Load Frequency Control of Two Area System Using PSO with Different Inertia Weight Strategies

M. S. V. Phani[1], Pavan Kumar .G[2]

[1, 2]Sagi Ramakrishna Raju Engineering College, Bhimavaram

Abstract: ***This paper presents the application of different inertia weight strategies to Particle swarm optimization (PSO) algorithm for a two area power system employing PID controller for automatic generation control. In automatic generation control (AGC) problem when we adopt conventional PSO algorithm there is a tradeoff between the global and local search. For different problems there should be different balances between the local search ability and global search ability. Considering this, different inertia weight strategies are brought in to the PSO algorithm which plays the role of balancing the global search and local search. Constant Inertia Weight, Random Inertia Weight and Linear Decreasing Inertia Weight are the different inertia weight strategies adopted and results are compared by applying to automatic generation control problem. The entire system is simulated using MATLAB/SIMULINK platform.***

Keywords: Two-area, Automatic generation control, Particle swarm optimization, inertia weight strategies.

1. Introduction

Synchronization of all regional grids will help in optimal utilization of scarce natural resources by transfer of power from resource centric regions to load centric regions. Further, this shall pave way for establishment of vibrant electricity market facilitating trading of power across regions. One nation one grid shall synchronously connect all the regions and there will be one national frequency. [1]

In a practically interconnected power system holding the frequency to a fixed point is not an easy task due to different kinds of uncertainties present due to load variations [2][4]. Several control strategies are proposed for AGC of power systems in order to maintain the system frequency and tie line power flow at their scheduled values during normal operation and also during random load perturbations. [3]

Whatever the control strategy adopted the main aim is to determine the gains of the PID controller that maintains the frequency of each area within limits and to keep the tie-line power flows within some pre-specified tolerances, by satisfying system constraints and overcoming uncertainties [5]. Particle swarm optimization (PSO) is commonly employed technique to tune the PID controller gain constants to fulfill the system requirements.

In contrast to the evolutionary computation techniques, Eberhart and kennedy developed a different algorithm through simulating social behavior [11]. This algorithm is called Particle swarm optimization (PSO) since it resembles a school of flying birds. As in other algorithms, a population of individuals exists, known as particles. Each particle adjusts its flying according to its own flying experience and its companions flying experience. Each particle represents a potential solution to a problem. Each particle is treated as a point in D-dimensional space. The ith particle is represented as

$$X_1=(x_{i1},x_{i2},\ldots..x_{iD})$$

The best previous position (the position given the best fitness value) of any particle is recorded and represented as

$$P1=(p_{i1}.p_{i2},\ldots..p_{iD})$$

The index of the best particle among all the particles in the population is represented by the symbol g. The rate of the position change (velocity) for particle i is represented as

$$V_1=(v_{i1}.v_{i2},\ldots..v_{iD})$$

The particles are manipulated according to the following equation:

$$V_1=v_{id}+c_1{*}rand()*(p_{id}-x_{id})+c_2{*}Rand()*(p_{gd}-x_{id}) \quad (1a)$$

$$X_{id}= x_{id}+v_{id} \quad (1b)$$

Where c1 and c2 are two positive constants, rand() and Rand() are two functions in the range[0,1]. The second part of equation (1a) is the “cognition” part, which represents the private thinking of particle itself. The part is the social part, which represents collaboration among the particles. The equation (1a) is used to calculate the particle’s new velocity and the distances of its current position from its own best experience (position) and the group’s best experience. Then the particle flies toward a new position according to (1b). The performance of each particle is measured according to a pre-defined fitness function, which is related to the problem to be solved.

