ISSN: 2348 - 2117

International Journal of Engineering Technology and Computer Research (IJETCR)
Available Online at www.ijetcr.org
Volume 3; Issue 5; September-October-2015; Page No. 39-45

Tuning of PID Controller using Soft Computing Methodology

Mrs. Bhawna Arora[1],Dr. Mukesh Gupta [2]

[1]Jagannath University, Jaipur, India

[2] JNIT, Jaipur, India

Abstract

This paper presented potential of soft computing technique for tuning of Proportional Integral Derivative (PID) controller [1]. The paper reveals that soft computing approach is much efficient as compared to other conventional approach of tuning of PID controller. The PID controller is the very commonly used compensating controller for in nonlinear systems. This controller is widely used in many different areas like aerospace, process control, manufacturing, and automation. The design of the controller is based on their precise mathematical models, which are usually very difficult to achieve owing to the complexity, nonlinearity, time varying and incomplete characteristics of the existing practical systems. Therefore, it needs an efficient method to control the different parameters of the plant.

There are various soft computing techniques which are used for tuning of PID controller. Tuning of PID parameters is important because these parameters have a great effect on the stability and performance of the control system. The aim of the work described in this paper is to illustrate the improvement of response of the system by optimizing the PID controllers using genetic algorithms. In the conclusion, by a comparative analysis between the conventional PID tuning methods and optimization carried out using genetic algorithms offers lesser oscillatory and better response.

The simulation outputs are the MATLAB results obtained for a step input to a third-order plant.

Keywords:*PID controller, Genetic Algorithm (GA) PID controller.*

INTRODUCTION:

The PID Controller was patented in 1939 by Albert Callender and Allan Stevenson of Imperial Chemical Limited of Northwich, England. The PID controller is most common controller used in industrial processes despite continuous advances in control theory. The main reason is due to their simplicity of operation, ease of design, inexpensive maintenance, low cost, and effectiveness for most linear systems. Recently, motivated by the rapidly developed advanced microelectronics and digital processors, conventional PID controllers have gone through a technological evolution, from pneumatic controllers via analog electronics to microprocessors via digital circuits.

Most conventional PID tuning methods require considerable technical experience to apply tuning formulas to determine the PID controller parameters. In practical applications, most of the industrial process exits to be non- linear, variability of parameters and uncertainty of model are very high, thus using conventional PID tuning methods the precise control of the process cannot be achieved. Due to this, PID controllers are rarely tuned optimally and thus required improved tuning technology. The above problems can be well addressed by the application of non-conventional methods for tuning of the PID controller .Most practical PID remains poorly tuned leading to deteriorated process performance .The conventional tuning methods require considerable technical experience and are time consuming and do not work well for non-linear, higher order and time- delayed systems and the ones that do not have a precise mathematical model. Non-conventional methods are especially useful for solving problems of computationally complicated and mathematicallyuntraceable. Hence the need arises for an optimization algorithm like Genetic Algorithm (GA) [2][3][4].

2. PID Controller

PID controller is structurally simple and exhibit robust performance over a wide range of operating conditions. In the absence of the complete knowledge of the process these types of controllers are the most efficient of choices. The three main parameters involved are Proportional (P), Integral (I) and Derivative (D). The proportional part is responsible for following the desired set-point, while the integral and derivative part account for the accumulation of past errors and the rate of change of error in the process respectively.

Corresponding author: Mrs. BhawnaArora

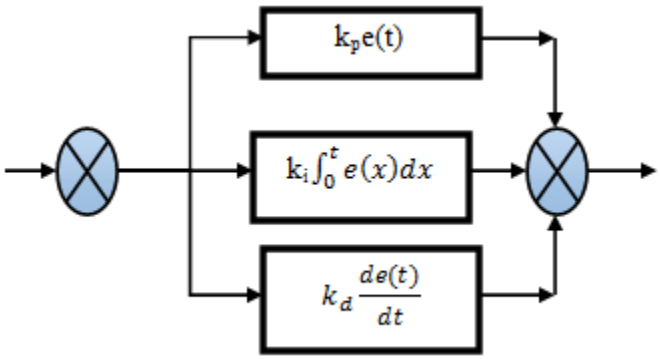

Fig. 1 Basic block diagram of a conventional PID controller

For the PID controller presented in Fig. 1, Output of the PID controller,

$$u(t) = K_p e(t) + K_i \int_0^t e(x)dx + K_d \frac{de(t)}{dt}$$

Where e(t) =Set point- Plant output,Kp = proportional gain, Ki = integral gain, Kd = derivative gain

The individual effects of these three terms on the closed-loop performance are summarized in Table (1). Note that this table serves as a first guide for stable open-loop plants only. For optimum performance Kp, Ki and Kd are mutually dependent in tuning.

