Optimal Tuning of PID Controller for AVR System using Modified Particle Swarm Optimization

G.SHABIB , A.G.MOSLEM ,AND A.M.RASHWAN
Department Of Electrical Engineering
High Institute Of Energy
South Valley University
ASWAN,EGYPT
engrashwan@yahoo.com

Abstract
This paper present a new method to determine the optimal tuning of the PID controller parameter of an AVR system using Modified particle swarm optimization (MPSO) algorithm. the AVR is not robust to variations of the power system parameters therefore, it is necessary to use PID controller To increase the stability and performance of the AVR system. Fast tuning of optimum PID controller parameter yield high quality solution. A new time domain performance criteria was also defined. Simulation comparison between the proposed method and genetic algorithm is done, the proposed method was indeed more efficient in improving the on line step response of an AVR system.

Keywords- AVR system ,PID controller, Particle swarm optimization.

1 INTRODUCTION
Even though several control theories have been developed significantly, we do see the widely popular use of proportional integral-derivative (PID) controllers in process control, motor drives, flight control, and instrumentation. The reason of this acceptability is for its simple structure which can be easily understood and implemented. Industries too can boast of the extensive use of PID controllers because of its robustness and simplicity. The past decades witnessed many advancing improvements keeping in mind the requirement of the end users. Easy implementation of hardware and software has helped to gain its popularity. Several approaches have been documented in literatures for determining the PID parameters of such controllers which is first found by Ziegler Nichols tuning [4] . Genetic Algorithm [5] neural network [2], fuzzy based approach[3], particle swarm optimization techniques [10,11] are just a few among these numerous works.

The genetic based methods have also been used in setting of the parameters of PID controllers. However, these methods needs more time to be performed. To overcome this, the real coded genetic algorithm (RCGA) has been suggested in [6]. The other problems of these methods is incapability in optimizing the objective functions, and coefficient are dependent to each other. Therefore, after a while, the GA is not able to produce a new population. As a result, the probability of reaching to the local optimal solution is increased.

Other method that recently has been used in designing of parameters of PID controller, is particle swarm optimization (PSO). This method is very capable in solving continuous non-linear optimization problem. This technique has a shorter calculation time and better convergence characteristics with respect to other stochastic methods [9].

In this paper, a method for designing the PID controller of AVR is presented. This method is based on the modified PSO. The suggested MPSO method is simulated on a network. The results of the simulation shows that when the MPSO method is used, the performance of the PID controller is significantly.

2 OVERVIEW OF PARTICLE SWARM OPTIMIZATION AND ITS MODIFICATION.
Natural creatures sometime behave as a Swarm. One of the main streams of artificial life researches is to examine how natural creatures behave as a Swarm and reconfigure the Swarm models inside the computer. Dr. Eberhart and Kennedy develop PSO, based on analogy of the Swarm of birds and fish school. Each individual exchanges previous experiences among themselves [7]. PSO as an optimization tool provides a population
based search procedure in which individuals called particles change their position with time. In a PSO system, particles fly around in a multi-dimensional search space. During flight each particle adjusts its position according to its own experience and the experience of the neighboring particles, making use of the best position encountered by itself and its neighbors. In the multidimensional space where the optimal solution is sought, each particle in the swarm is moved toward the optimal point by adding a velocity with its position. The velocity of a particle is influenced by three components, namely, inertial, cognitive, and social. The inertial component simulates the inertial behavior of the bird to fly in the previous direction. The cognitive component models the memory of the bird about its previous best position, and the social component models the memory of the bird about the best position among the particles. The particles move around the multidimensional search space until they find the optimal solution. The modified velocity of each agent can be calculated using the current velocity and the distance from Pbest and Gbest as given in the following equation:

\[
V_{ij}^{t+1} = \omega V_{ij}^t + C_1 r_1 (P_{best_{ij}}^{t-1} - X_{ij}^t) + C_2 r_2 (G_{best_{ij}}^{t-1} - X_{ij}^t)
\]

