Intelligent Control and Automation, **2013, 4, 126-137**
http://dx.doi.org/10.4236/ica.2013.42018 Published Online May 2013 (http://www.scirp.org/journal/ica)

Particle Swarm Optimization (PSO) Based Turbine Control

Ali Tarique[1], Hossam A. Gabbar[2*]
[1]Faculty of Engineering and Applied Science, University of Ontario Institute of Technology, Oshawa, Canada
[2]Faculty of Energy Systems and Nuclear Science, University of Ontario Institute of Technology, Oshawa, Canada
Email: *hossam.gaber@uoit.ca

Received October 22, 2012; revised March 14, 2013; accepted March 21, 2013

ABSTRACT

The steam turbine control system is strongly non-linear in all operating conditions. Proportional-Integral-Derivative (PID) controller that is currently used in control systems of many types of equipment is not considered highly precision for turbine speed control system. A fine tuning of the PID controller by some optimization technique is a desired objective to maintain the precise speed of the turbine in a wide range of operating conditions. This Paper evaluates the feasibility of the use of Particle Swarm Optimization (PSO) method for determining the optimal Proportional-Integral-Derivative (PID) controller parameters for steam turbine control. The turbine speed control is modelled in Simulink^{TM} with PID controller and the PSO algorithm is implemented in MATLAB to optimize the PID function. The PSO optimization technique is also compared with Genetic Algorithm (GA) and it is validated that PSO based controller is more efficient in reducing the steady-states error; settling time, rise time, and overshoot limit in speed control of the steam turbine control.

Keywords: PSO; PID Controller; GA; Optimization

1. Introduction

A steam turbine is a machine that converts thermal energy of pressurized steam to mechanical energy in the form of rotary motion. From traditional fossil fuel based power plant to nuclear power plant steam turbine is a widely used prime mover. It normally operates at 3600 rpm speed while in the nuclear industry it runs at 1800 rpm speed. The control of the turbine is indispensable, as turbines need to be run up slowly, to prevent damage as well as require precise speed control to follow the generator load variation. The control system and protection are separate although they have the same system with Controller/Governor, actuator with arm and series valves. The speed throughout the operation of the system needs to be controlled by the governor. The governor gets input signals from the generator, theturbine shaft, thesteam main output (input for turbine) pressure and thesteam outlet/extraction pressure of the turbine.

PID control was an essential element of a governor and it became the standard tool when process controls emerge in the 1940s. Today in the process control system, the usage of PID type control loops exceeded 95% [1]. PID controlsystem is designed to control the main steam-flow to the steam turbine in all operational conditions by means of the control equipment comprising of a governor, turbine throttle, admission, and or extraction control valves. Optimal performance of these components throughout the steam turbines operational life improves the life cycle of the system as well as the cost efficiency. Optimal performance in another way is even more important than before as previous turbines operate beyond their original life expectancy. After the development of digital processing technology turbine control systems are also replaced with digital controls achieving faster processing speed, lower cost and compactness. However, there exist some variables in continuous distributed control system (DCS) that experiences many unpredicted disturbance during operation. A PID control system is then combined with logic, sequential functions, selectors, and simple function blocks to build up a more efficient automation system that are used inenergy, transportation, and manufacturing sectors [2].

Steam turbine governing system is a nonlinear system whatsoever it is used in the mechanical governor or in the modern digital control governor. PID being one of the widely used controllers is in use for steam turbine governing since its development. The control strategy of

[*]Corresponding author.

the steam turbine governing system stays mainly at the control with PID, but additionally with the tuning of the PID components. One of the early methods of controller tuning is the Fuzzy logic method. But Fuzzy control doesn't necessary need the accurate mathematical model of the object, which fit to the nonlinear system. Intelligent integration fuzzy of control is adopted and simulation was done with PID controller. The simulation results enunciate that the performance of steam turbine governing system with intelligent integration fuzzy control is better than which with conventional PID control, carrying advantages such as small super-adjusting, high-speed response, short transition process time and no steady-state error and so on [3]. Optimization with a direct search method was thought a more effective in Nelder-Mead's method [4]. However, application of Nelder-Mead's method for the optimization of the automatic control systems with the fuzzy regulators does not result a precision optimum decision. A global optimization of the search space is required as stated by Sabanin *et al*. [5]. Golberg in his research depicted that optimization with Genetic Algorithm is a favoured method in complex control parameters [6]. Strategy of search of optimum solution in Genetic algorithms is based on hypothesis that the more fitness of an individual is higher with the higher probability that descendants got with its involvement [7]. Genetic algorithm is a stochastic optimization technique that is originally based on the theory of natural selection and evolutionary genetics. Lin *et al*. in their research has introduced GA technique to tune the gains of PID control for brushless DC motor [8]. GA technique has widely used to solve complex optimization problems; however recent research has identified some deficiencies in GA performance [9]. The crossover and mutation; the two main operations of GA cannot guarantee better fitness of offspring as the chromosomes in the population have similar structure and their average fitness are high toward the end of the evolutionary process [10]. Kennedy and Eberhart developed one of the modern heuristic algorithms known as PSO that has been stirred by the behavior of organisms, such as fish schooling and bird flocking [11]. Members of fish schooling or bird flocking follows an inherent rules to undertake a synchronized movement so that they don't collide. This phenomenon is used to optimize the complex solutions using PSO. PSO is characterized as easy to execute, and computationally proficient heuristic technique. This algorithm in comparison to GA has a faster speed and faster premature convergence [12]. It also has a flexible and well-balanced mechanism to improve the exploration abilities [13]. In PSO, the particles are updated according to their current positions and velocities [14]. The prospective solutions, called particles, fly through the problem space following the current optimum particles [15]. The position of a particle represents a candidate solution to the optimization problem. Each particle within the space changes its velocity and position according to rules originally motivated by behavioral models of bird flocking. Each time it identifies a better fitness value, stores it and represents it as pbest. It continues an iterative process and finally updated two best values; one is pbest and the other one gbest, the global best throughout the population.