Refer to equation (1a), the right side of which consists of three parts: the first part is the previous velocity of the particle; the second and third parts are the ones contributing to the change of the velocity of a particle. Without these two parts, the particles will keep on “flying” at the current speed in the same direction until they hit the boundary. PSO will not find acceptable solution unless there are acceptable solutions on their trajectories. But that is a rare case. On the other hand, refer to equation (1a) without the first part. Then the “flying” particles velocities are only determined by their current positions and their best position in history. The velocity itself is memory less. Assume at the beginning, the particle I will be “flying” at the velocity 0, that is, it will keep still until another particle takes over the global best position. At the same time, each other particle will be “flying” towards its weighted centroid of its own best position and the global best position of the population. As

Licensed Under Creative Commons Attribution CC BY

mentioned in [6], a recommended choice for constant c1 and c2 is integer 2 since it on average makes the weights for “social” and “cognition’’ parts to be 1. Under this condition, the particles statistically contract swarm to the current global best position until another particle takes over from which time all the particles statistically contract to the new global best position. Therefore, it can be imagined that the search process for PSO without the first part is a process where the search space statistically shrinks through the generations. It resembles a local search algorithm. This can be illuminated more clearly by displaying the “flying” process on a screen. From the screen, it can be easily seen that without the first part of equation (la), all the particles will tend to move toward the same position, that is, the search area is contracting through the generations. Only when the global optimum is within the initial search space, then there is a chance for PSO to find the solution. The final solution is heavily dependent on the initial seeds (population). So it is more likely to exhibit local search ability without the first part. On the other hand, by adding the first part, the particles have a tendency to expand the search space, that is, they have the ability to explore the new area. So they more likely have global search ability by adding the first part. Both the local search and global search will benefit solving some kinds of problems. There is a tradeoff between the global and local search. For different problems, there should be different balances between the local search ability and global search ability. Considering of this, a inertia weight w is brought into the equation (1) as shown in equation (2). This w plays the role of balancing the global search and local search, first time Shi and Eberhart [12] presented the concept of Inertia Weight by introducing Constant Inertia Weight. It can be a positive constant or even a positive linear or nonlinear function of time.

$$V_1= w*v_{id}+c_1*rand()*(p_{id}-x_{id})+c_2*Rand()*(p_{gd}-x_{id}) \quad (2a)$$
$$X_{id}= x_{id}+v_{id} \quad (2b)$$

As there is a large effect of initial velocity in the balancing of exploration and exploitation process of swarm, Inertia Weight (w) is used to control the velocity. In this paper, Inertia Weight for PSO is reviewed and experiments are carried out by applying different inertia weight strategies to automatic generation control (AGC) problem to compare different strategies of setting Inertia Weight.

2. Inertia Weight Strategies

Inertia Weight plays a major role in the process of providing balance between exploration and exploitation process. J.C. Bansal,P. K. Singh, Mukesh Saraswat, [9] proposed different inertia weight strategies in 2011.The use of inertia weight w, which typically decreases linearly from 0.9 to 0.4 during run, has improved performance in number of applications. The Inertia Weight determines the contribution rate of a particle’s previous velocity to its velocity at the current time step. The basic PSO, presented by Eberhart and Kennedy in 1995 [11], has no Inertia Weight. In 1998, first time Shi and Eberhart [12] presented the concept of Inertia Weight by introducing Constant Inertia Weight (CIW). They stated that a large Inertia Weight facilitates a global search while a small Inertia Weight facilitates a local search. Further, dynamical adjusting of Inertia Weight was introduced by many researchers which can increase the capabilities of PSO.

Constant Inertia Weight w = c
c=0.7(considered for experiments)

Eberhart and Shi [7] proposed a Random Inertia Weight strategy and experimentally found that this strategy increases the convergence of PSO in early iterations of the algorithm.

J. Xin, G. Chen, and Y. Hai [13] proposed Linearly Decreasing strategy; it enhances the efficiency and performance of PSO. It is found experimentally that Inertia Weight from 0.9 to 0.4 provides the excellent results. By reason of these values, the inertia weight can be interpreted as fluidity of the medium in which particles moves, showing that setting it to relatively high initial weight (e.g., 0.9) makes particles move in low viscosity medium and performs extensive exploration. Gradually reducing it to a much lower value (e.g., 0.4) makes the particle move in a high viscosity medium and performs more exploitation. In spite of its ability to converge optimum, it gets into the local optimum solving the question of more apices function. Linear decreasing inertia weight PSO (LDIW-PSO) algorithm have shortcoming of premature convergence in solving complex (multi peak) optimization problems due to lack of enough momentum for particles to do exploitation as the researchers have tried to address this shortcoming by modifying LDIW-PSO or proposing new PSO variants. Some of these variants have been claimed to outperform LDIW-PSO. The major goal of this paper is by simulation experimentally establish the fact that LDIW-PSO is very much efficient if its parameters, like velocity limits for the particles, are properly set.

