Using PSO and GA for Optimization of PID Parameters

Amir Hossein Fathi, Hamid Khaloozadeh, Mohammad Ali Nekoui, Reza Shisheie

Abstract

Proportional-Integral-Derivate (PID) controllers are widely used in industry because of their remarkable efficiency, simple structure and robust performance for a wide range of applications. Parameters tuning (K_p,K_i,K_d) of PID controller is necessary to satisfy the operation of the system. But many tuning methods such as Ziegler-Nichols methods do not work so perfectly as it is expected. Such methods have many disadvantages such as lack of precision, long run time and lack of stability. Therefore, Particle Swarm Optimization (PSO) algorithm and Genetic Algorithm (GA) are applied to system in order to optimize the PID controller parameters and improve the performance of PID controller system. The simulation results show that the PSO method is more effective in improving the step response characteristics such as: overshoot, rise time and settling time.

Keywords: Optimal Control, PID Parameters Tuning, PSO, GA

1. Introduction

In the last decades, process control technique has made great advances in industrial control systems. Numerous control methods such as adaptive control, neural network and fuzzy control have been studied; however, many alternative optimal control techniques have been proposed but PID controller still remains as the most popular controller used in industrial process [1-2]. The main advantages of such controller are simple structure, easy implementation and strong robustness for a wide range of applications. But the performance of PID controller is dependent to perfectly tuning of parameters. A variety of PID controllers methods have been developed such as Ziegler rules and Cohen-coon rules etc but all of them have their own restrictions. For example, Ziegler-Nichols [3-4] method works quiet well but it is laborious and time consuming which is applied to processes with long time constant or delay. Therefore, soft computing techniques like GA, PSO, and ACO have been applied to systems in order to improve controller performances.

Genetic Algorithms [5-6] (GA) was introduced for the first time by John Holland in 1970 and has been widely use to optimize parameters of PID controller [7] but it has some problems which are difficult to be overcome such as premature convergence, slow speed convergence and reduction in diversity. In order to overcome such disadvantages, we can apply PSO algorithm to tune parameters of PID controller.

Particle Swarm Optimization (PSO) originally was introduced by Kennedy and Eberhart in1995 [8] and it has been applied to a wide variety of applications and because of it is simplicity, easy implementation, it has been found to continuous in solving of continuous nonlinear optimization problem.

In this paper, we have applied PSO algorithm to optimize parameters of PID controller and the results of parameters tuning of PID controller are compared to GA. The result indicates that PSO algorithm is a reasonable and effective method for parameters optimization of PID controllers [9-10-11].

2. PID controller

2.1. Controller structure
PID controllers are simple and easy to implement and they are widely used in industry to solve different control problem. The standard PID controller structure is as shown in figure 1 and transfer function is described by following equation in the continuous s-domain (Laplace operator).

\[ G_{pid}(s) = P + I + D = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \]  
(1)

Or

\[ G_{pid}(s) = K_p \left( 1 + \frac{1}{T_i s} + T_d s \right) \]  
(2)

Where \( U(s) \) is control signal and \( E(s) \) is control error in s-domain, respectively; \( K_p \) is the proportional gain, \( K_i = \frac{K_p}{T_i} \) is the integration gain, and \( K_d = K_p T_d \) is the derivative gain. \( T_i \) is the integration time coefficient and \( T_d \) is referred to as the derivation time coefficient. In this context, the controller outputs in time domain is given by:

\[ u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \]  
(3)

Where \( u(t) \) and \( e(t) \) are the control output and error signals in time domain, respectively. The action of the proportional gain, \( K_p \), reduces the rise time of the system response and steady state error but it never eliminates error. On the other hand the integral gain, \( K_i \), reduces the steady state error and derivate gain, \( K_d \), improves stability of system and reduces the overshoot of system as well as improving transient response.

**Figure 1.** PID controller block diagram

Selecting appropriate gains, \( K_p, K_i \) and \( K_d \), which is adjusted in accordance with the specification of the system and satisfy the robustness of the closed loop performance over wide range of frequencies is the main of purpose of the PID controller optimizer. Figure 2 shows the structure of PID controller optimization process. We propose PSO and GA to tune value of three parameters repeatedly until they achieve an acceptance level of performance.

**Figure 2.** Block diagram of PID controller tuning
2.2. Performance indices

A performance index is a quantitative measure of the performance of the system. A system is considered an optimal control system when the system parameters are adjusted so that the index reaches an extreme value, commonly a minimum value [12].

A suitable fitness function (Performance index) is the integral of square error (ISE) which is defined as:

\[
ISE = \int_{0}^{T} e(t)^2 \, dt 
\]  
(4)

ISE is more convenient to minimize initial large amount of error.

