Advanced Optimization by PMSM of PSO and Neural Network

DONG HWA KIM
Department of Electronic and Control of Eng.,
Hanbat National University
16-1 Duckmyong Yuseong Daejeon, S.Korea
S.KOREA
koreahucare@gmail.com http://www.hucare.org

JIN ILL PARK
President, Research Center
Vision System Com.

Abstract: - This paper suggests novel optimal approach by progressive mapping search method (PMSM) of neural network aided particle swarm optimization (PSO) that can obtain global optimal solution easily and speed searching time up by PMSM. The PMSM by NN and PSO has an important role as navigation when PSO is going to search all areas to have an optimal solution, it can help to increase searching capability of PSO. That is, the PMSM by NN and PSO is also trained to capture the PSO-searched data in system. To prove the method, we use four test function: De Jong's function-1, Rosenbrock's valley (De Jong's function-2), Himmelblau function, Rastrigin's function-6. The PMSM method suggested in this paper is faster than the traditional PSO method in four test function. We also apply this optimal approach into AVR (Automatic Voltage Regulator) system in thermal power plant. The response is quite faster and more stable.

Key-Words: - PSO, Neural Network, Hybrid system, Optimization, Learning system, Artificial intelligence.

1 Introduction

Over the past several years, artificial intelligence such as fuzzy logic, neural networks, genetic algorithm have been giving a significant advance on intelligence and optimal solution tool [1, 2, 13, 14, 23]. They have also been a considerable interest in the past few years in exploring the applications of fuzzy and neural network systems, which combine the capability of fuzzy reasoning to handle uncertain information and the capability of artificial networks to learn from processes [24-26], to deal with nonlinearities and uncertainties of control systems [8-10].

Recently, many have been interesting in bio based intelligence such as IN (Immune network), PSO (Particle Swarm Optimization, GA (Genetic Algorithm), BF (Bacterial Foraging), and so on to have an optimization solution for their science and engineering areas [3, 4, 12, 26, 27] because of their robustness and flexibility against a dynamically changing system or complex system.

The PSO has some similarities with genetic algorithm (GA) in computing processing. That is, the PSO (Particle Swarm Optimization) is an algorithm for finding optimal regions of complex search spaces through interaction of individuals in a population of particles. It is usually using a parameter of two iterative equations, one for the positions and the other one for the velocities of the particles, with several parameters. It gives more "freedom" to the system but it is also then quite difficult to find the best parameters values to each case, although some researchers have been applying in engineering.

This kind of algorithm is still largely experimentally studied. There is still, however, no sure way to choose a priori the best parameters, the parameters of the velocity and position coefficients are also randomly tuned or selected at each time step when we apply to engineering area. There are different versions of particle swarm optimization algorithms, but we should study from an engineering and optimal approach point of view, to two question. That is, what kind of information each particle has access to, and how we speed it up for solution. To illustrate this, we study here two approaches, the hybrid system based on PSO. When we use this PSO, we can have an additional advantage comparatively simple in operation and it is easier to understand dynamic equation compared to other computational techniques [8, 9]. Especially, PSO shows a faster speed in computing processing as it

ISBN: 978-960-474-314-8

uses a smaller number to tune in parameters [7, 9]. Therefore, some researchers in industrial areas and sciences have been interesting to have solution with more simple and efficient algorithm. PSO can also be very useful to acquire a global optimization [6].

As the Proportional-Integral-Derivative (PID) controller has been widely used owing to its simplicity and robustness in power plant, its tuning technology is important as engineer. However, using only the P, I, D parameters, it is often very difficult to control a plant with complex dynamics for power plants having a high nonlinear characteristics. To get over these problem, recently, there has been a growing interest in the usage of intelligent approaches such as fuzzy inference systems, neural network, evolutionary algorithms, and their hybrid approaches [1, 18, 19]. In this paper, the NN is satisfactorily trained to change the performance of the PSO accurately depending on a situation to provide it. The variables of PSO can be suitably chosen depending on the application for which the system is intended to be developed. Section 2 in this paper describes the PSO algorithm and the role of NN used in this paper. Section 3 shows the entire scheme employing the NN trained PSO for optimal search methods. That is, how it works the PMSM system suggested in this session. Section 4 shows the experimental studies carried out using four test functions. Section 5 presents the conclusion.

2 Characteristics of PSO and NN in Hybrid System

2.1 Characteristics of PSO

The PSO conducts searches using a population of particles which correspond to individuals in GA [4, 5]. A population of particles is randomly generated initially. Each particle represents a potential solution and has a position represented by a position vector.

A swarm of particles moves through the problem space, with the moving velocity of each particle represented by a velocity vector. At each time step, a function representing a quality measure is calculated by using the results of crossover and mutation as input. Each particle keeps track of its own best position, which is associated with the best fitness it has achieved so far in a vector. Furthermore, the best position among all the particles obtained so far in the population is kept track as output. In addition to this global version, another local version of PSO keeps track of the best position among all the topological neighbors of a particle. At each time step, by using the individual best position, and global best position, a new velocity for particle is updated by equation (1).

