Design PID Controller tuning by using PSO-BFO Optimization technique

Mr. Lalit singh bisen[1], Dr. Dharmendra kumar singh[2]
M.Tech. Scholar[1], Associate professor[2]
Dr. C. V. Raman Institute of Science and Technology

Abstract – This paper Present to design method for determining the optimal proportional-integral-derivative (PID) controller parameters of plant system using the particle swarm optimization (PSO) algorithm and bacterial Foraging Optimization (BFO). There are several methods which are used to tune the controller parameters. They are categorized into two types known as classical methods and modern methods. In this paper the use of PSO method tuned the PID parameter to make them more general and to achieve the steady state error limit, also to improve the dynamic behaviour of the system. The performance and design criteria of automatic selection of controller constants are discussed below.

Keywords — plant system, PID controller, PSO and BFO Optimization.

1. Introduction

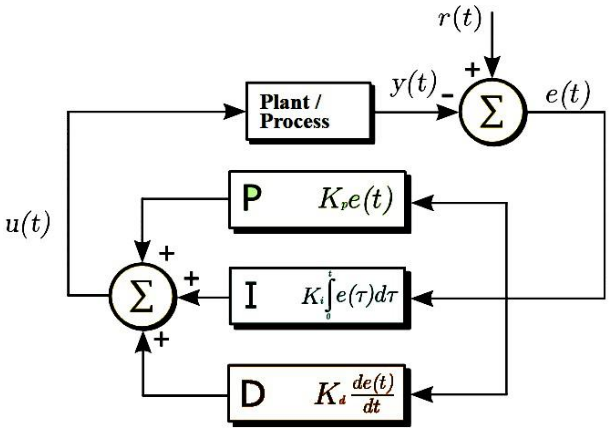
During the past decades, the process control techniques in the industry have made great advances. Numerous control methods such as adaptive control, neural control, and fuzzy control have been studied. Among them, the best known is the proportional-integral-derivative (PID) controller, which has been widely used in the industry because of its simple structure and robust performance in a wide range of operating conditions. Unfortunately, it has been quite difficult to tune properly the gains of PID controllers because many industrial plants are often burdened with problems such as high order, time delays, and nonlinearities. For these reasons, it is highly desirable to increase the capabilities of PID controllers by adding new features. Many artificial intelligence (AI) techniques have been employed to improve the controller performances for a wide range of plants while retaining their basic characteristics. AI techniques such as neural network, fuzzy system, and neural-fuzzy logic have been widely applied to proper tuning of PID controller parameters PID controller consists of Proportional, Integral and Derivative gains. The PID feedback control system is illustrated in Fig. where r, e, y are respectively the reference, error and controlled variables [1]. Where Kp is proportional gain, Ki is integral gain and Kd is derivative gain.

A PID controller is described by the following transfer function in the continuous s-domain:

$$Gc = P + I + D$$

$$Gc = K_p + \frac{ki}{s} + K_d S$$

$$Gc = Kp\,(1 + \frac{1}{Ti} S + T_d S)$$



- Fig - block diagram of a PID controller in a close loop

The PID control scheme is named after its three correcting terms, whose sum constitutes the manipulated variable (mv). The proportional, integral, and derivative terms are summed to calculate the output of the PID controller. Defining u(t) as the controller output, the final form of the PID algorithm is:

$$\mathrm{u}(t) = \mathrm{MV}(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}$$
………… (1)

Where,

K_p: Proportional gain, a tuning parameter

K_i: Integral gain, a tuning parameter

K_d: Derivative gain, a tuning parameter

e: Error SP - PV

SP: Set Point

PV: Process Variable

t: Time or instantaneous time (the present)

τ : Variable of integration; takes on values from time 0 to the present t.

Equivalently, the transfer function in the Laplace Domain of the PID controller is:

$L(s) = K_p + K_i/s + K_d s$

Where

s: Complex number frequency [2].

1. Swarm topology

Each particle i (total number of particles) has its neighbourhood N_i (a subset of P). The arrangement of the neighbourhoods is called the swarm topology, which can be represented by a graph. Usual topologies are: Fully connected topology, Circle topology and single sighted topology.

Characteristics of particle i at iteration t:

$x_i^{(t)}$ - the position

$p_i^{(t)}$ -the individual best position

$l_i^{(t)}$- the local best position of the neighbouring particles

$v_i^{(t)}$ - the velocity of the particle

At the beginning of the algorithm, the particle positions are randomly initialized, and the velocities are set to 0, or to small random values.

