Multimedia Tools and Applications (2024) 83:3021–3055
https://doi.org/10.1007/s11042-021-11266-4

1203: APPLICATIONS OF ADVANCED ARTIFICIAL INTELLIGENCE IN MULTIMEDIA AND INFORMATION SECURITY

A new adaptive tuned Social Group Optimization (SGO) algorithm with sigmoid-adaptive inertia weight for solving engineering design problems

Junali Jasmine Jena[1] · **Suresh Chandra Satapathy**[1]

Received: 25 August 2020 / Revised: 24 March 2021 / Accepted: 8 July 2021 /
Published online: 24 August 2021

Abstract
Evolutionary algorithms have found enormous applications in solving real-world problems due to their stochastic nature. They have a set of control parameters, which are used to perform certain operations to induce randomness, scalar displacement etc. Various works have been done for tuning these parameters, as appropriate parameter tuning can enhance the performance of algorithm greatly. Inertia weights based parameter tuning is one of the widely used techniques for this purpose. In this paper, we have reviewed some of the inertia weight strategies and applied them to Social Group Optimization (SGO) to study the changes in its performance and have performed a thorough analysis on the same. Following the analysis, the need of a more generalized inertia weight strategy was felt which could be used in parameter tuning for different variety of problems and hence Sigmoid adaptive inertia weight have been proposed. SGO with sigmoid-adaptive inertia weight (SGOSAIW) has been simulated on twenty-seven benchmark functions suite and further simulated on few mechanical and chemical engineering problems and compared to other similar algorithms for performance analysis. In eight-benchmark function suite, SGOSAIW obtained better minima except one i.e. 'Schwefel 2.26' with respect to other algorithms investigated in this work. In nineteen-benchmark function suite, SGOSAIW obtained better minima except one i.e. 'Noisy function'. Thus, the proposed algorithm yielded promising results which are well represented with suitable tables and graphs in the paper.

Keywords Sigmoid-adaptive inertia weight · Social group optimization · Single objective optimization · Engineering design problems · Evolutionary algorithms

✉ Suresh Chandra Satapathy
suresh.satapathyfcs@kiit.ac.in

Junali Jasmine Jena
junali.jenafcs@kiit.ac.in

[1] School of Computer Engineering, Kalinga Institute of Industrial Technology, Bhubaneswar, India

1 Introduction

Real world problems are complex in nature which needs intelligent algorithms for solving those [8]. Evolutionary Algorithms (EAs) are a group of intelligent algorithms which are used to find optimal solutions for problems which require non deterministic polynomial time to solve. Polynomial time solvable problems (P) are easy to solve and are easily verifiable for example addition, subtraction, multiplication etc. Non deterministic polynomial time solvable problems (NP) are hard to solve. NP-complete problems are those NP problems which are although tough, but verifiable such as game of Sudoku. NP-hard problems are those NP problems which are hard to solve as well as hard to verify like finding the next best move in game of Chess. Hence EAs are suitable dealing with NP problems. Using randomly initialized candidate solutions and an appropriate fitness function are the two main important features of EAs, which enable them to find solutions to difficult problems, though the solution may not be the best one but getting a near optimal solution can be expected. Hence, EAs find huge application to several problem domains such as engineering, life sciences, medicine etc.

According to No Free Lunch theorem, no evolutionary algorithm could be claimed to solve all types of optimization problems, and that has paved way for emergence of new evolutionary algorithms. These evolutionary algorithms achieve their goals by proper tuning of algorithmic parameters such as different operators, control parameters, different number of phases etc. Tuning of control parameters play a vital role to make the algorithm perform better for wide range of problems under investigation. In the literature, it is found that various works have been done to fine tune the control parameters. Many hybrid methods or adaptive methods have been suggested by researchers for different evolutionary algorithms. However, authors of this work feel that there must be viable inference, to be drawn, to ascertain the behavior of a particular algorithm whilst its control parameters are being modified. Why an algorithm gives better result for a particular problem? Why at a certain value of control parameter, an algorithm performs better? These are some of the basic questions which arise in mind of every reader and every researcher. Researchers should not only focus trying different possibilities of experimentation but also try to explain their perceptions and analysis about the same.

