CHAPTER 6

CONTROLLER OPTIMIZATION

This chapter describes about the tuning of controller parameters in PI controllers, that is, methods for finding proper values of $K_p$ and $K_d$.

6.1 INTRODUCTION

The PI (proportional plus integral) controller function is the most frequently used controller function in practical applications. The PI controller stems from a PID controller with the D-term (derivative) deactivated. The D-term is often deactivated because it amplifies random (high-frequent) measurement noise, causing abrupt variations in the control signal. In most practical applications the continuous-time PI controller is implemented as a corresponding discrete-time algorithm based on a numerical approximation of the integral term. Typically, the sampling time of the discrete-time controller is so small when compared to the dynamics (response-time or time-constant) of the control system. There is no significant difference between the behavior of the continuous-time PI controller and the discrete-time PI controller. Consequently, the sampling time does not contribute to tuning parameter.

The family of PID controllers is rigidly known as the building blocks of control systems owing to their simplicity and ease of implementation. Although the PID controllers have gained widespread usage across technological industries, it must also be pointed out that the unnecessary mathematical rigor, preciseness and accuracy involved with the design of the controllers have been a major drawback. Designing and tuning a PID controller appears to be conceptually intuitive, but can be hard in practice, if multiple objectives are to be achieved (Huailin Shu and Youguo Pi 2005). However various techniques and modifications to the conventional PID controllers have been employed in order to overcome these difficulties. This includes the use of
auto tuning PID controllers, adaptive PID controllers and also the implementation of compensation schemes to the PID controllers.

But as a whole the PID controllers are with some drawbacks of design of the controllers for electric drives. In simple PID controllers it is difficult to generate a derivative term in the output that has any significant effect on motor speed. It can be deployed to reduce the rapid speed oscillation caused by high proportional gain. However in many controllers, it is not used. The derivative action causes the noise (random error) in the main signal to be amplified and reflected in the controller output. Hence the most suitable controller for speed control is PI type controller. The proportional term does the job of fast acting correction which will produce a change in the output as quickly as the error arises. The integral action takes a finite time to act but has the capability to make the steady state speed error zero.

Genetic Algorithm which is adopted from the biological evaluation, is an efficient search technique that manipulates the coding representing a parameter set to search a near optimal solution through cooperation and competition among the potential solutions (Ravi and Balakrishnan 2011). This algorithm is highly relevant for industrial applications, because it is capable of handling problems with nonlinear constraints, multiple objectives and dynamic components. Genetic Algorithm is composed of two main elements which are strongly related to the problems being solved by the encoding scheme and the evaluation function. The encoding scheme is used to represent the possible solutions to the problem. Individual parameters can be encoded in some alphabets like binary strings, real numbers and vectors. While applying Genetic Algorithm practically, a population pool of chromosomes is installed and it is set to a random value. In each cycle of genetic evaluation, a subsequent generation is created from the chromosomes in the current population. The cycle of evaluation is repeated until a termination criterion is reached. Fitness value can be set as the termination criterion.
6.2. SYSTEM DESCRIPTION

The transfer function of the continuous system with controller is given as (Palaniswami and Sivanandam 2000):

\[
T(s) = \frac{G(s)G_c(s)}{1 + G(s)G_c(s)}
\]  

(6.1)

where \(G_c(s)\) is the transfer function of the continuous controller.

The transfer function of the discrete system with controller is given as (Ravichandran et al 2007):

\[
T(z) = \frac{G(z)G_c(z)}{1 + G(z)G_c(z)}
\]  

(6.2)

where \(G_c(z)\) is the transfer function of discrete controller.

6.2.1 Algorithm for the Design of PI Controller

The following performance criteria are selected for controller design (Manigandan et al 2005).

- Settling time \(\leq 3\) seconds
- Peak overshoot \(\leq 2\%\)
- Steady state error \(\leq 1\%\)

The following steps are considered for the design of PI controller.

- Read the open loop transfer function of the given higher order system.
- Form the closed loop transfer function.
- Obtain the step response of closed loop system.
• Check the response for the required specifications.
• If the specifications are not met, get a reduced order model and design a controller for the reduced order model.
• Obtain the initial values of the parameter \( K_p \) and \( K_i \) by pole zero Cancellation method.
• Cascade the controller with reduced order model and get the closed loop response with the initial values of the controller parameters.
• Find the optimum values for the controller parameters which satisfy the required specifications.
• By applying the optimum values, cascade this controller with the original system.
• Obtain the closed loop step response of the system with the controller.
• If the specifications are met give exit command else tune the parameters of the controller till they meet the required specifications.