In equation (1), positive constants and are uniformly distributed random numbers in [0, 1]. The term is limited to the range. If the velocity violates this limit, it is set at its proper limit. Changing velocity this way enables the particle to search around its individual best position, and global best position. The computation of PSO is easy and adds only a slight computation load when it is incorporated into GA. Furthermore, the flexibility of PSO to control the balance between local and global exploration of the problem space helps to overcome premature convergence of elite strategy in GA, and also enhances searching ability.

2.2 Dynamic Equation of PSO

The characteristic for hybrid system of PSO and GA have been studied [1, 14-20]. And also, many researchers have been studying hybrid system for real intelligent system [1, 16-20]. A number of approaches have been also proposed to implement mixed control structures that combine a PID controller with intelligent approaches [1, 20].

This paper focuses on novel hybrid system using NN and PSO. Position and speed vector of PSO is given by

$$\begin{aligned}
v_{f,g}^{(t+1)} &= w \cdot v_j^{(t)} + c_1^* rand()^* \left(pbest_{j,g} - k_{j,g}^{(t)}\right) \\
&\quad + c_2^* Rand()^* \left(gbest_g - k_{j,g}^{(t)}\right) \\
j &= 1,2,\ldots,n. \\
g &= 1,2,\ldots,m. \\
k_{j,g}^{(t+1)} &= k_{j,g}^{(t)} + v_{j,g}^{(t+1)}, \quad k_g^{\min} \le k_{j,g}^{(t+1)} \le k_g^{\max}
\end{aligned} \tag{1}$$

n: The number of agent in each group
m: The number of member in each group
t: Number of reproduction step
$v_{j,g}^{t}$: The speed vector of agent j in reproduction step of t^{th}. $v_g^{min} \le v_{j,g}^{t} \le v_g^{max}$
$k_{j,g}^{t}$: The position vector of agent j in reproduction step of t^{th}.
w: Weighting factor
$c1,\ c2$: Acceleration constant
$rand\ (\,), Rand\ (\,)$: Random value between 0 and 1
$pbest(j,g)$: Optimal position vector of agent j
$gbest$: Optimal position vector of group

The value of position vector and speed vector is determined by acceleration constant $c1, c2$. If these values are large, each agent moves to target position with high speed and abruptly variation. If vice versa,

ISBN: 978-960-474-314-8

agents wander about target place. As weighting factor w is for the searching balance of agent, the value for optimal searching is given by

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \quad , \qquad (2)$$

where $v_{\max}$: Max mum value of w (0.9),

$w_{\min}$: Minimum value of w (0.4),

$iter_{\max}$: The number of iterative number,

$iter$: The number of iterative at present.

The speed vector is limited by $v_g^{min} \le v_{j,g}^t \le v_g^{max}$. In this paper, the value of speed vector for each agent is limited with 1/2 to avoid abrupt variation of position vector. Computing algorithm for each step in hybrid system by PSO and NN is as shown in Fig.3.

2.3 The Characteristics of NN as Learning Tool in This paper

In this paper the adaptive linear learning method (Adaline), suggested by Widrow and Hoff [1962] is

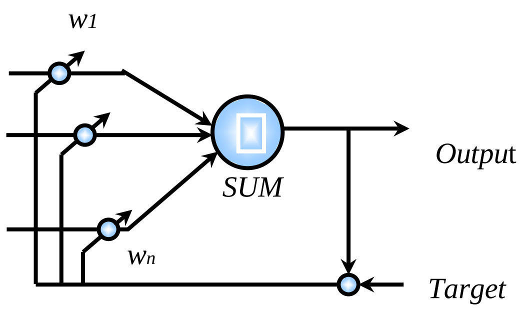

Fig.1. Adaptive linear learning.

use and it has the structure of learning diagram as shown in Fig.1.

This neural network is a supervised learning method and the input patterns $x_i = [x_1, x_2, \ldots, x_n]^{\mathrm{T}}$ are linearly independent. That is, the input-output relationship is linear in an Adaline. When the desired vector is $d_i = [d_1,\ d_2, \ldots, d_n]^T$ and the network output is y_i, the output is a weight linear combination of the input vectors plus a constant bias value as the following Equation:

$$y_i(t) = \left(\sum_{j=1}^{n} w_i x_i(t) + b_o \right) \qquad (3)$$

To find the optimal weights to get the desired output from Equation (3), a cost function P(w) is defined to measure the system's performance error by

$$P(w) = \frac{1}{2}\left(\sum_{j=1}^{n} (d_i - y_i)^2 \right) = \frac{1}{2}\left(\sum_{j=1}^{n} (d_i - w^T x_i)^2 \right). \qquad (4)$$

If P(w) is smaller, the better w will be, and P(w) is normally positive. However, it approaches zero when output y_i approaches target vector d_i for i=1, 2,…,n. Usually, the mean squared error method is used to minimize P(w). Eventually, running rule for

$$\Delta w_i = \alpha\left(d_i - w^T x_i\right). \qquad (5)$$

3 PMSM for Optimal Solution by NN Aided PSO

This paper suggests novel optimal solution by NN aided PSO learning algorithm. The NN has a role of function to increase memory and training function to avoid overlap search and data in the past region and data when the PSO is going to search.