\(i = 1, 2, \ldots, N_D, \quad j = 1, 2, \ldots, N_{par}\)

Using the above equation, a certain velocity, which gradually gets close to Pbest and Gbest, can be calculated. The current position (searching point in the solution space), each individual moves from the current position to the next one by the modified velocity in (1) using the following:

\[
X_{ij}^{t+1} = X_{ij}^t + V_{ij}^t
\]

\(i = 1, 2, \ldots, N_D, \quad j = 1, 2, \ldots, N_{par}\)

where,
- \(t\) Iteration count,
- \(V_{ij}\) Dimension \(i\) of the velocity of particle \(j\) at iteration \(t\),
- \(X_{ij}\) Dimension \(i\) of the position of particle \(j\) at iteration \(t\),
- \(w\) Inertia weight,
- \(C_1, C_2\) Acceleration coefficients,
- \(P_{best_{ij}}\) Dimension \(i\) of the own best position of particle \(j\) until iteration \(t\),
- \(G_{best_{ij}}\) Dimension \(i\) of the best particle in the Swarm at iteration \(t\),
- \(N_D\) Dimension of the optimization problem,
- \(N_{par}\) Number of particles in the swarm,
- \(r_1, r_2\) Two separately generated uniformly distributed random numbers in the range \([0, 1]\).

The following weighting function is usually utilized:

\[
\omega = \omega_{\text{max}} - \frac{\omega_{\text{max}} - \omega_{\text{min}}}{T_{\text{max}}} t
\]

Where, \(W_{\text{max}}, W_{\text{min}}\) Initial and final weights, \(T_{\text{max}}\) Maximum iteration number, \(t\) Current iteration number.

A new variation in the classical PSO since Pbest, is the best solution a particle found so far, it can be considered as the experience a particle acquired in past time. Experience usually helps people learn and accumulate new knowledge. Based on this observation, a new PSO in which the particles explore around its previous best is proposed and therefore the proposed modification in equation (2) is now expressed:

\[
X_{ij}^{t+1} = X_{ij}^t + P_{best_{ij}}^{t-1}
\]

### 3 LINEAR MODEL OF AN AVR SYSTEM

The role of an AVR is to keep constant the output voltage of the generator in a specified range. A simple AVR consists of amplifier, exciter, generator and sensor. The block diagram of AVR with PID controller is shown in Figure 1. The linear models of the elements of the AVR are given in Equations 5-8 [1].

**Amplifier model:**

The amplifier model is represented by a gain \(K_A\) and a time constant \(\tau_A\); the transfer function is

\[
\frac{V_A}{V_i} = \frac{K_A}{1 + \tau_A S}
\]

Typical values of \(K_A\) are in the range of 10 to 400. The amplifier time constant \(\tau_A\) is very small ranging from 0.02 to 0.1 s.
Typical values of $K_A$ are in the range of 10 to 400. The amplifier time constant $\tau_A$ is very small ranging from 0.02 to 0.1 s.

Exciter model:
The transfer function of a modern exciter may be represented by a gain $K_E$ and a single time constant $\tau_E$

$$\frac{V_E}{V_R} = \frac{K_E}{1 + \tau_E S}$$

Typical values of $K_E$ are in the range of 10 to 400. The time constant $\tau_E$ is in the range of 0.5 to 1.0 s.

Generator model:
In the linearized model, the transfer function relating the generator terminal voltage to its field voltage can be represented by a gain $K_G$ and a time constant $\tau_G$

$$\frac{V_L}{V_F} = \frac{K_G}{1 + \tau_G S}$$

These constants are load dependent, $K_G$ may vary between 0.7 to 1.0, and $\tau_G$ between 1.0 and 2.0 s from full load to no load.

