loop that adjusts the controller parameters on the basis of the operating conditions.

Gain scheduling can be regarded as a mapping from process parameters to controller parameters. It can be implemented as a function or table lookup [9].

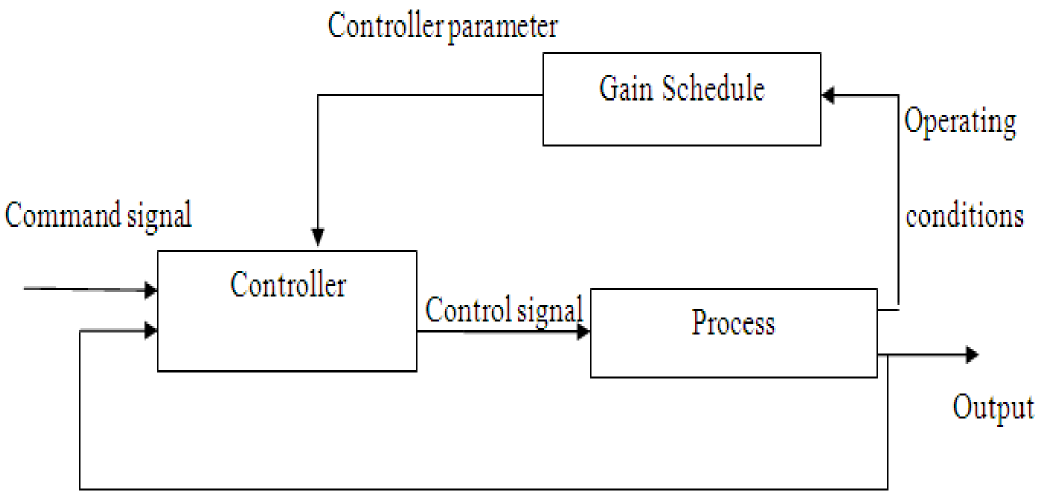

Fig. 1.2 Block diagram of a system with Gain Scheduling

1.4.1.2 Model Reference Adaptive System (MRAS)

The model reference adaptive system was originally proposed to solve a problem in which the performance specifications are given in terms of a reference model. This model tells how the process output ideally should respond to the command signal. A block diagram of the system is shown in fig.1.3.the controller can be thought of as consisting of two loops. The inner loop is an ordinary feedback loop composed of the process and the controller. The outer loop adjusts the controller parameters in such a way that the error, which is the difference between process output y and model output y_m is small

The key problem with MRAS is to determine the adjustment mechanism so that a stable system, which brings the error to zero, is obtained [9].

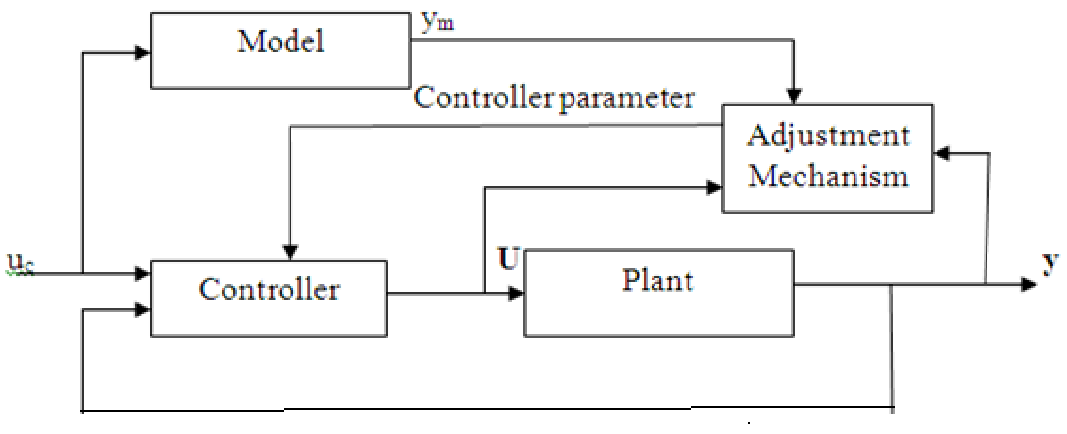

Fig.1.3 Block diagram of a Model Reference Adaptive System

1.4.1.3 Self Tuning Regulator

In this method the estimates of the process parameters are updated and the controller parameters are obtained from the solution of a design problem using the estimated parameters. Block diagram of self tuning regulator is shown in fig.1.4 the adaptive controller can be thought of as being two loops. The inner loop consists of the process and an ordinary feedback controller. The parameters of the controller are adjusted by the outer loop, which is composed of a recursive parameter estimator and a design calculation. It is sometimes not possible to estimate the process parameters without probing control signals or perturbations. The system may be viewed as an automation of process modeling and design, in which the process model and the control design are updated at each sampling period. A controller of this construction is called a self tuning regulator (STR) to emphasize that the controller automatically tunes its parameters to obtain the desired performance of closed loop system [9].

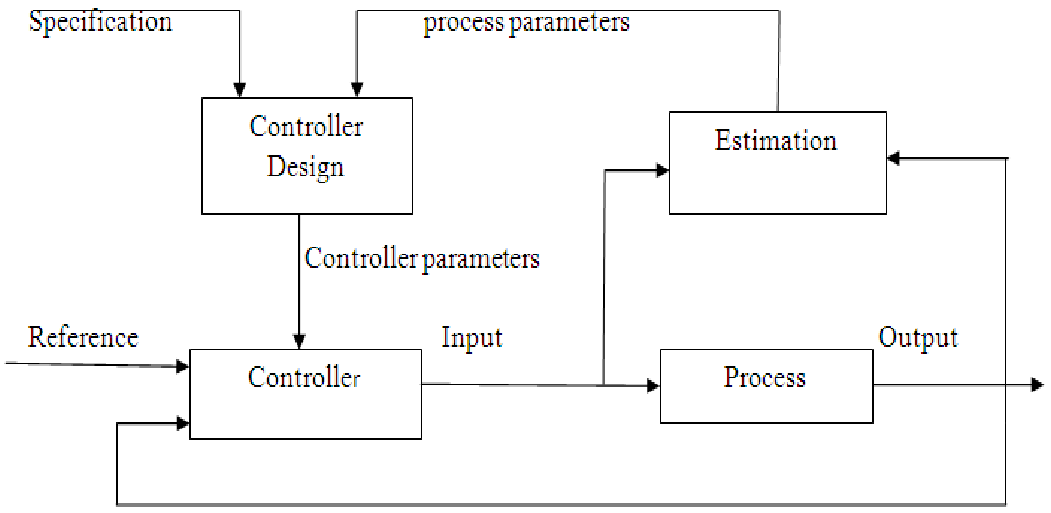

Fig.1.4 Block diagram of Self Tuning Regulator

1.4.1.4 Dual Control

Dual control theory uses the concept of abstract problem formulation and optimization theory. A major consequence of dual control is that uncertainties in the estimated parameters will be taken into account in the controller. The controller will also take special action when it has poor knowledge about the process. It attempts to drive the output to its desired value, but it will also introduce perturbations when the parameters are uncertain. This improves the quality of estimates and the future performance of closed loop system. It gives the correct balance between maintaining good control and small estimation errors [9].

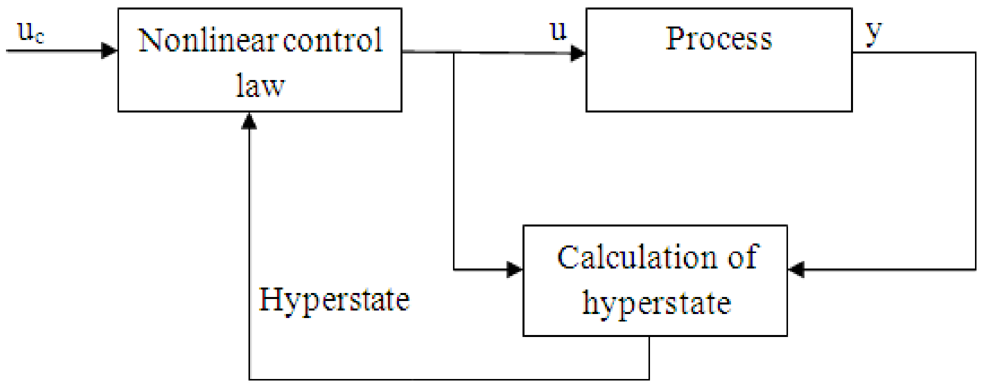

Fig.1.5 Block diagram of Dual Control

1.4.2 The Adaptive Control Problem

An adaptive controller has been defining as a controller with adjustable parameter and a mechanism for adjusting the parameters. The construction of an adaptive controller thus contains the following steps [10,13,15].

1) Characterize the desired behavior of closer loop system.
2) Determine a suitable control law with adjustable parameters.
3) Find a mechanism for adjusting the parameter.
4) Implement the control law.

In this thesis Model Reference Adaptive Control method has been studied.

1.5 Bio inspired Optimization:

An optimization algorithm is a numerical method or algorithm for finding the maxima or the minima of a function operating with certain constraints. Bio inspired optimization is a successor of artificial intelligence relying on Evolutionary computation, which is a famous optimization technique. Bio inspired intelligence combines elements of learning; adaptation and evolution to create programs that are, in some sense, intelligent. Bio inspired optimization research does not reject statistical methods, but often gives a complementary view. Bioinspired optimization finds its fundamental application in the area of fitness function design, methods for parameter control, and techniques for multimodal optimization. The importance of Bio inspired optimization lies in the fact that these techniques often find optima in complicated optimization problems more quickly than the traditional optimization methods.

The real beauty of bio inspired algorithms lies in the fact that it receives its sole inspiration from nature. They have the ability to describe and resolve complex relationships from intrinsically very simple initial conditions and rules with little or no knowledge of the search space Nature is the perfect example for optimization, because if we closely examine each and every features or phenomenon in nature it always find the optimal strategy, still addressing complex interaction among organisms ranging from microorganism to fully fledged human beings, balancing the ecosystem, maintaining diversity, adaptation, physical phenomenon like river formation, forest fire ,cloud, rain .etc..Even though the strategy behind the solution is simple the results are amazing. Nature is the best teacher and its designs and capabilities are extremely enormous and mysterious that researchers are trying to mimic nature in technology. Also the two fields have a much stronger connection since, it seems entirely reasonable that new or persistent problems in computer science could have a lot in common with problems nature has encountered and resolved long ago. Thus an easy mapping is possible between nature and technology. Bio inspired computing has come up as a new era in computing encompassing a wide range of applications, covering all most all areas including computer networks, security, robotics, bio medical engineering, control systems ,parallel processing ,data mining, power systems, production engineering and many more.

Formulating a design for bio inspired algorithms involves choosing a proper representation of problem, evaluating the quality of solution using a fitness function and defining operators so as to produce a new set of solutions [21,22].

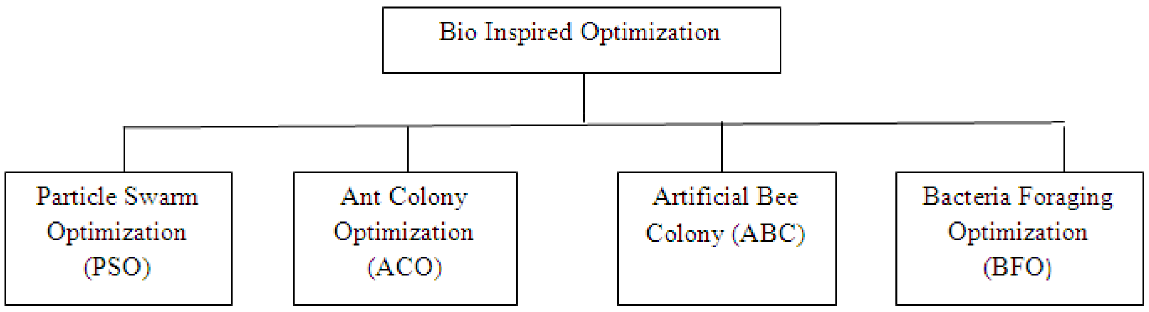
Bio Inspired Optimization
Particle Swarm Optimization (PSO)
Ant Colony Optimization (ACO)
Artificial Bee Colony (ABC)
Bacteria Foraging Optimization (BFO)

Fig.1.6 Types of Bio Inspired Optimization Techniques

In this thesis Particle Swarm Optimization has been studied.

1.5.1 Particle Swarm Optimization

Particle swarm optimization, first developed by Kennedy and Eberhart, is one of the modern heuristic algorithms. It was inspired by the social behavior of bird and fish schooling, and has been found to be robust in solving continuous nonlinear optimization problems. PSO is a robust stochastic evolutionary computation method based on the movement of swarms looking for the most fertile feeding location [33]. Particle swarm optimization is an extremely simple algorithm that seems to be effective for optimizing a wide range of functions. It is viewed as a mid-level form of A-life or biologically derived algorithm, occupying the space in nature between evolutionary search, which requires eons, and neural processing, which occurs on the order of

milliseconds. Social optimization occurs in the time frame of ordinary experience - in fact, it is ordinary experience. In addition to its ties with A-life, particle swarm optimization has obvious ties with evolutionary computation. Conceptually, it seems to lie somewhere between genetic algorithms and evolutionary programming. It is highly dependent on stochastic processes, like evolutionary programming [28,40].
PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA).In general. PSO implementation is easier than GA. Indeed, PSO only has one operator; velocity calculation, so the computation time is decreased significantly. The reason is PSO does not perform the selection and crossover operations in evolutionary process. Another difference between GA and PSO is the ability to control convergence. Crossover and mutation rates can affect the convergence of GA, but nothing can compare to the level of control achieved through manipulating of the inertial weight. The more decrease of inertial weight the more increase the swarm's convergence. This type of control allows determining the rate of convergence, and the level of 'stagnation' eventually achieved. Stagnation occurs in GA when all of the individuals have the same genetic code. In that case the gene pool is uniform, crossover has little or no effect on population and each successive generation is essentially same as the first. However, in the PSO, this effect can be controlled or prevented .Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied [24, 25, 33, 37]

This algorithm is based on the following scenario: a group of birds are randomly searching food in an area and there is only one piece of food. All birds are unaware where the food is, but they do know how far the food is at each time instant. The best and most effective strategy to find the food would be to follow the bird which is nearest to the food. Based on such scenario, the PSO algorithm is used to solve the optimization problem. In PSO, each single solution is a “bird” in the search space; this is referred to as a “particle”. The swarm is modeled as particles in a multi dimensional space, which have positions and velocities. These particles have two essential capabilities: their memory of their own best position and knowledge of the global best. Members of a swarm communicate good positions to each other and adjust their own position and velocity based on good positions.

The features of the particle swarm optimization method are as follows:

• The method is developed from research on swarm such as fish schooling and bird flocking. It can be easily implemented, and has stable convergence characteristic with good computational efficiency [31].

In a PSO system, particles fly around in a multi-dimensional search space adjusting its position according to its own experience and the experience of its neighboring particle. The goal is to efficiently search the solution space by swarming the particles towards the best fitting solution encountered in previous iterations with the intention of encountering better solutions through the course of the process and eventually converging on a single minimum or maximum solution. The performance of each particle is measured according to a pre-defined fitness function, which is related to the problem being solved. PSO has been regarded as a promising optimization algorithm due to its simplicity, low computational cost and good performance [34, 35].

CHAPTER 2

LITERATURE REVIEW

2.1 General

The reported literature reveals a rising interest toward optimized controller as well as adaptive controllers in process industry. Chemical reactors are ones of the most important plants in process industry. Their operation, however, is corrupted with various uncertainties. . The main difficulty in tuning of control is due to the disturbances and parameter uncertainties. Application of adaptive and optimized PID control approach can be some ways of overcoming all these problems.

The basic PID controllers have difficulty in dealing with problems that appear in complex non-linear processes. An adaptive system has maximum application when the plant undergoes transitions or exhibits non-linear behavior and when the structure of the plant is not known. This permits the controller to maintain a required level of performance in spite of any noise or fluctuation in the process.

With the advance of computational methods in the recent times tuning a PID controller is rather difficult and can be a time consuming process so, optimization algorithms are proposed to tune the PID controller parameters in order to find an optimal performance. There are several optimization algorithms which can be used for searching the optimal gain parameters.

This literature overview gives an impression of a number of the available methods for controlling like adaptive and optimized PID control design and discusses the advantages, disadvantages and their applicability.

2.2 Literature Review

Chemical reactors are the most influential and therefore important units that a chemical engineer will encounter. To ensure the successful operation of a continuous stirred tank reactor (CSTR) it is necessary to understand their dynamic characteristics. A good understanding will ultimately enable effective control systems design [1, 3].

The mathematical model of CSTR is constructed by two ordinary differential equations. The simple iteration method and Runge-Kutta's method were used for numerical solving of steady-state and dynamic analyses. These analyses show mainly nonlinearity of the system and results in the choice of the second order transfer function with relative order one as ELM [2, 3, 4].

A comparative study of tuning of PID controller using different conventional methods like ZieglarNicholas,Cohen-Coon, etc. is described and concluded that the traditional Ziegler-Nichols method can be used confidently for majority of systems [5,6,7].

PID controller is most popular control scheme that has been widely used throughout the chemical process industry but it doesn’t give satisfactory performance for nonlinear system so it is necessary to develop a new controller which accommodate the advantages of conventional PID controller and eliminate their drawbacks [8].

The disadvantage of PID Controller is that it cannot maintain the stability and it has higher response time i.e. the effluent temperature and concentration is not keep as a desired value. The PID controller does not ensure the stability of the process, and it is not suppressing the influence of external disturbances [8].

As compared to conventional controllers (PID Controllers), Adaptive Controllers are very effective to handle the situations where the parameter variations and environmental changes occur, therefore, the controller parameters are adjusted automatically to give a desired closed loop performance. The adaptive controller maintains constant performance in the presence of disturbances [9,10, 11].

The behavior of a system controlled by model reference adaptive control scheme using MIT rule and conclude that by increasing the adaptation gain till some extent, the settlingtime, peak time and rise time was decreased. Hence for suitable value of adaptation gain in MIT rule, the plant output follows the reference model [12].

By using adaptive control the entire dynamic characteristic of the system is improved also controller maintains constant dynamic performance in the presence of unpredictable and immeasurable variations [13].