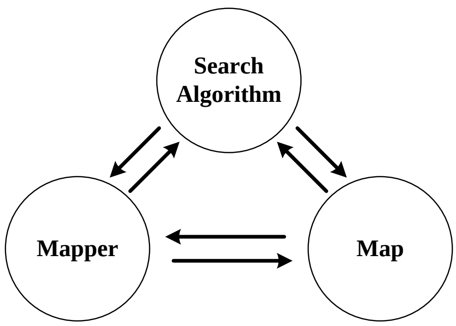

Fig.2. Learning structure of Progressive mapping search method (LSPM).

3.1 Navigation of a Mapping for Optimal Solution

The architecture of mapping has like a navigation of car to search by map. When PSO search optimal position and parameters, PSO should search all area without using map where to go. However, in this paper, PSO can search where and how to go to optimal parameter or region by using neural network. Therefore, it can be faster and global optimal solution than the traditional approaches.

3.2 Apply the PMSM for particle swarm optimization

To start the algorithm in Eq. (6), we need to select the initial values of θ_o and P_o. One way to avoid determining these initial values is to collect the first *n* data points and solve θ_n and $P_n$directly from

$$\begin{cases} \mathbf{P}_n = (\mathbf{A}_n^T \mathbf{A}_n)^{-1} \\ \boldsymbol{\theta}_n = \mathbf{P}_n \mathbf{A}_n^T \mathbf{y}_n \end{cases} \tag{6}$$

Begin;
  Generate random population of *N* solutions (particles);
  For each individual *i* : calculate fitness (*i*);
  Initialize the value of the weight factor, *w*;
  Generate the initial map based on all particles according to Eq.(6);
    For each particle;
      Set *pBest* as the best position of particle *i*;
      If fitness (*i*) is better than *pBest*;
      pBest(*i*) = fitness (*i*);
    End;
    Set *gBest* as the best fitness of all particles;
    For each particle;
      For each candidate particle *m*;
        Calculate candidate particle velocity;
        Update candidate particle position;
      End;
      Choose the best candidate based on the map according to Eq. (8);
    End;
    Update the value of the weight factor, *w*;
    Update the map based on new all particles according to Eq. (7);
  Check if termination = true;
End;

Fig.3. Computing process of PSO with PMSM algorithm.

where $[A_n : Y_n]$ is the data matrix composed of the first *n* data pairs. We can then start iterating the algorithm from the $(n+1)$ th data point.

In summary, the recursive least-squares estimator for the problem of $A\theta = Y$. Where the *k*th $(1 \le k \le m)$ row of $[A_n : Y_n]$, denoted by $[a_k^T : y_k]$, is sequentially obtained. It can be calculated as follows:

$$\begin{cases} \mathbf{P}_{n+1} = \mathbf{P}_n - \dfrac{\mathbf{P}_n \mathbf{a}_{n+1} \mathbf{a}_{n+1}^T \mathbf{P}_n}{1 + \mathbf{a}_{n+1}^T \mathbf{P}_n \mathbf{a}_{n+1}} \\ \boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n + \mathbf{P}_{n+1} \mathbf{a}_{n+1} (y_{n+1} - \mathbf{a}_{n+1}^T \boldsymbol{\theta}_n) \end{cases} \tag{7}$$

where *k* ranges from o to *m*-1, the estimator using all *m* data pairs.

$$\begin{cases} \min |\hat{\mathbf{y}}|, & \text{If global minimum} \\ \max |\hat{\mathbf{y}}|, & \text{If global maximum} \end{cases} \tag{8}$$

Fig.3 shows computing process by this equation with Matlab software

4. Experimental Verification by Test Function

4.1 Test Function for Proof

We use four test function to prove optimal algorithm suggested by PMSM of NN aided PSO.

4.1.1 De Jong's function 1

The simplest test function is De Jong's function 1. It is also known as sphere model. It is continuous, convex and unimodal.

$$f_1(x) = \sum_{i=1}^{n} x_i^2, \quad -5.12 \le x_i \le 5.12 \tag{9}$$

The global minimum:

$$f_1(x) = 0, \quad x_i = 0,\ i = 1, 2, \ldots n.$$

4.1.2 Rosenbrock's valley (De Jong's function 2)

Rosenbrock's valley is a classic optimization problem, also known as Banana function. The global optimum is inside a long, narrow, parabolic shaped flat valley. To find the valley is trivial, however convergence to the global optimum is difficult and hence this problem has been repeatedly used in assess the performance of optimization algorithms.

$$f_2(x)=\sum_{i=1}^{n-1}100\cdot\left(x_{i+1}-x_i{}^2\right)^2+\left(1-x_i\right)^2,\ -2.048\le x_i\le 2.048 \tag{10}$$

The global minimum:

$$f_{1b}(x)=0,\quad x_i=1,\ i=1,2,\dots n.$$

4.1.3 Himmelblau function

In mathematical optimization, the Himmelblau's function is a multi-modal function, used to test the performance of optimization algorithms.