Algorithm parameters:

w(t) - Inertia weight; a damping factor, usually decreasing from around 0.9 to around 0.4 during the computation

$\emptyset_1, \emptyset_2$ - Acceleration coefficients; usually between 0 and 4

Manipulating its velocity:

Many versions of the particle speed update exist, for example:

$$v_i^{(t+1)} = \omega^{(t)} v_i^{(t)} + \emptyset_1 c_1 (p_i^{(t)} - x_i^{(t)}) + \emptyset_2 c_2 (l_i^{(t)} - x_i^{(t)}) \quad \text{..... (2)}$$

The symbols u_1 and u_2 represent random variables with the C(0,1) distribution. The first part of the velocity formula is called "inertia", the second one "the cognitive (personal) component", the third one is "the social (neighbourhood) component". Position of particle i changes according to

$$x_i^{(t+1)} = x_i + v_i^{(t+1)} \quad \text{...... (3)}$$

2. PSO Algorithm

PSO is optimization algorithm based on evolutionary computation technique. The basic PSO is developed from research on swarm such as fish schooling and bird flocking. After it was firstly introduced in 1995, a modified PSO was then introduced in 1998 to improve the performance of the original PSO.

Basic Fundamentals of PSO Algorithm

The basic fundamentals of the PSO algorithm technique are stated and defined as follows.

Particle X(i): A candidate solution represented by a k-dimensional real-valued vector, where k is the number of optimized parameters; at iteration i, the jth particle X(i,j) can be described as :

$$X_{j(i)} = [x_{j,1(i)}; x_{j,2(i)}; \ldots\ldots\ldots; x_{j,k(i)}. \ldots.; x_{j,d(i)}] \quad \text{...... (4)}$$

Where: x_s are the optimized parameters

$X_k(i,j)$ is the k^{th} optimized parameter in the j^{th} candidate solution d represents the number of control variables.

Population: This is the set or say population of n particles at iteration i.

$$Pop(i) = [X_{i(j)}, X_{2(i)}, \ldots\ldots\ldots X_{n(i)}]^T \quad \text{...... (5)}$$

Where n represent the number of individual solutions.

Swarm: As the name swarm defines the group of particles, it is a disorganized population of moving particles that tends to cluster together and each particle seems to moving in a random direction with their velocity.

Particle velocity V(i): The velocity of the moving particles represented by a d-dimensional real-valued vector; at iteration i, the j^{th} particle $V_j(i)$ can be described as:

$$V_{j(i)} = [v_{j,1(i)}; v_{j,2(i)}; \ldots\ldots v_{j,k(i)}; \ldots\ldots; v_{j,d(i)}] \quad \text{...... (6)}$$

Weight factor w(i): This is a control parameter and sometimes it is called as inertia weight factor. the inertia factor decreases linearly from about 0.9 to 0.4 during a continuous run .

In general, this factor is set according to following equation**:**

$$W = w_{max} - (w_{max} - w_{min})/iter_{max} * iter \quad \text{...... (7)}$$

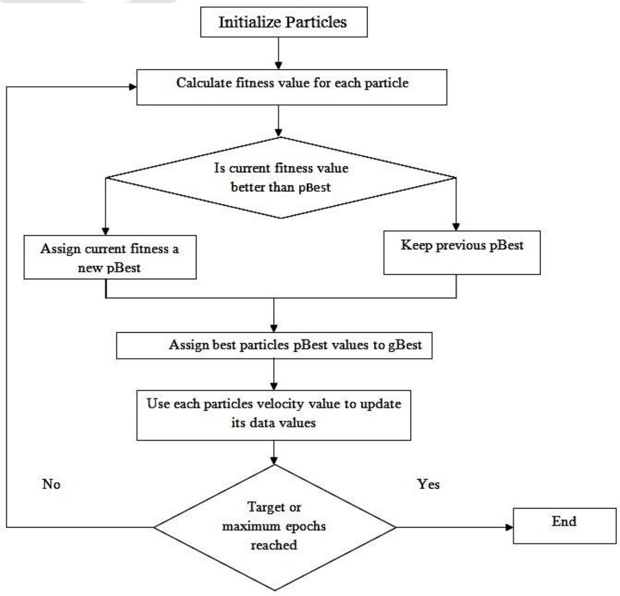
Individual best X^*(i): The best position that is associated with the best fitness encountered thus far is called individual best $X^*(i)$. For each particle in the swarm, $X^*(i)$ can be determined and updated during the search. For the j^{th} particle, individual best can be expressed as

$$X_j^*(i) = [x_{j,1(i)}, x_{j,2*(i)}, \ldots\ldots x_{j,d*(i)}]^T \quad \text{..... (8)}$$

In a minimization problem with only one objective function f, the individual best of the j^{th} particle $X_j^*(i)$ is updated. Otherwise individual best solution of the j^{th} particle will be kept as in the previous iteration.

Global best X^{**}(t): This is the best position among all of the individual best positions achieved so far.

Stopping Criteria: The search process will be terminated whenever one of the following criteria is satisfied [3].



- Fig - Flow diagram of PSO algorithm

3. BACTERIAL FORAGING OPTIMIZATION

Introduction Based on the research of foraging behaviour of E.coli bacteria Kevin M.Passino and Liu exploited a variety of bacterial foraging and swarming behaviour, discussing how to connect social foraging process with distributed non-gradient optimization.