Table 1: Different Inertia Weight Strategies

S.No	*Inertia Weight*	*Formula*
1.	Constant Inertia Weight	$\omega = c$ c=0.7 (considered for experiments)
2.	Random inertia Weight	$w = 0.5 - \frac{Rand()}{2}$
3.	Linear Decreasing Inertia Weight	$w_k = w_{max} - \frac{w_{max} - w_{min}}{iter_{max}} \times k$

3. Implementation of PSO-Based PID Tuning

PID controller consists of three separate parameters: proportional, integral and derivative with gains denoted by Kp, Ki, Kd. Appropriate setting of these parameters will improve dynamic response of a system, reduce over shoot eliminate steady state error and increase stability of the system. Proportional Integral (PI) controllers are the most often type used today in industry [5]. A control without derivative (D) mode is used when: fast response of the system is not required, large disturbances and noises are present during operation of the process and there are large

Licensed Under Creative Commons Attribution CC BY

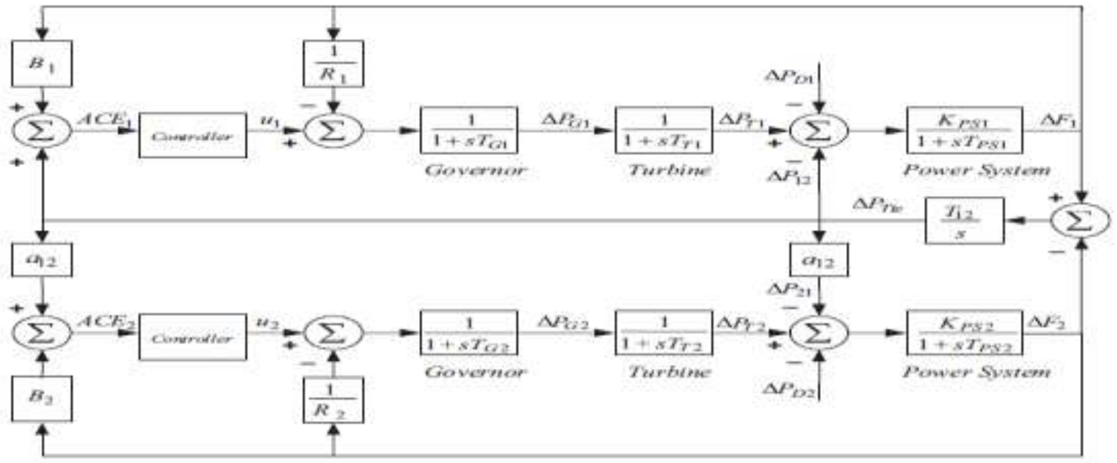

Figure 2: Transfer function model of two-area thermal system

transport delays in the system. Proportional Integral Derivative (PID) controllers are used when stability and fast response are required. Derivative mode improves stability of the system and enables increase in proportional gain and decrease in integral gain which in turn increases speed of the controller response.

PSO [17] is employed to tune PID gains (Kp, Ki, Kd). PSO firstly produces initial swarm of particles in search space represented by matrix. Each particle represents a candidate solution for PID parameters where their values are set in the range of 0 to 100.

4. PSO Algorithm

Step-1: Set up the control parameters of PSO optimization process that are population size, acceleration constants (C_1,C_2), convergence criterion, number of problem variables, lower and limits of variables and maximum number of iterations. Create an initial population of particles with random positions and velocities. The positions (X_{ki}) and velocity (V_{ki}) of initial swarm of the particle are randomly generated using lower and upper bounds of design variables. For ith particle position and velocity are generated as follows:

$$X_{0i}=X_{min}+(X_{max}-X_{min})*rand \quad (7)$$

$$V_{0i}=V_{min}+(V_{max}-V_{min})*rand \quad (8)$$

Step-2: For each particle calculate the value of fitness function.