2. Proposed Turbine Control Model

As shown in **Figure 1**, the proposed model consists of a

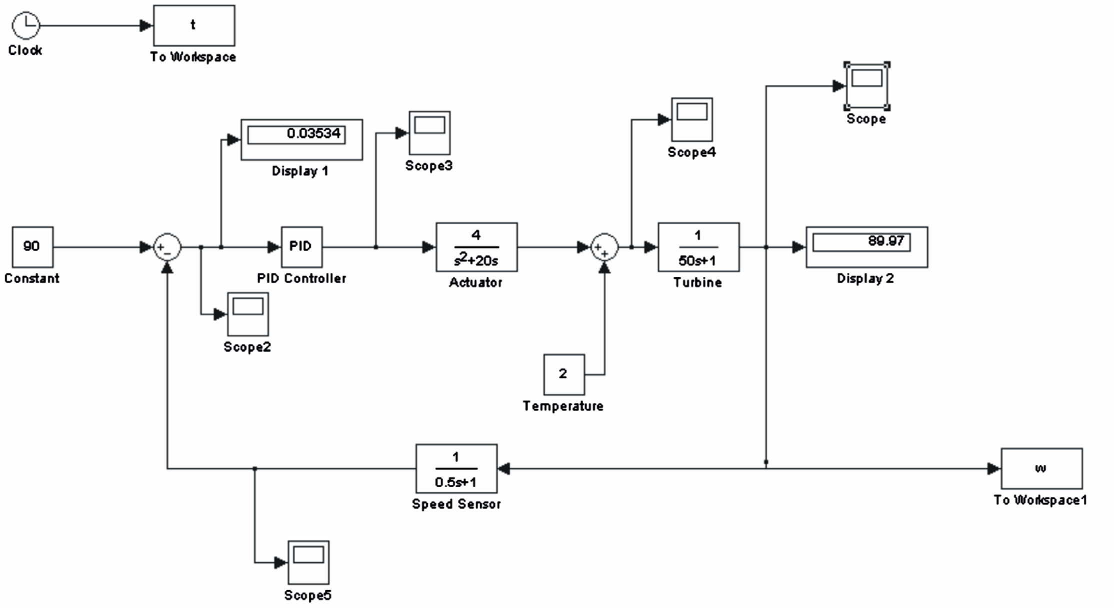

Figure 1. Model of turbine control with PID controller.

controller, actuator, the turbine and the process measuring equipments. In a steady state condition desired input and error value are the input of the controller. The controller compares the desired input and error function and develops a manipulated variable to feed to the actuator. The actuator controls the steam control valve to control the turbine speed. This PID controller is optimized with PSO and GA separately and the results are compiled in next section.

3. Controller Design

The proportional-integral-derivative (PID) controller is utilized in this design and latter tuned using PSO and GA. As shown in the flow diagram in **Figure 2**, the "error" value $e(t)$ is calculated as the difference between the desired set point $r(t)$ and measured process variable $y(t)$ at any instant of time t. The controller input $u(t)$ is determined from the Equation (1).

$$u(t)=K_p e(t)+K_I\int_0^t e(t)\,\mathrm{d}t+K_D\frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{1}$$

In this control algorithm, P corresponds the proportional term $K_p e(t)$, where K_p is the proportional gain of the controller. Similarly, I represent the integral term $K_i\int_0^t e(t)\mathrm{d}t$ and D stands for the derivative term $K_d\frac{\mathrm{d}e(t)}{\mathrm{d}t}$ of the controller. K_i and K_d are integral and derivative gain of the controller, respectively. The transfer function of the PID controller is as follows,

$$K_p+\frac{K_i}{s}+K_d s=\frac{K_p s+K_i+K_d s^2}{s} \tag{2}$$

Using these three terms, the controller by adjusting the process control input $u(t)$, minimizes the error value. The weighted sum of the three components is used to adjust the process by a control element, as in the steam turbine control a flow control valve.

4. Design Optimization Tools

In this paper, the PID controller is optimized to achieve the optimal behaviour of the plant. The optimizer is used to search for the optimal solution of the PID control gains K_p, K_i, and K_d. Intelligent optimization algorithms are appropriate choices in selecting an optimizer. The proposed methodology utilizes Particle Swarm Optimization (PSO) tools. To investigate the effectiveness of utilization of PSO in respect to Genetic Algorithm (GA), the numerous simulations of both the techniques have been performed and the results are compared.

