PID controller tuning using metaheuristic optimization algorithms for benchmark problems

Vishal Gholap, Chaitali Naik Dessai and V Bagyaveereswaran
School of Electrical Engineering, VIT University, Vellore - 632014, Tamil Nadu, India.
E-mail: vbagyaveereswaran@vit.ac.in

Abstract: This paper contributes to find the optimal PID controller parameters using particle swarm optimization (PSO), Genetic Algorithm (GA) and Simulated Annealing (SA) algorithm. The algorithms were developed through simulation of chemical process and electrical system and the PID controller is tuned. Here, two different fitness functions such as Integral Time Absolute Error and Time domain Specifications were chosen and applied on PSO, GA and SA while tuning the controller. The proposed Algorithms are implemented on two benchmark problems of coupled tank system and DC motor. Finally, comparative study has been done with different algorithms based on best cost, number of iterations and different objective functions. The closed loop process response for each set of tuned parameters is plotted for each system with each fitness function.

1. Introduction

PID controller is a traditional controller used almost in all applications to stabilize the system and get required closed loop responses. This is due to their robust nature and wide operating range [1]. The parameters provided to the controller must give the desired responses optimally. This is the reason we need to find optimal values of the tuning parameters. PSO, GA, SA is the algorithms used for finding those parameters. This paper deals with the comparison between the above mentioned algorithms. Comparison is done on the basis of time domain analysis, obtained tuned parameters, number of iterations, fitness function values [1] [2]. In this paper two different systems are considered. Two different objective functions are considered, one is dependent on ITAE and other is dependent on time domain analysis. Differences in the values of tuning parameters will be studied by using two different systems and cost functions.

2. Benchmark Problems

2.1 System 1 - Coupled Tank System

Consider a Coupled tank system as shown in the fig. Relation between input flow rate, tank output flow rate, fluid level and tank cross sectional area can be given as,
Figure 1. Coupled tank system

For tank 1,
\[ Q_1 - Q = A \frac{d}{dt} H_1 \]  
\[ ---(1) \]

For tank 2,
\[ Q_1 - Q_2 = A \frac{d}{dt} H_2 \]  
\[ ---(2) \]

\( Q \) is flow rate of input pump, \( Q_1 \) and \( Q_2 \) are output flow rates of tank 1 and 2, \( H_1 \) and \( H_2 \) indicate the levels up to which tank 1 and 2 are filled. \( A \) is tank cross-sectional area.

State space representation of the coupled tank system is,

\[
\begin{bmatrix}
    h_1 \\
    h_2
\end{bmatrix}
= \begin{bmatrix}
    -\frac{K_1}{A} & \frac{K_1}{A} \\
    \frac{K_1}{A} & -(K_1 + K_2) \frac{1}{A}
\end{bmatrix}
\begin{bmatrix}
    h_1 \\
    h_2
\end{bmatrix}
+ \begin{bmatrix}
    1 \\
    0
\end{bmatrix} q_i 
\]  
\[ ---(3) \]

Transfer Function of above state space representation is,

\[
G(s) = \frac{1}{K_2 \left( \frac{A^2}{K_1 K_2} s^2 + \left( \frac{A(2K_1 + K_2)}{K_1 K_2} \right) s + 1 \right)}
= \frac{1}{K_2 \left( S T_1 + 1 \right) \left( S T_2 + 1 \right)}
\]

Where \( T_1 T_2 = \frac{A^2}{K_1 K_2} \), \( T_1 + T_2 = \left( \frac{A(2K_1 + K_2)}{K_1 K_2} \right) \)
\[
K_1 = \frac{\alpha}{2\sqrt{H_1 - H_2}} \quad \text{and} \quad K_2 = \frac{\alpha}{2\sqrt{H_2 - H_3}}
\]