The adaptive controller exhibits superior performance in the presence of noise the convergence time is typically large and there is a large overshoot. To resolve these problems of adaptive controller, the proposed controller is redesigned by modifying the adaptation law. And a significant improvement is observed in the performance of the adaptive controller without excessive increase in the adaptation rate [14].

The MIT rule by itself does not guaranteeconvergence or stability. An MRAC designed usingthe MIT rule is very sensitive to the amplitudes of thesignals. The parameter adaption gain has to havesmall values. For greater values in absolute value of the offset leads also to instability [16].

An alternative approach to the MIT rule is to use Lyapunov based method, which avoids the stabilityproblems present in the gradient approaches [17, 20].

In case of MIT rule, if adaptation gain increases, the time response of the system is also improved for the chosen range of adaption gain and further system is unstable in the upper range. In Lyapunov rule, system is stable beyond the chosen range of adaptation gain. So with suitable value of adaptation gain in MIT rule and Lyapunov rule plant output can be made close to reference model. It can be concluded that performance using Lyapunov rule is better than the MIT rule [18].

Although by using adaptive controller settling time and overshoot is reduced as compared to conventional controller but by using optimized controller very good conversion can be achieved and at the same time the temperatures inside the reactor do not violate the safety constraints, even when there are large disturbances in the feed concentrations [13].

To enhance the capabilities of traditional PID parameter tuning techniques, several intelligent approaches have been suggested to improve the PID tuning, there are several optimization algorithms which can be used for searching the optimal gain parameters a very basic one is the random search. Randomly points are selected from the feasible set and these are evaluated the best one is selected as the optimum. A more sophisticate algorithm is, for example, Simulated Annealing (SA) [23], SA is an algorithm that inspired by annealing in metallurgy. Main idea of this method is to replace the current solution by a random “nearby” that is better than the current solution such as those using genetic algorithms (GA) [24,25] and the particle swarm optimization (PSO) [26,27]. With the advance of computational methods in the recent times, optimization algorithms are often proposed to tune the control parameters in order to find an optimal performance.

PSO method does not perform the selection and crossover operations in evolutionary processes, it can save some computation time compared with the GA method, thus proving that the PSO-PID controller is more efficient than the GA-PID controller [38].

Particle swarm optimization is an extremely simple algorithm that seems to be effective for optimizing a wide range of functions. We view it as a mid-level form of A-life or biologically derived algorithm, occupying the space in nature between evolutionary search, which requires eons, and neural processing, which occurs on the order of milliseconds [28].The PSO technique can generate a high-quality solution within shorter calculation time and stable convergence characteristic than other stochastic methods. PSO method is an excellent optimization methodology and a promising approach for solving the optimal PID controller parameters [29,30,31,32].

PSO presents multiple advantages to a designer by operating witha reduced number of design methods to establish the type of the controller, giving a possibility of configuring the dynamic behavior of the control system with ease, starting the design with a reduced amount of information about the controller (type and allowable range of the parameters), but keeping sight of the behavior of the control system. The performance of the above said method of tuning a PID controller can even be proved to be better than the method of tuning the controller after approximating the system to a FOPTD model, and using the traditional techniques, regarding which a rich literature is available. So this method of tuning can be applied to any system irrespective of its order and can be proved to be better than the existing traditional techniques of tuning the controller [33,34,35]..

The variables which characterize the quality of the final product in CSTR are often difficult to measure in real-time and cannot be directly measured using the feedback configuration. So, a virtual feedback control isimplemented to control the state variables using Extended Kalman Filter (EKF) in the feedback path. Since it is hard to determine the optimal or near optimal PID parameters using classical tuning techniques like Ziegler Nicholasmethod, so it is implemented with highly skilled optimization algorithm like ParticleSwarm Optimization (PSO) is used [33].

PSO method has more robust stability and efficiency and can solve the searching and tuning problems of PIDcontroller parameters more easily and quickly than the Ziegler-Nichols method [37].

Soft computing techniques are often criticized for two reasons: algorithms are computationally heavy and convergence to the optimal Solution cannot be guaranteed. PID controller tuning is a small-scale problem and thus computational complexity is not really an issue here. It took only a couple of seconds to solve the problem. Compared to conventionally tuned system, GA,EP, PSO an ACO tuned system has good steady state response and performance indices [36, 39].

2.3 Conclusion

The exhaustive literature review has been carried out. The detailed analysis of literature review has revealed that extensive work is being carried out in the field of CSTR. The advancement in evolutionary algorithms has facilitated the development of various control strategies for its implementation. Performance enhancement and increase in robustness encourages the use of evolutionary algorithms as it improves the both, in the presence of nonlinearities and varying operation conditions. Different controller configurations for concentration and temperature control of CSTR tried for better response. Reduction in computational effort and memory requirement can be achieved with optimization techniques.

CHAPTER 3

DEVELOPMENT OF MATHEMATICAL MODELING OF CSTR

3.1 General

The most important unit in a chemical process is generally a chemical reactor. A chemical reactor is a vessel where reactions are carried out purposefully to produce products from reactants by means of one or more chemical reactions. Chemical reactions are either exothermic (release energy) or endothermic (require energy input) and therefore require that energy either be removed or added to the reactor for a constant temperature to be maintained. Here, we consider a perfectly mixed, continuously stirred tank reactor (CSTR) as shown in Figure 3.1 Single, first-order exothermic irreversible reaction; $A \rightarrow B$ is taking place in CSTR [2].

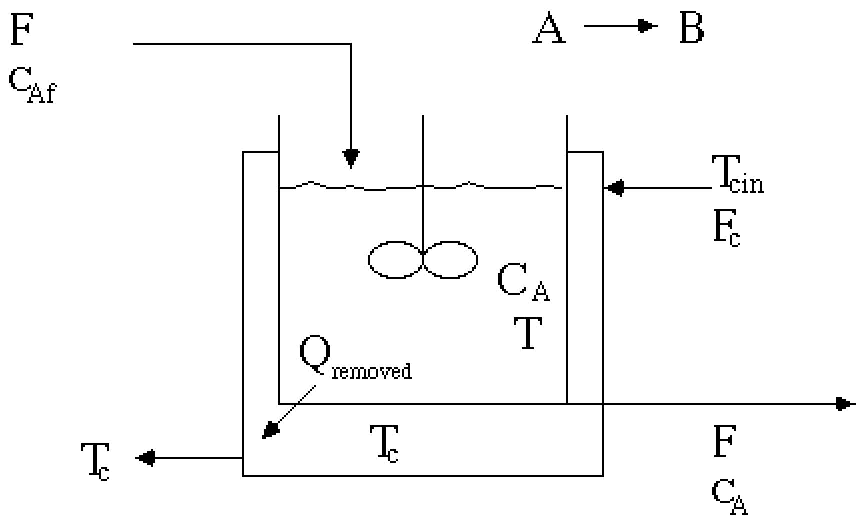

Fig.3.1 Continuous Stirred Tank Reactor with Cooling Jacket

In fig.3.1 a fluid stream is continuously fed to the reactor and another fluid stream is continuously removed from the reactor. Sincethe reactor is perfectly mixed, the exit stream has the same concentration and temperature as the reactor fluid. A jacket surrounding the reactor also has feed and exit streams. The jacket is assumed to be perfectly mixed and at lower temperature than the reactor. Energy then passes through the reactor walls into the jacket, remove the heat generated by reaction [1, 3, 4].

3.2 Mathematical Modeling

For simplicity it is assumed that the cooling jacket temperature can be directly manipulated, so that an energy balance around the jacket is not required. Following assumptions are also made [2,3]:

1) Perfect mixing
2) Constant volume
3) Constant parameter values

3.2.1 Overall Material Balance

The rate of accumulation of material in the reactor is equal to the rate of material in by flow minus the material out by flow.

$$\frac{dV\rho}{dt} = F_{in}\rho_{in} - F_{out}\rho \qquad (3.1)$$

Assuming a constant amount of material in the reactor$\left(\frac{dV\rho}{dt} = 0\right)$, we find that

$$F_{in}\rho_{in} = F_{out}\rho$$

If we also assume that the density remains constant, then

$$F_{in} = F_{out} = F$$

$$\text{And } \frac{dV}{dt} = 0$$

3.2.2 Balance on Component A

The balance on component A is

$$\frac{dVC_A}{dt} = FC_{Af} - FC_A - rV \qquad (3.2)$$

Where' r' is the rate of reaction per unit volume [2, 3].

3.2.3 Energy Balance

The energy balance equation is given by

$$\frac{d(V\rho c_p(T-T_{ref}))}{dt} = F\rho c_p(T_f - T_{ref}) - F\rho c_p(T - T_{ref}) + (-\Delta H)Vr - UA(T - T_j) \quad (3.3)$$

whereT_{ref} represents an arbitrary reference temperature for enthalpy [3,4].

3.3 State Variable Form of Dynamic Equations

Equation (3.1) and (3.2) can be written in the following state variable form

(since$\frac{dV}{dt} = 0$)

$$f_1(C_A, T) = \frac{dC_A}{dt} = \frac{F}{V}(C_{Af} - C_A) - r \quad (3.4)$$

$$f_2(C_A, T) = \frac{dT}{dt} = \frac{F}{V}(T_f - T) + \left(\frac{-\Delta H}{\rho c_p}\right) r - \frac{UA}{V\rho c_p}(T - T_j) \quad (3.5)$$

Here it is assumed that the volume is constant. The reaction rate per unit volume (Arrhenius expression) is

$$r = k_o e^{-\Delta E/RT}$$

Where it is assumed that the reaction is first-order [2].

Guess 1 - High concentration (low conversion), Low temperature. Here we consider an initial guess of CA =8 and T = 300 K.

So the steady-state solution for guess 1 is$\begin{bmatrix} C_{AS} \\ T_S \end{bmatrix} = \begin{bmatrix} 8.56 \\ 311.2 \end{bmatrix}$ that is, high concentration (low conversion) and low temperature [15].

3.4 Linearization of Dynamic Equations

The stability of the nonlinear equations can be determined by finding the following state-space form

$$\dot{X} = AX + BU \quad (3.6)$$

And determining the Eigenvalues of the **A** (state-space) matrix.

The nonlinear dynamic state equations are [13]

$$f_1(C_A,T) = \frac{dC_A}{dt} = -\frac{F}{V}C_A - kC_A + \frac{F}{V}C_{Af}$$

$$f_2(C_A,T) = \frac{dT}{dt} = \left(\frac{-\Delta H}{\rho c_p}\right)kC_A - \frac{F}{V}T - \frac{UA}{V\rho c_p}T + \frac{UA}{V\rho c_p}T_j + \frac{F}{V}T_f$$

Let the state, and input variables be defined in deviation variable form

$$x = \begin{bmatrix} C_A - C_{As} \\ T - T_s \end{bmatrix} u = [T_j]$$

3.5 Steady-State Solution

The steady-state solution is obtained when $\frac{dC_A}{dt} = 0$ and $\frac{dT}{dt} = 0$, that is

$$f_1(C_A,T) = 0 = \frac{F}{V}(C_{Af} - C_A) - k_o e^{-\Delta E/RT} C_A \qquad (3.7)$$

$$f_2(C_A,T) = 0 = \frac{F}{V}(T_f - T) + \left(\frac{-\Delta H}{\rho c_p}\right)k_o e^{-\Delta E/RT} C_A - \frac{UA}{V\rho c_p}(T - T_j) \qquad (3.8)$$

To solve these two equations, all parameters and variables except for two (CA and T) must be specified. Numerical values given in table1 can be use to solve for the steady-state values of C_A and T [2,3,5].

Table 3.1 Reactor Parameters[11]

Reactor Parameter	Description	Values
$\frac{F}{V}$ (hr-1)	Flow rate*reactor volume of tank	1
k_o (hr-1)	Exponential factor	$10e^{15}$
-ΔH (BTU/lbmol)	Heat of reaction	6000
E(BTU/lbmol)	Activation energy	12189
ρc_p (BTU/$ft^{3)}$	Density*heat capacity	500
T_f (K)	Feed temperature	312
(C_{Af} (lbmol/ft^3)	Concentration of feed stream	10
$\frac{UA}{V}$	Overall heat transfer coefficient/reactor volume	145
T_j (K)	Jacket temperature	300

3.6 Stability Analysis

Performing the linearization, following elements forA are obtained:

$$A_{11} = \frac{\partial f_1}{\partial x_1} = \frac{\partial f_1}{\partial C_A} = -\frac{F}{V} - k_s$$

$$A_{12} = \frac{\partial f_1}{\partial x_2} = \frac{\partial f_1}{\partial T} = -C_{As} k_s^{'}$$

$$A_{21} = \frac{\partial f_2}{\partial x_1} = \frac{\partial f_2}{\partial C_A} = \left(\frac{-\Delta H}{\rho c_p}\right) k_s$$

$$A_{22} = \frac{\partial f_2}{\partial x_2} = \frac{\partial f_2}{\partial t} = -\frac{F}{V}T - \frac{UA}{V\rho c_p} + \left(\frac{-\Delta H}{\rho c_p}\right) C_{As} k_s^{'}$$

Where we define the following parameters for more compact representation

$$k_s = k_o \exp\left(\frac{-\Delta E}{RT_s}\right)$$

$$k_s^{'} = \frac{\partial k_s}{\partial T} = k_o \exp\left(\frac{-\Delta E}{RT_s}\right)\left(\frac{-\Delta E}{RT_s^2}\right)$$

Or

$$k_s^{'} = k_s \left(\frac{\Delta E}{RT_s^2}\right)$$

From the analysis presented above, the state-space A matrix is

$$A = \begin{bmatrix} -\frac{F}{V} - k_s & -C_{As} k_s^{'} \\ \frac{-\Delta H}{\rho c_p} k_s & -\frac{F}{V} - \frac{UA}{V\rho c_p} + \left(\frac{-\Delta H}{\rho c_p}\right) C_{As} k_s^{'} \end{bmatrix} \tag{3.9}$$

$$A = \begin{bmatrix} -1.175 & -.08045 \\ -2.1 & -2.255 \end{bmatrix}$$

Where the elements of the **B** matrix are

$$B_{11} = \frac{\partial f1}{\partial U1} = \frac{\partial f1}{\partial T_j} = 0$$

$$B_{21} = \frac{\partial f2}{\partial U1} = \frac{\partial f2}{\partial T_j} = \frac{UA}{V\rho C_p}$$

The input matrix B is

$$B = \begin{bmatrix} 0 \\ \frac{UA}{V\rho C_p} \end{bmatrix} \tag{3.10}$$

$$B = [0; .29]$$

C= [0 1]

D=0

The stability characteristics are determined by the Eigenvalues of **A**, which are obtained by solving det (λI - A) = 0.

In Mat lab command we can write

$$\gg \mathrm{A} = [-1.175 \; -0.08045; \; -2.1 \; -2.255];$$

$$>> \; lambda = eig(\mathrm{A})$$

```
lambda =

  -1.0364

  -2.3936
```

Both of the Eigen values are negative, indicating that operating point is stable [2, 3].

3.7 Transfer Function Analysis

By evaluating the A and B, the transfer function that relate the input to output can be calculate by using Mat lab command.

$$\mathrm{G(S)} = \mathrm{C(SI - A)^{-1}\, B{+}D} \tag{3.11}$$

```
>> [num, den] =ss2tf (A, B, C, D)

num =   0   .29  .3407

den =    1.0000   3.43  2.48
```

So the transfer function of the process is

$$G_p(S) = \frac{.29S + .3407}{S^2 + 3.43S + 2.48}$$

3.8 Conclusion

In this chapter state space model of CSTR has been obtained by calculating mass balance and energy balance equation. After obtaining state space model of CSTR, its transfer function is obtained by using Matlab.

<u>CHAPTER4</u>
MODEL REFERENCE ADAPTIVE CONTROL

4.1 General

Model reference adaptive system is an important adaptive controller .It may be regarded as an adaptive servo system in which the desired performance is expressed in terms of a reference model, which gives the desired response to a command signalas shown in figure 2.1 the system has an ordinary feedback loop composed of the process and the controller and another feedback loop that changes the controller parameters. The parameters are changed on the basis of feedback from error, which is the difference between the output of system and the output of reference model. The ordinary feedback loop is called the inner loop and the parameter adjustment loop is called the outer loop. The mechanism for adjusting the parameters in a model reference adaptive system can be obtained in two ways by using a gradient method or by applying Lyapunov's stability theory.

MRAC is composed of four parts: a plant containing unknown parameters, a reference model for compactly specifying the desired output of control system, a feedback control law containing adjustable parameters

<u>**Plant:**</u> The plant is assumed to have a known structure, although the parameters are unknown.

<u>**Reference model:**</u> A reference model is used to specify the ideal response of adaptive control system to the external command. The choice of reference model has to satisfy two requirements.

1) It should reflect the performance specification in the control task, such as rise time, settling time, overshoot or equivalent frequency domain characteristics.
2) The ideal behavior specified by the reference model should be achievable for the adaptive control system.

<u>**Controller:**</u> The controller is usually parameterized by a number of numbers of adjustable parameters. This implies that there exists different set of controller parameters value for which the desired control task is achievable usually the control

law is liner in terms of adjustable parameters. Existing adaptive control designs normally require liner parameterization the controller in order to obtain mechanism with guarantee stability and tracking convergence.

<u>Adaption mechanism</u>: it is used to adjust the parameter in the control law. In MRAC system the adaption law searches for the parameter such that the response of plant under adaptive control becomes the same as that of the reference model. The adaption mechanism drives the tracking error to zero. This adaption mechanism is design to guarantee the stability of control system as well as convergence of the tracking error to zero.