Table 1: Effects of Independent P, I and D tuning

Closed loop response	Rise Time	Over- shoot	Settling Time	Steady State Error	Stability
Increase Kp	Decrease	Increase	Small Increase	Decrease	Degrade
Increase Ki	Small Decrease	Increase	Increase	Large Decrease	Degrade
Increase Kd	Small Decrease	Decrease	Decrease	Minor change	Improve

2.1. Standard performance measures

Table 2: Average values of standard performance measures

Parameter	Z-N Method	Iterative Method	Optimization by GA				
			MSE	IAE	ISE	ITAE	ITSE
%OV	38%	15%	10%	6%	11%	10%	8%
ST5%	1.53	1.54	1.47	1.01	1.45	0.745	1.37
RT	0.444	0.912	0.453	0.495	0.46	0.588	0.47
PT	0.613	3.43	0.576	0.622	0.58	0.836	0.6
SM	36.25	37.86	33.68	33.8	24.4	36.8	32

Percent Overshoot

Table 2 summarizes the average change of percent overshoots with respect to the time delay. Ziegler-Nichols rule [5] gives the biggest value for all time delay, consequently its average value is the biggest also, 38%. Here the difference between two tuning methods and GA methods can be seen: while tuning methods have almost the same pattern, its value decreasing as the time delay growing bigger, except for small value of delay where Ziegler-Nichols rule gives increasing values, the GA methods give almost constant value, around 10% as long the time delay is not too small, except for case optimized by ITAE, where the percent overshoot value fluctuates and decreasing as time delay increasing. Table 2 shows that GA produces much better percent overshoot than two others tuning methods, especially if optimized by IAE criterion. So it can be concluded that GA can be used to optimized percent overshoot. *Settling Time*

The value of settling time (5% criterion) all over the time delay is summarized by figure 3, where it can be seen that almost all methods, except for GA optimized by IAE and ITAE, fall under almost the same straight line with positive slope. It means that as the delay increasing, the settling time will increase linearly. In table 1 we can see that the settling time average value is not so different among all methods, in the range 1.37 second to 1.54 second, except for GA optimized by ITAE 0.745 second and IAE 1.01 second. However, because the others GA methods results are in agree with two tuning methods, we can't really differentiate these results. So it can be

said that the settling time is not optimized by GA methods.

Rise Time

The third variable is rise time, which plotted in figure 4 using logarithmic scale in y axis. We can see strong pattern which all the results, except for Iterative Method, have almost the same value all over the time delay. And it should not be surprising if we get almost the same average value for all result, around 0.44 second to 0.59 second, except for Iterative Method, 0.912 second. Another interesting thing is, in general, almost in all range of time delay the curves keep their ranking unchanged, with order from the biggest value: Iterative Method, ITAE, IAE, ITSE, ISE, MSE, and Ziegler-Nichols rule.From the average value (table 1), the best result is given by Ziegler-Nichols rule, 0.444 second and the worst one is Iterative Method 0.912 second, and all GA methods produce almost the same average value. But because GA results are not so different compared to Ziegler-Nichols rule, it can't be concluded that GA can optimize the rise time.

Peak Time

Almost the same pattern, as in rise time plots, is shown on the peak time plots on figure 3, except Iterative Method, in large delay range, tends to diverge, where all methods show almost the same values for all over time delay. But we must pay attention on GA optimized by ITAE because there is a range which its value is bigger than the others. Except for Iterative Method which is 3.43 second, all others methods produce peak time around 0.57 second to 0.83 second. The best values are given by GA optimized by MSE and ISE, 0.576 second. Like rise time, in general, almost in all range of time delay the curves keep their ranking unchanged, with order from the biggest value: Iterative Method, ITAE, IAE, Ziegler-Nichols, ITSE, MSE, and ISE. And because the GA methods produce peak time plots which only better than Iterative Method, not Ziegler-Nichols.