Let, \( H_1 = 18 \) cm, \( H_2 = 14 \) cm, \( H_3 = 6 \) cm, discharge coefficient \( \alpha = 9.5 \), \( A = 32 \) \( \text{cm}^2 \). Hence, the transfer function of a coupled tank system can be represented as,

\[
G(s) = \frac{0.002318}{s^2 + 0.201s + 0.00389^2}, \quad \text{---(4)}
\]

Adding PID controller to the block,

\[
G_{\text{rem}}(s) = \frac{0.002318(K_p s^2 + K_r s + K_i)}{s^3 + (0.201 + 0.0023K_p) s^2 + (0.0038 + 0.0023K_r) s + 0.0023K_i}
\]

Closed Loop Coupled Tank System with unity gain feedback:

![Diagram](image)

**Figure 2.** Closed loop system with unity feedback.

Tuning parameters using Auto-tuning in matlab,

\( K_p = 2.9566 \),

\( K_r = 0.0730 \),

\( K_D = 33.470 \),

\[ \text{---(6)} \]

### 2.2 System 2 - Electrical System

Consider a DC motor whose equivalent electrical representation in shown in the fig. \( V \) is the input voltage, \( L \) is the equivalent coil inductance, \( R \) is the equivalent resistance. ‘\( b \)’ friction constant , ‘\( J \)’ is inertia constant.
The motor torque, $T$ and armature current $i$ are related by, constant $K_t$. The back emf $e$ is related to the rotational velocity by the following equations:

$$T = K_t i$$ \quad \text{(7)}

$$e = K_e \dot{\theta}$$ \quad \text{(8)}

$K_t$ and $K_e$ in SI units are equal. Consider $K_t = K_e = K$

Combining, Newton’s Law with Kirchoff’s Law, we get,

$$J \ddot{\theta} + b \dot{\theta} = K i$$ \quad \text{(9)}

$$L \frac{d}{dt} i + Ri = V - K \theta$$ \quad \text{(10)}

State Space Representation,

$$\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{K}{J} \\ -\frac{R}{L} & -\frac{K}{L} \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} V$$

$$y = \dot{\theta} = \omega = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}$$

Consider,

$J=0.01$, $b=0.1$, $K=0.01$, $R=1$, $L=0.5H$

State Space representation becomes,

$$\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -10 & 1 \\ -0.02 & -2 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} V$$

$$y = \dot{\theta} = \omega = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}$$ \quad \text{(11)}

Transfer Function on the DC motor is,
\[
G(s) = \frac{0.005S^2 + 0.06S + 0.1001}{0.005S^3 + (0.06 + 0.01K_p)S^2 + (0.10 + 0.01K_p)S + 0.01K_i}
\]  \quad \text{--- (12)}

Adding PID controller to the block,

\[
G_{\text{eq}}(s) = \frac{0.01(K_pS^2 + K_iS + K_d)}{0.005S^3 + (0.06 + 0.01K_p)S^2 + (0.10 + 0.01K_p)S + 0.01K_i}
\]  \quad \text{--- (13)}

3. Fitness Functions

3.1 Integral Time Absolute Error

\[
\text{Obj1} = \text{ITAE} = \sum (1 - \text{output}) \times \text{time}
\]  \quad \text{--- (14)}

3.2 Time Domain Specifications

\[
\text{Obj2} = \left( \frac{1}{1 + e^{-\alpha}} \left( T_r + T_s \right) + \frac{e^{-\alpha}}{1 + e^{-\alpha}} \left( M_p + E_{SS} \right) \right)
\]  \quad \text{--- (15)}

Where,

- \(T_r\) = Rise Time,
- \(T_s\) = Settling Time,
- \(\alpha\) = weighting factor,
- \(M_p\) = Peak Overshoot,
- \(E_{SS}\) = Steady state error.

Value of \(\alpha\) lies in the range of -5 to 5. If, \(\alpha\) is negative then \(M_p\) and \(E_{SS}\) decreases. If, \(\alpha\) is positive then \(T_r\) and \(T_s\) decreases. Hence, to maintain balance in all these performance criteria, assume \(\alpha = 0\).