For designing the PI controller, the values of controller parameters \( K_p \) and \( K_i \) are obtained through existing tuning method. The GA is employed to obtain the optimized values of \( K_p \) and \( K_i \) to meet out the designs specifications.

6.3 EXISTING TUNING METHODS

The existing tuning methods that are mainly used in tuning of controllers are,

• Ziegler-Nichols (Z-N) method
• Magnitude Optimum (MO) method
• Symmetric Optimum (SO) tuning method
6.3.1 Ziegler–Nichols Tuning Method

A very useful empirical formula was proposed by Ziegler and Nichols in early 1942 (Ziegler and Nichols 1942). The tuning formula is obtained when the plant model is given by a first order plus dead time (FOPDT) which is by,

\[
G(s) = \frac{k}{1 + sT} e^{-\alpha s}.
\]  

(6.3)

In real time process control systems, a large variety of plants can be approximately modeled according to Equation (6.3). If the system model cannot be physically derived, experiments can be performed to extract the parameters for the approximate model (6.3). For instance, if the step response of the plant model can be measured through an experiment, the output signal can be recorded as sketched in Figure 6.1(a), from which the parameters of \(k\), \(L\), and \(T\) (or \(a\), where \(a = kL/T\)) can be extracted by the simple approach shown. More sophisticated curve fitting approaches can also be used. With \(L\) and \(a\), the Ziegler–Nichols formula in Table 6.1 can be used to get the controller parameters.

If a frequency response experiment can be performed, the crossover frequency \(\omega_c\) and the ultimate gain \(K_u\) can be obtained from the Nyquist plot as shown in Figure 6.1 (b). Let \(T_c = 2\pi/\omega_c\). The PID controller parameters can also be retrieved from Table 6.1. It applies for the design of P (proportional) and PI controllers in addition to the PID controller with the same set of experimental data from the plant. Since only the 180° point on the Nyquist locus is used in this approach. Ziegler and Nichols suggested it can be found by putting the controller in the proportional mode and increasing the gain until an oscillation takes place. The point is then obtained from measurement of the gain and the oscillation frequency.
Table 6.1 The Ziegler-Nichols Tuning formulae

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>From Step Response</th>
<th>From Frequency Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>K_p</td>
<td>T_i</td>
<td>T_d</td>
</tr>
<tr>
<td>P</td>
<td>T/L</td>
<td>Inf</td>
</tr>
<tr>
<td>PI</td>
<td>0.9 T/L</td>
<td>L/0.3</td>
</tr>
<tr>
<td>PID</td>
<td>1.2 T/L</td>
<td>2L</td>
</tr>
</tbody>
</table>

The advantages of the Ziegler-Nichols method is that the tuning rules are very simple to use but the Disadvantages are:

- The appropriate values of L and T can be obtained only if the curve obtained is S-shaped and if the curve is not S-shaped, deviations in values might occur.
- Further fine tuning is needed.
- Controller settings are aggressive, resulting in large overshoot and oscillatory responses.
- Poor performance for processes with a dominant delay.
• Closed loop is very sensitive to parameter variations.
• Parameters of the step response may be hard to determine due to measurement noise.

6.3.2 Magnitude Optimum (MO) and Symmetric Optimum (SO) Tuning methods

Magnitude optimum and Symmetric optimum are two loop shaping tuning methods extensively employed by the companies like Siemens. The first step in the application of these methods is to determine appropriate transfer function which models the process. Once the transfer function is determined, the controller is able to shape the open loop transfer function in a desired manner. MO tuning method was devised with the objective to obtain a control system with a frequency output characteristic as close to unity and as flat as possible for the maximum bandwidth. Its mathematical expression states the requirements posed on the closed loop transfer function $G_c(s)$:

$$G_c(s) = 0$$  \hspace{1cm} (6.4)

$$\lim_{\omega \to 0} \frac{d^n G_c(j\omega)}{d\omega^n} = 0$$  \hspace{1cm} (6.5)

for as many $n$ as possible.