Stability Margin

The last standard performance measure is stability margin. Stability margin is the maximum gain that can be set before system response goes into sinusoidal cycle. In the simulations, this is done by simply increasing value of Kc until sinusoidal cycle happens, and the stability margin of corresponding system is Kc at sinusoidal cycle.

This is the first result that shows consistency all over time delay range, which all curves fall under almost the same line. So this is the strongest patterns, and because the stronger the pattern, the less the ability of GA methods to optimize corresponding performance measures, we can't use GA methods to optimize the stability margin. Conversely, the GA methods are likely to produce less stable systems.

Like rise time and peak time, in general, almost in all range of time delay the curves keep their ranking unchanged, with order from the biggest value: Iterative Method, ITAE, Ziegler-Nichols, IAE, ITSE, ISE, and MSE. But unlike settling time, rise time, and peak time, stability margin values reduce as the time delay increases.

2.2. Performance Indices

We differentiate the term standard performance measures and performance indices here, where standard performance measures have already been discussed above, performance indices are: MSE, IAE, ISE, ITAE, and ITSE. Table 3, 4, 5 below summarizes performance indices from simulation results. As expected, the average values of GA performance indices are always smaller than its corresponding Ziegler-Nichols and Iterative Method. Moreover Ziegler-Nichols rule produces smaller average performance indices values than Iterative Method does for all time delay values range.

Table 3: Performance indices of Ziegler-Nichols tuning methods

Delay	Ziegler-Nichols				
	MSE	IAE	ISE	ITAE	ITSE
0.01	0.00097	2.9923	1.4612	0.0739	0.0206
0.025	.002626	6.927413	3.942144	0.432876	0.124957
0.05	0.004988	12.98842	7.487622	1.580628	0.452425
0.075	0.007197	18.53609	10.80309	3.249557	0.929874
0.1	0.009284	23.69959	13.93484	5.304512	1.517501
0.25	0.020422	48.31448	30.65349	20.82159	6.345217
0.5	0.037119	76.26794	55.71523	45.56416	17.73724
0.75	0.05416	108.6372	81.29453	91.98153	36.10692
1	0.072117	146.0676	108.247	174.114	65.18471
Av	0.02321	49.38123	34.12475	38.12475	14.26883

Table 4:Performance indices of Iterativetuning methods

Delay	Iterative Method				
	MSE	IAE	ISE	ITAE	ITSE
0.01	0.000927	2.730597	1.391095	0.061045	0.014833
0.025	0.002313	6.300772	3.471926	0.39184	0.089213
0.05	0.004376	11.6831	6.567769	1.452929	0.318439
0.075	0.006334	16.58511	9.507315	2.979748	0.642704
0.1	0.008226	20.99624	12.34746	4.801577	1.035597
0.25	0.019092	40.69953	28.65714	15.58065	4.507248
0.5	0.037603	71.60299	56.44149	37.68745	16.79849
0.75	0.057392	123.8553	86.44149	146.8766	42.31687
1	0.077665	176.087	116.5754	298.2008	81.12556
Av	0.02377	52.2822	35.6784	56.44808	16.31655

Table 5:Performance indices of tuning through Genetic Algorithm

Delay	Optimization by GA				
	MSE	IAE	ISE	ITAE	ITSE
0.01	0.00081	1.623776	0.840777	0.011395	0.005388
0.025	0.001884	3.701027	2.827366	0.218351	0.038659
0.05	0.003708	7.09663	5.57464	0.301924	0.155625
0.075	0.005508	10.75044	8.266998	0.754851	0.350289
0.1	0.0073	13.86607	10.95693	1.526731	0.622276
0.25	0.017983	34.18455	26.98306	7.69427	3.854827
0.5	0.035612	67.45696	53.45713	30.00848	15.15463
0.75	0.053071	100.0041	79.65463	63.58332	33.54903
1	0.070379	134.0671	105.6145	111.3298	58.69364
Av	0.021806	41.41674	32.68734	23.93657	12.4916

3. GA-PID controller

The Genetic Algorithm (GA)[6] can be seen as a search algorithm based on the concept of natural selection and genetic inheritance. It searches an optimal solution to the problems by manipulating a population of string that represents different potential solutions. A string structure in GA represents each parameter. This is similar to the chromosome structure in natural genes [6]. The population is a group of strings. After mutation and crossover between strings in a given population, a new generation of strings evolves. For each generation, all the populations are evaluated based on their fitness. An individual with larger fitness has a higher chance of evolving into the next generation. The features of GA are different from other search techniques in several aspects. The algorithm works with a population of strings. By searching many peaks simultaneously, GA reduces the possibilities of trapping into a local minimum. Coding of parameters is used instead of using the parameters themselves which is useful to evolve the next state with minimum computations. Finally, fitness of each string is evaluated to guide its search. There is no need for computations of derivatives or other auxiliary knowledge.