Steps for BFO approach

The step involving for finding the best value or best position is given below including chemotactic step, reproduction step, and elimination step.

Step1 Initialize parameter p,S,N_c ,N_s ,N_{re} ,N_{ed} ,P_{ed} C(i)(i=1,2....S) α^i

Step2 Elimination-dispersal loop: l= l+1

they will break into two parts due to the presence of high temperature. Due to the sudden change in the local environment where a bacterium population lives may occur due to various reasons like change in temperature may kill a group of bacteria that are currently in a region with a high concentration of nutrient gradients.

Step3 Reproduction loop: k=k+1

The cycle of the bacteria formation will be continuing and they will try to make the swarm size constant up to a certain value [4]. The least healthy bacteria sooner or later dies but the healthier bacteria splits up to two bacteria two maintain the swarm size constant..

Step4 Chemotaxis loop

Chemotaxis loop: *j=j*+1

[a] For *i* =1, 2…S take a chemotactic step for bacterium *i* as the iteration for the value of swarm.

[b] Compute fitness function, J (j, k, l).

Suppose (j, k, l) i_q represents i^{th} bacterium at j^{th} chemotactic, k^{th} reproductive and l^{th} elimination-dispersal step. C (i) is the size of the step taken in the random direction specified by the tumble (run length unit).

[c] Let J_{last} = Last value J (j, k, l) to save this value since we may find a better cost via a run.

[d] Tumble: generate a random vector

[e] Move: Let

$$\Theta_{i(j+1,k,l)} = \Theta_{i(j,k,l)} + C(i)\left[\frac{d(i)}{(dT(i)d(i))^{1/2}}\right] \qquad \text{...... (9)}$$

These results in a step of size C(i) in the direction of the tumble for bacterium i.

[f] Compute J(j+1,k,l) for finding the new position of bacterium.

[g] Swim

I. Let n=0 (which is the counter for swim length)
II. While n<N_s (not gone to the higher level)

- Let n=n+1
- If J(j+1,k,l) <J_{last} (if doing better)
- J_{last} = J(j+1,k,l) and let

$$\theta_{i(j+1,k,l)} = \theta_{i(j,k,l)} + C(i)\left[\frac{d(i)}{(dT(i)d(i))^{1/2}}\right] \qquad \text{............ (10)}$$

And use this $\theta_{i(j+1,k,l)}$ to compute the new values of J which will be iterated by their respected nature.
[h] Else let n=N_s. This is the end of the while statement.

Step 5 If j<N_c go to step 4. In this case continue chemotaxis since the life of the bacteria is not over.

Step6 Reproduction: Reproduction step states that the bacteria energised by taking food and it will break into two parts equally, hence there is a new birth of bacteria.

For the given k and l and for each i=1,2,......,S,
let

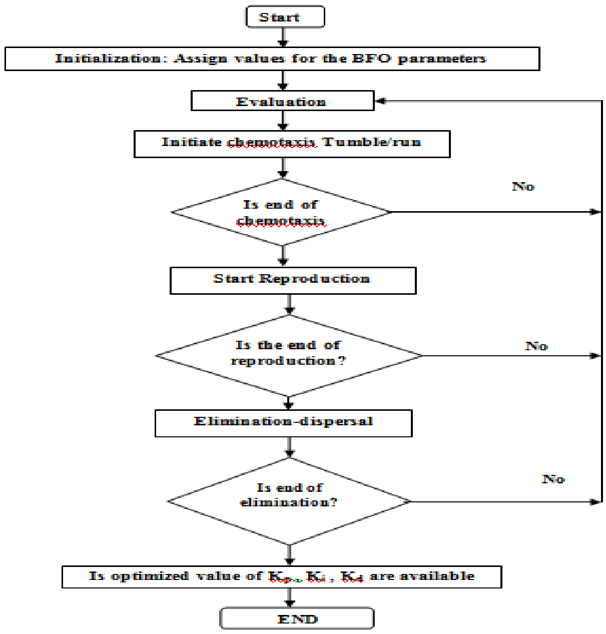
$$\boldsymbol{J_{health}} = \sum_{j=1}^{Nc+1} J(j,k,l) \qquad \text{... ... (11)}$$

J_{health} denotes the health of the bacterium during each step and each position when they are moving in (j,k,l) coordinates.

According to above equation if the bacteria with highest J_{health} values die and the remaining bacteria with best value splits.

Step7 If k<N_{re} go to step 3. In this case, we have not reached the number of specified reproduction steps, so we start the next generation of the chemotactic loop.

Step8 Elimination Dispersal: For i=1,2,3.... S, Eliminate and Disperse each bacterium (these keeps the number of bacteria in the population constant) [4].