Step-3: Compare the fitness of all particles with global best (g_{best}). If any of the particles is better than g_{best}, and then replace $g_{best.}$

Step-4: Update the velocity and positions of all particles. The velocity of i_{th} particle is updated as

$$V_{k+1i}=V_{ki}+C_1*rand(P_{bestki}-X_{ki})+C_2*rand(G_{besti}-X_{ki}) \quad (9)$$

Where Vki is the velocity of ith particle at time k.c1,c2 are acceleration constants.r1,r2 are random variables. Pbestki is the personal best position of ith particle at time k. Gbesti is the global best position of ith particle. Xki is the position of ith particle at time k. The position of the particle is updated as

$$X_{k+1i}= X_{ki}+ V_{k+1i} \quad (10)$$

Step-5: Repeat the steps from 2 to 4 until the desired fitness is reached.

The fitness function considered here is based on error criterion. This work utilizes performance indices as objective function. Controller performance is evaluated in terms of integral square error (ISE), integral absolute error(IAE),integral time multiplied by absolute error(ITAE). PID controller is tuned based on the minimum value of performance index. Algorithm of PSO is shown in figure 3.

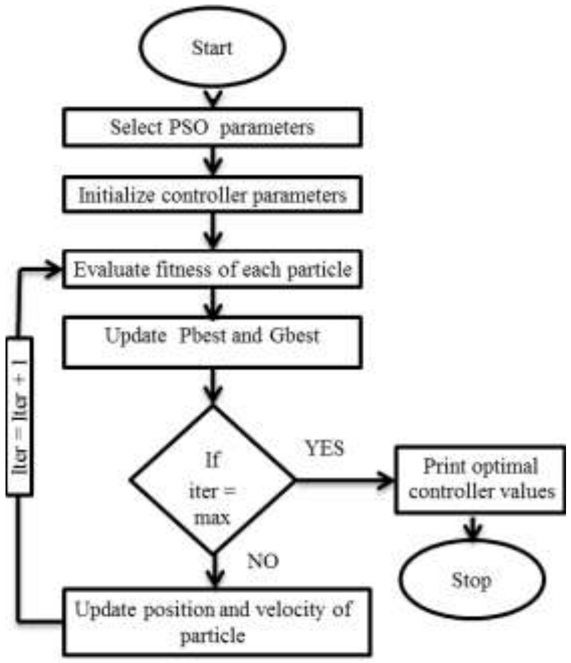
Start
Select PSO parameters
Initialize controller parameters
Evaluate fitness of each particle
Update Pbest and Gbest
If iter = max
YES
NO
Print optimal controller values
Stop
Update position and velocity of particle
Iter = Iter + 1

Figure 3: Flow chart of PSO

Licensed Under Creative Commons Attribution CC BY

6.6 Comparison of three inertia weight strategies

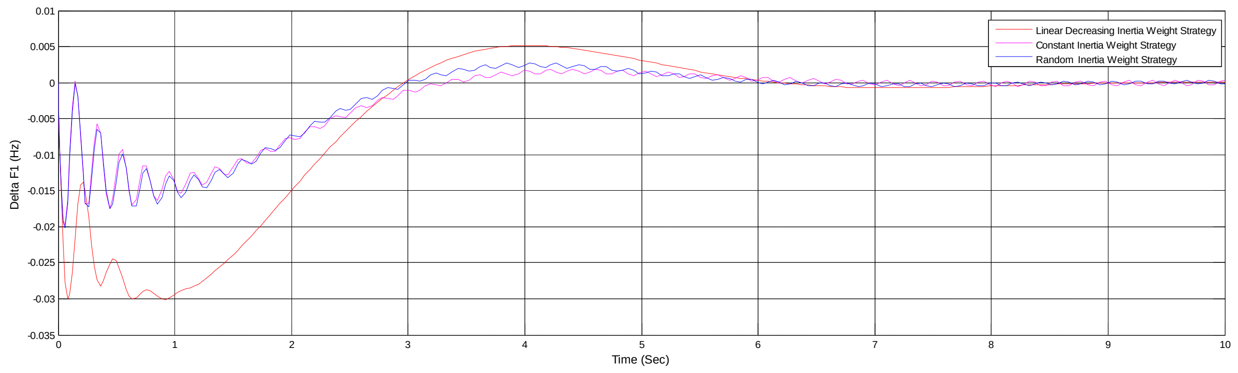

Figure 49: Dynamic response for 10% step load increase area-1 (ΔF1) with different inertia weight strategies