An evolutionary algorithm has two important parts – Exploration phase and Exploitation phase. So, tuning of parameters should be done basing upon the position of parameter, whether it is present in the exploration or exploitation phase. Parameters of exploration phase help the algorithm to make sharp convergence towards the optimum, whereas of exploitation phase, help algorithm to search the vicinity of the present solution space for optimal solution. Let's take the example of PSO, which is a widely used evolutionary algorithm. It has three control parameters i.e. 'w', 'c1' and 'c2'. The inertia weight 'w' [16] refers to exploration phase and the algorithm performs better for higher values of 'w' [38] or linear decreasing value of 'w' [17]. The reason lies behind the fundamental philosophy of PSO. It was inspired from flock of birds flying towards a certain goal. In PSO, every bird tries to follow the leader. So, for a particular bird, initial velocity is higher, but as it nears the destination, velocity gradually decreases. Hence, 'Linear decreasing inertia weight based PSO' [17] performed better than PSO and it was one of the most popular technique and has got numerous citations.

Social Group Optimization (SGO) [36, 49], proposed by Satapathy and Naik has performed well in various optimization and real-world problems. The algorithm has an advantage of having only one control parameter 'C', i.e. self-introspection parameter, which is

actually good, as dealing with too many control parameters increases the computational complexity. Moreover, SGO algorithm is based on a very simplified approach i.e. Random walk based on the best candidate solution and the fitness value of a randomly chosen candidate solution. No complex techniques are involved which makes it easier to be understood by the researchers of other fields and hence appropriate for easy applications. Complex techniques needs more complex logic to be implemented, but in SGO simple random walk technique has been used in both the phases. SGO has successfully been applied to several domains. Rajnikanth and Satapathy [43] used SGO and fuzzy-tsallis entropy for segmentation of ischemic stroke lesion in brain MRI images. Dey et al. [11] used SGO for segmentation and evaluation of skin melanoma images. Praveen et al. [42] used SGO for task scheduling and resource allocation in cloud environment. SGO was also used for task scheduling and resource allocation by Dey et al. [12]. Naik et al. [37] used a modified SGO for short term hydro-thermal scheduling. Similarly, Fang et al. [19] used improved SGO for transformer fault diagnosis. Das et al. [9] used SGO for structural monitoring in the field of civil engineering.

While going through the state of the arts of the SGO, we find that no works based on application of inertia weight techniques could be found in the literature and this has been addressed in this paper. After analyzing the performance of some of the inertia weight strategies with SGO, we felt the need of a better inertia weight technique which can properly tune the parameter for variety of problems, and hence Sigmoid Adaptive Inertia Weight have been proposed and simulated with SGO (SGOSAIW) upon several benchmark functions and few engineering domain applications and results obtained were satisfactory. Highlights of the works carried out in this paper are -

- Simulated SGO with some of the inertia weight techniques and analyzed their behavior.
- Following the analysis, we felt the need of a better performing inertia weight strategy, which could be applicable to wide range of problems and hence proposed a new inertia weight technique known as Sigmoid-adaptive inertia weight
- Sigmoid adaptive inertia weight was implemented with SGO (SGOSAIW) and compared to two well-known and recently proposed, adaptive evolutionary algorithms i.e. ATLBO [51] and ASF-BA [48]
- SGOSAIW has also been applied to variety of engineering design problems such as Three bar truss problem, Cantilever beam problem, Process design and synthesis problems and has been compared to some recent state of the arts techniques.

Rest of the paper have been organized as - Review of some related works have been provided in Sect. 2 followed by application of inertia weight strategies to SGO in Sect. 3. The description of proposed SGOSAIW have been provided in Sect. 4 followed by simulation and results in Sect. 5. Section 6 highlights the Conclusion and future aspects of the research work.