$$f_H(x)=\left(x_1^2+x_2-11\right)^2+\left(x_1+x_2^2-7\right)^2,\qquad -5\le x\le 5 \tag{11}$$

The global minimum:

$$f_H(3{,}2)=0,\ f_H(-3.78,-3.28)=0.0054,$$
$$f_H(-2.81,3.13)=0.0085,\ f_H(3.58,-1.85)=0.0011$$

4.1.4 Rastrigin's function 6

Rastrigin's function is based on function 1 with the addition of cosine modulation to produce many local minima. Thus, the test function is highly multimodal. However, the location of the minima is regularly distributed.

$$f_6(x)=10\cdot n+\sum_{i=1}^{n}\left(x_i^2-10\cdot\cos(2\cdot\pi\cdot x_i)\right),\ -5.12\le x_i\le 5.12 \tag{12}$$

The global minimum:

$$f_{1b}(x)=0,\quad x_i=0,\ i=1,2,\dots n.$$

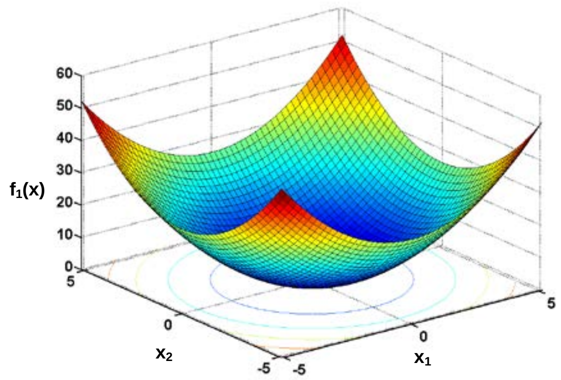

Fig.4(a) 3D-maping at initial point of De Jong's function 1

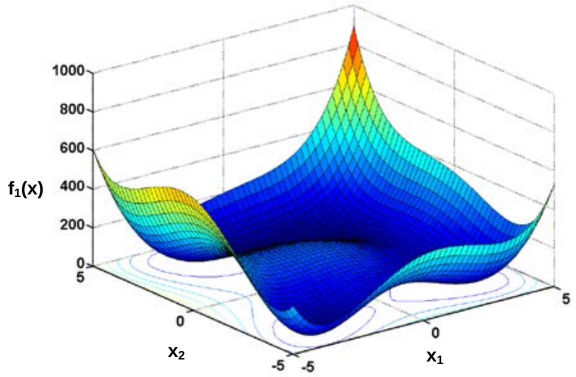

Fig.4(b) 3D-maping at initial point of Rosenbrock's valley (De Jong's function 2).

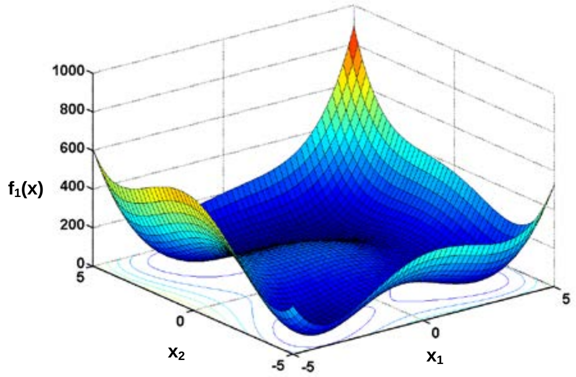

Fig.4(c) 3D-maping at initial point of Himmelblau's function.

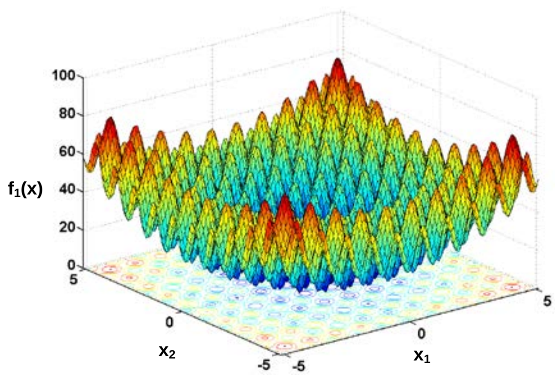

Fig.4(d). 3D-maping at initial point of Rastrigin's function 6

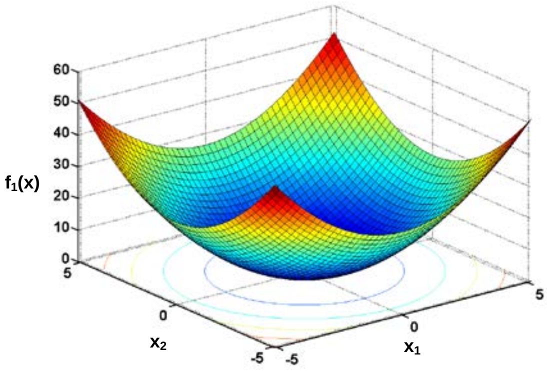

Fig.5(a). Experimental results after 100 iterations: De Jong's function 1.