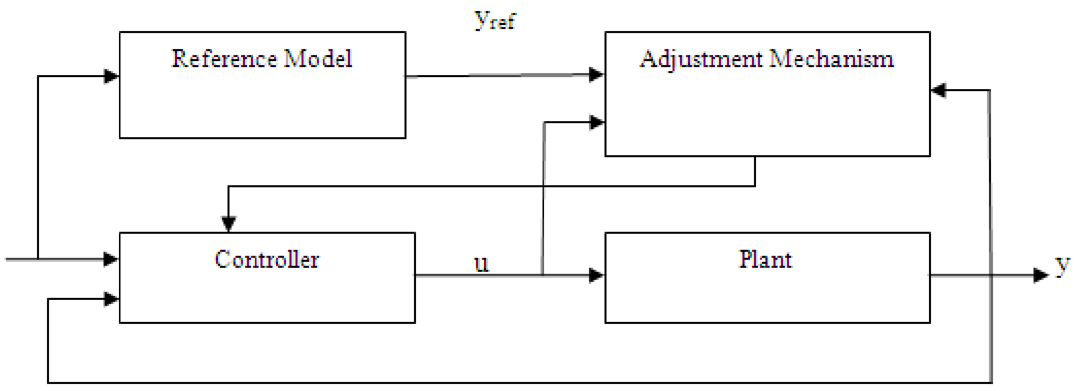

Figure 4.1 Block Diagram of a Model Reference Adaptive System

In the MRAS the desired behavior of the system is specified by a model, and the parameters of the controller are adjusted on the error, which is the difference between the output of closed loop system and the model [9,10,18,19].

4.2 MIT Rule

The MIT rule is the original approach to MRAC.The name is derived from the fact that it was developed at the instrumentation laboratory at MIT.

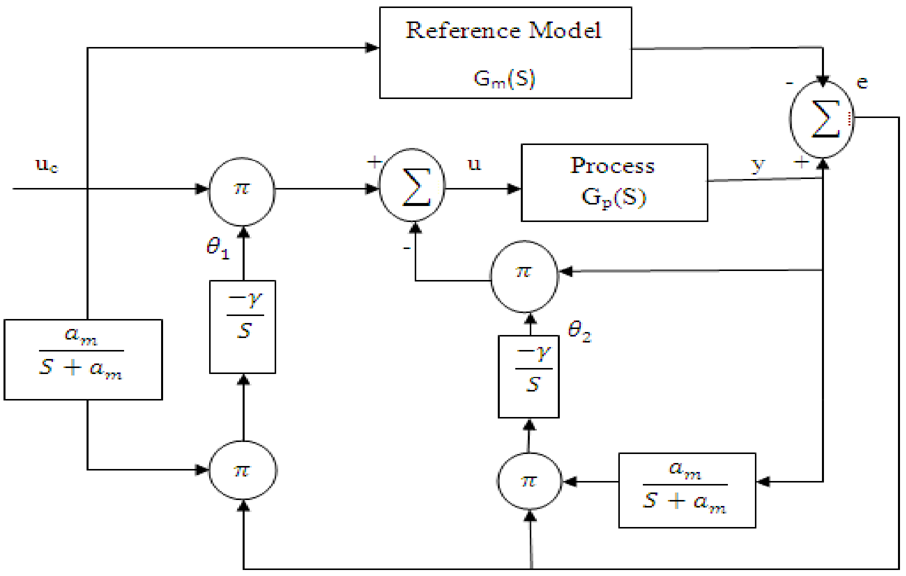

Figure 4.2 Block diagram of MRAS Based on MIT Rule

Here, we have considered a closed loop control system in which the controller has one adjustable parameter θ the desired closed loop response is specified by a model whose output is y_m.Let e be the error between the output y of closed loop system and the output y_m of the model. The convergence of the modeling error to zero for any given u_c is assured when y exactly follows the output of the model (y_m),

The modeling error e is given by equation (4.1)

$$e = y - y_m \tag{4.1}$$

One possibility is to adjust the parameter in such a way that the loss function is

$$J(\theta) = \frac{1}{2}e^2(\theta) \tag{4.2}$$

To find out how to update the parameter theta, an equation needs to be formed for the change in theta. If the goal is to minimize this cost related to the error, it is sensible to move in the direction of the negative gradient of J. This change in J is assumed to be proportional to the change in theta. Thus, the derivative of theta is equal to the negative change in J. The result for the cost function chosen above is:

$$\frac{d\theta}{dt} = -\gamma \frac{\partial J}{\partial \theta} \tag{4.3}$$

$$= -\gamma\, e \frac{\partial e}{\partial \theta}$$

This relationship between the change in theta and the cost function is known as the MIT rule. The MIT rule is central to adaptive nature of the controller. The partial derivative term$\frac{\partial e}{\partial \theta}$is called the sensitivity derivative of the system. This shows how the error is dependent on the adjustable parameter,θ.There are many alternatives to choose the loss function F,like it can be taken as mod of error also. Similarly $\frac{d\theta}{dt}$ can also have different relations for different applications.

$$F(\theta) = |e|$$

Sign-sign
algorithm:

$$\frac{d\theta}{dt} = -\gamma \text{sign} \left(\frac{\partial e}{\partial \theta}\right) \text{sign}(e) \tag{4.4}$$

Or it may be chosen as

$$\frac{d\theta}{dt} = -\gamma \left(\frac{\partial e}{\partial \theta}\right) \text{signe}$$

Where signe= 1 for e› 0

$= 0\ fore = \ 0$

$= -1 fore‹\ 0$

The sign-sign algorithm used in telecommunications where simple implementation and fast computations are required [10,12,13,14,16].

4.3 Lyapunov StabilityTheory

The Lyapunov stability theory can be used to describe the algorithms for adjusting parameters in Model Reference Adaptive control system.

Let the first order system is described by

$$\frac{dy}{dt} = -ay + \ b\, u \tag{4.5}$$

Where u is the controller output or manipulated variable.

Similarly the reference model is described by

$$\frac{dy_m}{dt} = \ -a_m y_m + \ b_m r \tag{4.6}$$

Where r is the reference input.

For the above mentioned system the error is givenby

$$e(t) = y - y_m$$

Nowthechangeinerrorwithrespecttotimecanbewrittenas

$$\frac{de}{dt} = -a_m e - \left(b\theta_2 + a - a_m\right)y + (b\theta_1 - b_m)u_c \qquad (4.7)$$

The Lyapunov function is described $V(e, \theta_1, \theta_2)$.

This function should be positive semi definite and is zero when error is zero. For stability according to Lyapunov theorem the derivative $\frac{dV}{dt}$ must be negative. The derivative $\frac{dV}{dt}$ requires the values of $d\theta_1/dt$ and $d\theta_2/dt$ .If the parameters are updated then

$$\frac{d\theta_1}{dt} = -\gamma u_c \qquad (4.8)$$

$$\frac{d\theta_2}{dt} = \gamma y e \qquad (4.9)$$

$$\frac{dV}{dt} = -a_m e^2 \qquad (4.10)$$

$So\ dV/dt$is negative semidefinite. This shows that$V(t) \leq V(0)$ and e, θ_1, θ_2 must be bounded.

The main advantage of Lyapunov design is that it guarantees a closed-loop system. For a linear, asymptotically stable governed by a matrix A ,a positive symmetric matrix Q yields a positive symmetric matrix P by the equation

$$A^T P + PA = -Q \qquad (4.11)$$

This equation is known as Lyapunov equation.

The main drawback of Lyapunov design is that there is no systematic way of finding a suitable Lyapunov function V leading to a specific adaptive law.

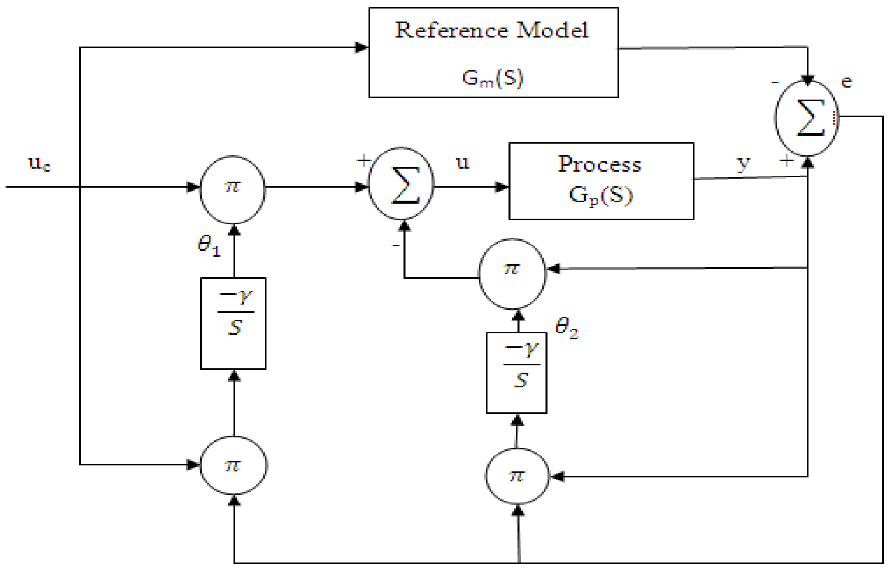

Fig. 4.3 Block diagram of MRAS based on LyapunovTheory

4.3.1Lyapunov design of MRAC:

- Determine the controller structure.
- Derive the error equation.
- Find a lyapunov equation.
- Determine adaption law that satisfies the lyapunov theorem [10,11,15,17,20].

4.4 Development of Simulink Model of Model Reference Adaptive Control

4.4.1 MIT Rule:

Simulink model of Model Reference Adaptive Control by using MIT rule is shown in fig.no 4.4

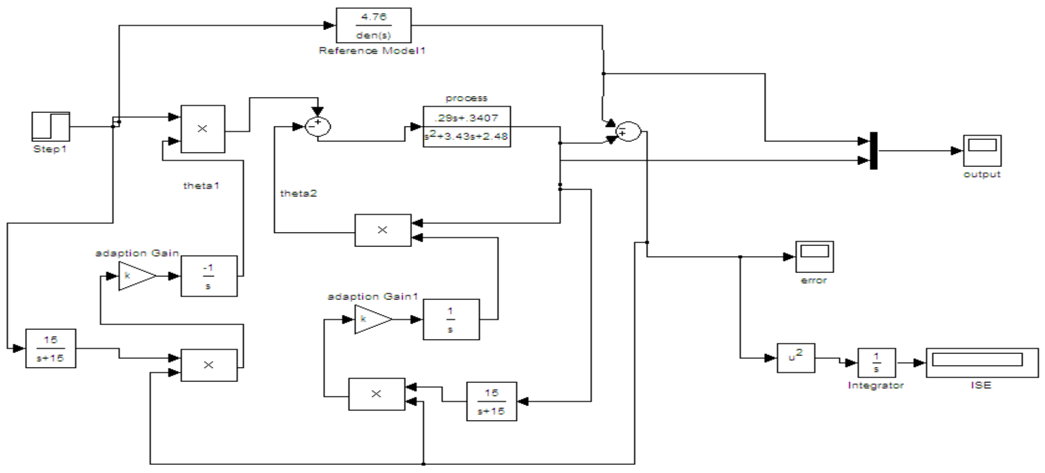

Fig.4.4 Matlab Simulink diagram of MIT Rule

4.4.2 Lyapunov Rule

Simulink model of Model Reference Adaptive Control by usingLyapunovrule is shown in fig.no 4.5

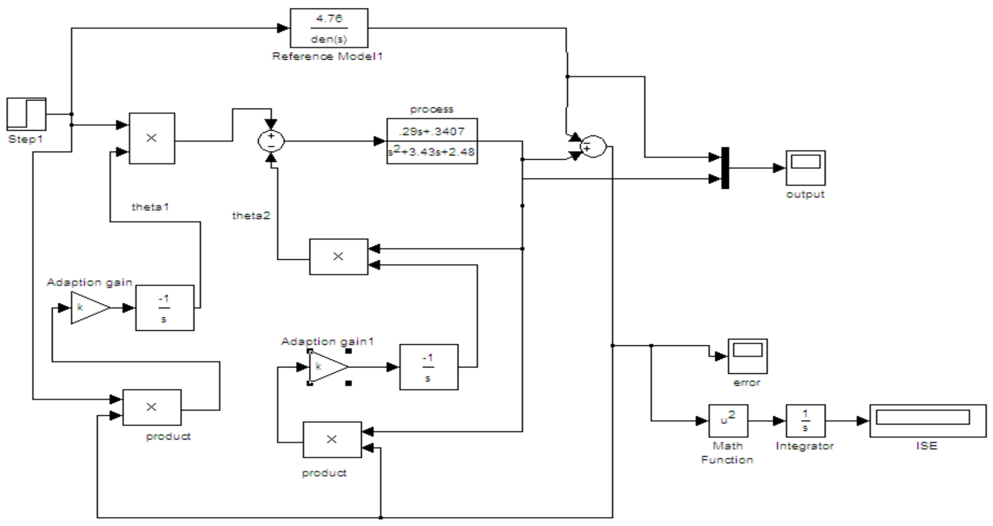

Fig.4.5 Matlab Simulink diagram of Lyapunov Rule

4.5 Conclusion

The purpose of this chapter is to describe the mechanism of adjusting the parameters in model reference adaptive control. Two techniques of adjusting the parameters i.e. MIT rule and Lyapunov rule have been described. Also Matlab/Simulink model of MRAC has been developed to indicate why adaption is useful.

CHAPTER 5

PID CONTROLLER

5.1 General

The PID controller is also called as three mode controller. In industrial practice, it is commonly known as proportional-plus-reset-plus-rate controller. The combination of proportional, integral and derivative mode is one of the most powerful but complex controller operations.The PID controller is the most common general purpose controller in the today's industries. It can be used as a single unit or it can be a part of a distributed computer control system. Over 30 years ago, PID controllers were pneumatic-mechanical devices, whereas now a day they are implemented in software based techniques like ANN, Fuzzy Logic, Genetic Algorithm and most popular Optimization techniques.

After implementing the PID controller, we have to tune the controller; and there are different approaches to tune the PID parameters like P, I and D. The Proportional (P) part is responsible for following the desired set-point while the Integral (I) and Derivative (D) part account for the accumulation of past errors and the rate of change of error in the process or plant, respectively [5,6].

5.2 PID Control

PID controller consists of three types of control i.e. Proportional, Integral and Derivative control.

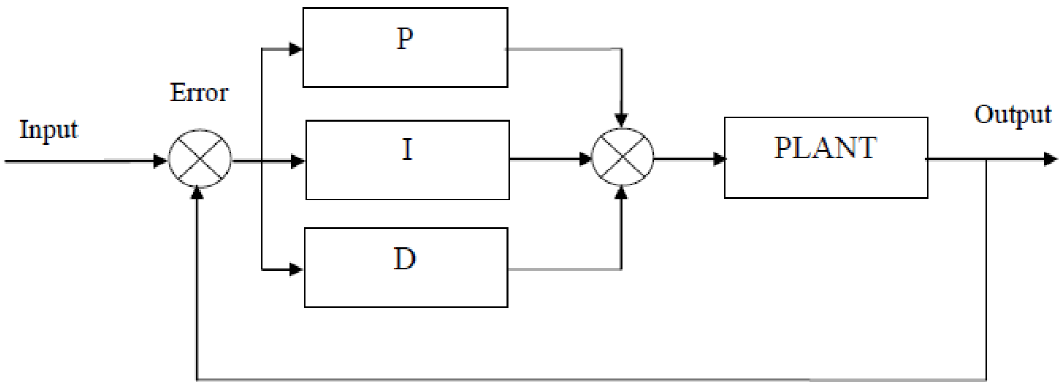

Fig.5.1 Block diagram of PID Controller

Proportional Control (P)

The proportional controller output uses a ‘proportion’ of the system error to control the system. However, this introduces an offset error into the system or plant.

$$\mathrm{P} = \mathrm{Kp} * \mathrm{Error} \qquad (5.1)$$

Integral Control (I)

The integral controller output is proportional to the amount of time; there is an error present in the system. The integral action removes the offset introduced by the proportional control but introduces a phase lag into the system.

$$\mathrm{I} = \mathrm{Ki} * \int \mathrm{error\,dt} \qquad (5.2)$$

Derivative Control (D)

The derivative controller output is proportional to the rate of change of the error. Derivative control is used to reduce or eliminate overshoot and introduces a phase lead action that removes the phase lag introduced by the integral action.

$$\mathrm{D} = \mathrm{K_d} * \frac{\mathrm{d(error\,)}}{\mathrm{dt}} \qquad (5.3)$$

The PID controller is simple and easy to implement. It iswidely applied in industry to solve various controlproblems. PID controllers have been used for decades. The transfer function of PID controller is describedby the following equation in the continuous *s*-domain(Laplace operator) [5,6,8].

The system transfer functions in continuous s-domain are given as

$$P = K_p\,, I = \frac{K_i}{S}\ and\ D = K_d\,S$$

$$G_C\,(S) = P + I + D = K_p + \frac{K_i}{S} + K_d\,S$$

$$G_C(S) = K_p\left(1 + \frac{1}{T_i\,S} + T_d S\right) \qquad (5.4)$$

Where *Kp* is the proportional gain, *Ki* is the integration coefficient and *Kd* is the derivative coefficient *Ti* is known as the integral action time or reset time and Td is the derivative action time or rate time [36].

Output of the PID controller in timedomain is given by

$$u(t) = K_p\,e(t) + K_i \int_0^t e(\tau) d\,\tau + K_d\,\frac{de\,(t)}{dt} \qquad (5.5)$$

Where *u*(*t*) and *e*(*t*) are the control and tracking errorsignals in time domain, respectively.

The proportional part of the PID controller reduces error responses to disturbances. The integral term of the error eliminates steady state error and the derivative term of error dampens the dynamic response and there by improves stability of the system.

The parameter settings of a PID controller for optimalcontrol of a plant depend on the plant’s behavior. To design the PID controller the engineer can appropriately choose the combination of and to simultaneously takecare of the transient response as well as the steady-state error. In the design of a PID controller, the three gains of PID must be selected in such a way that the closed loop system has to give the desired response. The desiredresponse should have minimal settling time with a smallor no overshoot in the step response of the closed loop system [5,6,7,8].

5.3 Tuning of PID Controller

5.3.1 Open Loop Tuning

Open loop tuning methods are where the feedback controller is disconnected and the experimenter excites the plant and measures the response. The key point here is that since the controller is now disconnected the plant is clearly now no longer strictly under control. If the loop is critical, then this test could be hazardous. Indeed if the process is open-loop unstable, then we will be in trouble before we begin.For many process control applications, open loop type experiments are usually quick to perform, and deliver informative results. If the system is steady at set point, and remains so, then we have no information about how the process behaves.