2 Related works

Several evolutionary algorithms have emerged based on mathematical derivations or inspired from nature, and various inertia weight strategies have been applied for parameter tuning for increasing the algorithm efficiency and efficacy. This section gives a comprehensive view of such kind of inertia weight strategies in various algorithms. Liu et al.

[31] have proposed a Modified PSO having a chaotic based non-linear inertia weight and found significant improvement in the performance. Pawan and Prakash [41] investigated a range of inertia weight for PSO and proposed IWPSO with some effective modifications, which enhanced its performance. Shukla et al. [51] proposed an Adaptive TLBO and found increased convergence rate in the algorithm. Yue and Zhang [55] proposed a modified hybrid bat algorithm and got better results while using it for multilevel image segmentation. Rauf et al. [48] proposed ASF-Bat algorithm and also found promising results. Kiani et al. [29] proposed a Chaotic inertia weight PSO and found better results in estimating solar cell parameters. Huang et al. [25] used time varying inertia weight based PSO for task scheduling in cloud and obtained better results. Chen et al. [6] used adaptive inertia weights to propose an Improved pigeon-inspired optimization and observed improvement in performance. Singh et al. [50] used inertia constant strategy on Mean grey-wolf optimizer found enhanced performance in the algorithm. Orouskhani et al. [40] proposed an average inertia weighted cat swarm optimization and found better results. Olivas et al. [39] proposed an ACO with dynamic parameter adaptation and witnessed improved performance. Hu et al. [24] proposed Improved whale optimization based on inertia weights and observed that performance of the algorithm improved greatly. Rani et al.[44] proposed Modified cuckoo search algorithm with weighted sum for optimization of linear antenna array synthesis. Chou and Ngo [7] proposed inertia weight based Modified firefly algorithm for multidimensional optimization in structural design and achieved improved performance. In the literature, it is found that PSO has been extensively studied with varieties of inertia weights. Harrison et al. [22], Rathore and Sharma [47], Bansal et al. [2], Imran et al. [26] and Chauhan et al. [3] have reviewed several such works on inertia weight based PSO, which can be referred to and can be used for improving other optimization algorithms.

Though there are several other parameters which affect the optimality of the algorithm, but in this paper we have focused on algorithmic dependent parameter only like 'C' in SGO, 'w' in PSO etc. When an EA have more algorithmic dependent parameters, we have to also perform the correlation analysis among the parameters which will help us in knowing their mutual behavior when there value changes. More parameters mean more complexity in parameter tuning. There are other parameters such as number of population, number of function evaluation, number of iterations etc. which are basic to every EA and hence not algorithm dependent. Table 1 provides the information about some of the EAs along with their algorithm dependent variables.

Also several optimization algorithms have found their applications in solving some of the constrained optimization problems such as engineering design problems and a few such engineering optimization problems are considered in our work to implement our proposed algorithm. Mirjalili et al. [35] used Salp Swarm algorithm to solve various engineering design problems and got better result. Similarly, Dhiman and Kaur [13] used Spotted hyena optimizer for the same and observed improvement in the performance. Chen et al. [4], proposed a balanced Whale optimization algorithm using levy flight and chaotic local search to solve constrained optimization problems and the results obtained were promising. Azqandi et al. [1] used an enhanced time evolutionary optimization and de Paula Garcia et al. [10] used Genetic algorithm along with a rank based technique to solve constrained engineering design problems the performance was found better than some of the state of the arts techniques. Overall idea is that, few algorithms will perform better than the others in some cases where as others will perform better in some different cases. So, there has always been a scope of improvement and experimentation by using several combinations of techniques and hence in this paper, we have tried to improve the performance of the conventional SGO using some inertia weight strategy.