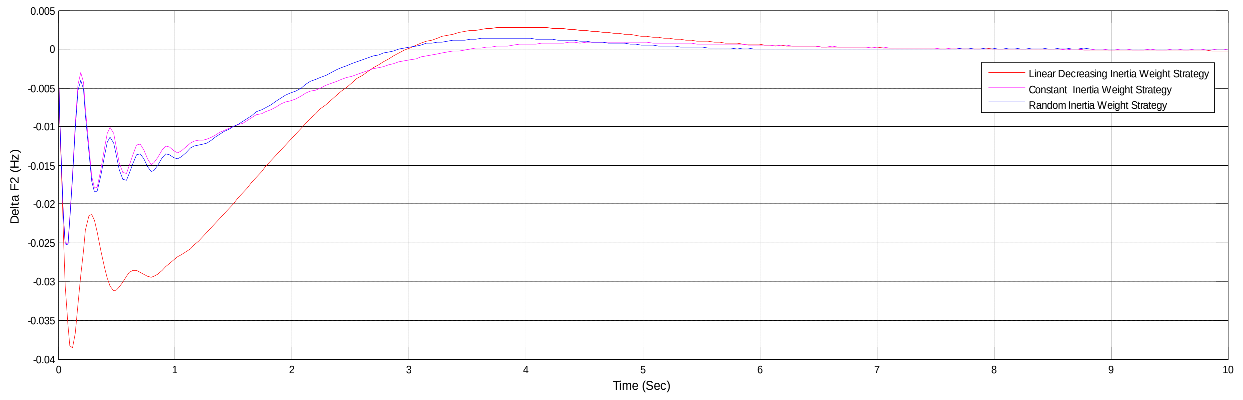

Figure 50: Dynamic response for 10% step load increase area-2 (ΔF2) with different inertia weight strategies

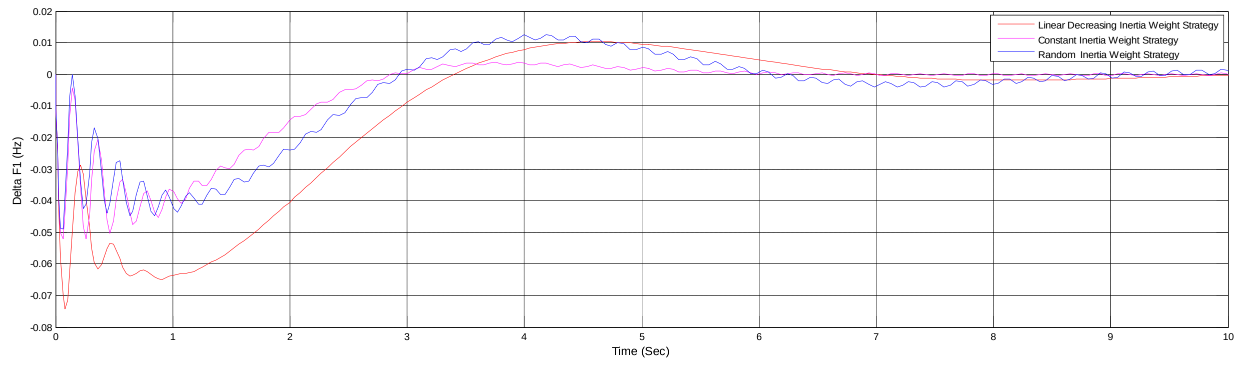

Figure 51: Dynamic response for 25% step load increase area-1 (ΔF1) with different inertia weight strategies