4. Optimization Algorithms

4.1 Genetic Algorithm

Genetic algorithm is based on biological evolution. This algorithm randomly generates individuals in the beginning. Here, the individuals are different solutions in the problem space. They are modified in every iteration. Initially, individuals are selected at random to be parents and reproduce new set of solutions for the next iteration. The new generation evolved is considered to be better solution than the earlier ones. Now, the individuals selected for regeneration are based on their fitness function. As the number of iteration increase, we move toward more optimal solution. When the required fitness of the function or the performance measure is achieved or the number of specified iteration is exceeded, the algorithm stops and gives the optimal solution. The required fitness function value depends on whether we have to maximize or minimize the fitness function or performance measure. [3][4]
Algorithm Flow Diagram:

4.2 Particle Swarm Optimization Algorithm

This optimization technique is based on bird flocking. All the particles together locate the optimal solution point in the problem space. Problem space is set of all possible solutions. PSO gets better results in less number of iterations and it has been implemented in many applications. Particles are potential solutions which fly through problem space [10]. Each particle in the swarm stores its own best fitness function value. We call it as particle best (pbest). Another best value which is tracked is the global best value gbest. This is the best fitness function value amongst all the particles in the swarm. Initially, positions of all particles are initialized randomly in space with small initial velocities. The particle position and velocity are updated and fitness function value is found out at the end of iteration. [8]

Algorithm steps:-

- Initializing positions and velocities of particles randomly within the problem space.
- Finding the fitness function value for each particle in the population.
- Compare this fitness function value with the current pbest value and update it if necessary.
- Compare pbest values of all particles and update the gbest value.
- Carry out iterations from step 2 to 5 until minimum fitness function is obtained.

Governing equations [9]:-

\[ V_i(t+1) = wV_i(t) + r_1c_1(P_i - X_i(t)) + r_2c_2(G - X_i(t)) \] \hspace{1cm} (16)

\[ X_i(t+1) = X_i(t) + V_i(t+1) \] \hspace{1cm} (17)

Here,

\[ X_i(t) = \text{position vector of particle in search/problem space} \]

\[ X_i(t+1) = \text{new position vector of the particle in search/problem space} \]
\( V_i(t) \) = Velocity vector of particle in search space. Dimensions of \( V \) and \( X \) are same. It denotes the movement of particle in sense of direction.

\( V_i(t+1) \) = new velocity vector of the particle in search/problem space.

\( P_i(t) \) = Personal best of the particle.

\( G_j(t) \) = Global best amongst all the particles.

\( \omega \) = real valued inertia coefficient.

\( c_1, c_2 \) = real valued acceleration coefficients. It is a acceleration coefficient.

\( r_1, r_2 \) = random numbers uniformly distributed in the range of zero to one.

\( P_i(t) - X_i(t) \) is the vector joining personal best and position vector of particle.

\( G_j(t) - X_j(t) \) is the vector joining global best and position vector.

The particle moves somewhat parallel to vectors (17), (18) and \( V_i(t) \). This gives us new position of the particle. Adding all these vectors we get new velocity of the particle in the problem space as in equation (17). Adding \( X_i(t) \) and \( V_i(t+1) \) we get new position vector as in equation (18).

4.3 Simulated Annealing (SA)

Annealing is the process of heating a material and allowing it to cool down slowly. This is done to decrease defects in the material and attain minimum energy state. This state is called as ‘ground state’ [5][7]. This algorithm is used to solve unconstrained and bound constrained problems. A new point is created in every iteration. The new point distance from the current point is dependent on the probability distribution. Scale is proportional to the temperature. The algorithm rejects the points which increase the fitness function value and accept points that minimize the value. On a contrary it accepts certain points which increase the fitness function value. This helps the algorithm to avoid getting trapped in local minima. As, temperature decrease, search space also decreases [6].