Naturally if the response is not sigmoidal or ‘S’ shaped and exhibits overshoot, or an integrator, then this tuning method is not applicable. This method implicitly assumes the plant can be adequately approximated by a first order transfer function with time delay,

$$G_p = \frac{Ke^{-\theta S}}{TS+1} \tag{5.6}$$

where K is gain, θ is the dead time or time delay, and T is the open loop process time constant. Once we have recorded the open loop input/output data, and subsequently measured the times T and θ, the PID tuning parameters can be obtained directly from the given tables for different classical methods.

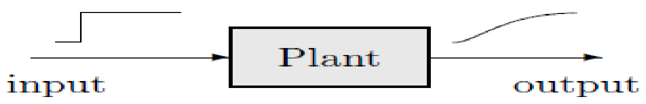

Figure 5.2 Block diagram of plant with variable output

The method is based on computing the times t_1 and t_2 at which the 35.3% and 85.3% of the system response is obtained as shown in the figure:

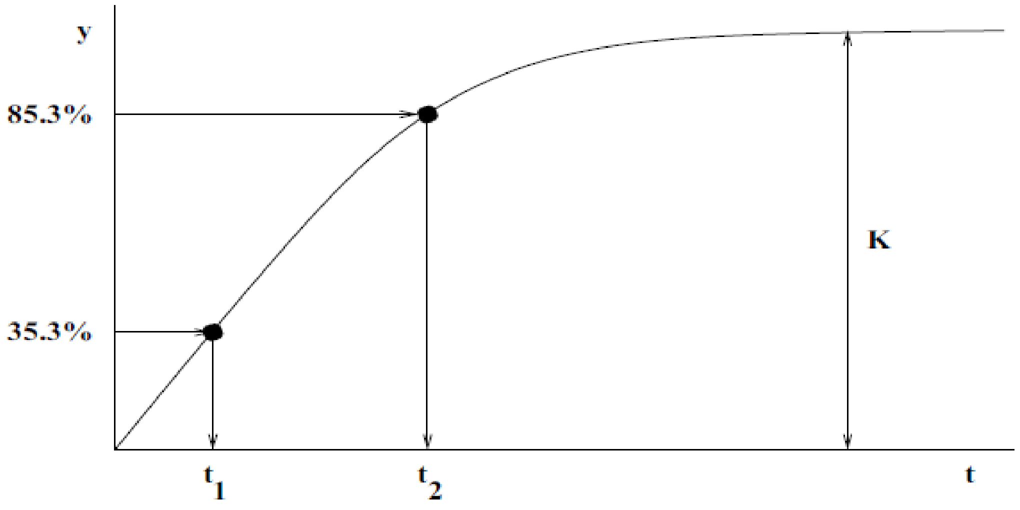

Figure 5.3 System response for first order time delay transfer function

After computing the t_1 and t_2 times, the time delay (θ) and process time constant (T) can be obtained from the following equations [5,6,7] :

$$\theta = 1.3t_1 - 0.29t_2 \tag{5.7}$$

$$T = .67(t_2 - t_1) \tag{5.8}$$

5.3.1.1 Ziegler-Nichols open loop Tuning Method

The PID tuning parameters as a function of the open loop model parameters K, T and $\theta$from equation (5.6) as derived by Ziegler-Nichols. They often form the basis for tuning procedures used by controller manufacturers and process industry. The methods are based on determination of some features of process dynamics. The controller parameters are then expressed in terms of the features by simple formulas. The method presented by Ziegler-Nichols is based on a registration of the open-loop step response of the system, which is characterized by two parameters. first determined, and the tangent at this point is drawn. The intersections between the tangent and the coordinate axes give the parameters T andθ. A model of the process to be controlled was derived from these parameters. This corresponds to modeling a process by an integrator and a time delay. Ziegler and Nichols have given PID parameters directly as functions of T and θ.the behaviour of the controller is as can be expected. The decay ratio for the step response is close to one quarter. It is smaller for the load disturbance. The overshoot in the set point response is too large.

Table 5.1 Ziegler-Nichols open loop method

Controller		**K_p**	**T_i**	**T_d**
Ziegler-Nicholas Tuning Method (Open Loop)	P	$\frac{T}{K\theta}$	-	-
	PI	$\frac{.9\,T}{K\theta}$	$\frac{\theta}{0.3}$	-
	PID	$\frac{1.2\,T}{K\theta}$	$2\,\theta$	$0.5\,\theta$

5.3.2 Closed Loop Tuning Methods

5.3.2.1 Ziegler-Nichols Tuning Method

The control system performs poor in characteristics and even it becomes unstable, if impropervalues of the controller tuning constants are used. So it becomes necessary to tune the controllerparameters to achieve good control performance with the proper choice of tuning constants.Controller tuning involves the selection of the best values of k_p Ti and Td (if a PID algorithm is being used). This is often a subjective procedure and is certainly process dependent. It is widely accepted method for tuning the PID controller. The method is straightforward. First, set thecontroller to P mode only. Next, set the gain of the controller (k_p) to a small value. Make a small set point (or load) change and observe the response of the controlled variable. If k_p is low the response should be sluggish. Increase k_p by a factor of two and make another small change in the setpoint or the load. Keep increasing k_p (by a factor of two) until the response becomes oscillatory. Finally, adjust k_p until a response is obtained that produces continuous oscillations. This is known asthe ultimate gain (K_u). Note the period of the oscillations (P_u). The steps required for the method aregiven below. We have to set the integral and derivative coefficients are zero. Gradually increase the proportional coefficient from 0 to until the system just begins to oscillate continuously. The proportional coefficient at this point is called the ultimate gain K_u and the period of oscillation at this point is called ultimate period P_u.The K_u is the gain margin of the system and;

$$P_u=\frac{2\pi}{\omega cg} \qquad (5.9)$$

where, the ω cg is the gain crossover frequency. Gain margin is the reverse of amplitude ratio.

The Ziegler-Nichols continuous cycling method is one of the best known closed loop tuning strategies. The controller gain is gradually increased (or decreased) until the process outputcontinuously cycles after a small step change or disturbance. At this point, the controller gain is selected as the ultimate gain, Ku, and the observed period of oscillation is the ultimate period, Pu.Ziegler and Nichols originally suggested PID tuning constants as a function of the ultimate gain and ultimate period [5,7,8].

Table 5.2 Ziegler-Nichols closed loop method

Controller		**K_p**	**T_i**	**T_d**
Zieglar-Nicholas TuningMethod(closed Loop)	**P**	$.5K_u$	-	-
	PI	$.45K_u$	$\frac{P_u}{1.2}$	-
	PID	$.6K_u$	$\frac{P_u}{2}$	$\frac{P_u}{8}$

Table 5.3 Modified Ziegler-Nichols closed loop method

Controller		**K_p**	**T_i**	**T_d**
Modified Zieglar-Nicholas TuningMethod(closed Loop)PID control	**No overshoot**	$.2K_u$	$\frac{P_u}{2}$	$\frac{P_u}{2}$
	Some overshoot	$.33K_u$	$\frac{P_u}{2}$	$\frac{P_u}{3}$

5.4 Performance Analysis

To analyze and design a control system, we must define and measure its performance. Based on the desired performance of control system, the system parameter may be adjusted to provide the desired response. Because control systems are inherently dynamic, their performance is usually specified in terms of both the transient response and steady state response.

The transient response is the response that disappears with time. The steady state response is the responses that exist for a long time following an input signal initiation. The design specification for control system normally includes time domain specification and performance indices [5].

5.4.1Time Domain Specification

Specifications for a control system design often involve certain requirements associated with the time response of the closed-loop system. The requirements are specified by the behavior of the controlled variable or by the control error on well defined test signals.

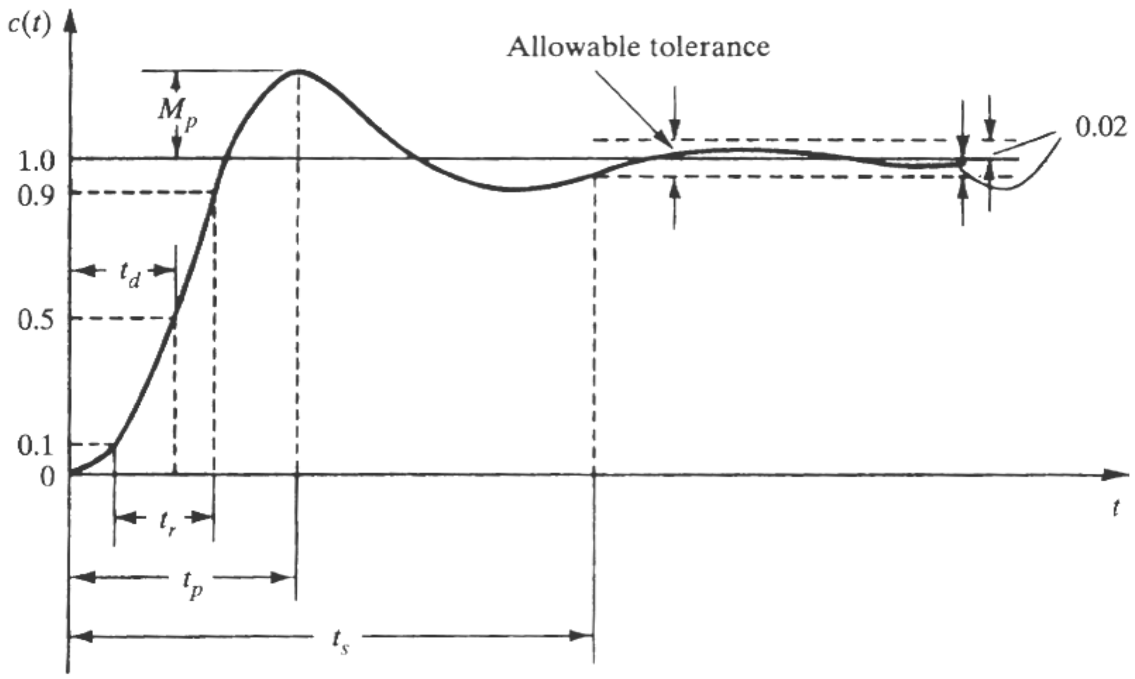

Fig.5.4 Step response of a second order system

Time response indices includes:

1. Delay time (T_d)
2. Rise time (T_r)
3. Peak time (T_p)
4. Maximum overshoot (M_p)
5. Settling time (T_s)

1.Delay Time (T_d): The delay time is the time needed for the response to reach half of its final value very first time.

2. Rise Time (T_r): The rise time is the time required for the response to rise from 10% to 90%, 5% to 95% or 0% to100% of its final value. For underdamped second order systems, the 0% to 100% rise time is normally used. For overdamped systems the 10% to 90% rise time is common.

3. Peak Time (T_p): The peak time is the time required for the response to reach the first peak of the overshoot.

4. Maximum Overshoot) (M_p): The maximum percent overshoot is the normalized difference between the maximum peak value and final steady state output and is expressed in per cent value. It is defined as

$$M_p = \frac{c(t_p)-c(\infty)}{c(\infty)} *100\% \tag{5.10}$$

5. Settling Time (T_s): The settling time is the time required for the response curve to reach and stay within 2% of the final value. In some cases, 5%instead of 2% is used as the percentageof the final value. The settling time is the largest time constant of the system.

5.4.2 Performance Indices

Modern control theory assumes that the systems engineer can specify quantitatively the required system performance. Then a performance index can be calculated or measured and used to evaluate the system's performance. A quantitative measure of the performance of a system is also necessary for the operation of modern control systems, for automatic parameter optimization of a control system, and for the design of optimum systems. A system is considered an optimum control system when the system parameters are adjusted so that the index reaches an extremum value, commonly a minimum value [5,33].

To optimize the performance of a closed-loop control system, it can be try to adjust the control system parameters to maximize or minimize some performance index. Some common performance indices are:

1. **Integral Square Error**: It is defined as

$$ISE = \int_0^T e^2 \, (t)dt$$

The upper limit T is a finite time chosen arbitrarily so that the integral approaches a steady state value.

ISE is more suitable to minimize large amount oferrors. The squared error is mathematically more convenient for analytical and computational purposes.

2. **Integral Absolute Error**: It is defined as

$$IAE = \int_0^T |e(t)|dt$$

It is useful for computer simulation studies.

3. **Integral Time Square Error**: It is defined as

$$ITSE = \int_0^T te^2\,(t)dt$$

4. **Integral Time Absolute Error**: It is defined as

$$ITAE = \int_0^T t|e(t)|dt$$

It provides the best selectivity of the performance indices.

The performance criterion which are examined in this thesis are rise time, peak time, settling time, integral square error (ISE) and maximum overshoot.

5.5 Conclusion

In this chapter tuning procedure of PID controller by the Zeigler-Nicholas method, together with its analytical approach has been discussed. Also various types of performance indices which are necessary to describe a closed loop system like, rise time, settling time, peak time, maximum overshoot along with different types of error indices has been studied.

CHAPTER 6

PARTICLE SWARM OPTIMIZATION

6.1 General

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr.Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling. It is an approach for solving continuous and discreteoptimization problems [28,29].

It is inspired by social behaviors in flocks of birds and schools of fish. In particle swarm optimization (PSO), a set of software agents called particles search for good solutions to a given continuous optimization problem. Each particle is a solution of the considered problem and uses its own experience and the experience of neighbor particles to choose how to move in the search space. In practice, in the initialization phase each particle is given a random initial position and an initial velocity. The position of the particle represents a solution of the problem and has therefore a value, given by the objective function. While moving in the search space, particles memorize the position of the best solution they found. At each iteration of the algorithm, each particle moves with a velocity that is a weighted sum of three components: the old velocity, a velocity component that drives the particle towards the location in the search space where it previously found the best solution so far, and a velocity component that drives the particle towards the location in the search space where the neighbor particles found the best solution so far. PSO has been applied to many different problems and is another example of successful artificial/engineering swarm intelligence system [37].

6.2 Implementation of PSO Controller

PSO isderived from the social-psychological theory, and has been found to be robust in complex systems. Each particleis treated as a valueless particle in g-dimensional searchspace, and keeps track of its coordinates in the problem space associated with the best solution (evaluating value)and this value is called P*best.* The overall best value and its location obtained so far by any particle in the groupthat was tracked by the global version of the particleswarm optimizer *gbest*. The PSO concept consists of changing the velocity of each particle toward its Pbest and gbest locations at each time step. As example, the jth particle is represented as $x_j= (x_{j.1}, x_{j.2}, . . . ,x_{j.g})$ in the g-dimensional space. The best previous position of the jth particle is recorded and represented as $Pbest_j= (Pbest_{j.1}, Pbest_{j.2}, . . . ,Pbest_{j.g})$. The index of best particle among all particles in the group is represented by the $gbest_g$. The rate of the position change (velocity) for particle j is represented as $v_j= (v_{j.1}, v_{j.2} . . . v_{j.g})$. The modified velocity and position of each particle can be calculated using the current velocity and distance from $Pbest_{j.g}$ to $gbest_g$ as shown in the following formulas:

$$v_{j.g}^{(t+1)} = w.v_{j.g}^{(t)} + c_1 * rand\,( ) * (Pbest_{j.g} - x_{j.g}^{(t)}) + c_2 * rand\,( ) * (gbest_g - x_{j.g}^{(t)}) \qquad (6.1)$$

$$x_{j.g}^{(t+1)} = x_{j.g}^{(t)} + v_{j.g}^{(t+1)} \qquad (6.2)$$

j=1, 2…... n

g=1, 2... m

where

n- number of particles in a group;

m- number of members in a particle;

t- pointer of iterations(generations);

$v_{j.g}^{(t)}$ −velocity of particle j at iteration t,

$v_g^{min} \leq v_{j.g}^{(t)} \leq v_g^{max}$

w- inertia weight factor;

c_1, c_2-acceleration constant;

rand( ) - random number between 0 and 1;

$x_{j.g}^{(t)}$-current position of particle j at iteration t;

$Pbest_j$ - Pbestof particle j;

gbest -gbestof the group;

The parameter v_g^{max} determined the resolution, or fitness, with which regions were searched between thepresent position and the target position. If v_g^{max} is too high, particles might fly past good solutions but if v_g^{max} is too low, particles may not explore sufficiently beyond local solutions.

The constant $c_1$and $c_2$represent the weighting of the stochastic acceleration terms that pull each particle toward pbest and gbest.

The process of the PSO algorithm can be summarized as follows:

Step (1): Initialization of a group at random while satisfying constraints.

Step (2): Velocity and Position updates while satisfying constraints.

Step (3): Update of Pbest and gbest.

Step 4) Go to Step 2 until stopping criteria is satisfied.

In the subsequent sections, the detailed implementation strategies of the PSO are described:

6.2.1 Initialization

In the initialization process, a set of individuals is created at random. Therefore, individual j's position at iteration 0 can be represented as the vector (x_j^0 = (P_{j1}^0 ….$P_{j.g}^0$)) where g is the number of generators. The velocity of individual j(v_j^0 = (v_j^1 ….$v_{j.g}^0$)) corresponds to the generation update quantity covering all generators. The elements of position and velocity have the same dimension. The following strategy is used in creating the initial velocity:

$$(P_{imin} - \Sigma) - P_{ji}^0 \leq (P_{imax} + \Sigma) - P_{ji}^0 \qquad (6.3)$$

Where Σ is a small positive real number. The velocity of element i of individual j is generated at random within the boundary. The developed initialization scheme always guarantees to produce individuals satisfying the constraints while maintaining the concept of PSO algorithm. The initial $Pbest_j$ of individual j is set as the initial position of individual j and the initial gbest is determined as the position of an individual with minimum payoff.