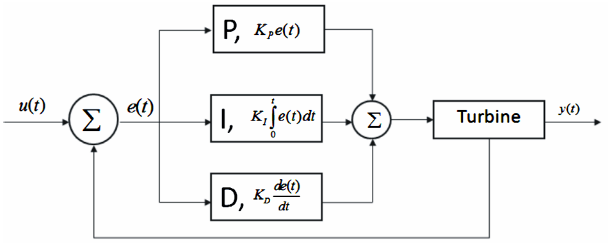

Figure 2. Block diagram of a PID control.

4.1. Genetic Algorithm

Genetic Algorithm (GA) utilizes evolution operations, such as, selection, crossover and mutation. In GA, the design variable $\boldsymbol{X}$ consists of PID control gains, as shown in the following equation.

$$\boldsymbol{X}=\begin{bmatrix}\boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_n\end{bmatrix} \tag{3}$$

and

$$\boldsymbol{x}_j=\begin{bmatrix}K_p & K_i & K_d\end{bmatrix}_j \text{ for, } j=1,2,3,\cdots,n \tag{4}$$

A population of n sets of these design variables are send forward to evaluate the objective function y by genetic algorithm (GA).

$$y_j=f\left(\boldsymbol{x}_j\right) \text{ for, } j=1,2,3,\cdots,n \tag{5}$$

After each of the n objective functions are evaluated, the resultant set $\boldsymbol{Y}$ of objective function values are checked whether the convergence criteria are met.

$$\boldsymbol{Y}=\begin{bmatrix}y_1 & y_2 & \cdots & y_n\end{bmatrix} \tag{6}$$

If not, the n objective function values are returned back to GA. Based on these values, GA generates next generation of design variable set by performing genetic operations, namely, selection, crossover, and mutation. This process continues until the convergence is reached as shown in **Figure 3**.

4.2. Particle Swarm Optimization

In PSO, each particle contains these three components K_p, K_i, and K_d and updates the components in each iteration to find the Pbest and Gbest. Finally, the program runs to converge to the optimal solution. PSO has many similarities with evolutionary computation techniques like Genetic Algorithms (GA). As shown in **Figure 4** the system is initialized with a population of random solutions and searches for optima by updating generations. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.

It is demonstrated that PSO has advantages over other methods in respect to run time, cost and better result. Another reason that PSO is attractive is that there are few parameters to adjust. One version, with slight variations, works well in a wide variety of applications. Particle

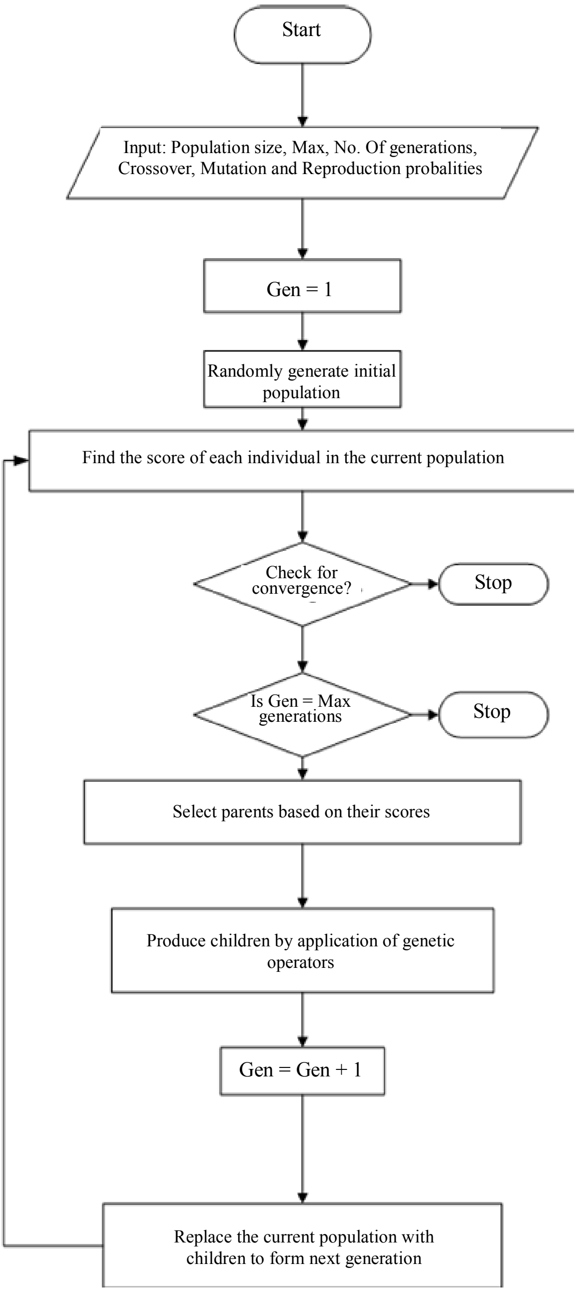
Start
Input: Population size, Max, No. Of generations, Crossover, Mutation and Reproduction probalities
Gen = 1
Randomly generate initial population
Find the score of each individual in the current population
Check for convergence?
Stop
Is Gen = Max generations
Stop
Select parents based on their scores
Produce children by application of genetic operators
Gen = Gen + 1
Replace the current population with children to form next generation

Figure 3. Genetic algorithm used for turbine control optimization.