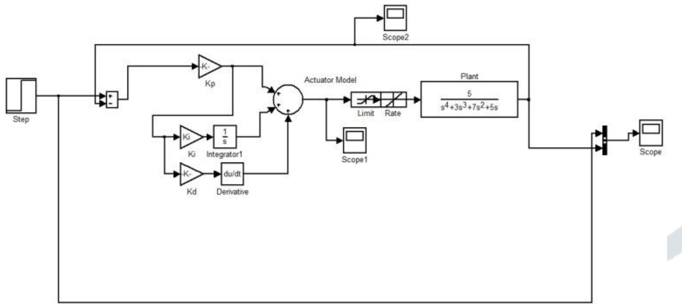


- Fig. - Flow diagram of BFO algorithm

4. Simulation and results

As the figure shows that manipulating the values of gain by PSO PID technique or PSO BFO technique it gives the required results. In both PSO as well as in BFO there are two important factor i.e. alpha and beta which is the base of the gain values. Alpha and Beta are the two constant terms which provides the system stability.

Matlab modelling of plant model using controller



- Fig - Matlab modelling for plant with PID controller

Functions are used for designing PID controller ISE, IAE, ITAE and ITSE .We set the following parameters

Dimension of search space =3;
The number of bacteria =10;
Number of chemotactic steps =10;
Limits the length of a swim =4;
The number of reproduction steps =4;
The number of elimination-dispersal events =2;
The number of bacteria reproductions per generation =s/2;
The probability that each bacteria will be eliminated/dispersed =0.25;
c(:,1)=0.5*ones(s,1); the run length.
We use the following PSO parameters
C1=1.2;
C2= 0.5;
W=0.9;

5. COMPARISONS FOR THE DIFFERENT VALUES OF ALPHA AND BETA

TABLE 1 For PSO when n =50 alpha = 10 and Beta = 5,10,15,20

S. no	n	α	β	Kp	Ki	Kd	Mp %	Ts	Tr
1	50	10	5	0.96	0	0.78	22.5	9.27	1.48
2	50	10	10	0.74	0	0.86	0.66	7.03	2.05
3	50	10	15	0.81	0	0.83	1.94	6.89	1.88
4	50	10	20	0.56	0	0.81	0	7.71	3.22

According to Table 1 it maintains the value of alpha and regulating the value of beta to get the required result of Kp,Ki and Kd.

The response in Fig.1(a) shows that in PSO when the number of iteration is 50 and manipulating the value of beta by keeping the value of alpha at 10 i.e. constant from that all the factors like maximum overshoot, rise time, settling time will be determined.

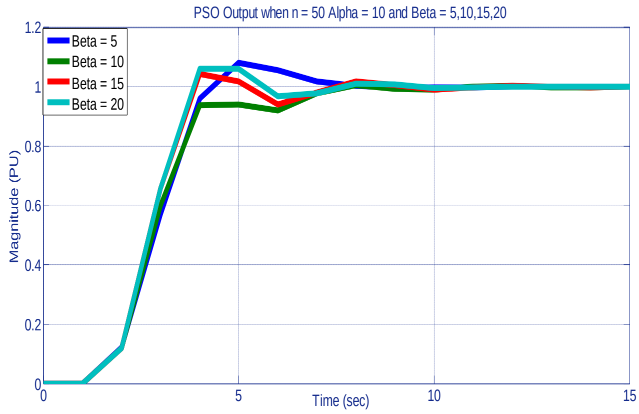


Fig. 1(a) output response when n =50 Alpha = 10 and Beta = 5,10,15,20

When n=50 and applying PSO algorithm, by keeping the value of alpha constant at 10 and by varying the value of beta from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp= 0.682,0.7463,0.8188 and 0.5631 and Kd = 0.7854,0.8605,0.8333 and 0.8155. It concludes that when Beta = 5 (that is minimum) the maximum overshoot is greater than all the other values of Beta, as well as the value settling time is also maximum in this case. The rise time of Beta = 20 is greater than all the values of Beta.

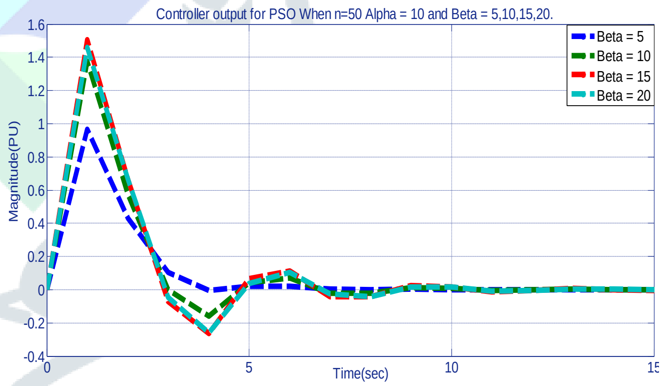


Fig. 1(b) Controller output response when n =50 Alpha = 10 and Beta = 5,10,15,20

The aboveFig.1(b) shows that controller output when n=50 and applying PSO algorithm, by keeping the value of alpha constant at 10 and by varying the value of beta from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of the controller output.