6.2.2 Velocity Update

To modify the position of each individual, it is necessary to calculate the velocity of each individual in the next stage, which is obtained from (5.1). In this velocity updating process eq.6.4 is:

$$\text{w}=\frac{w_{\max} - w_{min}}{iter_{max}} * iter \qquad (6.4)$$

Where:

w_{min} = initial weight

w_{max} = final weight

$iter_{max}$ = maximum iteration number

$iter$ = current number of iterations

6.2.3 Position Update:

The position of each individual is modified by (5.1). The resulting position of an individual is not always guaranteed to satisfy the inequality constraints due to over/under velocity. If any element of an individual violates its inequality constraint due to over/under speed then the position of the individual is fixed to its maximum/minimum operating point. Therefore, this can be formulated as follows:

$$P_{ji}^{t+1} = \begin{cases} P_{ji}^{t} + v_{ji}^{t+1} \ if \ P_{ji.min} \leq P_{ji}^{t} + v_{ji}^{t+1} \leq P_{ji.max} \\ P_{ji.min} \ if \ P_{ji}^{t} + v_{ji}^{t+1} \leq P_{ji.max} \\ P_{ji.max} \ if \ P_{ji}^{t} + v_{ji}^{t+1} \geq P_{ji.max} \end{cases} \tag{6.5}$$

6.2.4 Update of Pbest and gbest

The Pbestof each individual at iteration is updated as follows:

$$\mathrm{Pbest}_j^{t+1} = x_j^{t+1} \text{ if } \mathrm{TC}_j^{t+1} < \mathrm{TC}_j^{t} \tag{6.6}$$

$$\mathrm{Pbest}_j^{t+1} = \mathrm{Pbest}_j^{t} \text{ if } \mathrm{TC}_j^{t+1} < \mathrm{TC}_j^{t} \tag{6.7}$$

Where TC_J isthe objective function evaluated at the position of individual j. additionally, g$best$at iteration $t+1$ is set as the best evaluated position among $Pbest_j^{t+1}$.

6.2.5 Stopping Criteria

The PSO is terminated if the iteration approaches to the predefined maximum iteration [29, 32, 34, 37].

It is a very simple concept, and paradigms can be implemented in a few lines of computer code. It requires only primitive mathematical operators, and is computationally inexpensive in terms of both memory requirements and speed. Early testing has found the implementation to be effective with several kinds of problems. In the past several years, PSO has beensuccessfully applied in many research and applicationareas. It is demonstrated that PSO gets better results in afaster, cheaper way compared with other methods. Another reason that PSO is attractive is that there are fewparameters to adjust. Particleswarm optimization has been used for approaches thatcan be used across a wide range of applications, as well as for specific applications focused on a specificrequirement [31, 33, 34].

6.2.6 PSO Controller Flow Chart

The flowchart of the Particle Swarm Optimization system is shown in figure 6.1 [34].

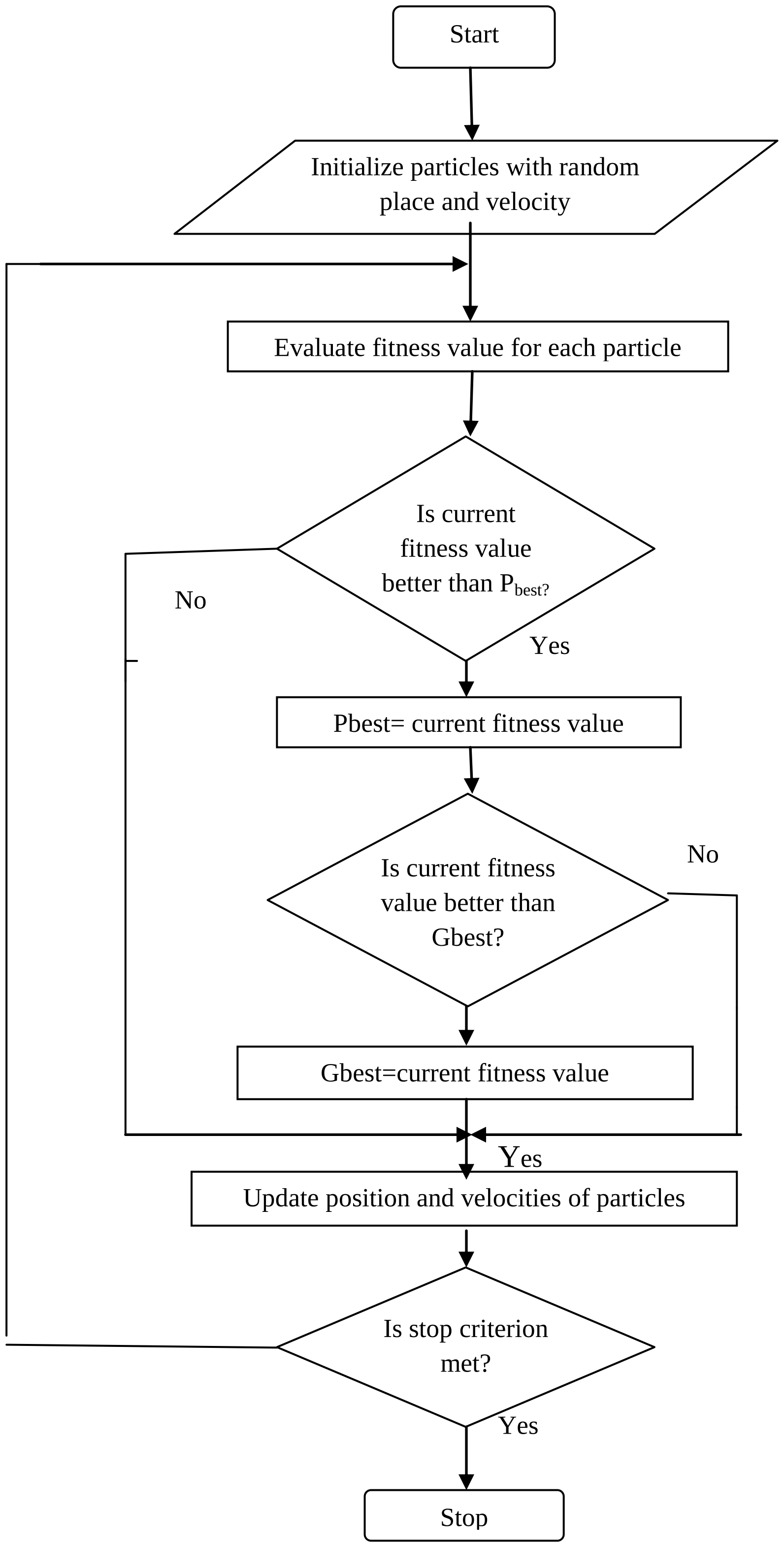
Start
Initialize particles with random place and velocity
Evaluate fitness value for each particle
Is current fitness value better than $P_{best?}$
No
Yes
Pbest= current fitness value
Is current fitness value better than Gbest?
No
Gbest=current fitness value
Yes
Update position and velocities of particles
Is stop criterion met?
Yes
Stop

Fig. 6.1 flowchart of PSO

6.3 PSO-PID Controller

PSO method is an excellent optimization methodology and a promising approach for solving the optimal PID controller parameters. The PID controller is simple and easy to implement. It is widely applied in industry to solve various control problems. Therefore, The PID controller using the PSO algorithm is developed to improve response of system. It is also called the PSO-PID controller. The PSO algorithm was mainly utilized to determine three optimal controller parameters k_p,k_i,k_d, such that the controlled system could obtain a good step response output.

To apply the PSO method for searching the controller parameter, "individual" is used to replace the "particle" and the "population"is used to define the "group". The three controller parameters k_p,k_i and $k_d$composed an individual K by $K \equiv [k_p,k_i,k_d]$;hence there are three members in an individual. These members are assigned as real values. If there are n individuals in a population, then the dimension of a population is n x 3. A set of good control parameters k_p,k_i and $k_d$can achieve a good step response and result in minimization of performance criteria in the time domain including the settling time (t_s), rise time (t_r),maximum overshoot (Mp) , and integral square error (ISE). In the same time, the evaluation value, F which is reciprocal of the performance criterion W (K).

$$F=\frac{1}{W(K)} \tag{6.7}$$

It employs the smaller *W (K)* the value of individual *K*, thehigher its evaluation function. In order to limit the evaluation value of each individual of the population within a reasonable range, the ***Routh-Hurwitz*** criterion must be utilized to test theclosed-loop system stability before evaluating theevaluation value of an individual. The feasible individualand small value of *W (K)* if the individual satisfied the ***Routh-Hurwitz*** criterion stability test applied to thecharacteristic equation of the system.

The searching procedure of the proposed PSO-PID controller is shown as follows:

Step 1: Specify the lower and upper bounds of the three controller parameters and initialize randomly the individuals of the population including searching points, velocities, pbest sand gbest.

Step 2: For each initial individual K of the population, employ the Routh-Hurwitz criterion to test the closed-loop system stability and calculate the values of the four performance criteria in the time domain, namely M_p, E_{ss}, t_r and t_s.

Step 3: Calculate the evaluation value of each individual in the population using the evaluation value, f given by (6.7).

Step 4: Compare evaluation value of each individual with its Pbest. The best evaluation value among the Pbest is denoted as gbest.

Step 5: Modify the member velocity *v* of each individual*K* according to (6.8)

$$v_{j.g}^{(t+1)} = w.v_{j.g}^{(t)} + c_1 * \text{rand}\,(\,) * (\text{pbest}_{j.g} - k_{j.g}^{(t)}) + c_2 * \text{rand}\,(\,) * (\text{gbest}_g - k_{j.g}^{(t)}) \quad (6.8)$$

j=1, 2…... n

g=1, 2...... m

Step 6:if $v_{j,g}^{(t+1)} > v_g^{max}$ then $v_{j,g}^{(t+1)} = v_g^{max}$

if$v_{j,g}^{(t+1)} < v_g^{max}$ then $v_{j,g}^{(t+1)} = v_g^{max}$

Step 7: Modify the member position of eachindividual *K* according to (6.9)

$$k_{j,g}^{(t+1)} = k_{j.g}^{(t)} + v_{j.g}^{(t+1)} \quad (6.9)$$

Such that $k_g^{min} = k_{j,g}^{(t+1)} = k_g^{max}$

Wherek_g^{min} and k_g^{max} the lower and upperbounds, respectively, of member *g* of the Individual *K*.For example, when *g* is 1, the lower and upper bounds ofthe controller parameter k_g^{min} and k_g^{max} respectively.

Step 8: If the number of iterations reaches the maximum then, go to Step 9. Otherwise, go to Step 2.

Step 9: The individual that generates the latest gbest is an optimal parameter. [30,32,33,34].

6.3.1 PSO–PID Controller Flowchart

The flowchart of the Particle Swarm Optimization based PID control system is shown in figure 6.2 [32].

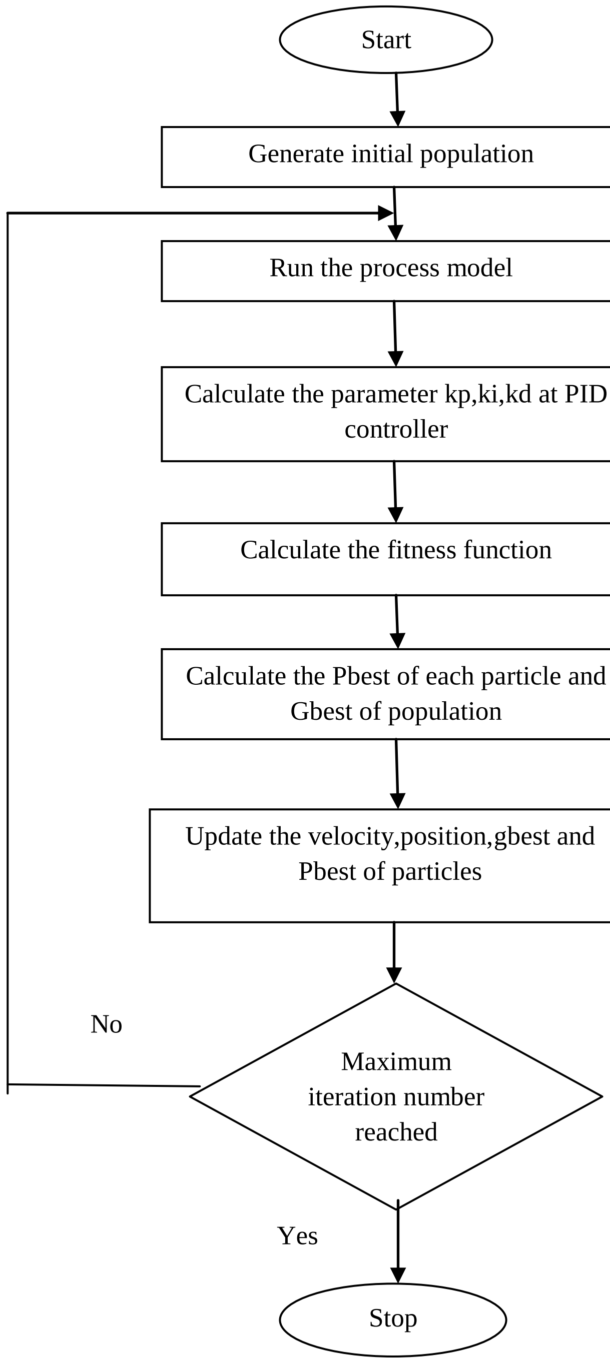
Start
Generate initial population
Run the process model
Calculate the parameter kp,ki,kd at PID controller
Calculate the fitness function
Calculate the Pbest of each particle and Gbest of population
Update the velocity,position,gbest and Pbest of particles
No
Maximum iteration number reached
Yes
Stop

Fig.6.2 Flowchart of PSO-PID control system

6.4 Implementation of PSO-PID Controller

The optimized PID tuning parameters of the CSTRsystem is obtained using Particle Swarm Optimization.PSO algorithm is implemented on Matlab environment.

6.4.1 PSO Parameters

To start up with PSO, certain parameters need to be defined. Selection of these parameters decides to a great extent the abilityof global minimization. The maximum velocity affects the abilityof escaping from local optimization and refining globaloptimization. The size of swarm balances the requirement ofglobal optimization and computational cost.Initializing the valuesof the parameters is given below:

Weight / Inertia of the system w= 0.5.

Acceleration constants c1 and c2= l.5.

Swarm population = 100.

Dimension of the search-space = 3 (k_p, ki, kd)

6.4.2 Calculation of Fitness Function

A particular point in the search-space is the best point for which the fitness function attains an optimum value. In this case, four components are taken to define the fitness function. The fitness function is a function of steady-state error, peak overshoot, rise time and settling time. However, the contribution of these component functions towards the original fitness function is determined by a scaling factor. Scaling factor (β) is chosen as 1 in this application.

The chosen fitness function is expressed as

$$F = \left(1 - e^{\beta}\right)(M_p + E_{ss}) + e^{-\beta}(T_s - T_r) \quad (6.10)$$

Where

F=fitness function

M_p =Peak overshoot

E_{ss}= Steady State Error

T_s=Settling Time

T_r= Rise Time

β =Scaling factor

The best fitness function value obtained after all iterations is -0.0445.

Matlab code for fitness calculation is shown in Appendix B.

Matlab code for PSO-PID controller is shown in Appendix C

6.5 Conclusion

In this chapter, particle swarm optimization technique of swarm intelligence along with tuning of PID controller with PSO has been developed. It is clear from above discussion that PSO technique can generate a high-quality solution within shorter calculation time and stable convergence characteristic than other stochastic methods. The main advantages of the PSO algorithm are summarized as: simple concept, easy implementation, robustness to control parameters, and computational efficiency when compared with mathematical algorithm and other heuristic optimization techniques.

CHAPTER 7

SIMULATION RESULTS AND THEIR COMPARATIVE ANALYSIS

7.1 General

In the following section, the simulation models developed in the previous chaptersare simulated .concentration and temperature control of CSTR is obtained by using different controlling mechanism. The results obtained are plotted to depict their effectiveness. Finally the results obtained from the different controllers are compared in terms of performance – overshoot, rise time, settling time, peak time, integral square error (ISE).

7.2 **Open Loop Responseof CSTR**

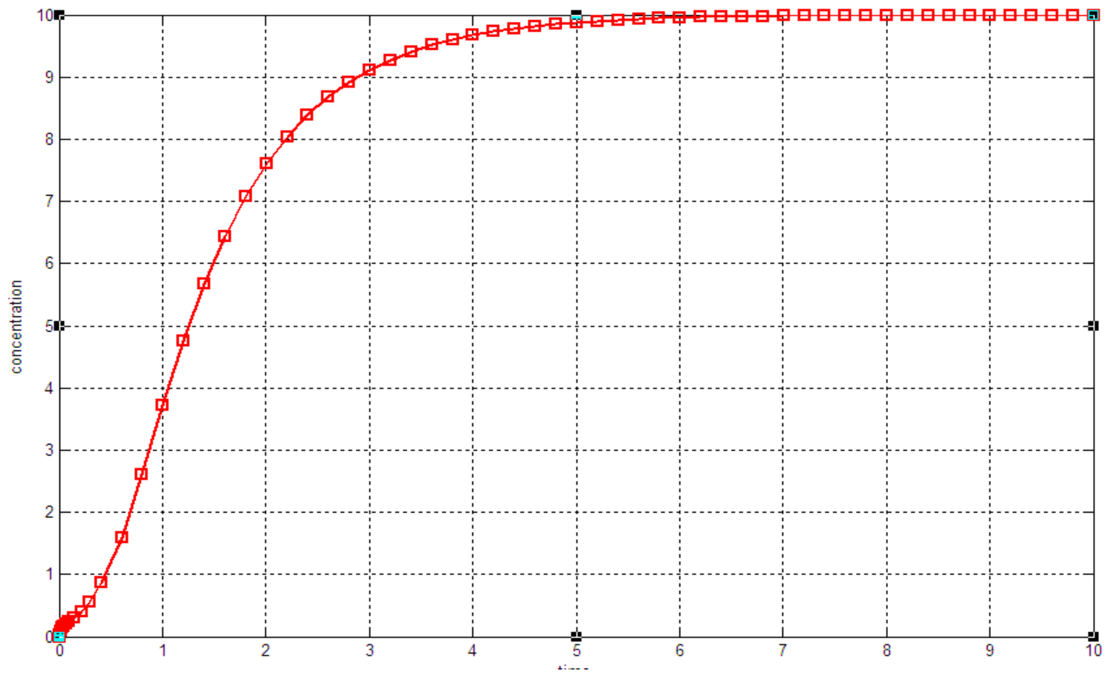

Fig.7.1 Open-loop response of concentration in CSTR

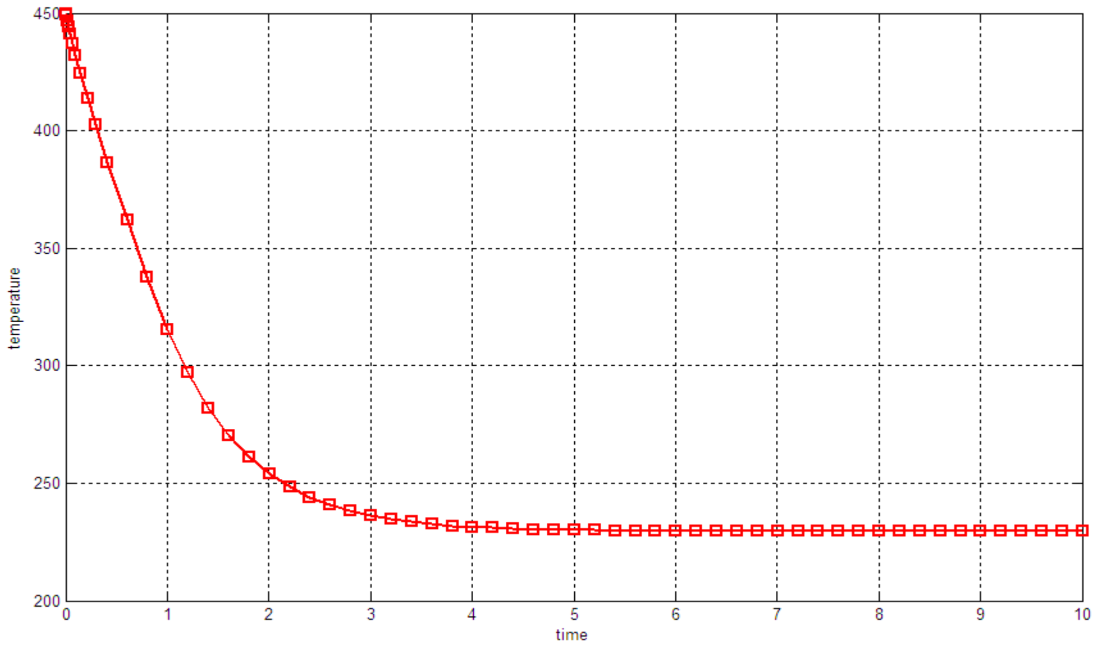

Fig.7.2 Open-loop response of temperature in CSTR

7.3 Response of CSTR with Model Reference Adaptive Control

The simulation model of the CSTR is simulated using the model reference adaptive control. For this purpose two methods of MRAC i.e. MIT rule and Lyapunov rule are used.