Let the desired open loop transfer function is:

$$G_{ol}(s) = \frac{\omega_n^2}{s(s + 2\xi\omega_n)}$$  \hspace{1cm} (6.6)

Where, $\xi$ is the damping of the closed loop system and $\omega_n$ determines the closed loop dynamics, that is, the speed of response. For example, the PI controller is employed when it is possible to approximate the model of the process with the transfer function:
\[ G_p(s) = \frac{K}{(1 + T_1 s)(1 + T_2 s)} \]  
(6.7)

With \( T_2 < T_1 \). By analyzing Equations (6.4) to (6.7) and by setting \( \xi = 0.707 \), PI controller parameters are calculated (Åström and Hägglund, 1995).

\[ K_p = \frac{T_1}{2KT_2} \]  
(6.8)

\[ T_1 = T_1, \text{ with } \omega_n = 0.707/T_2 \]  
(6.9)

The dominant pole is cancelled by the PI controller zero, and the closed loop dynamics are determined with the smaller time constant \( T_2 \) of the process. MO design method optimizes the closed loop transfer function \( G_c(s) \) between the reference and the output signal. It often cancels the process poles by the controller zeros, which can lead to poor performance of the control system in response to load disturbance. The objective of the SO method, which was originally proposed by Kessler (1958), is to obtain an open loop transfer function.

\[ G_{o2}(s) = \frac{a\omega_c^2(s + \frac{\omega}{a})}{s^2(s + a\omega_c)} \]  
(6.10)

Where, \( \omega_c \) is the gain crossover frequency and \( a \) is related to the phase margin of the control system through:

\[ \gamma = 2a \tan \left( \frac{a-1}{a+1} \right) \]  
(6.11)

or conversely through:

\[ a = \frac{1+\sin \gamma}{\cos \gamma} \]  
(6.12)
The method maximizes the phase margin of the control system and leads to symmetrical phase and amplitude characteristics, as can be observed in Figure 6.2.

![Image of Figure 6.2 showing gain and phase characteristics of control system tuned according to symmetric optimum method](image)

**Figure 6.2** Gain and phase characteristics of control system tuned according to symmetric optimum method

The second multiplicand in equation (6.10) has the transfer function of a phase-lead network, which provides required phase uplifting at the frequency $\omega_c$. Figure 6.2 shows the amplitude and gain characteristics of the control system with the open loop transfer function given in equation (6.10). In the example, the parameter ‘$a$’ was set to 4. For example, if the process can be modeled with the transfer function $G_2(s)$:

$$G_2(s) = \frac{K_2}{s(1 + T_2s)}$$  \hspace{1cm} (6.13)

It is suitable for the application of the SO tuning method. The procedure leads to a PI controller with the following settings (Peric, 1979):

$$K_p = \frac{1}{aK_2T_2}$$  \hspace{1cm} (6.14)

$$T_I = a^2T_2$$  \hspace{1cm} (6.15)
and \[ \omega_c = \frac{1}{aT_2}. \]

The common choice for the parameter \(a\) is 2 which gives the phase margin of the control system \(\gamma \approx 37^\circ\). The SO method is designed to give a good response to load disturbance, but the response of the control system to set-point change has large overshoot. The overshoot is commonly reduced through the usage of a two-degree-of-freedom controller or with a prefilter. The MO and SO tuning methods are widely used in the cascade control systems, especially to control motor drives (Peric 1979, 1989) and (Deur 1999).

As the controller tuning methods are available for second order system hence model order reduction is preferred. The main objective of model order reduction is to design a controller of lower order which can effectively control the original higher order system. In the proposed scenario, there are two common approaches for controller design. First approach is to obtain the controller on the basis of reduced order model called process reduction. In the second approach, the controller is designed for the original higher order system and then the closed loop response of higher order controller with original system is reduced pertaining to unity feedback. Proportional Integral (PI) controller is designed for lower order model with the help of proposed cross multiplication of polynomials model order reduction method. This method is based on the minimization of the error index criterion between the desired response and actual response pertaining to a unit step input. The controller parameters \(K_p\) and \(K_i\) are obtained from the reduced order model with the help of pole zero cancellation technique. Finally this designed PI controller is connected to the original higher order system to get the desired specification.

6.4 GENETIC ALGORITHM (GA)

The basic principles of Genetic Algorithm were first proposed by Holland as a biological process in which stronger individual is likely to be the winner in a competing environment. Genetic Algorithm uses a direct analogy of such natural
evolution to do global optimization in order to solve highly complex problems. It presumes that the potential solution of a problem is an individual and can be represented by a set of parameters (Sheroz Khan and Salami Femi Abdulazeez 2008). These parameters are regarded as the genes of a chromosome and can be structured by a string of concatenated values. This form of variables representation is defined by the encoding scheme. The variables can be represented by binary, real numbers, or other forms depending on the application data. Its range i.e., the search space is usually defined by the problem. Genetic Algorithm has been successfully applied to many different problems. It has also been applied to machine learning, dynamic control system using learning rules and adaptive control (Pivonka 2002).