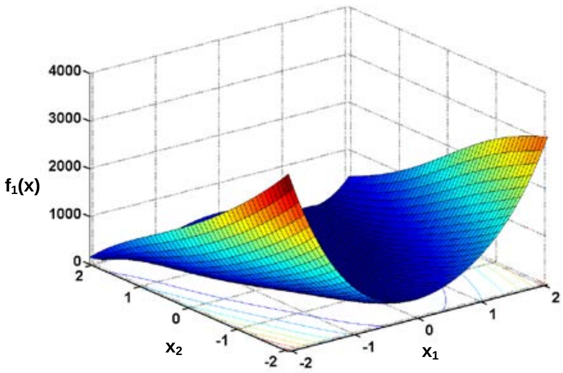

Fig.5(b). Experimental results after 100: Rosenbrock's valley.

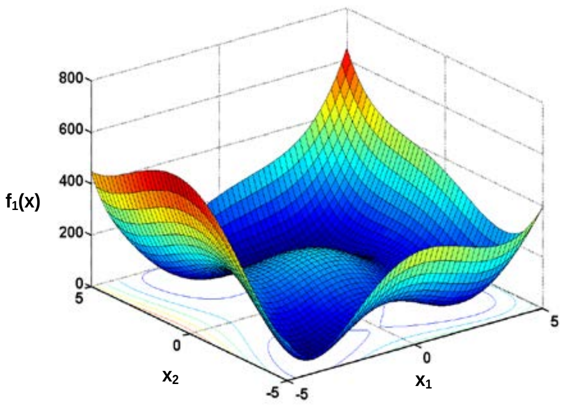

Fig.5(c). Experimental results after 100: Himmelblau's function.

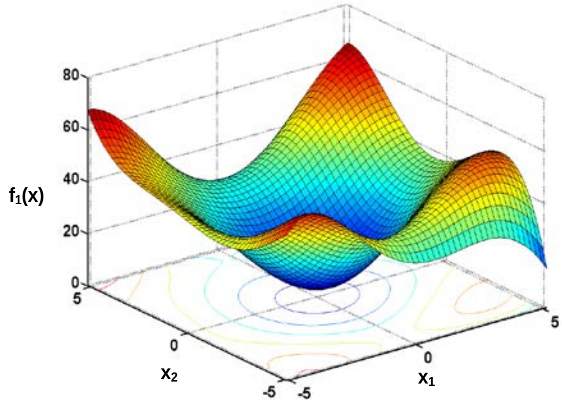

Fig.5(d). Experimental results after 100: Rastrigin's function 6.

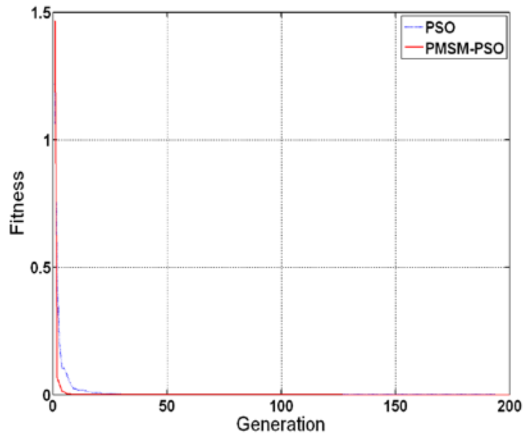

Fig.6(a). Optimal speed between PSO and PMSM-PSO: De Jong's function 1.

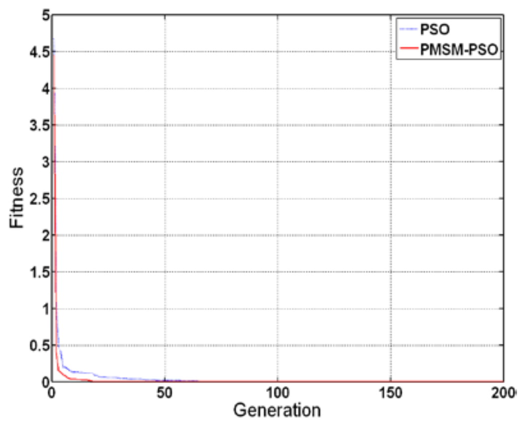

Fig.6(b). Optimal speed between PSO and PMSM-PSO: Rosenbrock's valley.

ISBN: 978-960-474-314-8

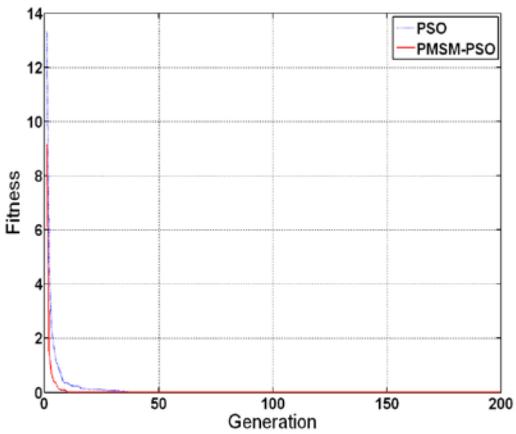

Fig.6(c). Optimal speed between PSO and PMSM-PSO: Himmelblau's function.