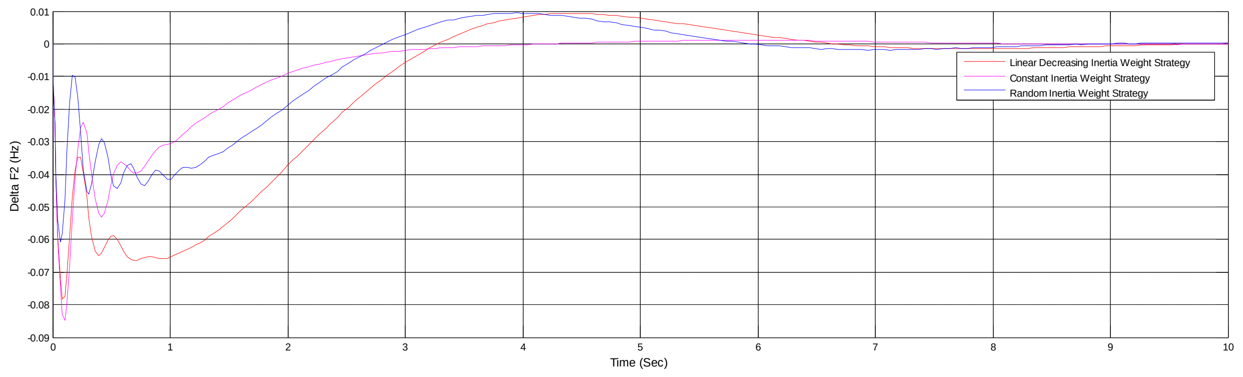

Figure 52: Dynamic response for 25% step load increase area-2 (ΔF2) with different inertia weight strategies

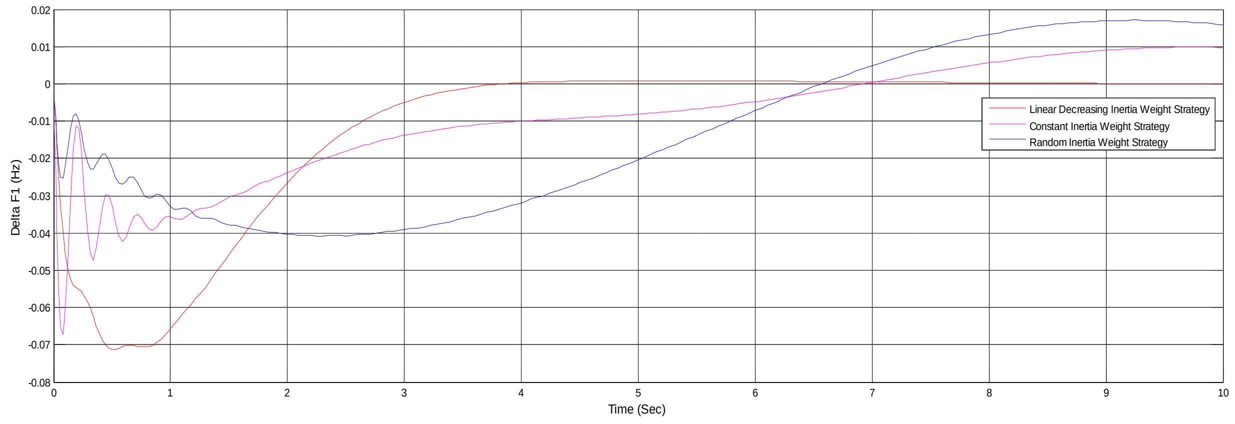

Figure 53: Dynamic response (ΔF1) for 10% step load increase area-1 & 25% step load increase area-2 with different inertia weight strategies