Sensor model:
The sensor is modeled by a simple first-order transfer function, given by

$$\frac{V_S}{V_t} = \frac{K_R}{1 + \tau_R S}$$

$\tau_R$ is very small, ranging from 0.001 to 0.06 s.

4 PID CONTROLLER
A. Fundamentals of PID controller

The PID controller is simple and easy to implement. It is widely applied in industry to solve various control problems. It is used to improve the dynamic response as well as to reduce or eliminate the steady-state error. The derivative controller adds a finite zero to the open-loop plant transfer function and improves the transiencresponse. The integral controller adds a pole at the origin, thus increasing system type by one and reducing the steady-state error due to a step function to zero. The PID controller transfer function is

$$\frac{E(s)}{U(s)} = K_p + \frac{K_i}{s} + K_d s.$$  (9)

where $U(s)$ and $E(s)$ are the control (controller output) and tracking error signals in S-domain, respectively; $K_p$ is the proportional gain, $K_i$ is the integration gain, and $K_d$ is the derivative gain. The proportional part of the PID controller reduces error responses to disturbances. The integral term of the error eliminates steady-state error and the derivative term of error dampens the dynamic response and thereby improves stability of the system. The parameter settings of a PID controller for optimal control of a plant (process) depend on the plant’s behavior. To design the PID controller the engineer must choose the tuning way of design parameters to improve the transient response as well as the steady-state error. In the design of a PID controller, the three gains of PID must be selected in such a way that the closed loop system has to give the desired response. The desired response should have minimal settling time with a small or no overshoot in the step response of the closed loop system.
B. Performance Estimation of PID Controller

In general, the PID controller design method using the integrated absolute error (IAE), or the integral of squared-error (ISE), or the integral of time-weighted-squared-error (ITSE) is often employed in control system design because it can be evaluated analytically in the frequency domain [8]. The three integral performance criteria in the frequency domain have their own advantages and disadvantages. For example, a disadvantage of the IAE and ISE criteria is that its minimization can result in a response with relatively small overshoot but a long settling time because the ISE performance criterion weights all errors equally independent of time. Although the ITSE performance criterion can overcome the disadvantage of the ISE criterion, the derivation processes of the analytical formula are complex and time-consuming [8]. The IAE, ISE, and ITSE performance criterion formulas are as follows:

\[
\text{IAE} = \int_{0}^{\infty} |r(t) - y(t)| \, dt = \int_{0}^{\infty} |e(t)| \, dt
\]

(10)

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

(11)

\[
\text{ITSE} = \int_{0}^{\infty} e^2(t) \, dt
\]

(12)

In this paper, a new performance criterion in the time domain is proposed for evaluating the PID controller. A set of good control parameters \(K_p, K_i, \) and \(K_d\) can yield a good step response that will result in performance criteria minimization in the time domain. These performance criteria in the time domain include the overshoot \(M_p\), rise time \(t_r\), settling time \(t_s\), and steady-state error \(E_{ss}\). Therefore, a new performance criterion is defined as follows:

\[
W(K) = (1+ M_p) \times 0.5 \times (t_s + t_r)
\]

(13)

C. Evaluation Function Definition

In the meantime, we defined the evaluation function \(f\) given in (14) as the evaluation value of each individual in population. The evaluation function \(f\) is a reciprocal of the performance criterion \(W(K)\) as in (13). It implies the smaller the value of individual \(K\), the higher its evaluation value:

\[
f = \frac{1}{W(K)}
\]

(14)

In order to limit the evaluation value of each individual of the population within a reasonable range, the Routh–Hurwitz criterion must be employed to test the closed-loop system stability before evaluating the evaluation value of an individual. If the individual satisfies the Routh–Hurwitz stability test applied to the characteristic equation of the system, then it is a feasible individual and the value of \(W(K)\) is small. In the opposite case, the \(W(K)\) value of the individual is penalized with a very large positive constant.