swarm optimization has been used for approaches that can be applied across a wide range of applications, as well as for specific applications focused on a specific requirement.

$$V_i^{k+1} = wV_i^k + c_1 rand_1(\cdots)\times\left(pbest_i - s_i^k\right) + c_2 rand_2(\cdots)\times\left(gbest - s_i^k\right) \quad (7)$$

where, v_i^k : velocity of agent *i* at iteration *k*,
w: weighting function,
c_j: weighting factor,

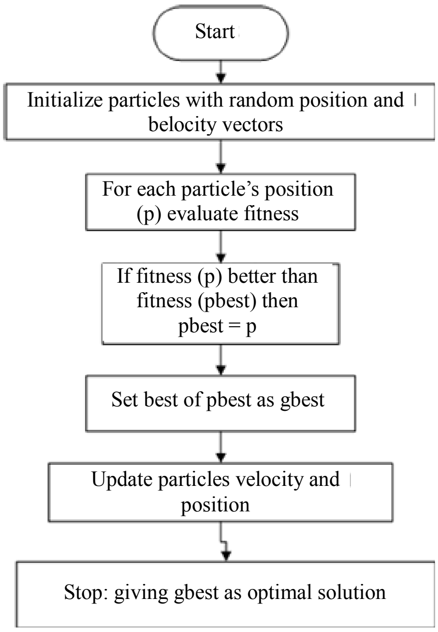
Start
Initialize particles with random position and belocity vectors
For each particle's position (p) evaluate fitness
If fitness (p) better than fitness (pbest) then pbest = p
Set best of pbest as gbest
Update particles velocity and position
Stop: giving gbest as optimal solution

Figure 4. Proposed PSO algorithm for turbine control optimization.

rand: uniformly distributed random number between 0 and 1,
s_i^k : current position of agent *i* at iteration *k*
pbest$_i$: pbest of agent *i*,
gbest: gbest of the group.

5. Proposed Algorithm

In Matlab, the particle swarm optimization (PSO) generates the set of swarm particles including the proportional gain K_p, integral gain K_i, and differential K_d of PID control algorithm. The PID control gains are then sent to the Simulink environment where the steam turbine model is located. The model is simulated with the PID gains sent to Simulink and the results are sent back to Matlab workspace. The three performance measures including settling time t_{stl}, rise time t_{rise} and overshoot d_{st} are evaluated from the simulation results. The fitness function value f is calculated from the equation as follows

$$f = \frac{t_{stl}}{t_{stl0}} + \frac{t_{rise}}{t_{rise0}} + \frac{d_{st}}{d_{st0}} \quad (8)$$

where, t_{stl0} denotes the settling time, t_{rise0} the rise time and d_{st0} the overshoot of step response of the nominal steam turbine simulation. The optimization terminates if the convergence criteria is satisfied otherwise the fitness function values are sent back to SPO and sets of swarm particle are created for next loop of iteration.

In the case of genetic algorithm (GA), similar procedure of SPO, as mentioned above, is continued with same set of design variables of PID control gains.

6. Simulation Results

In this research, a simple PID controller is designed and simulated using the methodology shown in **Figure 2**. Using particle swarm optimization (PSO) and genetic algorithm (GA), the control system is optimized separately to investigate their individually performances. The objective of the multi-criteria optimization problem is to minimize settling time t_{stl}, rise time t_{rise} and overshoot d_{st}.Both optimization algorithms search for the optimal values of PID control gain, namely, K_p, K_i, and K_d. In the baseline case, the PID control gains K_p, K_i, and K_d take the value of 0.5, 0.0001, and 0.05, respectively. In the baseline case, settling time t_{stl}, rise time t_{rise} and overshoot d_{st} take the value of 382.77, 27.20, and 50.813, respectively. The result sets including rise time, settling time, overshoot/undershoot, peak time and with the other optimization parameters for PSO and GA are listed in **Tables 1** and **2**, respectively. In order to make both optimizations comparable, number of design variables in GA takes the value of 20 and so is that of swarm in PSO. The optimization is continued until 35 iterations in both cases.

Figure 5 depicts the simulation result of response of baseline case of the PID controller for step input. **Figure 6** represents one of the simulation results of the response after optimizing the PID controller with PSO. The settling time is improved by 4.37% by decreasing its value to 366.06 sec from the corresponding baseline value of 382.77 sec. To avoid the computational cost and optimization complexity, a narrow band of variation for the design variable set is chosen. Also the PID controller designed is a basic model of the conventional form.

Table 1. Result set of PSO.

PSO Results	*Values*
Rise Time	2.5495e + 001
Settling Time	3.6431e + 002
Minimum Settling Time	6.5225e + 001
Maximum Settling Time	1.3735e + 002
Overshoot	5.2664e + 001
Undershoot	0
Peak	1.3735e + 002
PeakTime	6.7791e + 001
Time for optimization	891.431
Number of iterations	35
Objective function value	2.9255e + 000
Number of swarm particles	20
K_p	5.5753e − 001
K_i	4.0198e − 005
K_d	6.7017e − 002

Table 2. Results set of GA.