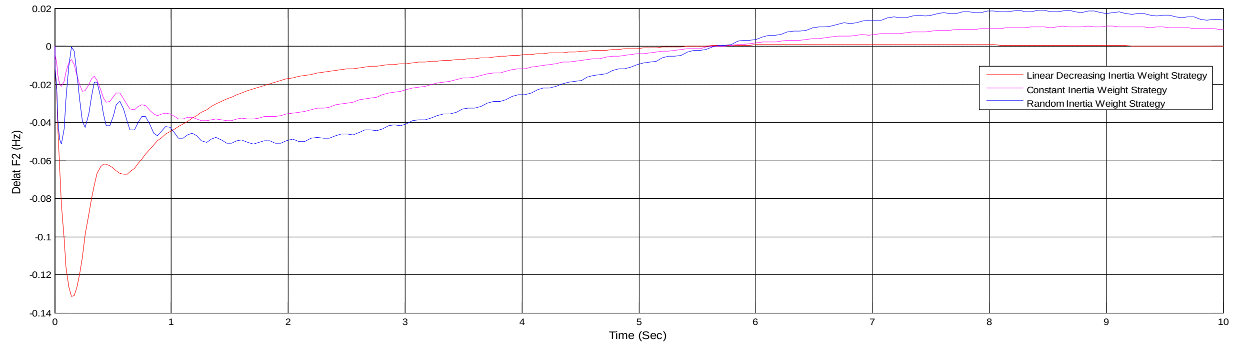

Figure 54: Dynamic response (ΔF2) for 10% step load increase area-1 & 25% step load increase area-2 with different inertia weight strategies

Table 2: Dynamic Response of PID tuned by PSO algorithm with different inertia weight strategies at 10% step disturbance of load

S. No	Dynamic Response	Constant Inertia Weight	Random Inertia Weight	Linear Decreasing Inertia Weight
1	First Peak of Δf1	-0.0855	0.0987	-0.073
2	First Peak of Δf2	-0.0363	-0.0256	-0.0115
3	Min ISE	4.534×10^{-4}	4.837×10^{-4}	1.27×10^{-4}
4	Steady State error of Δf1	3.9×10^{-4}	1.85×10^{-4}	1.807×10^{-4}
5	Steady State error of Δf2	4.21×10^{-4}	4.22×10^{-4}	1.650×10^{-4}

Table 3: Dynamic Response of PID tuned by PSO algorithm with different inertia weight strategies at 25% step disturbance of load

S. No	Dynamic Response	Constant Inertia Weight	Random Inertia Weight	Linear Decreasing Inertia Weight
1	First Peak of Δf1	-0.0633	-0.0884	-0.0515
2	First Peak of Δf2	-0.084	-0.0907	-0.0486
3	Min ISE	0.0031	0.0050	0.00110
4	Steady State error of Δf1	0.2×10^{-4}	4.31×10^{-4}	1.31×10^{-4}
5	Steady State error of Δf2	1.4×10^{-4}	4.67×10^{-4}	1.67×10^{-4}

Table 4: Dynamic Response of PID tuned by PSO algorithm with different inertia weight strategies at area1=10% and area2=25% step disturbance of load

S. No	Dynamic Response	Constant Inertia Weight	Random Inertia Weight	Linear Decreasing Inertia Weight
1	First Peak of Δf1or	-0.077	-0.0584	-0.0471
2	First Peak of Δf2	-0.087	-0.0652	-0.01292
3	Min ISE	0.0104	0.0152	0.0087
4	Steady State error of Δf1	-9.2×10^{-4}	7.652×10^{-4}	4.808×10^{-4}
5	Steady State error of Δf2	-8.286×10^{-4}	5.57×10^{-4}	4.82×10^{-3}

5. Conclusion

In this paper, an attempt has been made to apply Different inertia weight strategies in PSO algorithm for PID controller tuning for AGC of an interconnected power system. Firstly, a two area thermal system is considered and the superiority of the proposed approach is demonstrated by comparing the results of three inertia weight strategies such as Constant Inertia Weight, Random Inertia Weight and Linear Decreasing Inertia Weight in PSO algorithm. Then, analysis is carried out that demonstrates the robustness of the optimized controller parameters to wide variations in operating loading condition. The results obtained from the simulations show that the Linear decreasing inertia weight strategy achieves better dynamic performances compare to other inertia weight strategies when applied for the same power system. Finally, the dynamic response of the power system under random step load changes has been verified with five different inertia weight strategies. It is observed that Linear decreasing inertia weight strategy and next Global-Local best inertia weight strategy gives a better performance compare to others.