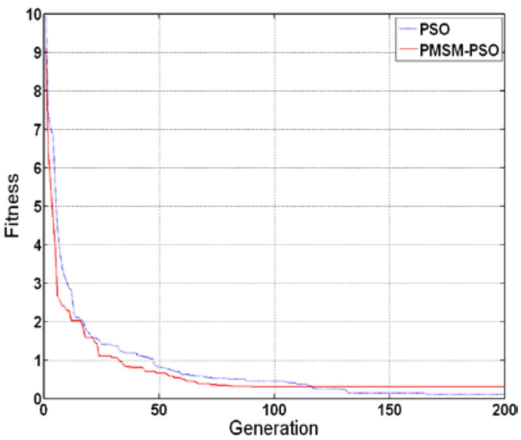

Fig.6(d). Optimal speed between PSO and PMSM-PSO: Rastrigin's function 6.

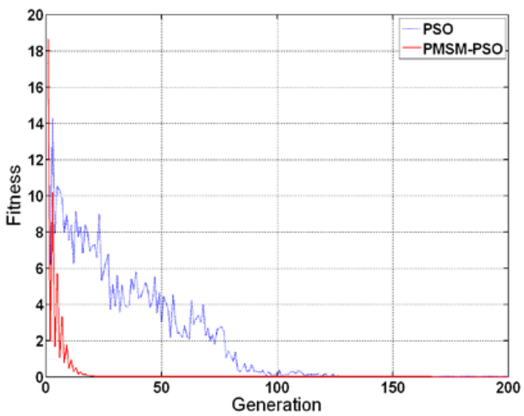

Fig.7(a). Optimal speed between PSO and PMSM-PSO: De Jong's function 1.

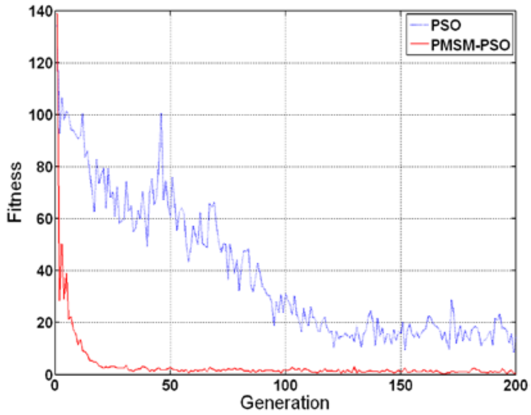

Fig.7(b). Optimal speed between PSO and PMSM-PSO: Rosenbrock's valley.

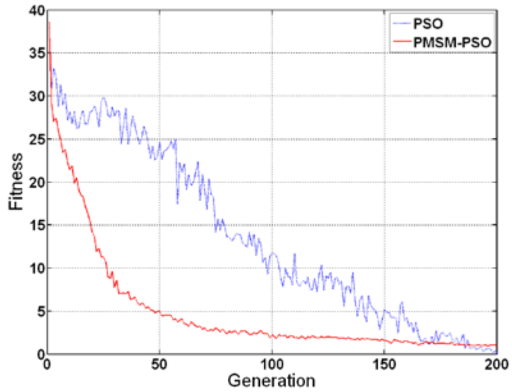

Fig.7(c). Optimal speed between PSO and PMSM-PSO: Himmelblau's function.

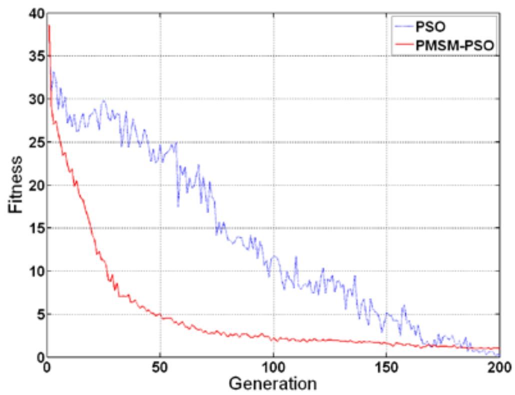

Fig.7(d). Optimal speed between PSO and PMSM-PSO: d Rastrigin's function 6.