5 PID CONTROLLER DESIGN BASED ON MPSO ALGORITHM

The structure of each particle in MPSO algorithm for designing PID controller is

\[
X = [k_p, k_i, k_d]_{n \times 3}
\]

where

\[k_p, k_i \text{ and } k_d = \text{The coefficient of PID controller which are to be designed using the MPSO algorithm.}\]

\[n = \text{Number of population.}\]

In order to coefficient of PID controller using MPSO algorithm the following steps should be performed.

1. Specify the upper and lower limits using the Routh-Hurwitz algorithm.
2. Choose \(k = 1\).
3. Generate \(X_j^k\) randomly in the specified limit.
4. Compute the evaluation function \(f(x_j)\) using (14) and assign it to \(P_{best_j}^k\), for \(j = 1, 2, ..., n\)
5. Determine \(G_{best}^k\) from \(G_{best}^k = \min (\{P_{best_j}^k\})\)
6. Compute \(v_j^k\) from (1).
7. If \(V_{ij}^k > V_{i}^{max}\), then \(V_{ij}^k = V_{i}^{max}\).
   If \(V_{ij}^k < V_{i}^{min}\), then \(V_{ij}^k = V_{i}^{min}\).
8. Compute \(X_j^k\).
9. Update \(P_{best}\) according to the following For \(j = 1: n\)
If \( f(P_{best}^j) \geq f(X_{j}^{k+1}) \) then:

\[
P_{best}^j = X_{j}^{k+1}.
\]

Step 11: Update Gbest according to the following
For \( j = 1:n \)
If \( f(G_{best}) \geq \min (f(P_{best}^j)) \) then:

\[
G_{best} = P_{best}^j
\]

Step 12: Set \( k = k+1 \).
Step 13: Determine Gbest for the X optimal or continue,
as if \( k \neq k_{max} \) go to step 6 else \( X_{optimal} = G_{best} \)
Step 14: End
In our case, the optimal coefficients for PID controller is \( G_{best} \), so that
\[
G_{best} g1 = k_p , G_{best} g2 = k_i \text{ and } G_{best} g3 = k_d.
\]

Inspired from practical requirements, the lower bounds of the three controller parameters are zero and their upper bounds are set to \( k_{p_{max}} = 1.5, k_{i_{max}} = 1.5 \) and \( k_{d_{max}} = 1.5 \).
The following parameters are used for carrying out the MPSO-PID design:

- The members of each particle are \( k_p, k_i \) and \( k_d \).
- Population size = 50.
- Inertia weight factor \( w \) is set as (3) where \( w_{\text{max}} = 0.9 \) and \( w_{\text{min}} = 0.4 \).
- The limit of change in velocity is set to maximum dynamic range of the variables on each dimension.
- Acceleration constants \( c_1 = 2 \) and \( c_2 = 2 \).
- Maximum iteration is set to 1500.

### 6 SIMULATION

In order to compare the quality of MPSO with PSO and also with GA, two AVR systems are simulated. The specifications of the two systems are given in Table 1. These parameters are according to the block diagram given in Figure 1.

#### Table 1. the parameters of the two simulated AVR systems

<table>
<thead>
<tr>
<th>System</th>
<th>( K_A )</th>
<th>( \tau_A )</th>
<th>( K_E )</th>
<th>( \tau_E )</th>
<th>( K_G )</th>
<th>( \tau_G )</th>
<th>( K_R )</th>
<th>( \tau_R )</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVR 1</td>
<td>10</td>
<td>0.1</td>
<td>1</td>
<td>0.4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.01</td>
</tr>
<tr>
<td>AVR 2</td>
<td>400</td>
<td>0.01</td>
<td>0.2</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.001</td>
</tr>
</tbody>
</table>

#### A. RESULTS OF SIMULATION OF AVR SYSTEM 1

Fig. 2 shows the original terminal voltage step response of the AVR system without a PID controller. To simulate this case, we found that \( MP = 50.6 \% , \text{ ESS} = 0.09 , \text{ tr} = 0.27 \text{ sec} , \text{and ts} = 6.98 \text{ sec} \).