TABLE 2 For PSO when n =200 alpha = 10 and Beta = 5,10,15,20

s. no	n	α	β	Kp	Ki	Kd	Mp %	Ts	Tr
1	200	10	5	0.721	0	0.85	0.27	7.12	2.11
2	200	10	10	0.735	0	0.86	0.47	7.08	2.08
3	200	10	15	0.843	0	0.78	2.42	8.13	1.81
4	200	10	20	0.846	0	0.60	6.75	7.08	1.74

The Table 2 shows that by maintaining the value of alpha and regulating the value of beta it gives the required result of Kp, Ki and Kd.

The response in Fig.2(a) shows that in PSO when the number of iteration is 200 and manipulating the value of beta by keeping the value of alpha at 10 i.e. constant from that all the factors like maximum overshoot, rise time, settling time will be determined.

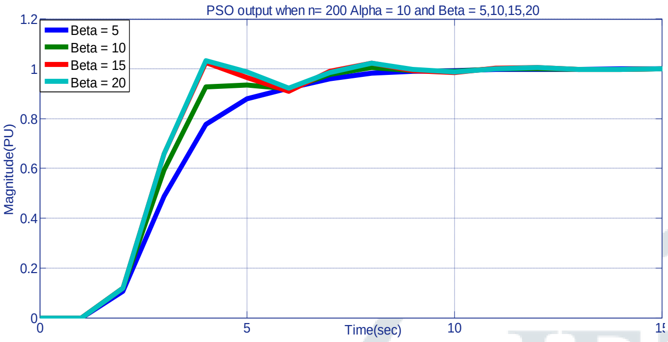


Fig. 2(a) Output response when n =200 Alpha = 10 and Beta = 5,10,15,20

The above figure shows that when n=50 and applying PSO algorithm, by keeping the value of alpha constant at 10 and by varying the value of beta from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp= 0.7214, 0.7359 0.8436 and Kd = 0.8552, 0.8675, 0.7856and 0.6073. It concludes that when Beta = 10 then the maximum overshoot is greater than all the other values of Beta, as well as the value settling time maximum when Alpha = 10 and Beta =15 in this case. The rise time when Alpha =10, Beta = 20 is greater than all the values of Beta.

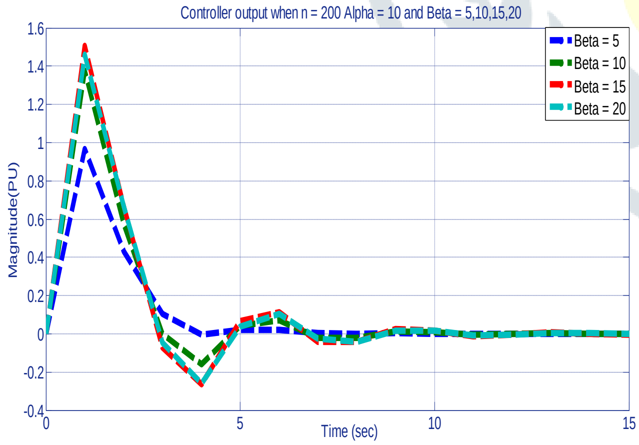


Fig. 2(b) Controller output response when n =200 Alpha = 10 and Beta = 5,10,15,20

The above figure 2(b) shows that controller output when n=50 and applying PSO algorithm, by keeping the value of alpha constant at 10 and by varying the value of beta from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp= 0.7214, 0.7359 0.8436 and Kd = 0.8552, 0.8675, 0.7856and 0.6073.

TABLE 3 For PSO when n =50 alpha = 5,10,15,20 and Beta = 10

s.no	n	A	β	Kp	Ki	Kd	Mp %	Ts	Tr
1	50	5	10	0.66	0	0.41	7.95	6.95	2.01
2	50	10	10	0.74	0	0.86	0.66	7.03	2.05
3	50	15	10	0.83	0	0.67	4.41	6.97	1.78
4	50	20	10	0.83	0	0.58	6.10	7.09	1.76

The Table 3 shows that by maintaining the value of beta and regulating the value of alpha it gives the required result of Kp, Ki and Kd

The response in Fig.3(a) shows that in PSO when the number of iteration is 50 and manipulating the value of alpha and by keeping the value of beta at 10 i.e. constant from that all the factors like maximum overshoot, rise time, settling time will be determined.