ISBN: 978-960-474-314-8

4.1.5 Application of AVR system

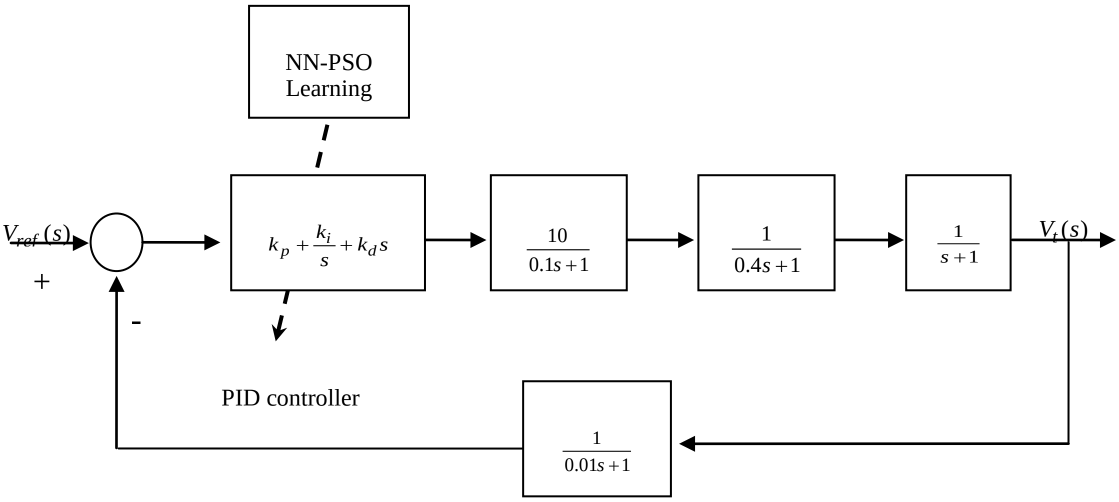

Fig. 8. AVR control system in power plant.

In this session, we apply this optimal approach to AVR system of power plant as shown in Fig.8. The transfer function of PID controller of AVR system is given by [19-24]

$$G(s) = k_p + \frac{k_i}{s} + k_d s \, , \tag{12}$$

and block diagram of AVR system is shown as Fig. 8. The performance index of control response is defined by

$$\begin{aligned} \min F(k_p, k_i, k_d) &= \frac{e^{-\beta} \cdot t_s / \max(t)}{\left(1-e^{-\beta}\right) \cdot \left|1 - t_r / \max(t)\right|} + e^{-\beta} \cdot Mo + ess \\ &= \frac{e^{-\beta} \cdot \left(t_s + \alpha_2 \cdot \left|1 - t_r / \max(t) \cdot Mo\right|\right)}{\left(1-e^{-\beta}\right) \cdot \left|1 - t_r / \max(t)\right|} + ess \\ &= \frac{e^{-\beta} \cdot \left(t_s / \max(t) + \alpha \cdot Mo\right)}{\alpha} + ess \end{aligned} \tag{13}$$

$\alpha = \left(1-e^{-\beta}\right) \cdot \left|1 - t_r / \max(t)\right|$

k_p, k_i, k_d : Parameter of PID controller

β : Weighting factor

Mo : Overshoot

t_s : Settling time (2%)

ess : Steady-state error

t : Desired settling time.

In equation (13), if the weighting factor, β increases, rising time of response curve is small, and β decreases, rising time is big. Performance criterion is defined as $Mo = 50.61\%$, $ess = 0.0909$, $t_r = 0.2693(s)$, $t_s = 6.9834(s)$.

Simple crossover and dynamic mutation of GA is used and the number of individuals is 50, 200, and initial value of crossover and mutation are 0.6, 0.5, respectively.

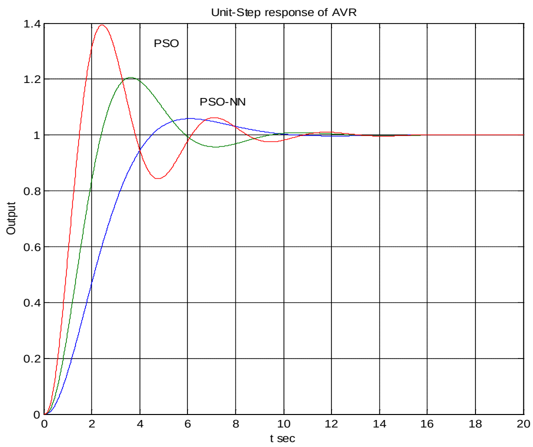

Fig.9. Response of PID controller with PSO, PSO-NN in AVR system.

5. Experiments and Discuss

In 3D mapping at initial point of Fig.4, some results are good. However, in case of initial point of Rastrigin's mapping, Fig.4(d) is so much complicated. That means it is difficult to search optimal solution. After training 100 iteration, Fig.5(d) has quite good response in optimal solution. This paper also compare training speed between traditional PSO and PSO using PMSM approach. Fig.7 and Fig.8 show both results. The optimal algorithm with PMSM by NN aided PSO in this paper is faster to final target. In AVR control system of Fig.8, it is more stable than tradition PSO as showed in Fig.9.

6. Conclusions

In this paper, as we intend to implement a completely application in the industrial area and we have been obtaining optimal solution that it is simpler to implement PSO compared to GA or so, we chose PSO as the optimization technique. Especially, computing speed in industrial fields is very important in order to obtain fast response and stability.

This paper suggests the novel search method that can speed up and to obtain global optimal in PSO as employing efficient training of a NN into PSO. The NN trained PSO can has milestone during search for optimal solution as if car has a navigation or map to search route or way to get final destination. From Fig.5, 6, 7 show that PMSM method has been successfully trained as hybrid system. The NN can perform increasing of the PSO performance and hybrid system of NN and PSO can provide faster optimal solution. The present system suggested in this paper can be further improved by making the entire PSO completely trained by others tool such as different type of neural network or FNN.