Another fitness function is the integral of the absolute error (IAE) as follow:

\[
IAE = \int_{0}^{T} |e(t)| \, dt 
\]  
(5)

This index is especially useful for computer simulation study.

Another widely used fitness function is as:

\[
MSE = \frac{1}{n} \sum_{i=1}^{n} [e(t_i)]^2 
\]  
(6)

The mean of square of the error is mathematically more convenient for analytical and computational purposes.

In this paper the performance ISE is applied to estimate the PID parameters using PID control tune by PSO and GA algorithms and result are compared in transient time \( t_{r} \), settling time \( t_{s} \) and overshoot \( M_p \) of the closed loop system.

3. Tuning methods

3.1. Genetic algorithm (GA)

Genetic Algorithm (GA) is a stochastic optimization algorithm that mimics the process of natural evolution. The initial information of the Genetic Algorithm to start calculation is far away from the near correct solution or is simply arbitrary; therefore, achieving the best solution depends completely to the environment and evolution operators (i.e. selection, crossover and mutation). The algorithm begins the calculation from several independent nods and searches the space which prevents the algorithm from local minimal and suboptimal convergence. In many different research areas such as system identification, control robotics, pattern recognition and fuzzy logic controller design, GA has good performance.

Basically, genetic algorithm consists of three main operators: selection, crossover and mutation.

Selection is a process which selects some fine individuals from the current population to generate matting pool. Some common methods for selection are: Roulette Wheel selection, Stochastic Universal sampling, Tournament selection etc.

Crossover is composing a pair with every two individuals of the population and exchange parts of their chromosomes. By combining the chromosomes of two individuals, new chromosomes are generated and integrated into the population.

Altering one or several gene values for each individual of the population is called mutation.

In this paper genetic algorithm is applied to a system to tune the parameters of PID controller and improve controller performance to ensure optimal control performance at the nominal operation condition [13-14-15].
The steps involved in creating and implementing a genetic algorithm are as follows:

1. Generate an initial, random population of individual (Chromosomes) for fix size (according to congenital methods \( K_p, K_i, K_d \) ranges declared).
2. Evaluate the value of the fitness function.
3. Select the fitness members of the populations.
4. Go to the crossover and mutation operation and make up the new cluster.
5. Go to number 2 until get best value.
6. Select the chromosome with highest fitness as optimal PID controller parameter.

### 3.2. Particle swarm optimization (PSO)

PSO algorithm is one of the evolutionary computational optimization which is based on natural system developed in 1995 [8], inspired from the simulation of social behavior of birds flocks, school of fish or swarm of bees. It has been proven that PSO has simplicity, robustness against nonlinearity, non-differentiability and high dimensionality. The advantages of PSO compared to other optimization techniques are as follow: easy implementation, few parameters to be adjusted and the fact that no gradient information is required [16].

Each particle generated by the PSO algorithm has the potential to be the best solution for the problem. At the beginning, each particle of population is distributed randomly through the search space. The velocity and position are updated by the following two best values. The best value achieved for each particle is best solution which is called personal best \((p_{best})\) and best value achieved so far among all particles is called global best \((g_{best})\). The modified velocity and position of each particle can be calculated via the present velocity and distance form \(p_{best}\) and \(g_{best}\) as shown in following two equations which inertia weight \((w)\) is added to [17-18-19-20].

\[
v_{id}^{t+1} = wv_{id}^t + c_1 r_1 (p_{id} - x_{id}^t) + c_2 r_2 (g_{id} - x_{id}^t) \quad (7)
\]

\[
x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \quad (8)
\]

Where \(c_1\) and \(c_2\) are acceleration constants and, \(r_1\) and \(r_2\) are random values in the range \([0, 1]\). \(v_{id}^t\) and \(x_{id}^t\) represent the velocity and position of the \(i\)th particle with \(d\) dimensions at iteration \(t\). \(p_{id}\) and \(g_{id}\) are the best values of positions which are achieved respectively for the \(i\)th particle and all particles so far. The concept of PSO searching mechanism is illustrated in figure 3.

![Figure 3. The movement of particles](image)

\(c_1\) and \(c_2\) represent the cognitive and social component which lead each particle toward \(p_{best}\) and \(g_{best}\) position. Low values cause roaming far from the target regions before being tugged back and high
value results in abrupt movement toward the target region. Hence, the acceleration constant $c_1$ and $c_2$ are set to 2 according to past experience [21].