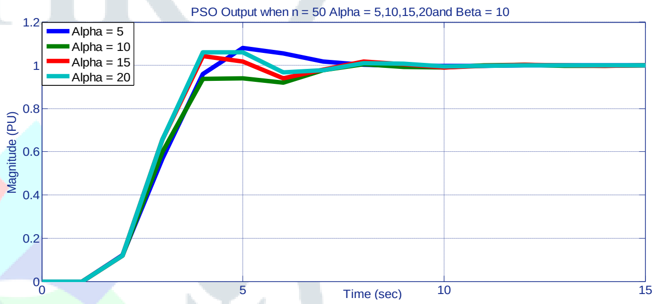


Fig 3(a) For PSO when n =50 Alpha = 5,10,15,20 and Beta = 10

The above figure shows that when n=50 and applying PSO algorithm, by varying the value of Alpha from 5 to 20 (in 4 interval) and by keeping the value of Beta constant at 10, by applying PSO algorithm it will gives the value of Kp= 0.6685, 0.7463, 0.8362, 0.8321 and Kd = 0.4111, 0.8605, 0.6740 and 0.5850. It concludes that when Beta = 5 then the maximum overshoot is greater than all the other values of Beta, as well as the value settling time and rise time is maximum when Alpha =10.

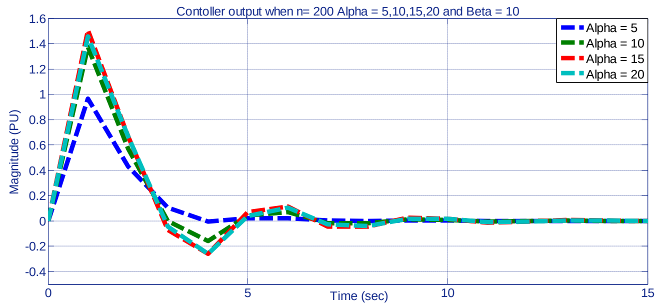


Fig 3(b) Controller output response for PSO when n =50 Alpha = 5,10,15,20 and Beta = 10

The above figure 3(b) shows that controller output when n=50, by keeping the value of Beta constant at 10 and by varying the value of Alpha from 5 to 20 (in 4 interval) and then applying PSO algorithm it will gives the value of Kp and Kd.

TABLE 4 For PSO when n = 200 alpha = 5,10,15,20 and Beta = 10

s. no	n	α	β	Kp	Ki	Kd	Mp %	Ts	Tr
1	200	5	5	0.8809	0	0.6724	8.1781	8.3174	1.6991
2	200	10	10	0.8403	0	0.7790	2.3438	8.1187	1.8221
3	200	15	15	0.8747	0	0.8018	5.7273	8.3137	1.7434
4	200	20	20	0.9762	0	0.4032	23.085	9.3541	1.4851

The Table 4 shows that by maintaining the value of alpha and regulating the value of beta it gives the required result of Kp, Ki and Kd.

The response in Fig.4(a) shows that in PSO when the number of iteration is 200 and manipulating the value of alpha and by keeping the value of beta at 10 i.e. constant from that all the factors like maximum overshoot, rise time, settling time will be determined.

Fig. 4(a) For PSO when n =200 alpha = 5,10,15,20 and Beta = 10

The above figure shows that when n=200 and by varying the value of Alpha from 5 to 20 (in 4 interval) and by keeping the value of beta constant i.e. 10 and by applying PSO algorithm it will gives the value of Kp= 0.5316, 0.7359 0.8437 and 0.8403 and Kd = 0.1855, 0.8675, 0.7855 and 0.7359. It concludes that when Alpha = 20 then the maximum overshoot is greater than all the other values of Beta, as well as the value of settling time is maximum when Alpha = 15 for and the rise time when Alpha =10, Beta = 20 is greater than all the values of Beta.

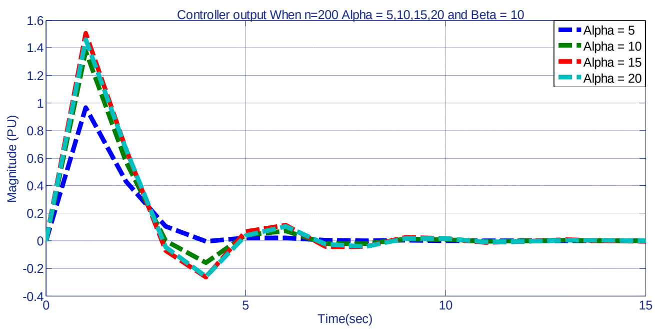


Fig 4(b) Controller output response for PSO when n =50 alpha = 5,10,15,20 and Beta = 10

The above figure 4(b) shows that the controller output , when n=200 by keeping the value of Beta constant at 10 and by varying the value of Alpha from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp and Kd.

TABLE 5 For PSO when n = 200 alpha = 5,10,15,20 and Beta = 10

s. no	n	α	B	Kp	Ki	Kd	Mp %	Ts	Tr
1	200	5	10	0.53	0	0.81	0	7.90	3.52
2	200	10	10	0.73	0	0.86	0.47	7.08	2.08
3	200	15	10	0.84	0	0.78	2.43	8.13	1.81
4	200	20	10	0.84	0	0.7	3.31	8.10	1.80

The Table 5 shows that by maintaining the value of beta and regulating the value of alpha it gives the required result of Kp, Ki and Kd.