Acknowledgement

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2009-013-D00046).

References:

[1] Dong Hwa Kim, "GA–PSO based vector control of indirect three phase induction motor", Applied Soft Computing, vol.7, issue 2, March 2007. Elsevier.

[2] J.-X. Du, D.-S. Huang, J. Zhang, and X.-F. Wang, "Shape matching using fuzzy discrete particle swarm optimization," IEEE, 2005.

[3] Kennedy, J., Eberhart, R. C., and Shi, Y., Swam intelligence San Francisco: Morgan Kaufmann Publishers, 2001

[4] Kennedy, J., and Eberhart, R. C., "Particle swarm optimization", in Proc. IEEE International Conf. on Neural Networks, IEEE Service Center, Piscataway, NJ, pp. 39-43, 1995.

[5] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms", Advanced Engineering Information, vol. 19, pp. 43-53, 2005.

[6] B.R. Secrest and G.B. Lamont, "Visualizing particle swarm optimization – Gaussian particle swarm optimization," Swarm Intelligence Symposium, 2003, SIS'03. Proc. of the 2003 IEEE, pp. 198-204, April 2003.

[7] Sidhartha Panda and Narayana Prasad Padhy, "Comparison of particle swarm optimization and genetic algorithm for FACTS-based controller design," Applied Soft Computing, 2007.

[8] Zwe-Lee Gaing, "A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System," IEEE Trans. On Energy Conversion, vol. 19, no. 2, June 2004.

[9] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in Proc. IEEE World Congress on Computational Intelligence, May 1998, pp. 69–73.

[10] Zwe-Lee Gaing, "A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System", IEEE Trans. Energy. Conv. Vol. 19. pp. 384-391. No. 2, June, 2004.

[11] H. Yoshida, K. Kawata, and Y. Fukuyama, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment", IEEE Trans. Power Syst., Vol. 15, pp. 1232-1239, Nov. 2000.

[12] Angeline, P.J., "Using Selection to Improve Particle Swarm Optimization", IEEE int. Conf. Pp. 84-89, 4-9, May, 1998.

[13] Dong Hwa Kim, "Robust PID controller tuning using multi-objective optimization based on clonal selection of immune algorithm," Proc. Int. Conf. Knowledge-based intelligent information

and engineering systems. Springer-Verlag. pp. 50-56, 2004.

[14] Dong Hwa Kim, "Comparison of PID Controller Tuning of Power Plant Using Immune and genetic algorithm. Measurements and Applications," Ligano, Switzerland, 29-31 July 2003.

[15] Chia-Feng Juang, "A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design", Systems, Man and Cybernetics, Part B, IEEE Trans. Vol. 34, pp. 997-1006, 2, April, 2004.

[16] D. B. Fogel, Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. Piscataway, NJ: IEEE Press, 1995.

[17] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in Proc. IEEE World Congress on Computational Intelligence, May 1998, pp. 69–73. [28] D. J. Montana and L. Davis, "Training feedforward networks using genetic algorithms," in Proc. Int. Conf. Artifical Intelligence, Detroit, MI, 1989, pp.762–767.

[18] D. Whitley, "Genetic algorithms and neural networks," in Genetic Algorithms Engineering and Computer Science, G. Winter, J. Periaux, M. Galan, and P. Cuesta, Eds. New York: Wiley, 1995, pp. 191–201.

[19] O. Cordón, F. Herrera, F. Hoffmann, and L. Magdalena, Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases, Singapore: World Scientific, 2001.

[20] Teng Fong-Chwee, "Self-tuning PID controllers for dead time process,"IEEE Trans., vol. 35, no. 1, pp. 119-125, 1988.

[21] Ya-Gang Wang, "PI tuning for processes with dead time,"AACC2000, Chicago, Illinois, June, 2000.

[22] Dong Hwa Kim, "Intelligent Tuning of the 2-DOF PID Controller On the DCS for Steam Temperature Control of Thermal Power Plant," IEEE Industrial Application Society. I&CPS 2002, Savannah, GA, USA, May 5 - 8, 2002.

[23] Dong Hwa Kim, "Auto-tuning of reference model based PID controller using immune algorithm," IEEE international conference on evolutionary computation, Hawaii, May 12 - 17, 2002.

[24] Cing-Hung Lee, Ching-Cheng Ten, "Calculation of PID controller parameters by using a fuzzy neural network," ISA transaction pp/ 391-400, 2003.

[25] Chung-Liang Lin, Huai-Wen Su, "Intelligent control theory in guidance and control system design:an Overview," Proc. Natul. Sci., Counc. ROC(A), vol. 24, no. 1, pp. 15-30, 2000.

[26] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control system engineering: A survey," Control Eng. Practice, vol. 10, pp. 1223–1241, 2002.

[27] R. Ketata, D. De Geest and A. Titli, "Fuzzy controller: design, evaluation, parallel and hierarchical combination with a PID controller," Fuzzy Sets and Systems, vol. 71, pp. 113-129, 1995.