**Genetic Algorithm Based Tuning of the PI Controller**

In this thesis, Genetic Algorithm approach is used for the following two purposes:

- To reduce the error value between given higher order and obtained reduced order models.
- To determine the optimized value of PI controller parameters namely $K_p$ and $K_i$.

Genetic Algorithm is a stochastic global search method that mimics the process of natural evolution. The genetic algorithm starts with no knowledge of the correct solution and depends entirely on responses from its environment and evolution operators (i.e. reproduction, crossover and mutation) to arrive at the best solution. By starting at several independent points and searching in parallel, the algorithm avoids local minima and converging to sub optimal solutions. In this way, GA has been shown to be capable of locating high performance areas in complex domains without experiencing the difficulties associated with high dimensionality, as may occur with gradient descend techniques or methods that rely on derivative information (O’Mahony et al 2000). A genetic algorithm is typically initialized with a random
population consisting of between 20-100 individuals. This population (mating pool) is usually represented by a real-valued number or a binary string called a chromosome. How well an individual performs a task is measured is assessed by the objective function. The objective function assigns each individual a corresponding number called its fitness. The fitness of each chromosome is assessed and a survival of the fittest strategy is applied. In this thesis, the magnitude of the error will be used to assess the fitness of each chromosome. The flowchart for PID Controller tuning using GA is shown in Figure 6.3. There are three main stages of a genetic algorithm; these are known as reproduction, crossover and mutation.

**Initialization of parameters**

To start with the Genetic Algorithm, certain parameters need to be defined. These include population size, bit length of chromosome, number of iterations, selection, crossover and mutation types. Selection of these parameters is the ability of the controller designer. The range of the tuning parameters is considered between 0 and 10 (Mudi and Pal 1997).

Initializing values are detailed as follows:

- Population type : Double vector
- Population size : 100
- Bit length of the considered chromosome : 6
- Number of generations : 100
- Selection function : Tournament selection
- Crossover type : Single point crossover
- Crossover function : Intermediate
Crossover probability : 1.0

Mutation type : Uniform mutation

---

**Figure 6.3**  GA based PI Controller Tuning Flowchart

In each generation, the genetic operators are applied to selected individuals ($K_p$ and $K_i$) from the current population in order to create a new population (Mohammed Hassan and Waleed Sharif 2000). Generally, the three main genetic
operators of reproduction, crossover and mutation are employed. By using different probabilities for applying these operators, the speed of convergence can be controlled. Crossover and mutation operators must be carefully designed, since their choice highly contributes to the performance of the whole Genetic Algorithm (Ching-Chih Tsai and Chi-Huang Lu 1998).

Reproduction

A part of the new population can be created by simply copying without change, selected individuals from the present population. Also new population has the possibility of selection by already developed solutions. There are a number of other selection methods available and it is up to the user to select the appropriate one for each process. Reproduction crossover fraction is taken as 0.8

Crossover

New individuals are generally created as offspring of two parents (i.e., crossover being a binary operator). One or more so called crossover points are selected (usually at random) within the chromosome of each parent, at the same place in each (Ismail Yusuf et al 2010). The parts delimited by the crossover points are then interchanged between the parents. The individuals resulting in this way are the offspring. Beyond one point and multiple point crossover, there exist some crossover types. The so called arithmetic crossover generates an offspring as a component wise linear combination of the parents in latter phases of evolution. It is more desirable to keep individuals intact and so it is a good idea to use an adaptively changing crossover rate: higher rates in early phases and a lower rate at the end of the Genetic Algorithm (Saravanakumar and Wahidha Banu 2006).

Mutation

A new individual is created by making modifications to one selected individual. The modifications can consist of changing one or more values in the
representation or adding/deleting parts of the representation. In Genetic Algorithm, mutation is a source of variability and too great a mutation rate results in less efficient evolution, except in the case of particularly simple problems (Mohammed Obaid Ali et al 2009). Hence, mutation should be used sparingly because it is a random search operator; otherwise, with high mutation rates, the algorithm will become little more than a random search. Moreover, at different stages one may use different mutation operators. At the beginning, mutation operators resulting in bigger jumps in the search space might be preferred (Ali Reza Mehrabian and Morteza Mohammad Zaheri 2003). Later on, when the solution is close by a mutation operator leading to slighter shifts, the search space could be favoured.

**Summary of Genetic Algorithm Process**

In this section, the process of Genetic Algorithm will be summarized as a flowchart and is shown in Figure 6.4. The summary of the process will be described below.