The response in Fig. 5(a) shows that in BFO when the number of iteration is 50 and manipulating the value of beta and by keeping the value of alpha at 10 i.e. constant from that all the factors like maximum overshoot, rise time, settling time will be determined.

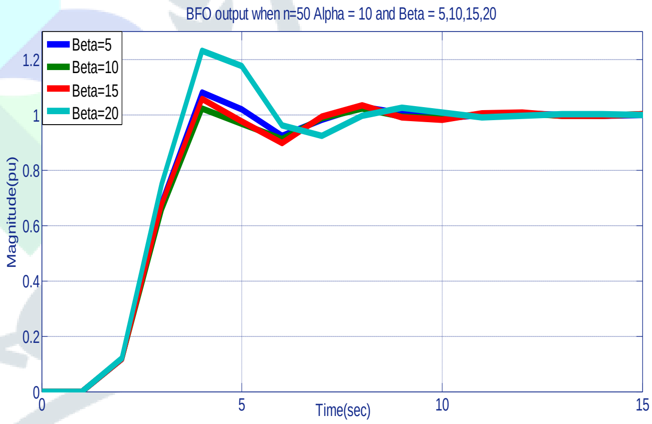


Fig 5(a) For PSO when n =50 alpha = 10 and Beta = 5,10,15,20

The above figure shows that when n=50 and keeping the value of Alpha constant i.e. 10 and by varying the value of Beta from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp= 0.8809, 0.8403 0.8747 and 0.9762 and Kd = 0.6724, 0.7790, 0.8018 and 0.4032. It concludes that when Beta = 20 then the maximum overshoot as well as settling time is greater than all the other values of Beta, and the rise time when Beta= 15 is greater than all the values of Beta.

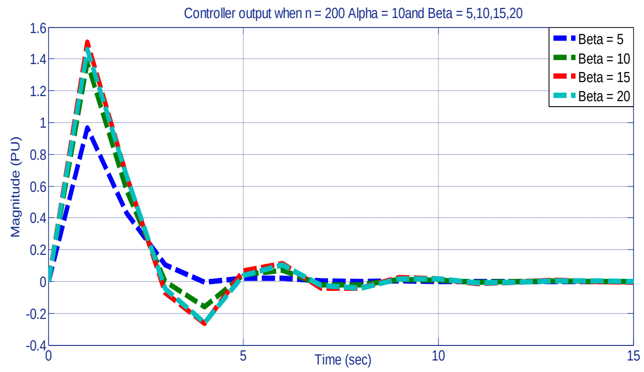


Fig 5(b) Controller output response for PSO when n =50 alpha = 10 and Beta = 5,10,15,20

The above figure 5(b) shows that controller output when n=50 and applying PSO algorithm, by keeping the value of alpha constant at 10 and by varying the value of beta from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp and Kd and the fluctuation in the output of controller is totally depends upon the value of Alpha and Beta.

TABLE 6 For PSO when n = 200 alpha = 5,10,15,20 and Beta = 10

s.no	n	α	β	Kp	Ki	Kd	Mp %	Ts	Tr
1	200	10	5	0.16	0	0.55	0	21.7	11.2
2	200	10	10	0.81	0	0.78	10.4	7.01	1.76
3	200	10	15	0.54	0	0.74	4.97	8.19	2.39
4	200	10	20	0.92	0	0.91	0	42.7	22.9

The Table 6 shows that by maintaining the value of beta and regulating the value of alpha it gives the required result of Kp, Ki and Kd

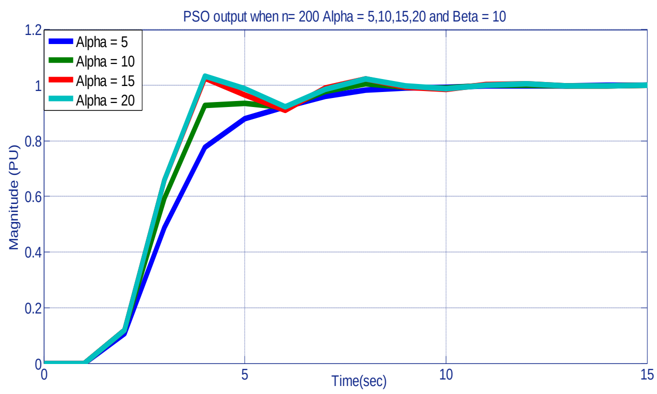


Fig. 6(a) For PSO when n =200 alpha = 5,10,15,20 and Beta = 10

The above figure shows that when n=200 and by varying the value of Alpha from 5 to 20 (in 4 interval) and by keeping the value of beta constant i.e. 10 and by applying PSO algorithm it will gives the value of Kp= 0.5316, 0.7359 0.8437 and 0.8403 and Kd = 0.1855, 0.8675, 0.7855 and 0.7359. It concludes that when Alpha = 20 then the maximum overshoot is greater than all the other values of Beta, as well as the value of settling time is maximum when Alpha = 15 for and the rise time when Alpha =10, Beta = 20 is greater than all the values of Beta.