Table 1 List of Control Parameters of some Evolutionary Algorithms

Evolutionary Algorithms	*Algorithm dependent parameter(s)*
Teaching Learning Based Optimization (TLBO) [45]	T_f – Teaching Factor
Bat Algorithm (BA) [53]	α- Pulse loudness β- Pulse rate
Pigeon Inspired Optimization (PIO) [15]	F- Rate of velocity change
Grey Wolf Optimization (GWO) [34]	A and C– Exploration and Exploitation parameters
Ant Colony Optimization (ACO) [14]	τ- Pheromone vaporization factor
Cuckoo Search Algorithm (CSA) [21]	P_a – Switching probability
Artificial Bee Colony (ABC) [27]	P_N - Number of bees to number of food sources ratio
Firefly Algorithm (FA) [54]	α- Step size factor γ- Absorption coefficient
Whale Optimization Algorithm (WOA) [33]	a- distance control parameter b- logarithmic spiral constant
Social Group Optimization (SGO) [49]	C - Self Introspection parameter

After reviewing the literature thoroughly, we come to a conclusion that inertia weight techniques are doing fairly good and it motivated us to explore the same with Social group optimization (SGO) and later the proposed SGO with sigmoid-adaptive inertia weight (SGOSAIW) has been simulated upon numerical benchmark functions and engineering design problems to verify the effectiveness of the proposed approach.

3 Application of different inertia weight strategies to SGO

In this section, the basic SGO is presented and followed by that, seven varieties of inertia weight parameter tuning techniques are discussed and those are simulated for few benchmark functions to verify their effectiveness in solving optimization problems.

3.1 SGO algorithm

SGO is inspired from social behavior of individuals in a group. It has two phases. They are '*Improving phase*' and '*Acquiring phase*'. Improving phase is the exploration phase of the algorithm and acquiring phase is the exploitation phase of the algorithm, whose equations are described in Eqs. (1), (2) and (3), respectively, where 'P_{ij}' is the 'i^{th}' person having 'j' number of traits, 'gbest' is the best person in the whole group, '$f(P_i)$' is the fitness of 'i^{th}' person and 'rand' is any random number generated in the range of (0,1).

$$Xij = c * P_{ij} + rand * (\text{gbest}(j) - P_{ij}) \tag{1}$$

In improving phase, a person interacts with the best person and tries to improve himself, as shown in Eq. (1), but percentage of improvement depends on 'rand'. It indicates that, as no person is same, so their rate of improvement will also be not same. But, an individual improvement not only depends on the good impact of best person, but also on how much he is introspecting himself, which is represented by 'c' in Eq. (1),. In SGO algorithm, it was considered to be '0.2', which is a constant.

If f (P_i) < f (P_r)

$$Xij = P_{i,j} + rand_1 * \left(P_{i,j} - P_{r,j}\right) + rand_2 * (\text{ gbest}(j) - P_{i,j}) \quad (2)$$

else

$$Xij = P_{i,j} + rand_1 * \left(P_{r,j} - P_{i,j}\right) + rand_2 * (\text{ gbest}(j) - P_{i,j}) \quad (3)$$

As, all individuals cannot have same rate of self-introspection, so in Sect. 3.2, we have tried to vary the self-introspection parameter according to various inertia weight functions. In acquiring phase, every individual interact with a random person of the group and tries to learn from them, as given in Eqs. (2) and (3). More clear description can be found in the base paper of SGO [36].

3.2 SGO with inertia weight strategies

In this section, some of the basic inertia weights adjustment strategies have been discussed along with their respective equations. These techniques were applied to vary the value of 'C' i.e. self-Introspection parameter of SGO and their performance was compared to the original SGO algorithm, where the value of 'C' is a constant i.e. 0.2. To study the effect, convergence graphs of SGO versus SGO with adjusted inertia weights, have been provided, after simulating those on Ackley and Griewank benchmark functions to find their optimal value, with Dimension 50. Ackley is a complex function, where the EAs find it harder to find the global optimum and mostly gets trapped in local optimum. Griewank is a simple function, where EAs easily converge to the global optimum. Taking both the variety of functions helped us in analyzing the performance of the algorithms with better clarity. The inertia weight strategies which performed better than SGO were simulated with other benchmark functions and their result has been provided in Sect. 5.1. Table 2 shows the values of the basic parameters used for simulation of SGO and SGO with various inertia weight strategies. In the description, 'I' represents the current iteration number, 'Imax' represents the value of maximum number of Iterations, 'C(I)' represents the value of 'C' at iteration 'I', 'Cmin' is the minimum value of 'C' and 'Cmax' is the maximum value of 'C'. Summary of the performance of these inertia weight strategies when implemented with SGO, has been provided in Table 3.