The parameter $w$ in formula (7) is inertia weight that increases the overall performance of PSO. It is reported that a larger value of inertia weight encourage global exploration while a small one tends to promote local exploration. Suitable selection of inertia weight ($w$) usually provides a balance between the global and local exploration and reduce the average number of iteration to locate the optimum solution. To achieve a high performance, we linearly decrease the value of inertia weight from about 0.9 to 0.4 during a run. In general, the inertia weight is set according to the following equation:

$$w = w_{\text{max}} - \frac{\text{iter}}{\text{iter}_{\text{max}}} \cdot \left( w_{\text{max}} - w_{\text{min}} \right)$$  \hspace{1cm} (9)

Where $\text{iter}_{\text{max}}$ is the maximum number of iteration in evolution process, $w_{\text{max}}$ is maximum value of inertia weight, $w_{\text{min}}$ is the minimum value of inertia weight and $\text{iter}$ is current value of iteration.

Based on principle of PSO mentioned above, we can summarize the search procedure as below:

1. Specifying the parameters of PID controller such as: swarm size, iteration, acceleration constant, inertia weight factor, lower and upper bounds of the three controller parameters and initializing the individuals of population including searching point, velocities, $p_{\text{best}}$ and $g_{\text{best}}$ position randomly.
2. Calculating the evaluation value of each individual in the population using the evaluation function $(f)$ given by (4).
3. Comparing the value of each individual to its $p_{\text{best}}$. The best value among all $p_{\text{best}}$ is called $g_{\text{best}}$.
4. Update the velocity of each particle by (7).
5. Update the particle position by (8).
6. Update the control parameter.
7. If the number of iterations reaches the maximum, then go to next step. Otherwise, go to number 2.
8. The last $g_{\text{best}}$ has a set of optimal PID controller parameter which is expected to be achieved.

4. Simulation

4.1. Simulation plant

The transfer function of the system is [22]:

$$G(s) = \frac{1}{s^3 + 6s^2 + 5s}$$  \hspace{1cm} (10)

The step response of the above system is shown in figure 4. The various methods used to improve the system performance are mentioned in section 3. It can be observed that the rise time of the step response is very large. A PID controller has to be designed such that it improves the following time response parameters.

4.2. Simulation parameters

4.2.1. Parameter’s initial value of PSO

The parameters of PSO are as follow: Number of particles is 15 and number of functional evaluation is 40, $c_1$ and $c_2$ are 2, $w_{\text{max}}$ is 0.9 and $w_{\text{min}}$ is 0.1, $v_{\text{max}}$ is 25% of range of parameter and dimension of particles are 3.
4.2.2. Parameter’s initial value of GA

The maximum number of generation is 100, the size of population is 55, crossover is 0.8, and mutation probability is 0.06.

4.3. Simulation results

On simulation the following results were obtained. First the system was tuned using Zeigler-Nichols. The values of $k_p$, $k_i$, and $k_d$ obtained are 18, 12.8, 6.32 respectively. The following step response was obtained as shown in figure 5.

As we can see that there is reduction in rise time. The steady state error is zero. But the system has very high overshoot and the settling time has increased and also the response is still sluggish.

The system was again tuned using GA based PID controller. The values of $k_p$, $k_i$, and $k_d$ obtained are
20.2, 24.5 respectively. The following response was obtained as shown in figure 6. As seen in the above figure the settling time and rise time of GA-based PID controller are shorter than those of Ziegler-Nichols based PID controller. The overshoot is also smaller than that those of Ziegler-Nichols based PID controller.

The system is now simulated using PSO based PID controller. The values of $k_p$, $k_i$ and $k_d$ obtained are, 13.07, 0.042, 20.14 respectively. The response is shown in figure 7.

![Figure 7. Step response of PSO-based PID](image)

As seen from above figure the response has completely improved. The settling time and overshoot have reduced and the steady state error is zero too. The results obtained from all the methods have been summarized in Table 1.

<table>
<thead>
<tr>
<th>Tuning methods</th>
<th>overshoot ($M_p$) in %</th>
<th>Settling time ($t_s$) in sec</th>
<th>Rise time ($t_r$) in sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ziegler-Nichols</td>
<td>61</td>
<td>10.4</td>
<td>0.511</td>
</tr>
<tr>
<td>GA-based PID</td>
<td>12</td>
<td>1.97</td>
<td>0.292</td>
</tr>
<tr>
<td>PSO-based PID</td>
<td>4</td>
<td>2.93</td>
<td>0.355</td>
</tr>
</tbody>
</table>

5. Conclusion

In this work, tuning parameters of PID controllers has been made by two different evolutionary computing techniques with initial knowledge are taken from conventional tuning method Ziegler-Nichols. The PID controller was designed for ISE performance index. Simulation results have shown that the PSO method has more robust stability and efficiency, and solve the searching and tuning problems of PID controller parameters more easily and quickly than GA method.

6. References