GA Results	*Values*
Rise Time	2.5676e + 001
Settling Time	3.6717e + 002
Minimum Settling Time	6.5296e + 001
Maximum Settling Time	1.3745e + 002
Overshoot	5.2701e + 001
Undershoot	0
Peak	1.3745e + 002
Peak Time	6.8290e + 001
Time for Optimization	9.5064e + 002
Number of Iterations	35
Objective Function Value	2.9404e + 000
Number of Design Variable	20
K_p	5.4999e − 001
K_i	9.1943e − 005
K_d	5.4930e − 002

Comparison of PSO and GA results are graphically represented in **Figures 7**-**10**. The optimization process has been carried out with different number of swarm/ population to compare both optimization algorithms. **Figure 7** depicts that, for same number of swarm/population, the elapsed time to run the optimization algorithm is less in PSO in comparison to GA. Moreover, the settling and rise time of PSO are much lower than GA as shown in **Figures 8** and **9**, respectively. The results confirm that optimization of PID using PSO more efficient compared to GA.

Figure 10 represents the response of over overshoot versus number of warm and population size for PSO and GA, respectively. A close observation of the figure reveals that the overshoot of the step response in PSO, however, is higher with lower number of swarm. This does not affect the advantage of PSO over GA, since the practical implementation of PSO is with much higher number of swarms.

7. Conclusion

In this paper, a particle swarm optimization (PSO) and a genetic algorithm (GA) are separately implemented to optimize the gains of a proportional-integral-derivative (PID) algorithm to control a steam turbine. The results of both algorithms have been investigated and compared. The research has been performed with a simple PID controller to investigate the effectiveness of the algorithm

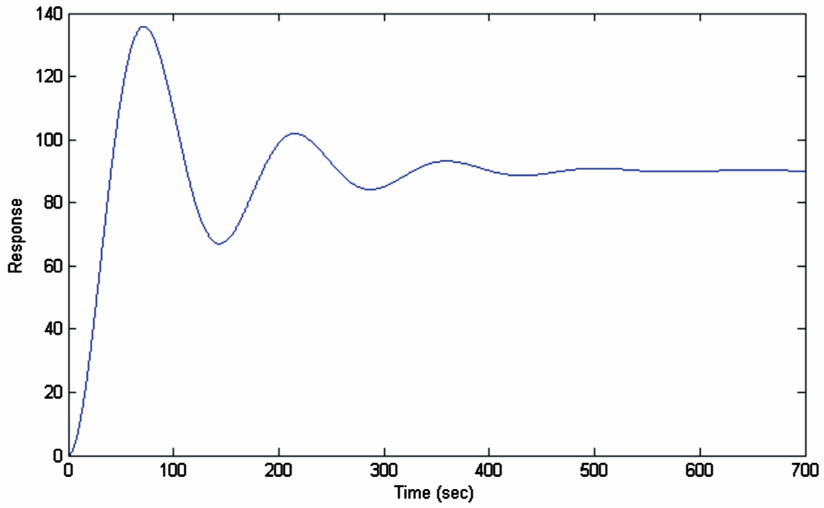

Figure 5. PID simulation result of time history of response without optimization.

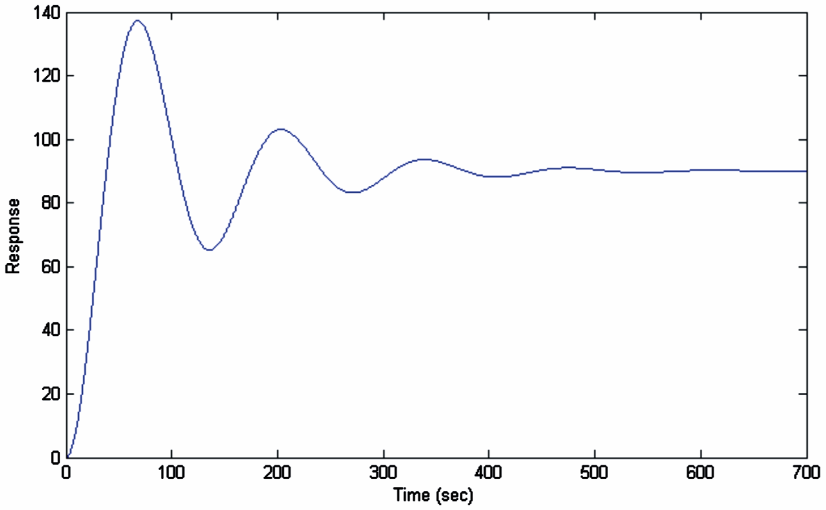

Figure 6. Simulation result of time history of optimal response (using PSO with, number of swarm = 20).