PID Algorithm Flow Diagram:-

![Figure 5. Simulated annealing algorithm flow diagram [6]](image)

5. Simulation and analysis

PID controller tuning parameters are found out for the systems mentioned in equations (4) & (12) by using PSO, GA and SA. This paper considers two different fitness/cost functions (equations (14) & (15)) for tuning. In this case, we consider system 1 and ITAE as the fitness function to be reduced. On applying PSO algorithm, GA and SA, we get tuning parameters for PID controller as shown in table. When controller is
provided to the system with these tuning parameters, we get response as shown in the fig. Hence, the
following cases are provided with their respective responses.

![Figure 6. Closed loop response using system 1 and ITAE](image)

![Figure 7. Closed loop response using system 1 and obj2](image)

**Table 1.** Tuning Parameters

|        | System 1 |         | System 2 |         |
|--------|----------|---------|----------|---------|
|        | Obj1     | Obj2    | Obj1     | Obj2    |
| **GA** |          |         |          |         |
| $K_p$  | 26.404   | 9.716   | 8.614    | 6.515   |
| $K_i$  | 0.945    | 0.183   | 21.762   | 14.4    |
| $K_D$  | 21.613   | -1.196  | -1.354   | 0.024   |
| **PSO**|          |         |          |         |
| $K_p$  | 66.614   | 82.91   | 149.6    | 249.99  |
| $K_i$  | 0.0001   | 0.0001  | 249.9    | 249.9   |
| $K_D$  | 249.99   | 249.99  | 12.34    | 18.09   |
| **SA** |          |         |          |         |
| $K_p$  | 64.101   | 56.04   | 133.796  | 88.632  |
| $K_i$  | 0.115    | 0.51    | 225.788  | 27.16   |
| $K_D$  | 249.887  | 218.781 | 10.495   | 50.369  |
Figure 8. Closed loop response using system 2 and ITAE

Figure 9. Closed loop response using system 2 and obj 2
PSO and SA algorithm give similar tuning parameters as compared to GA for both the systems and objective functions.

Table 2. Minimum Cost Table

| System 1 | System2 |
|----------|---------|
| Min. Cost with obj1 | Min. Cost with obj2 | Min. Cost with obj1 | Min. Cost with obj2 |
| GA       | 129.54  | 28.68   | 1.28     | 1.727   |
| PSO      | 20.4133 | 7.54    | 0.0096   | 0.073   |
| SA       | 20.79   | 5.07    | 0.0136   | 0.027   |

SA and PSO provide us with minimum fitness function value as compared to GA.

Table 3. Iteration Table

| System 1 | System2 |
|----------|---------|
| No. of Iteration With obj1 | No. of Iteration With obj2 | No. of Iteration With obj1 | No. of Iteration With obj2 |
| GA       | 100     | 92      | 100      | 92      |
| PSO      | 100     | 100     | 100      | 100     |
| SA       | 5439    | 2201    | 1914     | 1528    |

It is the nature of simulated annealing algorithm that it will converge slowly. But it will give good results. One iteration is small but more iteration is required.

The time domain analysis for system 1 and 2 is given in table 4 and table 5. We can see that rise time is less for system 1 with obj1 as compared to obj2 with all three algorithms. Settling time required for PSO and SA is less as compared to GA. Peak overshoot is same for all algorithms. Peak time is less for simulated annealing.

Table 4. System 1 Time Domain Analysis

| Rise Time (sec) | Settling Time (sec) | Overshoot (percent) | Peak Time (sec) |
|-----------------|---------------------|---------------------|-----------------|
| 6.0043          | 32.3338             | 25.1048             | 13.7000         |
| 18.1244         | 68.2591             | 8.4378              | 40.7000         |
| 2.9533          | 10.5748             | 3.2779              | 7.2000          |
| 15.7573         | 69.1034             | 14.0005             | 46.7000         |
| 1.7546          | 7.5283              | 17.8960             | 4.1000          |
| 2.1255          | 9.3623              | 15.9880             | 4.9000          |