3.1 Work Flow of GA

A GA is typically initialized with a randomly generated population consisting of candidate individuals [7]. Each individual in the population is usually represented by a real- valued number or a binary string. Such strings are called as chromosomes. A set of chromosome or individual is a whole population. Performance of each individual is measured and assessed by the objective function [8]. The objective function assigns each individual a corresponding number called its fitness value. If the termination criteria are not met with the current population then a new individuals are created with genetic operators. A survival of the fittest strategy is

applied on individuals. The fittest parents are found out by reproduction or selection operator. New individuals are generated by performing operations such as crossover and mutation on the individuals whose fitness has just been measured. The fitness of the offspring is then computed. The off springs are inserted into the population replacing the parents or low-fitness individuals producing a new generation. This cycle is performed until the termination criterion is reached. Such a single population GA is powerful and performs well on a wide variety of problem. Every iteration of a GA loop is referred to a generation and it stops when termination criteria get satisfied.

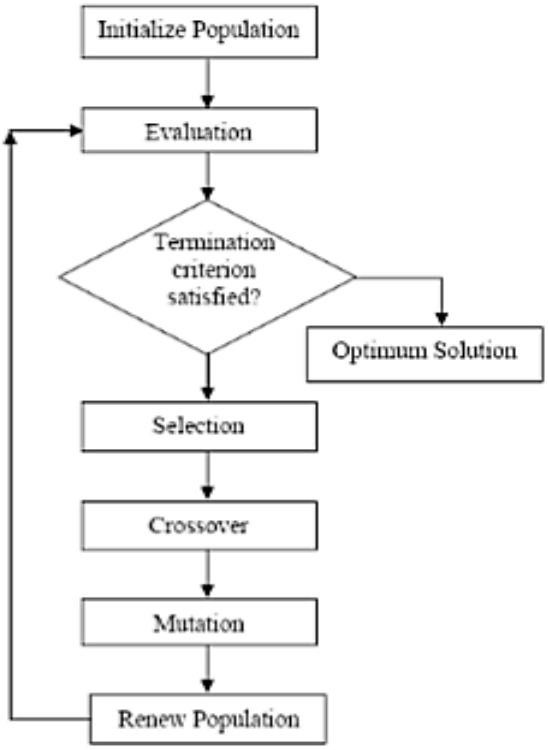

Fig.2: Work1 flow of Genetic Algorithm

4. Simulation Result

We implement the proposed algorithm on a system with transfer function given as follows.

The step response of the closed loop system of conventional PID controller & GA-PID controller are shown in figure below:

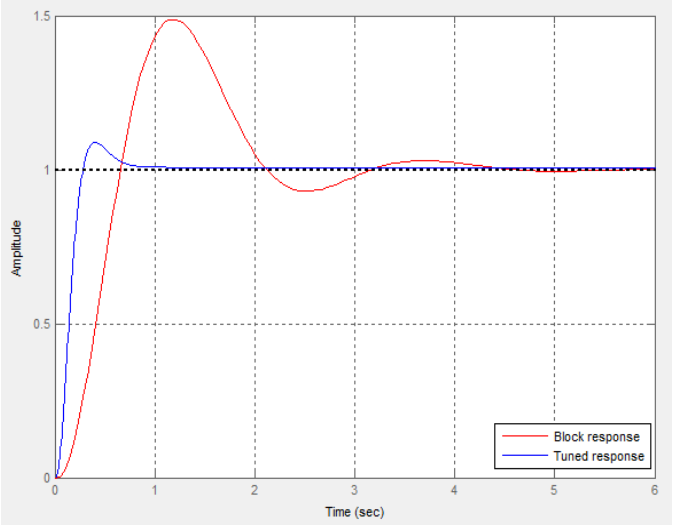

Fig.3: Iterative tuned PID controller response for K_p=0.7,K_I=0.6,K_d= .001

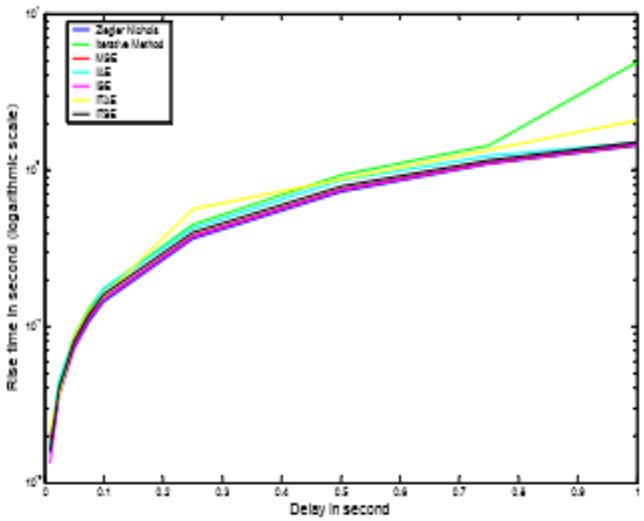

Fig 4: The comparison of rise time (RT).

The size of population of GA is often chosen between [20,100]. For our simulation, we chose the size of population as 40. The number of generation is often chosen between [100,500]. For our case, we chose number of generations equal to 300. The parameter ranges of GA-PID controller are $K_p \in [0, 20]$, $K_i \in [0, 1]$ and $K_d \in [0, 5]$.

5. Comparison of result

Here, the output response of the given third order system of conventional PID Controller has an overshoot of more than 40% &Settling time is also close to 4 seconds while using iterative tuned PID Controller Settling time & an overshoot is reduced to 0.6 sec and 12% respectively as shown in figure 3. While using GA-PID Controller further reduce the settling time to 0.2 sec and overshoot within 5% shown in figure 5.

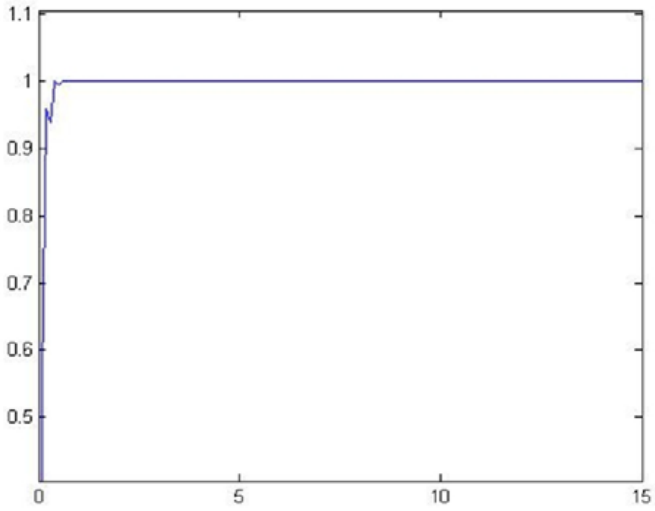

Fig. 5:Output response of the GA-PID controller

References

1. B.G. Liptak: Instrumentation Engineer's Handbook: Process Control, Third Edition, CRC Press, 1995
2. Liu Fan, ErMengJoo. "Design for Auto-tuning PID Controller Based on Genetic Algorithms". IEEE Conference on Industrial Electronics and Applications (ICIEA 2009)
3. Visioli.A "Tuning of PID controllers with fuzzy logic" IEEE Proc.-Control Theory Appl., Vol.148, No.1, January 2001
4. Chaiyaratana, N.; Zalzala, A.M.S. "Recent developments in evolutionary and genetic algorithms: theoryandapplications"Genetic Algorithms in Engineering Systems: Innovations and Applications,

Second International Conference On (Conf. Publ. No. 446) 2-4 Sep 1997 Page(s):270 – 277.

5. Gopal M. "Digital control and state variable methods".

6. B. Nagaraj, S. Subha, B. Rampriya. "Tuning Algorithms for PID Controller Using Soft Computing Techniques".

7. S.S. Ge, member ,IEEE, T.H. Lee,Member,IEEE and G.Zhu . "Genetic algorithm tuning of Lyapunov-based controllers: An Application to a Single-Link flexible Robot System". IEEE transaction on industrial electronics.

8. J. G. Ziegler, N. B. Nichols, "Optimum setting for automatic controllers", Trans. ASME, Vol. 64, pp. 759-768, 2