The steps involved in creating and implementing a genetic algorithm:

- Generate an initial, random population of individuals for a fixed size.
- Evaluate their fitness.
- Select the fittest members of the population.
- Reproduce using a probabilistic method.
- Implement crossover operation on the reproduced chromosomes.
- Execute mutation operation with low probability.
- Repeat step 2 until a predefined convergence criterion is met.

The convergence criterion of a genetic algorithm is a user specified condition. For example, for the maximum number of generations or when the string
fitness value exceeds a certain threshold, they are considered as terminating conditions.

Figure 6.4 GA Process Flowchart

Genetic Algorithm versus Traditional Methods

Genetic algorithm is substantially different to the more traditional search and optimization techniques. The five main differences are:

- Genetic algorithm searches a population of points in parallel, not from a single point.
- Genetic algorithm do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the direction of the search.
• Genetic algorithm use probabilistic transition rules and not deterministic rules.
• Genetic algorithm work on an encoding of a parameter set and not the parameter set itself (except where real-valued individuals are used).
• Genetic algorithm may provide a number of potential solutions to a given problem and the choice of the final is left up to the user.

6.5 NUMERICAL ILLUSTRATION

Consider the 3rd order interval system stated in Ismail et al (1997) is described by its transfer function.

\[ G(s) = \frac{[1,2]s^2 + [3,4]s + [8,10]}{[6,6]s^3 + [9,9.5]s^2 + [4.9,5]s + [0.8,0.85]} \]

The corresponding reduced (2\textsuperscript{nd} order) model obtained through proposed order reduction method (as detailed in chapter 3) is obtained as,

\[ G_r(s) = \frac{[−1.0704,−0.4562]s + [1.888,2.258]}{[1.1]s^2 + [0.9137,0.9379]s + [0.1888,0.1919]} \]

By applying midpoint theorem, the reduced order interval system coefficients transfer function is given by

\[ G_r(s) = \frac{−0.7633s + 1.9196}{s^2 + 0.8843s + 0.3525} \]

The figure 6.5 shows the step response of the reduced order interval system without controller. The controller parameters \( K_p \), \( K_i \) and \( K_d \) are determined from this step response by using the conventional Ziegler Nichols Method tuning method as shown in table 6.2
Figure 6.5  Step Response of Reduced Order Interval System without Controller

Table 6.2  Parameter values for various controllers by Ziegler-Nichols method

<table>
<thead>
<tr>
<th>Controller</th>
<th>$K_p$</th>
<th>$K_i$</th>
<th>$K_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>0.71</td>
<td>Inf</td>
<td>0</td>
</tr>
<tr>
<td>PI</td>
<td>1.29</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>PID</td>
<td>1.71</td>
<td>0.944</td>
<td>0.765</td>
</tr>
</tbody>
</table>

Using these above controller parameters, the step response of the reduced order system with Z-N PID Controller is shown in Figure 6.6.
Figure 6.6  Step response of reduced interval system with Z-N PID Controller

The best population may be plotted to give an insight into how the genetic Algorithm converged to its final values as illustrated in figure 6.7.

Figure 6.7  PID controller parameters tuning process using GA
The GA tuning of the reduced order transfer function results in the following Controller parameter values: $K_d = 1.29377$, $K_p = 2.22880$ and $K_i = 1.0020$. Based on the above computed values the step response of the reduced order system with GA tuning for a PID controller is shown in figure 6.8

![Step Response of the Reduced Order System after GA Tuning](image)

**Figure 6.8**  Step Response of the Reduced Order System after GA Tuning

**Table 6.3**  Comparison of Time domain specifications for various controllers

<table>
<thead>
<tr>
<th>Transfer Function</th>
<th>Time Domain Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rise Time</td>
</tr>
<tr>
<td>Higher order system</td>
<td>3.75</td>
</tr>
<tr>
<td>System with Z – N PID controller</td>
<td>0.45</td>
</tr>
<tr>
<td>System with Genetic PID controller</td>
<td>3.44</td>
</tr>
</tbody>
</table>
6.6 SUMMARY

In this chapter Genetic algorithm based PI controller tuning for the reduced order system is proposed. The GA based PI Controller is attached with the original higher order system and the closed loop response is observed for stabilization process. The steady state performance of proposed PI controller with the help of GA has been analyzed. It is observed that the Genetic Algorithm based PI controller produces an output which is several times ahead of that of PI in the rise time analysis. The set point tracking and disturbance rejection is obtained in the Genetic Algorithm PI controller.