AVR system 1 with the specifications given in Table 1 is simulated for different situations: with GA, PSO and suggested MPSO. The results of the simulation for different algorithms are shown in Table 2. For AVR system 1, the performance of the PID's obtained from the different algorithms, are shown in Figure 3. The transient response for the different algorithms is also illustrated in Figures 4.

As can be seen in Table 2 and Figure 3 and 4, the oscillations of the transient response of the AVR system is improved significantly compared to other algorithms.

As shown from figure 4, the peak overshooting and the steady state error in MPSO algorithm is reduced compared to other two methods.

#### Table 2. The simulation results and the PID's parameters obtained from different algorithms for AVR system 1.

<table>
<thead>
<tr>
<th>Method</th>
<th>( K_p )</th>
<th>( K_i )</th>
<th>( K_d )</th>
<th>( MP ) ( (%) )</th>
<th>( \text{ESS} )</th>
<th>( \text{Ts} )</th>
<th>( \text{Tr} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>0.32</td>
<td>0.28</td>
<td>0.23</td>
<td>9.0</td>
<td>0.0</td>
<td>5.2</td>
<td>1.2</td>
</tr>
<tr>
<td>PSO</td>
<td>0.69</td>
<td>0.42</td>
<td>0.27</td>
<td>2.0</td>
<td>0.0</td>
<td>1.3</td>
<td>0.5</td>
</tr>
<tr>
<td>MPSO</td>
<td>0.45</td>
<td>0.29</td>
<td>0.13</td>
<td>0.09</td>
<td>0.0</td>
<td>0.7</td>
<td>0.3</td>
</tr>
</tbody>
</table>

#### B. RESULTS OF SIMULATION OF AVR SYSTEM 2

In this section, similar to Section A, AVR system 2 with the specifications given in Table 1 are simulated. The comparison between the MPSO algorithm and PSO (and GA) for the AVR system is performed in Table 3. The results of the simulation for this system are also illustrated in Figures 5 and 6.

Comparing the results of simulation the two AVR systems demonstrates that the MPSO algorithm is not sensitive to the parameter variations of the system.
Table 3. The simulation results and the PID's parameters obtained from different algorithms for AVR system 2.

<table>
<thead>
<tr>
<th>Method</th>
<th>Kp</th>
<th>Ki</th>
<th>Kd</th>
<th>Mp (%)</th>
<th>Ess</th>
<th>Ts</th>
<th>Tr</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>0.92</td>
<td>0.05</td>
<td>0.51</td>
<td>7.32</td>
<td>0.0</td>
<td>1.11</td>
<td>0.3</td>
</tr>
<tr>
<td>PSO</td>
<td>0.78</td>
<td>0.05</td>
<td>0.30</td>
<td>2.32</td>
<td>0.0</td>
<td>0.62</td>
<td>0.24</td>
</tr>
<tr>
<td>MPSO</td>
<td>0.96</td>
<td>0.02</td>
<td>0.73</td>
<td>0.2</td>
<td>0.0</td>
<td>0.20</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Fig. 2 Terminal voltage step response of an AVR Without PID controller

Fig. 3 The performance of AVR system (1)

Fig. 4 The transient performance of AVR system 1

Fig. 5 The performance of AVR system (2)
There are several methods such as GA and PSO algorithms for designing the parameters of the PID controllers. The aim of this paper is to find design the parameters of the PID controller using the MPSO algorithm. The suggested algorithm, is compared to GA and PSO algorithms, the MPSO algorithm has the better performance and needs less time to be performed. The results of the simulating two AVR systems validates the proposed algorithm. The results also shows that the change in the system parameters does not affect the quality of the suggested MPSO algorithm.

REFERENCES