7.3.1.MIT rule

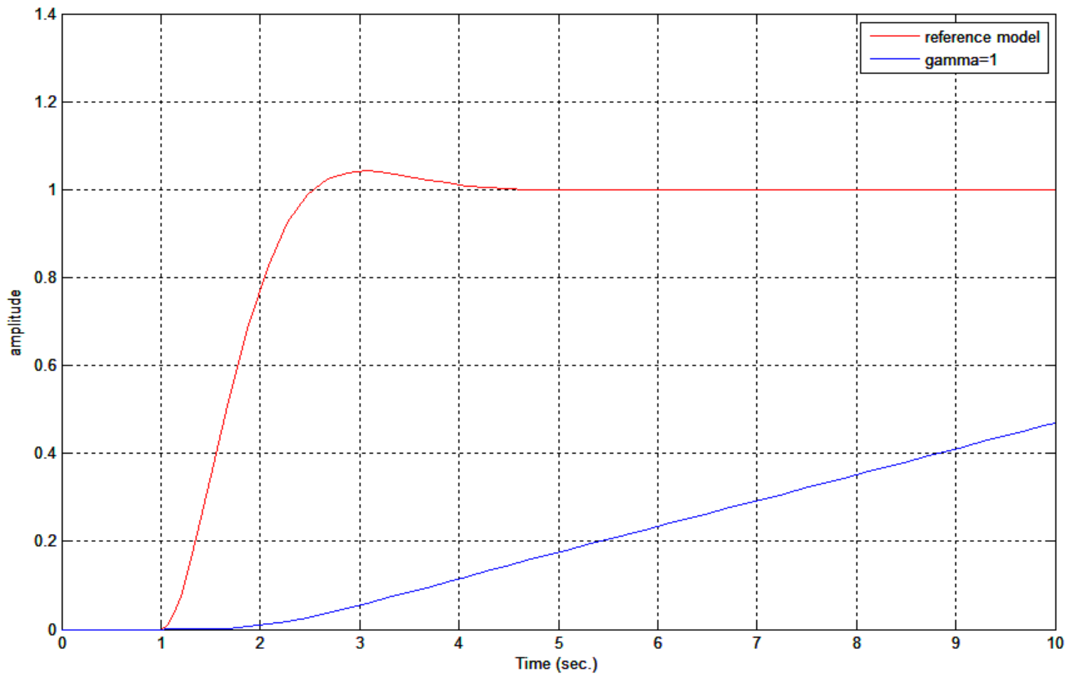

Fig.7.3 Output of CSTR with MIT Rule when gamma=1

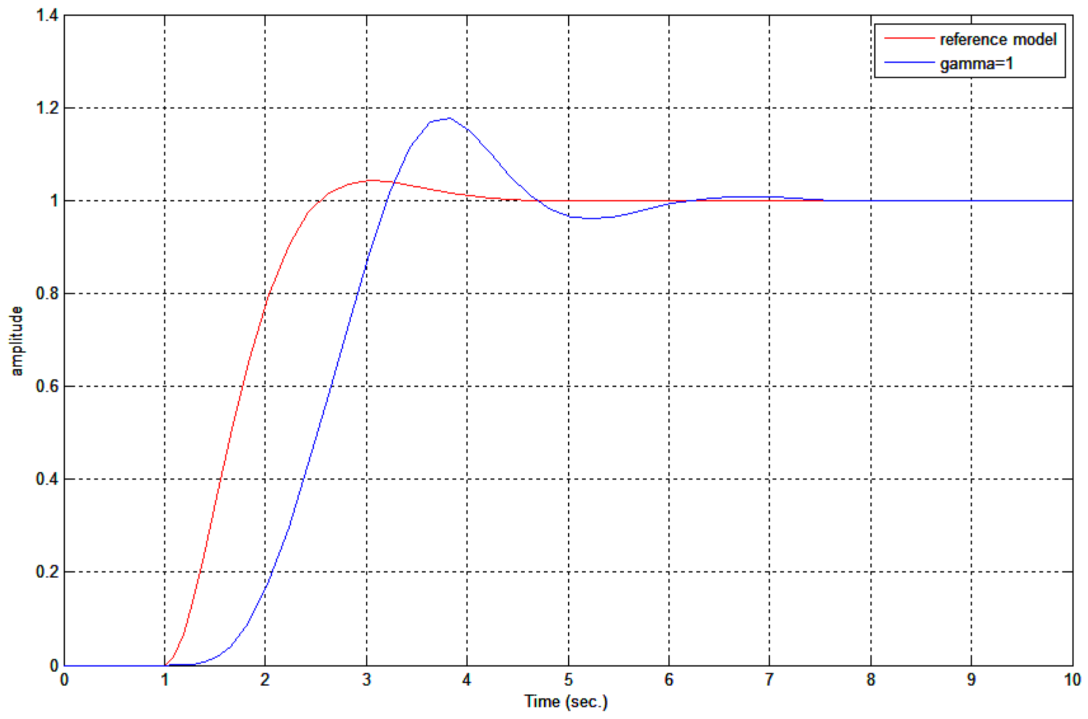

Fig.7.4 Output of CSTR with MIT Rule when gamma=10

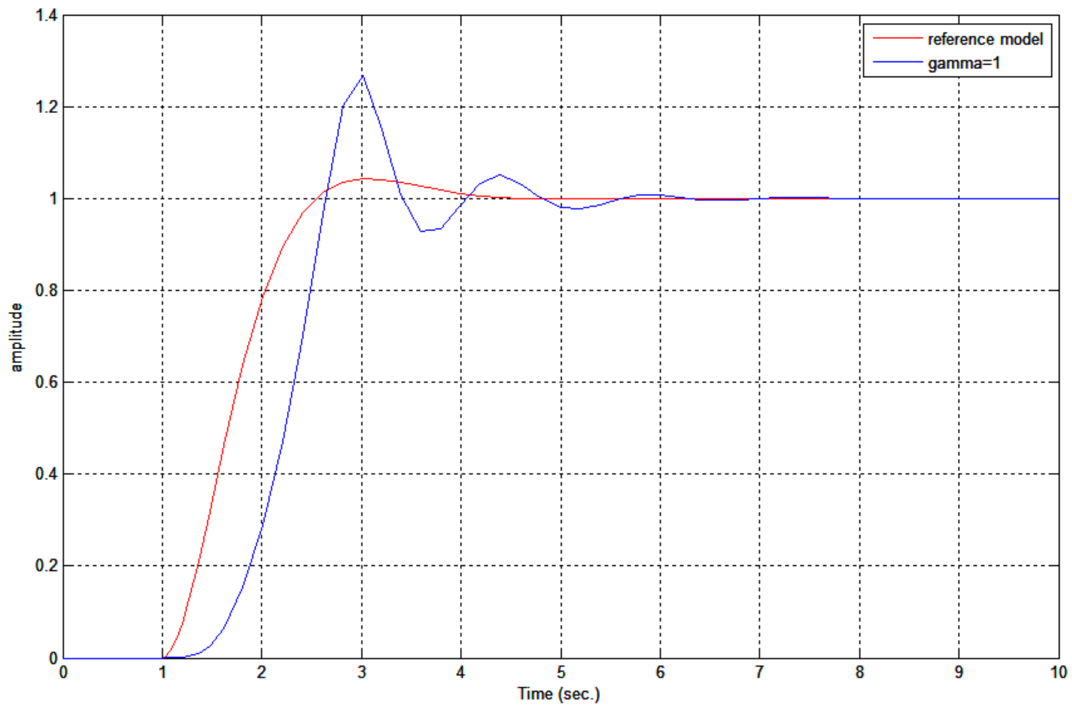

Fig.7.5 Output of CSTR with MIT Rule when gamma=50

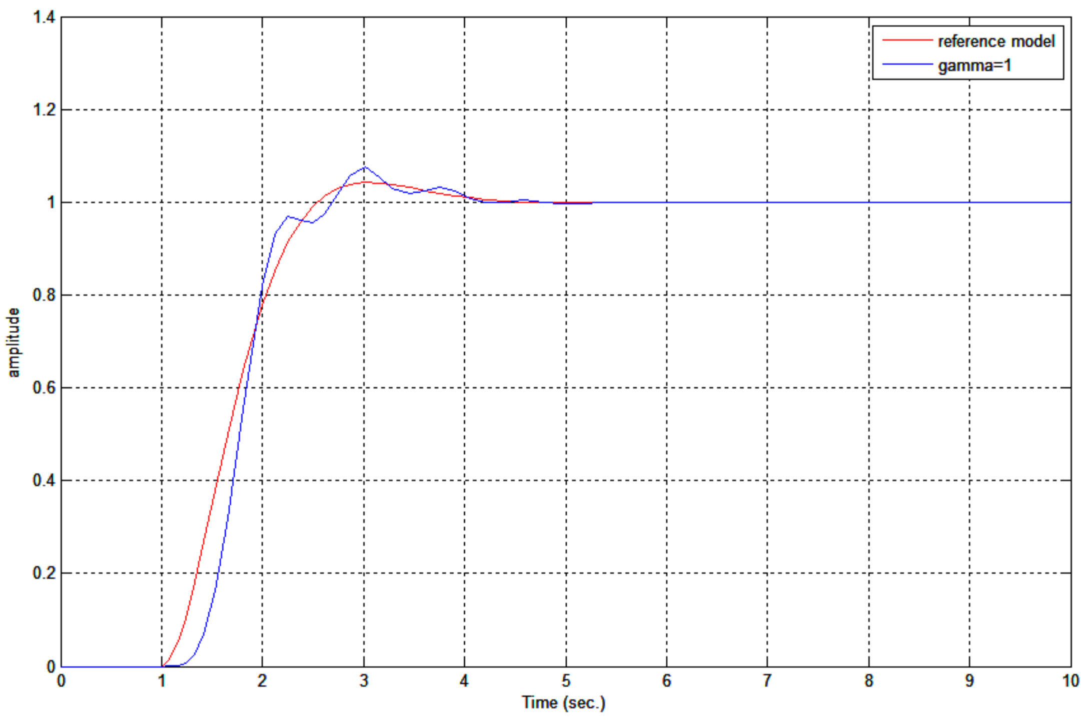

Fig.7.6 Output of CSTR with MIT Rule when gamma=100

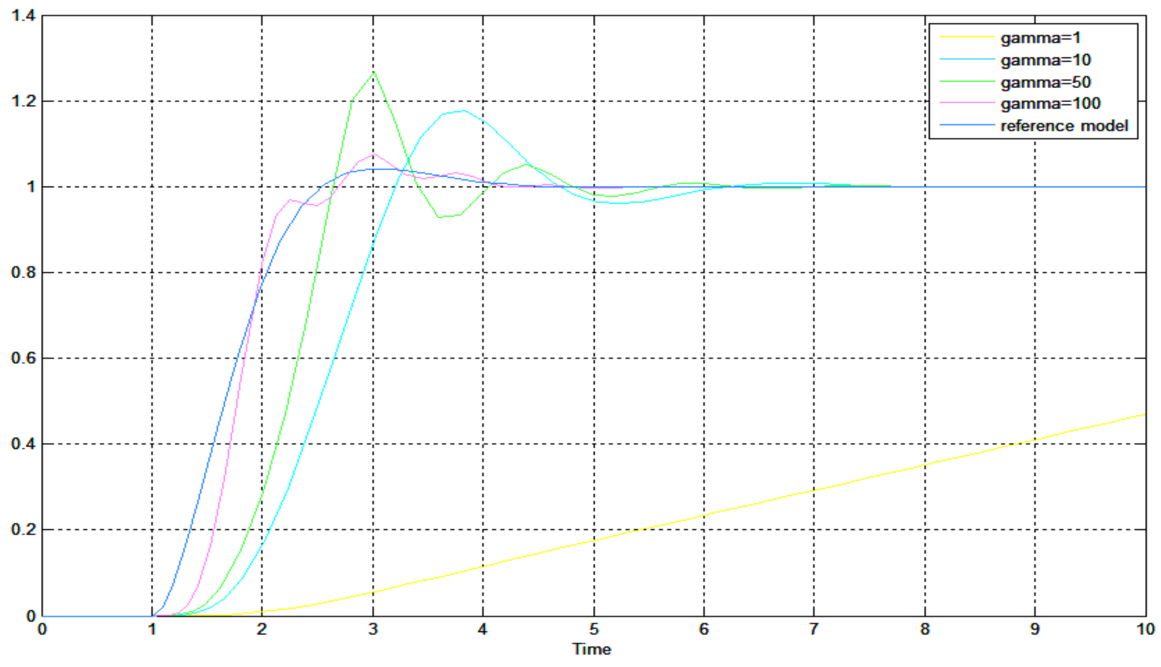

Fig.7.7 CSTR output with MIT rule for different values of adaption gain

7.3.2 Lyapunov Rule

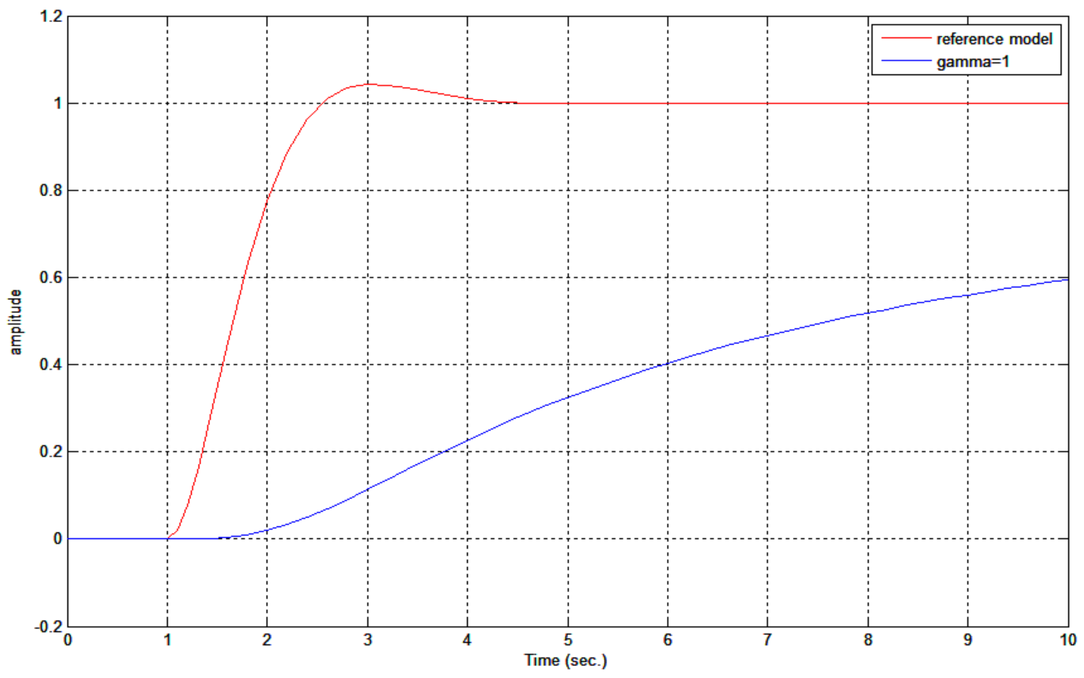

Fig.7.8 Output of CSTR with Lyapunov Rule when gamma=1

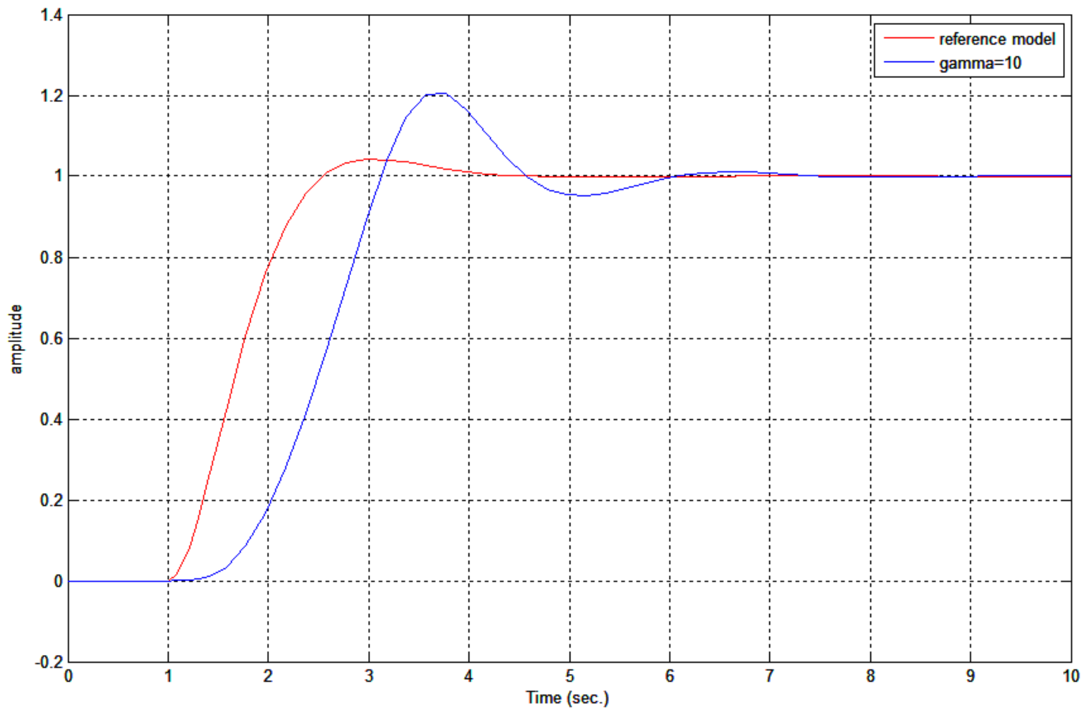

Fig.7.9 Output of CSTR with Lyapunov Rule when gamma=10

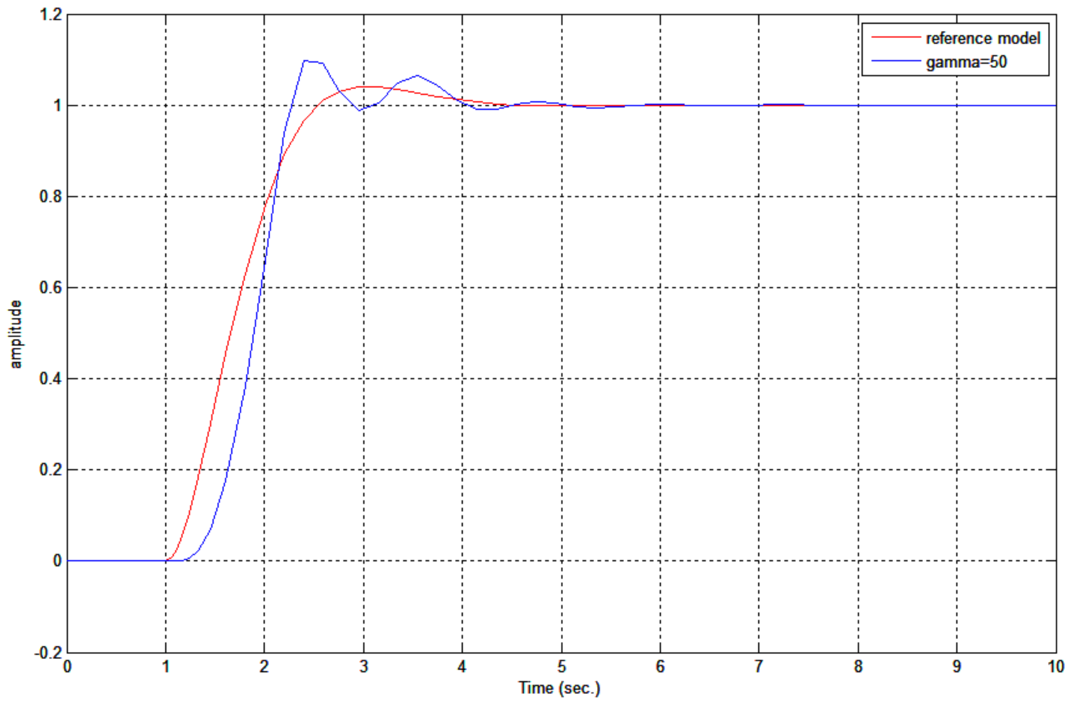

Fig.7.10 Output of CSTR with Lyapunov Rule when gamma=50

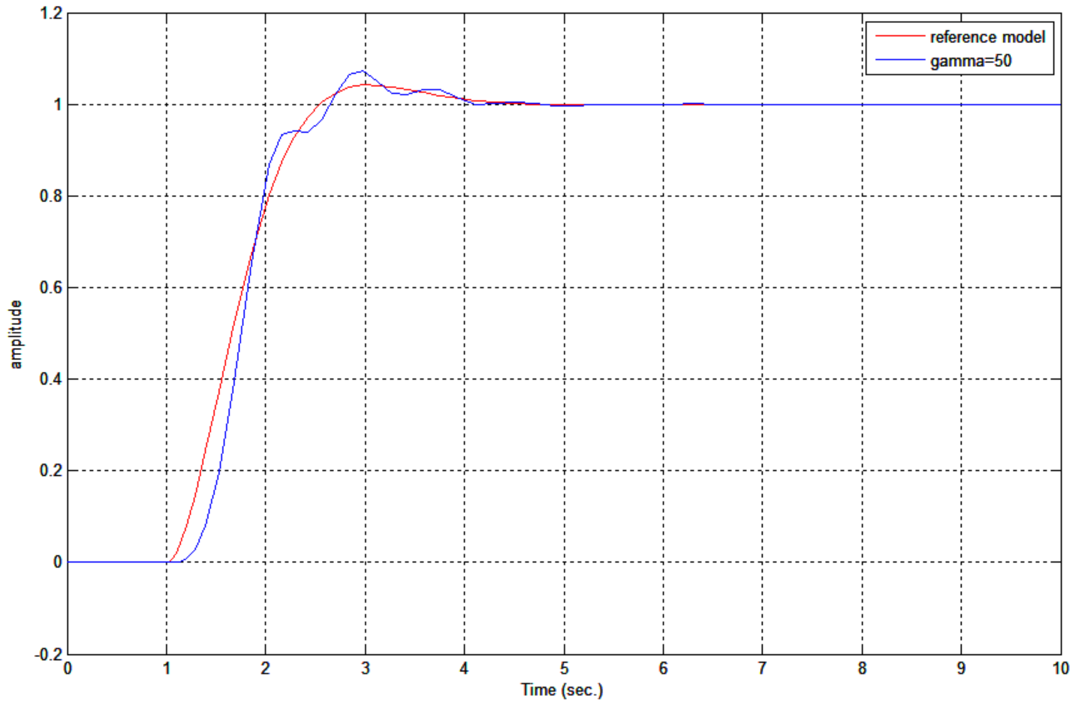

Fig.7.11 Ooutput of CSTR with Lyapunov Rule when gamma=100

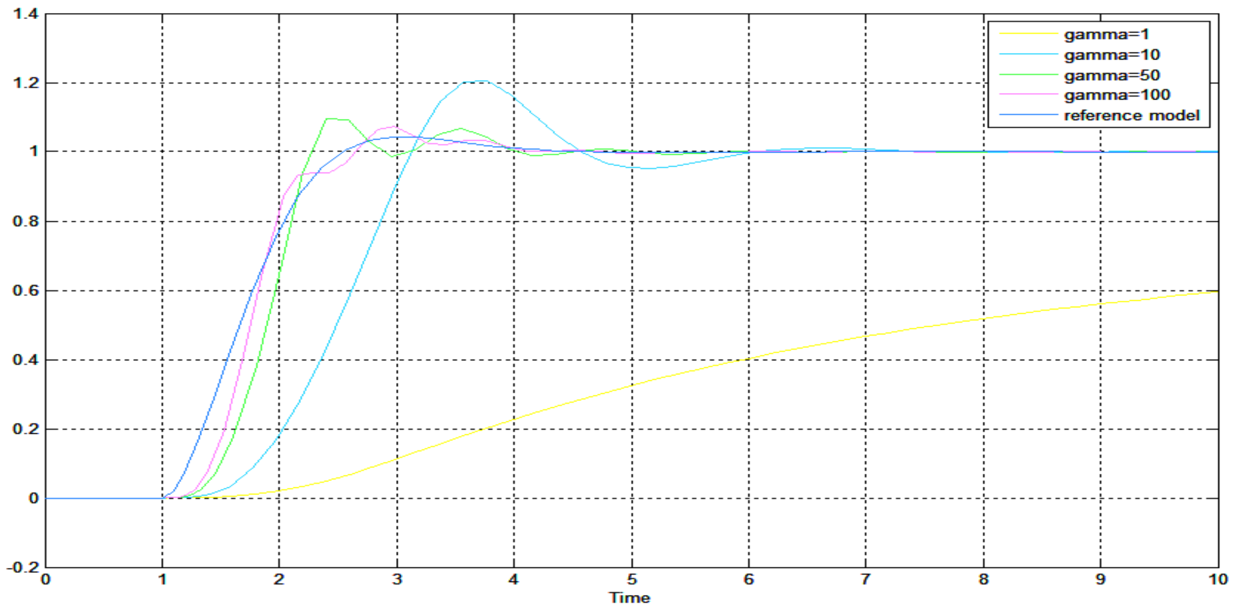

Fig.7.12 CSTR output with Lyapunov Rule for different values of adaption gain

7.4 Response of CSTR with PID Controller (Ziegler-Nicholas Method)

The simulation model of the CSTR is simulated using the PID controller. For tuning of PID controller Ziegler- Nicholas method is used.

Table 7.1 PID tuning parameters using Zeigler-Nicholas method

Tuning Method	K_p	K_i	k_d
ZNT	9.2675	37.911	-.678

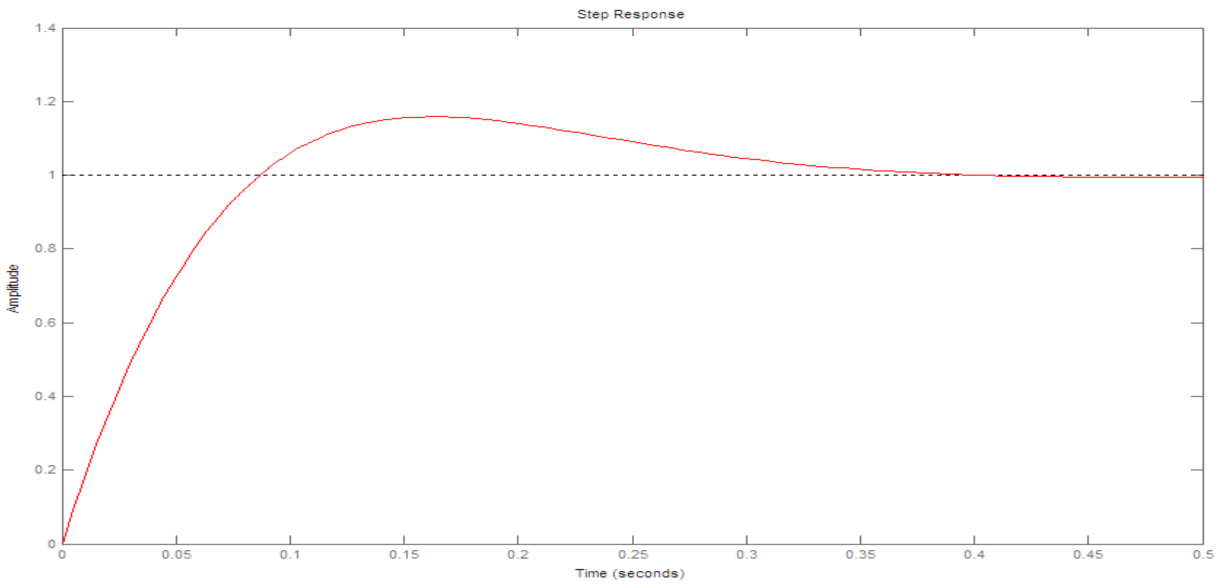

Fig.7.13 CSTR output with PID controller using Ziegler-Nicholas method

7.5 Response of CSTR with PSO-PID Controller

The simulation model of the CSTR is simulated using the particle swarm optimization method. Using PSO, parameters of PID controller are tuned.

Table 7.2 Optimized PID tuning parameters of PSO for different iterations

No. of iterations	K_p	K_i	k_d
1	0.051459	0.49844	0.17948
2	0.055658	0.5149	0.10265
3	0.057726	0.63898	0.062685
4	0.05612	0.76378	0.081139
5	0.053816	0.85906	0.063554

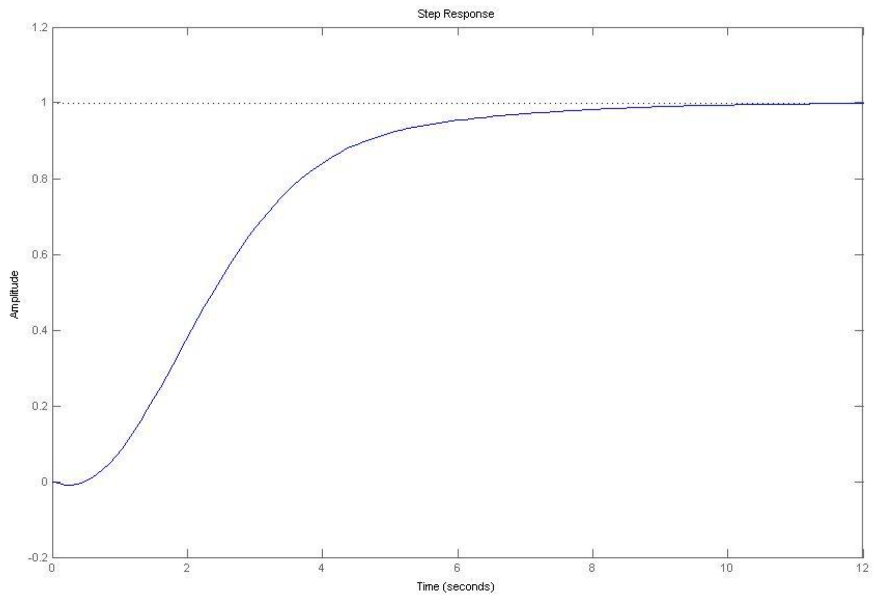