Table 5. System 2 Time Domain Analysis
Table 5. System 2 Time Domain Analysis

|                 | GA Obj1 | GA Obj2 | PSO Obj1 | PSO Obj2 | SA Obj1 | SA Obj2 |
|-----------------|---------|---------|----------|----------|---------|---------|
| Rise Time (sec) | 0.4186  | 1.2879  | 0.0869   | 0.0854   | 0.0846  | 0.0852  |
| Settling Time (sec) | 0.7801  | 2.1620  | 0.1714   | 0.6092   | 0.6295  | 0.4191  |
| Overshoot (percent) | 2.2348  | 0.0638  | 0.3639   | 0.0798   | 2.3695e-05 | 2.2204e-14 |
| Peak Time (sec) | 1       | 3.8000  | 0.2000   | 2        | 5.7000  | 23.3000 |

6. Conclusion

Tuning parameters are found for both the systems with both the fitness functions using GA, PSO, and SA. Comparison for all three algorithms is done with the obtained tuning parameters, time domain analysis, number of iterations, and value of cost functions. Closed loop Responses for each combinations are plotted. Each system is stable and settles down. System with SA PID parameters settles down quickly as compared to PSO and SA, but its peak overshoot is more. In system with PSO tuned parameters peak overshoot is least. PSO and SA are better as compared to GA as System with GA settles slowly with more overshoot and greater minimum cost function value PSO algorithm converges quickly as compared to SA also the cost minimization is better in PSO. Each algorithm may not give same results after implementing it with same system and objective function.

References

[1] Puralachetty, Mansa Madhavi and Vinay Kumar pamula 2016 Differential evolution and particle swarm optimization algorithms with two stage initialization for PID controller tuning in coupled tank liquid lvel system IEEE Int. Conf. on Advanced Robotics and Mechatronics (ICARM) 507 - 11
[2] Nagrath I J and Gopal M 2006 Control systems engineering 4th edition (New Age International)
[3] Kumar, Himanshu, Raghav Kumar, Jyoti Yadav, Asha Rani, and Vijiander Singh 2016 Genetic Algorithm based PID controller for blood pressure and Cardiac Output regulation IEEE Int. Conf. on Power Electronics Intelligent Control and Energy Systems (ICPEICES) 1 - 6
[4] Rout M K, Sain D, Swain S K, and Mishra S K 2016 PID controller design for cruise control system using genetic algorithm, Int. Conf. on Electrical Electronics and Optimization Techniques (ICCEOT) pp 4170 - 74.
[5] Bansal, Ruchi, Mohit Jain, and Bharat Bhushan 2014 Designing of Multi-objective Simulated Annealing Algorithm tuned PID controller for a temperature control system 6th IEEE Power India Int. Conference (PICON) 1 - 6
[6] Ping, Xiao, Tian Li, and Gao Hong 2013 Research on Four-wheel-Steering Automobile Based on Simulated Annealing PID Algorithm 3rd Int.Con. on Intelligent System Design and Engineering Applications(ISDEA) 1175 - 78
[7] Dashahi, Mohammad J, Kambiz Shojae G, Mohammad S, Seyedkashi H and Behnam T M 2010 Novel simulated annealing algorithm in order to optimal adjustment of digital PID controller 11th Int. Con. on Control Automation Robotics & Vision (ICARCV) 1766 - 71
[8] Gaing and Zwe-Lee 2003 Particle swarm optimization to solving the economic dispatch considering the generator constraints IEEE transactions on power systems 18 (3) 1187 - 95
[9] Jeyakumar D N, Jayabarathi T and Raghunathan T 2006 Particle swarm optimization for various types of economic dispatch problems International Journal of Electrical Power & Energy Systems 28 (1) 36 - 42
[10] Prabakaran S, Road Laxshminarayana Rao 2016 Application of Particle Swarm Optimization Algorithm for solving Power Economic Dispatch with Prohibited Operating zones and Ramp-rate limit Constraints International Journal of Emerging Technologies and Engineering (IJETE) 3(3) 234 - 45