Appendix A
Nominal parameters of the system investigated are:
Two area system [6,16,17]:
f = 60 Hz, $B_1 = B_2 = 0.425$ p.u. MW/Hz; $R_1 = R_2 = 2.4$ Hz/p.u.; $T_{G1} = T_{G2} = 0.08$ s; $T_{T1} = T_{T2} = 0.3$ s; $K_{PS1} = K_{PS2} = 120$ Hz/p.u.; $T_{PS1} = T_{PS2} = 20$ s; $T_{12} = 0.545$ p.u.; $a_{12} = -1$.

References

[1] Elgerd OI. Electric energy systems theory. An introduction. 2nd ed. New Delhi: Tata McGraw-Hill; 2007.
[2] Kundur P. Power system stability and control; 8th reprint. New Delhi: Tata McGraw-Hill; 2009.
[3] Bevrani H. Robust power system frequency control. New York: Springer; 2009.
[4] Elgerd OI, Fosha CE. Optimal megawatt-frequency control of multi-area electrical energy systems. IEEE Trans PAS 1970;89(4):556–63.
[5] Y. Gao, X. An, and J. Liu., “A Particle Swarm Optimization Algorithm with Logarithm Decreasing Inertia Weight and Chaos Mutation”, In Computational Intelligence and Security, 2008. CIS’08. International Conference on, volume 1, pages 61–65. IEEE, 2008.
[6] Rout UK, Sahu RK, Panda S. Design and analysis of differential evolution algorithm based automatic generation control for interconnected power system. Ain Shams Eng J 2013;4(3):409–21.
[7] R.C. Eberhart and Y. Shi., “Tracking and optimizing dynamic systems with particle swarms”, In Evolutionary Computation, 2001. Proceedings of the 2001 Congress on, volume 1, pages 94–100. IEEE, 2002.
[8] Panda S, Mohanty B, Hota PK. Controller parameters tuning of differential evolution algorithm and its applicationto load frequency control of multisource power system. Int J Electr Power Energy Syst 2014; 54:77–85.
[9] J.C. Bansal,P. K. Singh, Mukesh Saraswat, “Inertia Weight Strategies in Particle Swarm Optimization”,978-1-4577-1123-7/11/$26.00_c 2011 IEEE.
[10] Rabindra Kumar Sahu, Sidhartha Panda, Umesh Kumar Rout, Dillip Kumar Sahoo .Teaching learning based optimization algorithm for automatic generation control of power system using 2-DOF PID controller. Electrical Power and Energy Systems 77 (2016) 287–301.
[11] J. Kennedy, R.C. Eberhart, et al.,“Particle swarm optimization”,In Proceedings of IEEE international conference on neural networks, volume 4, pages 1942–1948. Perth, Australia, 1995.
[12] Y. Shi and R. Eberhart., “A modified particle swarm optimizer”, In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence. The1998IEEE International Conference on, pages 69–73.IEEE, 2002.
[13] J. Xin, G. Chen, and Y. Hai., “A Particle Swarm Optimizer with Multistage Linearly-Decreasing Inertia Weight”, In Computational Sciences and Optimization, 2009. CSO2009. International Joint Conference on, volume1, pages 505–508. IEEE, 2009.
[14] A.Nikabadi, M.Ebadzadeh , “Particle swarm optimization algorithms with adaptive Inertia Weight : A survey of the state of the art and a Novel method”, IEEE journal of evolutionary computation ,2008
[15] R.F. Malik, T.A. Rahman, S.Z.M. Hashim, and R. Ngah, “New Particle Swarm Optimizer with Sigmoid Increasing Inertia Weight”, International Journal of Computer Science and Security (IJCSS), 1(2):35, 2007.
[16] Ali ES, Abd-Elazim SM. Bacteria foraging optimization algorithm based load frequency controller for interconnected power system. Electr Power Energy Syst 2011;33:633–8.
[17] Mohanty B, Panda S, Hota PK. Hybrid BFOA–PSO algorithm for automatic generation control of linear and nonlinear interconnected power systems. Appl Soft Comput 2013;13:4718–30.

Licensed Under Creative Commons Attribution CC BY