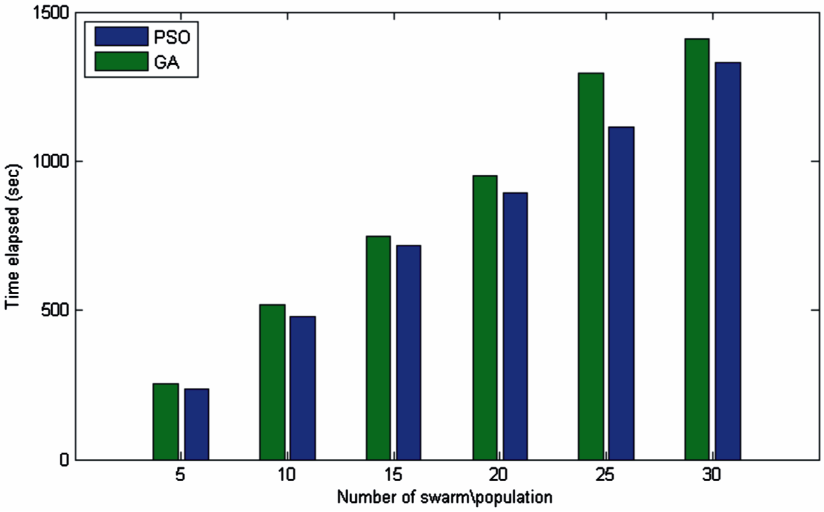

Figure 7. Comparison of PSO and GA for turbine control (time elapsed).

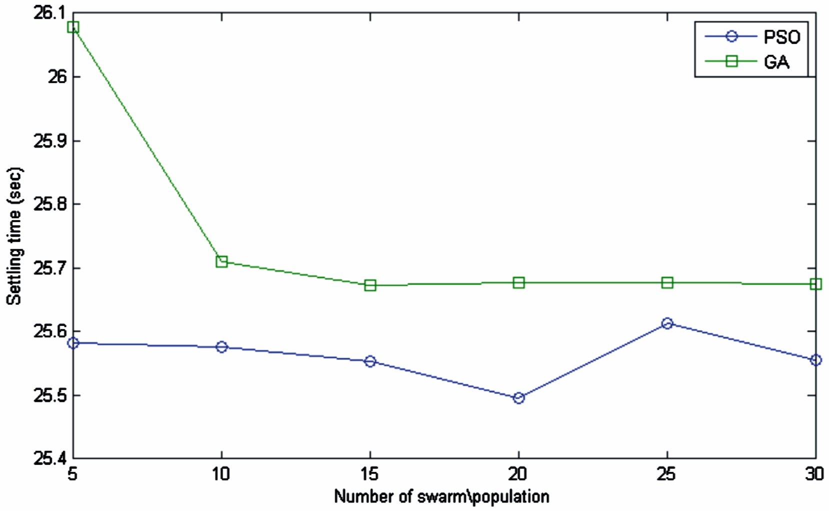

Figure 8. Comparison of PSO and GA for turbine control (setting time).

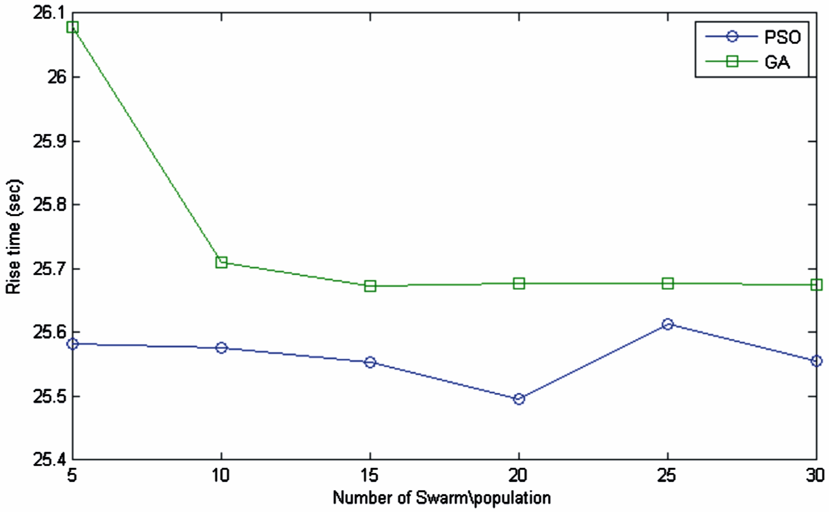

Figure 9. Comparison of PSO and GA for turbine control (rise time).

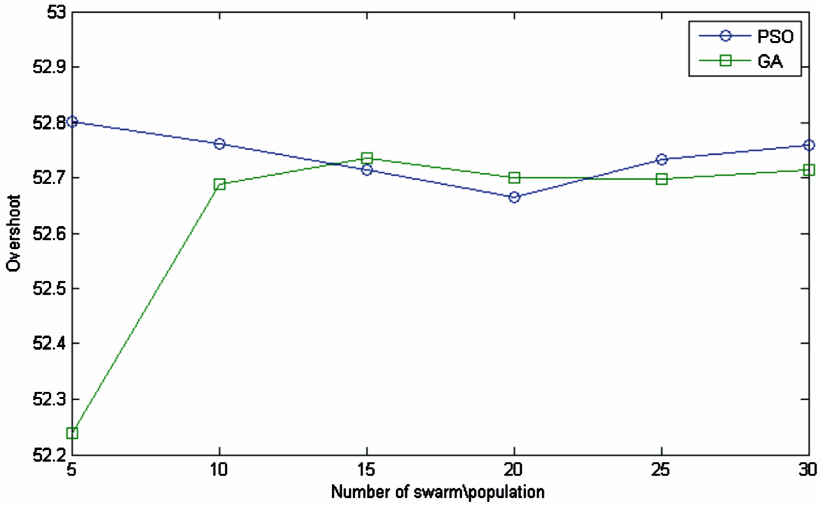

Figure 10. Comparison of PSO and GA for turbine control (overshoot).

while avoiding complexity and computational cost of the work. The goal of this paper was to evaluate the usefulness of PSO algorithm coupled with PID controller for steam turbine control and compare its function with a GA based PID controller. The numerical results validate the effectiveness of using PSO algorithm coupled with PID controller for the steam turbine control. The PSO-based PID controller was able to improve the optimization objective function by minimizing its value 0.51% lower than that of GA with spending 6.23% less time than GA. The PSO-based PID controller is highly recommended over GA-based PID controller. Future work may be performed with a more precise and practical PID controller for steam turbine control to optimize the output for better performance.