Fig.7.14 Step response of CSTR with PSO-PID Controller

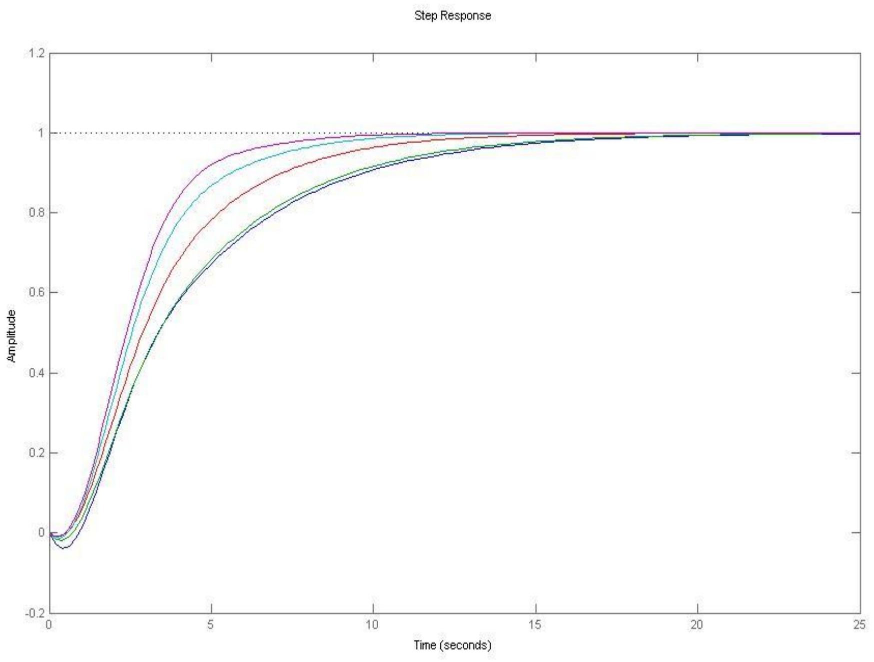

Fig.7.15 Plot of PSO parameters for different iterations

7.6 Comparative Analysis of Results of ZNT, MRAC and PSO-PID Controller

A comparative analysis has been made by using different types of controlling mechanism. Firstly controlling of CSTR is done by MRAC, then controlling is done be tuning of PID controller. For tuning of PID controller two methods are used one is conventional method of tuning i.e. Zeigler-Nicholas and other is evolutionary method of tuning i.e. Particle Swarm Optimization is used**.**

Table7.3 Comparison results

Performance specification	MIT Rule				Lyapunov Rule				ZNT	PSO
	$\gamma = 1$	$\gamma = 10$	$\gamma = 50$	$\gamma = 100$	$\gamma = 1$	$\gamma = 10$	$\gamma = 50$	$\gamma = 100$		
Rise time (sec.)	-	3.275	2.658	2.785	-	3.15	2.3	2.6	.36	4.47
Peak time (sec.)	-	3.83	3.0	3.017	-	3.8	2.4	3	1.0	17.0
Maximum Overshoot (%)	-	17.83	26.74	7.5	-	20.34	9.65	7.3	7.1	0
Settling time (sec.)	-	7.0	6.25	4.5	-	6.4	4	3.9	1.4	8.65
ISE	5.023	.4052	.2147	.0204		.3799	.0464	.01412	.16	.003

7.7 Conclusion

The modeling, analysis, design and the simulation of the CSTR with different types of controllers has been done in the MATLAB/Simulink environment. A thorough comparative analysis has been carried out on CSTR performance with different controllers. It has been shown that the individual controllers have their own merits and demerits. The choice of selection of controller for a particular application should be based on typical requirement. When the requirement is of simplicity and ease of application, a Z-N tuned PID controller is of a good choice. When the need is of less integral square error and fast dynamic response, then, the model reference adaptive control can be selected. When the requirement is of both intelligent response and good steady state performance with minimum overshoot and least error, optimization based PSO-PID controller is a better choice.