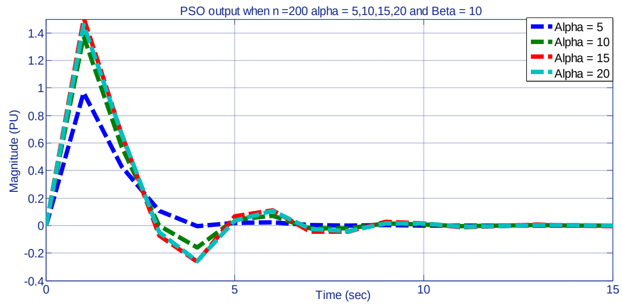


Fig 6(b) Controller output response for PSO when n =200 Alpha = 5,10,15,20 and Beta = 10

The above figure 6(b) shows that the controller output , when n=200 by keeping the value of Beta constant at 10 and by varying the value of Alpha from 5 to 20 (in 4 interval) and by applying PSO algorithm it will gives the value of Kp and Kd.

6. Conclusion

From the closed discussion it is seen that by applying PSO algorithm it provides optimal values for PID parameters for better system performance. Using PSO it can be seen that the best overshoot is achieved many times along with good rise time as well as settling time. BFO algorithm is next optimization technique applied for optimization of PID parameters for stability enhancement of plant model. Overshoot, rise time and settling time are achieved in specified range but as compare to PSO it is not.

References

[1]Dingyu Xuet and D.P. Atherton, "A reduction based pid controller design algorithm for a class of plant model",u.k. proceeding of 23 conference on decision and control dec 1993.

[2]Anitha Mary.x and l.sivakumar "a reduced order transfer function models for alstom gasifier using genetic algorithm "international journal of computer applications (0975 – 8887) volume 46– no.5, may 2012 .

[3] Yogendra Kumar Soni and Rajesh Bhatt "PSO optimized reduced order PID Controller design"International Journal Of Advanced Resarch in Computer Engineering & Technology.Volume 2 Issue 3,March-2013,pp 989-992 ISSN:2278-1323.

[4] S. Morkos and H. Kamal " Optimal Tuning of PID Controller using Bacteria Foraging based Particle Swarm Optimization Algorithm," Int. J. of Computers, Communications & Control, ISSN 1841 9836, E-ISSN 1841-9844 Vol. VII, No. 1 (March), pp. 101-114, 2012.

[5] Vinay Gupta, AshisPatra, —Design of Self-Tune PID Controller for Governor Control to Improve Dynamic

Characteristics ―International Conference (ICITM-2011) to be held on 3rdJuly2011 at Ranchi, India

[6] S.R.Vaishnav, Z.J.Khan, ―Design and Performance of PID and Fuzzy Logic Controllers with Smaller Rule Set for Higher Order System‖, International Conference on Control & Automation (ICCA'07), HongKong, china, 2007, pp.1469-1472.

[7] J.G. Ziegler and N.B.Nichols, ―Optimal Setting for Automatic Controller‖, Trans. ASME, Vol.64, pp.759-768, 1942.

[8] M. Zhuang and D.P. Athertong, ―Automatic Tuning of Optimum PID Controller‖, IEE proceeding part-D: control Theory and ApplicationVol.140, No.3, pp.216-224, 1993.

[9] K.J. Astrom and T. Haglund, ―PID Controller: Theory, Design and Tuning‖, 2nded.Research Tringle park, NC: Instrum.Soc.Amer, 1995.

[10] K.H. Ang, G. Chong and Y. Li, ―PID Control System Analysis, Design and Technology‖, IEEE Transaction on Control System Technology, Vol.13, No.4, 2005pp.559-576.

[11] Ritu Gopal and Dr. Dharmendra kumar singh - "Performance Analysis of Induction Motor For Sudden Load Disturbance Using PI And FUZZY Controller" IJRITCC | June 2015, Volume: 3 Issue: 6, ISSN: 2321-8169

[12] Gaurav Singh Yadav, Amit Agrawal, Dr. Dharmendra Kumar Singh - *"Power Flow Problem Reduced Using Unified Power Flow Controller" International Journal of Science and Research (IJSR) ISSN: 2319-7064 Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438*

Acknowledgment

I would like to express my sincerest gratitude to H.O.D. of EEE Department of Dr C.V. Raman University, who has contributed, directly or indirectly, in accomplishing this paper.

Author Profile

Lalit Singh Bisen: M.Tech (power system) is pursuing from Dr. C.V. Raman University Bilaspur (C.G.). He has done B.E. (Electrical and Electronic Engineering) From C.V.R.U. Bilaspur (C.G.), India

Dr. Dharmendra Kumar Singh: H.O.D. Electrical and Electronics department, Dr. C.V. Raman university, bilaspur. He has done his Phd from Dr. C.V. Raman university, bilaspur, (C.G.), India