REFERENCES

[1] K. J. Astrom and T. Hagglund, "PID Controllers, Theory, Design and Tuning," 2nd Edition, Instrument Society of America, 1995.

[2] O. Aydogdu and M. Korkmaz, "A Simple Approach to Design of Variable Parameter Nonlinear PID Controller," *International Conference on Advancement in Information Technology*, Singapore, 2011.

[3] H. B. Liu, "Intelligent Coordinated Control of Power-Plant Main Steam Pressure and Power Output," *Journal of Systems Engineering and Electronics*, Vol. 15, No. 3, 2004, pp. 350-358.

[4] J. A. Nelder and R. A. Mead, "A Simplex Method for Function Minimization," *Computer Journal*, Vol. 7, No. 4, 1964, pp. 308-313.

[5] V. R. Sabanin, N. I. Smirnov and A. I. Repin, "Optimization of Settings of Regulative Devices in ASR," *Collection of Labours of Conference of Control*, MEI, 2003, pp. 144-148.

[6] D. E. Goldberg, "Genetic Algorithms in Search Optimizations and Machine Learning," Wesly, Addison, 1989.

[7] G. K. Voronovskiy, K. V. Makhotilo, S. N. Petrashev and S. A. Sergeev, "Genetic Algorithms, Artificial Neuron Networks and Problems of Virtual Reality," Basis, Kharkov, 1997.

[8] C. L. Lin, H. Y. Jan and N. C. Shieh, "GA-Based Multi Objective PID Control for a Linear Brushless DC Motor," *IEEE/ASME Transactions on Mechatronics*, Vol. 8, No. 1, 2003, pp. 56-65.

[9] M. Nasri, H. Nezamabadi-Pour and M. Maghfoori, "A PSO-Based Optimum Design of PID Controller for a Linear Brushless DC Motor," *World Academy of Science, Engineering and Technology*, Vol. 20, 2007, pp. 211-215.

[10] D. B. Fogel, "Evolutionary Computation toward a New Philosophy of Machine Intelligence," *IEEE*, New York, 1995.

[11] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *Proceedings of IEEE of International Conference on Neural Networks*, Vol. 4, Perth, 1995, pp. 1942-1948. doi:10.1109/ICNN.1995.488968

[12] A. Banks, J. Vincent and Ch. Anyakoha, (2007) "A Review of Particle Swarm Optimization. Part I: Background and Development," *Natural Computing*, Vol. 6, No. 4, pp. 467-484. doi:10.1007/s11047-007-9049-5

[13] M. A. Abido, "Optimal Design of Power-System Stabilizers Using Particle Swarm Optimization," *IEEE Transactions on Energy Conversion*, Vol. 17, No. 3, 2002, pp. 406-413. doi:10.1109/TEC.2002.801992

[14] J. Yang and A. Bouzerdoum, "A Particle Swarm Optimization Algorithm Based on Orthogonal Design," *IEEE Congress on Evolutionary Computation*, 18-23 July 2010.

[15] PSO Tutorial, 2012.
http://www.swarmintelligence.org/tutorials.php

Appendix

Different Optimization Results:

Without Optimization:

Output value:

K_p	0.5
K_i	0.0001
K_d	0.05

Results:

RiseTime:	2.7200e + 001
SettlingTime	3.8277e + 002
SettlingMin	6.7015e + 001
SettlingMax	1.3587e + 002
Overshoot	5.0813e + 001
Undershoot	**0**
Peak	1.3587e + 002
PeakTime	7.1789e + 001

With PSO Optimization:

***PSO with number of swarm particles* = 5**

Time for optimization: 237.5975
Number of iterations = 35

RiseTime:	2.5582e + 001
SettlingTime	3.6606e + 002
SettlingMin	6.5199e + 001
SettlingMax	1.3756e + 002
Overshoot	5.2802e + 001
Undershoot	**0**
Peak	1.3756e + 002
PeakTime	6.8127e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.5351e − 001
K_i	9.9860e − 005
K_d	6.0240e − 002

PSO with number of swarm particles = 10
Time for optimization: 479.0345
Number of iterations = 35
Objective function value = 2.9345e + 000

RiseTime:	2.5577e + 001
SettlingTime	3.6586e + 002
SettlingMin	6.5221e + 001
SettlingMax	1.3751e + 002
Overshoot	5.2762e + 001
Undershoot	**0**
Peak	1.3751e + 002
PeakTime	6.8137e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Outputvalue:

K_p	5.5387e − 001
K_i	9.1432e − 005
K_d	6.2754e − 002

PSO with number of swarm particles = 15
Time for optimization: 718.9101
Number of iterations = 35
Objective function value = 2.9313e + 000