CHAPTER 8

MAIN CONCLUSION AND SUGGESTION FOR FURTHER WORK

8.1 General

The modeling of Continuously Stirred Tank Reactor with different typeof controller has been successfully carried out using MATLAB/Simulink environment. Different types of controller i.e Ziegler Nicholas, Model Reference Adaptive Control, Particle Swarm Optimization are used for simulation study for assessment of the performance of CSTR.The present chapter summaries all the investigations carried out right and accordingly main conclusion are derived and suggestions for further work are also presented.

8.2 Main Conclusion

The MATLAB/Simulink environment has been extensively used for simulation of model of CSTR using various controllers. The mathematical model of CSTR is obtained by solving the differential equations. The response of thetemperature and concentration of the reactant for the step change in coolant flow rate is obtained.

The proposed adaptive controller is tested by using Matlab Simulink program and its performance is compared to both conventional controller and PSO based PID controller, Adaptive controllers are very effective where parameters are varying. The controller parameters are adjusted to give desired result.

Time response is studied for CSTR system using MIT rule and Lyapunov rule with varying the adaptation gain. It is observed that in case of MIT rule, if adaptation gain increases the time response of the system also is improved for the chosen range of adaption gain and further system is unstable in the upper range. In Lyapnove rule, system is stable beyond thechosen range of adaptation gain. So with suitable value of adaptation gain in MIT rule and Lyapunov rule plant output can be made close to reference model. It can be concluded that performance using Lyapunov rule is better than the MIT rule. The simulation shows that very good conversion can be achieved and at the same time the temperature inside the reactor do not violate the safety concentrations, even when there are large disturbances in the feed concentrations. The proposed process control system increases the safety of operations by reducing the impact from external disturbances. This will decrease the risk of unnecessary shutdowns of the process operation and also reduce the power consumption in industrial interactive thermal process by effective recycling of heat.

PID controller tuning is done using a standard tuning algorithm (Ziegler-Nichols Tuning),and a more advanced swarm intelligence approach(Particle Swarm Optimization). By comparing these two methods, it is found that PSO algorithm is the best implemented. Also, the PID controller parameters obtained from PSO algorithm gives better tuning resultthan the Z-N tuning methods. This is also validated by checking for the robustness of PSO algorithm and it is concluded that the system exhibited best performance with PSO.

Though PSO algorithm is much advanced and simpler than other artificial intelligence based approach, it has its own short-comings. In this application, PSO algorithm is taken and applied for single set of data. Generally this can be seen as alimitation in terms of not being able to analyse multiple sets of data.

8.3 Suggestion for Further Work

The proposed PSO based PID controller and adaptive controller shows better results for controlling of CSTR.In future both adaptive control and PSO based control can be implemented with other artificial intelligence techniques,i.e adaptive control can be applied with neural network ,fuzzy control etc.Similarly PSO can be applied with these artificial intelligence techniques. Apart from this other swarm intelligence based algorithms like Artificial Bee Colony Algorithm, Ant Colony Algorithm can be applied for controlling of CSTR.

An effort to further reduce the complexity may be investigated.

REFERENCES

[1] Jiri Vo jtesek ,PetrDostal, "simulation analysis of Continuous Stirred Tank reactor", IEEE transactions, Vol 24, pp 27-31, December, 2008.

[2] Dr. M.J.Willis," continous stirred tank reactor models", march2000.

[3] Dale E. Seborg, Thomas F. Edgar, Duncan A. Mellichamp," Process dynamics and control",Wiley,2004.

[4] B.Wayne Bequette, "Process Control, Modeling, Design and Simulation", Prentice-Hall of India Private Limited, India,2003.

[5] Richard C.Dorf,Robert H. Bishop," Modern control system ",Pearson,Eleventh Edition.

[6] Mohammad Shahrokhi,Alireza Zomorrodi," Comparison of PID Controller Tuning Methods".

[7] Zahra'a F. Zuhwar,"The Control of Non Isothermal CSTR Using Different Controller Strategies", Iraqi Journal of Chemical and Petroleum Engineering, Vol.13 (3),pp. 35- 45,2012.

[8] Glan Devadhas.G and Pushpakumar.S, "An Intelligent Design of PID Controller for a Continuous Stirred Tank Reactor", World Applied Sciences Journal 14(5).pp.698-703, 2011.

[9] Karl.J.Astrom, Dr. Bjorn Wittenmark," adaptive control",Addison-Wasley Company,second edition.

[10] Rahul Upadhyay, Rajesh Singla." Analysis of CSTR Temperature Control with Adaptive and PID Controller (A Comparative Study), IACSIT International

Journal of Engineering and Technology, Vol.2 (5), pp.453-458, 2010.

[11] S.Jegan,K.Prabhu,"Temperature control of CSTR process using Adaptive Control", International Conference on Computing and Control Engineering (ICCCE), 2012.

[12].K.Prabhu, Dr. V. Murali Bhaskaran,"Optimization of a Control Loop Using Adaptive Method",International Journal of Engineering and Innovative Technology (IJEIT), Vol.1(3),pp.133-138, 2012.

[13] Pankaj Swarnkar,Shailendra Jain, R.K.Nema," Effect of Adaptation Gain in Model Reference Adaptive Controlled Second Order System",ETASR – Engineering, Technology & Applied Science Research Vol. 1(3),pp. 70-75, 2011

[14]Rahul Upadhyay and Rajesh Singla, "Application of adaptive control in a process control", International Conference on Education Technology and Computer.pp.323-327, 2010.

[15]ComanAdrian, AxenteCorneliu, BoscoianuMircea,"the simulation of the adaptive systems using the MIT rule", 10th WSEAS Int. conf. on mathematical methods and computational techniques in electrical engineering (MMACTE),pp.301-305,2008.

[16] R.Aruna,M.SenthilKumar,D.Babiyola,"Intelligence based and Model based Controllers to the Interactive Thermal Process,International Conference on VLSI, Communication & Instrumentation (ICVCI),pp. 24-28,2011.

[17] Rajiv Ranjan, Dr. Pankaj Rai,"Performance analysis of a second order system using MRAC, International Journal Of Electrical Engineering Technology(IJEET), Vol.3(3), pp. 110-120,2012.

[18] Vojtesek.J,Dostal,P.,” simulation of adaptive control of CSTR”, International Journal of Simulation & Modeling,vol.8(3),pp.133-144,2009.

[19] R.Aruna, M.Senthil Kumar,”Adaptive control for interactive thermal process”, International Conference on Emerging Trends in Electrical and Computer Technology ICETECT, 2011.

[20] T. van den Boom and B de Schutter, “Lecture notes Optimization in Systems and Control”, TU Delft, 2009.

[21] Binitha S, S Siva Sathya,” A Survey of Bio inspired Optimization Algorithms”, International Journal of Soft Computing and Engineering (IJSCE),Vol.2(2),2012

[22] G. Glan Devadhas, S. Pushpakumar, S.V. Muruga Prasad,” Intelligent Computation of Controller Using Optimisation Techniques for a Nonlinear Chemical Process”, International Journal of Research and Reviews in Soft and Intelligent Computing (IJRRSIC) Vol.1(3), pp.49-55, 2011.

[23] T. O..Mahony, C J Downing and K Fatla., “Genetic Algorithm for PID parameter Optimization: Minimizing Error Criteria”, Process Control and Instrumentation, University of Stracthclyde, pp.148- 153,2000.

[24] A.Varsek, T. Urbacic and B. Filipic, Genetic Algorithms in Controller Design and Tuning, IEEE Trans. Sys. Man and Cyber, Vol. 23(5), pp1330-1339, 1993.

[25] Gaing, Z.L.,” A particle swarm optimization approach for optimum design of PID controller in AVR system”, IEEE Transaction on Energy Conversion, Vol.19(2), pp.384-391,2004.

[26] Zhao, J., Li, T. and Qian, J., Application of particle swarm optimization algorithm on robust PID controller tuning. Advances in Natural Computation,

pp.948-957,2005.

[27] James Kenned, Russell Eberhart," Particle Swarm Optimization",*Proc. IEEE Int.* Conf. Neural Networks, Vol.4, pp.1942-1948, 1995.

[28] Mohammad Ali Nekoui, Mohammad Ali Khameneh,Mohammad Hosein Kazemi,"Optimal Design of PID Controller for a CSTR System Using Particle Swarm Optimization", International Power Electronics and Motion Control Conference, (EPE-PEMC), 2010.

[29] Mehdi Nasri, Hossein Nezamabadi-pour,Malihe Maghfoori,"A PSO-Based Optimum Design of PID Controller for a Linear Brushless DC Motor", World Academy of Science, Engineering and Technology, 2007.

[30] Y. Yan,W.A. Klop,M. Molenaar, P. Nijdam,"Tuning a PID controller: Particle Swarm Optimization versus Genetic Algorithms"2010.

[31] Mahmud Iwan Solihin,Lee Fook Tack,Moey Leap Kean,"Tuning of PID controller Using Particle Swarm Optimization (PSO)", International Conference advanced Science,Engineering and Information Technology, 2011.

[32] Geetha. M,Balajee.K.A, Jovitha Jerome," Optimal Tuning of Virtual Feedback PID Controller for a Continuous Stirred Tank Reactor (CSTR) using Particle Swarm Optimization (PSO) Algorithm", IEEE-International Conference On In Engineering, Science And Management (lCAESM), 2012.

[33] S.M.GirirajKumar,Deepak Jayaraj,Anoop.R.Kishan,"PSO based Tuning of a PID Controller for a High Performance Drilling Machine", International Journal of Computer Applications,Vol.1(19),2010.

[34] Mohamed .M. Ismail,"Adaptation of PID Controller using AI Techniques for

Speed Control of Isolated Steam Turbine”.International Journal of Control, Automation and Systems,Vol.1(1), 2012 .

[35] B. Nagaraj, P. Vijayakumar,”A comparative study of PID controller tuning using GA, EP, PSO & ACO”, Journal of Automation, Mobile Robotics & Intelligent Systems,Vol.5(2),2011

[36] Wan Azhar Wan Yusoff,Nafrizuan Mat Yahya,Azlyna Senawi,”Tuning of Optimum PID Controller Parameter Using Particle Swarm Optimization Algorithm Approach”.

[37] Zwe-Lee Gaing,”A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System”, IEEE Transactions on Energy Conversion,Vol.19(2), 2004

[38] S. Easter Selvan, Sethu Subramanian, S. Theban Solomon,” Novel Technique for PID Tuning by Particle Swarm Optimization”.

[39] Pranay Lahoty, Girish Parmar,” A Comparative Study of Tuning of PID Controller using Evolutionary Algorithms”,International Journal of Emerging Technology and Advanced Engineering”,Vol.3(1),2013.

APPENDIX A

<u>STATE SPACE MODEL OF CSTR:</u> cstr_ss_model.m

```
function smdl=cstr_ss_model
% Measured Product
 Ca=7.59; % mol/lit
% Concentration (Cd
% Reactor Temperature (T)
 T=313.17;
% Volumetric Flow rate (q)
 q=100;% Llmin
% Reactor Volume (V)
 V=100;
% Feed Concentration (CAf)
 Caf=1; % mol/lit
% Feed Temperature (Tf)
 Tf=312; % K
 TJ=300;
% Coolant Temperature (T'f) 350 K
% Coolant Flow rate( qc) 100 Llmin
 delH=6000;
% Heat of Reaction (6.H) 2e5 cal/mol
% Reaction rate constant(ko) 7.2elO min"
 k0=10e15;
% Activation energy
 ER=5963.6;
% term(E/R)
% Heat transfer term (hA) 7e5 cal/(min.K)
 rowcp=500;
 UAV=145;
% Liquid Density(p, Pc)
% Specific Heat capacity (Cp)
 ks=.175;
 kg=.0106;
 A11=(-q/V)-ks;
 A12=-kg*Ca;
 A21=(-delH*ks/(rowcp));
 f1=-q/V;
 f2=((-delH/(rowcp))*Ca*kg);
 f3=(-UAV/rowcp);
 A22=(f1+f2+f3);
 B11=0;
 B12=(UAV/rowcp);
C11=0;
 C12=1;

 AMAT=[A11 A12;A21 A22];
 BMAT=[B11 ;B12];
 CMAT=[C11 C12];
 DMAT=0;
 smdl=ss(AMAT,BMAT,CMAT,DMAT);
```

APPENDIX B

CALCULATION OF FITNESS FUNCTION: fitness_ process.m

```
function final_fit=fitness_ process(papra,err)


 rs_tm=papra.RiseTime;
 st_tm=papra.SettlingTime;
 stmin_tm=papra.SettlingMin;
 stmax_tm=papra.SettlingMax;
 over_tm=papra.Overshoot;
 under_tm=papra.Undershoot;
 peak_val=papra.Peak;
 peak_tm=papra.PeakTime;
 f1=1-exp(1);
 f2=peak_val+err;
 f3=exp(-1);
 f4=st_tm-rs_tm;
 final_fit=(f1*f2)+(f3*f4);
```

APPENDIX C

FINAL RUN DESIGN:final_run_design.m

```
 clear all
 close all
 clc
 warning off;
 design_name='final_cstr1';
 sim(design_name,10);
 amp_cnt=cont.signals.values;
 t_cnt=cont.time;
 figure,plot(t_cnt,amp_cnt,'r-s','linewidth',2);
 xlabel('time');
 ylabel('concentration');
 grid on;

  amp_temp=temp.signals.values;
 t_temp=temp.time;
 figure,plot(t_temp,amp_temp,'r-s','linewidth',2);
 xlabel('time');
 ylabel('temperature');
 grid on;

 No_of_iter= 5;
 intr_value = 0.5;
 cf_value = 1.5;
 int_pop= 100;

index = 1;
intial_ctrl_value1=[];
for k2=1:int_pop
        intial_ctrl_value1(index, 1, 1) = rand/2;
        intial_ctrl_value1(index, 1, 2) = rand/2;
        intial_ctrl_value1(index, 1, 3) = rand/2;
        index = index + 1;

end
intial_ctrl_value1(:, 4, 1) = 1000;
intial_ctrl_value1(:, 2, :) = 0;

intial_ctrl_value=intial_ctrl_value1;

for iter=1 : no_of_iter

    for i= 1 : int_pop
        intial_ctrl_value(i, 1, 1) = intial_ctrl_value(i, 1, 1) + intial_ctrl_value(i, 2, 1)/1.3;
        intial_ctrl_value(i, 1, 2) = intial_ctrl_value(i, 1, 2) + intial_ctrl_value(i, 2, 2)/1.3;
        intial_ctrl_value(i, 1, 3) = intial_ctrl_value(i, 1, 3) + intial_ctrl_value(i, 2, 3)/1.3;

        kp=intial_ctrl_value(i, 1, 1);
        ki=intial_ctrl_value(i, 1, 2);
        kd=intial_ctrl_value(i, 1, 3);



        mdl=cstr_ss_model;
        [papra err final_mdl]=pid_process(mdl,kp,ki,kd);
        final_fit=fitness_process(papra,err);
        fit_val(i)=final_fit;



        if (final_fit < intial_ctrl_value(i, 4, 1) & ~isnan(final_fit))
            intial_ctrl_value(i, 3, 1) = intial_ctrl_value(i, 1, 1);
            intial_ctrl_value(i, 3, 2) = intial_ctrl_value(i, 1, 2);
            intial_ctrl_value(i, 3, 3) = intial_ctrl_value(i, 1, 3);
            intial_ctrl_value(i, 4, 1) =final_fit;
        end
    end
    [temp, gbest] = min(intial_ctrl_value(:, 4, 1));

    for i = 1 : int_pop
  intial_ctrl_value(i, 2, 1) = (rand*intr_value*intial_ctrl_value(i, 2, 1) +
cf_value*rand*(intial_ctrl_value(i, 3, 1) - intial_ctrl_value(i, 1, 1)) +
cf_value*rand*(intial_ctrl_value(gbest, 3, 1) - intial_ctrl_value(i, 1, 1)));
  %x velocity component
  intial_ctrl_value(i, 2, 2) = (rand*intr_value*intial_ctrl_value(i, 2, 2) +
cf_value*rand*(intial_ctrl_value(i, 3, 2) - intial_ctrl_value(i, 1, 2)) +
cf_value*rand*(intial_ctrl_value(gbest, 3, 2) - intial_ctrl_value(i, 1, 2)));
  %y velocity component
  intial_ctrl_value(i, 2, 3) = (rand*intr_value*intial_ctrl_value(i, 2, 3) +
cf_value*rand*(intial_ctrl_value(i, 3, 3) - intial_ctrl_value(i, 1, 3)) +
cf_value*rand*(intial_ctrl_value(gbest, 3, 3) - intial_ctrl_value(i, 1, 3)));
 %y velocity component
          end
     [temp, gbest] = min(intial_ctrl_value(:, 4, 1));
    kp=intial_ctrl_value(gbest, 1, 1);
     ki=intial_ctrl_value(gbest, 1, 2);
     kd=intial_ctrl_value(gbest, 1, 3);
     [papra err final_mdl]=pid_process(mdl,kp,ki,kd);

     final_pid_val(iter,1:4)=[iter kp ki kd];

    design_out{iter}=final_mdl;

    end

fprintf('%15.13s %15.13s %15.13s %15.13s','iteration','kp','ki','kd');
fprintf('\n');

for k3=1:no_of_iter

fprintf('%15.13s %15.13s %15.13s %15.13s',num2str(final_pid_val(k3,1)),...
                                        num2str(final_pid_val(k3,2)),...
                                      num2str(final_pid_val(k3,3)),...
                                      num2str(final_pid_val(k3,4)));



fprintf('\n');
end
%%
figure,
for k3=1:no_of_iter
    step(design_out{k3});
  hold on;
end

  figure,step(design_out{end});

  %% pid parameter based PSO

  FINAL_KP=final_pid_val(end,2)
  FINAL_KI=final_pid_val(end,3)
  FINAL_KD=final_pid_val(end,4)
```