RiseTime:	2.5553e + 001
SettlingTime	3.6534e + 002
SettlingMin	6.5231e + 001
SettlingMax	1.3744e + 002
Overshoot	5.2713e + 001
Undershoot	**0**
Peak	1.3744e + 002
PeakTime	6.7923e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.5502e − 001
K_i	7.0110e − 005
K_d	6.3870e − 002

PSO with number of swarm particles = 20
Time for optimization: 891.431
Number of iterations = 35
Objective function value = 2.9255e + 000

RiseTime:	2.5495e + 001
SettlingTime	3.6431e + 002
SettlingMin	6.5225e + 001
SettlingMax	1.3735e + 002
Overshoot	5.2664e + 001
Undershoot	**0**
Peak	1.3735e + 002
PeakTime	6.7791e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.5753e − 001
K_i	4.0198e − 005
K_d	6.7017e − 002

PSO with number of swarm particles = 25
Time for optimization: 1113.3763
Number of iterations = 35
Objective function value = 2.9364e + 000

RiseTime:	2.5613e + 001
SettlingTime	3.6630e + 002
SettlingMin	6.5247e + 001
SettlingMax	1.3747e + 002
Overshoot	5.2732e + 001
Undershoot	**0**
Peak	1.3747e + 002
PeakTime	6.8137e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.5247e − 001
K_i	8.5223e − 005
K_d	5.8013e − 002

PSO with number of swarm particles = 30
Time for optimization: 1330.8598
Number of iterations = 35
Objective function value = 2.9328e + 000

RiseTime:	2.5556e + 001
SettlingTime	3.6553e + 002
SettlingMin	6.5219e + 001
SettlingMax	1.3751e + 002
Overshoot	5.2758e + 001
Undershoot	**0**
Peak	1.3751e + 002
PeakTime	6.7932e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.5478e − 001
K_i	8.9964e − 005
K_d	6.6389e − 002

Genetic Algorithm (*GA*)
A with number of design variable = 5
Time for optimization: 2.5569e + 002
Number of iterations = 35
Objective function value = 2.9580e + 000

RiseTime:	2.6078e + 001
SettlingTime	3.7175e + 002
SettlingMin	6.5763e + 001
SettlingMax	1.3701e + 002
Overshoot	5.2238e + 001
Undershoot	**0**
Peak	1.3701e + 002
PeakTime	6.9249e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.3571e − 001
K_i	9.1649e − 005
K_d	5.1280e − 002

GA with number of design variable = 10
Time for optimization: 5.1639e + 002
Number of iterations = 35
Objective function value = 2.9426e + 000

RiseTime:	2.5710e + 001
SettlingTime	3.6764e + 002
SettlingMin	6.5325e + 001
SettlingMax	1.3744e + 002
Overshoot	5.2238e + 001
Undershoot	**0**
Peak	1.3744e + 002
PeakTime	6.8437e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.4869e − 001
K_i	9.9708e − 005
K_d	5.4953e − 002

GA with number of design variable = 15
Time for optimization: 7.4615e + 002
Number of iterations = 35
Objective function value = 2.9411e + 000

RiseTime:	2.5673e + 001
SettlingTime	3.6724e + 002
SettlingMin	6.5273e + 001
SettlingMax	1.3749e + 002
Overshoot	5.2736e + 001
Undershoot	**0**
Peak	1.3749e + 002
PeakTime	6.8264e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.4996e − 001
K_i	9.7845e − 005
K_d	5.3235e − 002

GA with number of design variable = 20
Time for optimization: 9.5064e + 002
Number of iterations = 35
Objective function value = 2.9404e + 000

RiseTime:	2.5676e + 001
SettlingTime	3.6717e + 002
SettlingMin	6.5296e + 001
SettlingMax	1.3745e + 002
Overshoot	5.2701e + 001
Undershoot	**0**
Peak	1.3745e + 002
PeakTime	6.8290e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.4999e − 001
K_i	9.1943e − 005
K_d	5.4930e − 002

GA with number of design variable = 25
Time for optimization: 1.2953e + 003
Number of iterations = 35
Objective function value = 2.9403e + 000

RiseTime:	2.5676e + 001
SettlingTime	3.6716e + 002
SettlingMin	6.5298e + 001
SettlingMax	1.3744e + 002
Overshoot	5.2697e + 001
Undershoot	**0**
Peak	1.3744e + 002
PeakTime	6.8291e + 0011

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.4999e − 001
K_i	9.0923e − 005
K_d	5.4967e − 002

GA with number of design variable = 30
Time for optimization: 1.4081e + 003
Number of iterations = 35
Objective function value = 2.9407e + 000

RiseTime:	2.5675e + 001
SettlingTime	3.6720e + 002
SettlingMin	6.5289e + 001
SettlingMax	1.3746e + 002
Overshoot	5.2715e + 001
Undershoot	**0**
Peak	1.3746e + 002
PeakTime	6.8290e + 001

Range on Dimension K_p: [0.450000, 0.550000]
Range on Dimension K_i: [0.000090, 0.000110]
Range on Dimension K_d: [0.045000, 0.055000]

Optimized Output Value:

K_p	5.4999e − 001
K_i	9.5538e − 005
K_d	5.4950e − 002