3.2.1 Linear decreasing inertia weight

Eberhart and Shi [17] proposed a linearly decreasing inertia weight technique, defined in Eq. (4), which was one of the most cited and popular technique, as it converged faster

Table 2 Values of the basic parameters

Parameter Name	Parameter Value
Population	50
Iteration(I)	50
Cmin	0.1
Cmax	0.9

Table 3 Summary of the performance of some inertia weight strategies when simulated with SGO on few numerical benchmark functions

Inertia Weight Strategy	*Performance*	*Remarks*
Linear Decreasing Inertia Weight [17] (SGOLDIW)	Poor Convergence and Optimality [Fig. 1]	Higher 'C' value in the beginning may not be suitable for SGO
Linear Increasing Inertia Weight [56] (SGOLIIW)	Convergence is good but Optimal solution not reached [Fig. 2]	Though lower 'C' value seems suitable but it cannot overcome local optimum
Sigmoid Decreasing Inertia Weight [32] (SGOSDIW)	Poor Convergence and Optimality [Fig. 3]	Higher 'C' value in the beginning may not be suitable for SGO
Sigmoid Increasing Inertia Weight [32] (SGOSIIW)	Very good Convergence and Optimality achieved, performs better for simple functions [Figs. 3 and 8]	Lower 'C' value seems suitable and sigmoidally increasing helps in reaching optimality
Oscillating Inertia Weight [28] (SGOOIW)	Poor Convergence and Optimality [Fig. 4]	Alternating change in 'C' value may not be suitable for SGO
Chaotic Inertia Weight [20] (SGOCIW)	Poor Convergence and Optimality [Fig. 5]	Randomly changing 'C' value may not be suitable for SGO
Adaptive Inertia Weight [38] (SGOAIW)	No adaptation of 'C' value [Fig. 6]	May be the adaptation formula not working for 'C'
Adaptive Chaotic Inertia Weight [5] (SGOACIW)	Adaptive 'C' values and good convergence and optimality, performs better for complex functions [Figs. 7 and 8]	Adaptation formula worked for 'C' but no effect of chaotic maps found
Sigmoid Adaptive Inertia Weight (Proposed) (SGOSAIW)	Adaptive 'C' values and better convergence and optimality, performs better for both simple and complex functions [Fig. 8]	Adaptation formula worked for 'C' and provided better convergence for complex functions. Sigmoidally increasing 'C' value helped in better convergence for simple functions

than the original PSO. Figure 1a, b shows the convergence graphs of applying the same to vary the values of 'C' in SGO and it clearly shows that SGO with linear decreasing inertia weight does not perform better. Though, it obtains the optimal result finally in both the cases, but convergence is too poor with respect to SGO. Figure 1c shows, how value of 'C' changes for SGO with linear decreasing inertia weight and for SGO it is constant. It clearly shows that, having greater 'C' value in the beginning, is making the convergence graph, convex in nature. Hence, greater 'w' value was suitable for PSO because 'w' signifies weight associated to inertia of the birds and initially speed of the birds should be greater and it should gradually decrease as they near the goal, but greater 'C' value may not be suitable for SGO because, 'C' signifies learning rate and initially learning rate of a person is less and as knowledge increases, learning rate also increases gradually.

$$C(I) = \frac{\mathrm{Im}\, ax - I}{\mathrm{Im}ax}(C\mathrm{max} - C\mathrm{min}) + C\mathrm{min} \tag{4}$$

3.2.2 Linear increasing inertia weight

Zheng et al. [56] proposed a Linear increasing inertia weight, whose equation is defined in Eq. (5). From the experimentation, it was found that PSO performed better for weight value ranging among 0.4 to 0.9. Hence in Eq. (5), weight increases from 0.4 onwards. But for SGO, 0.1 has been used as minimum. So, '0.4' was replaced with '0.1' in the simulation.

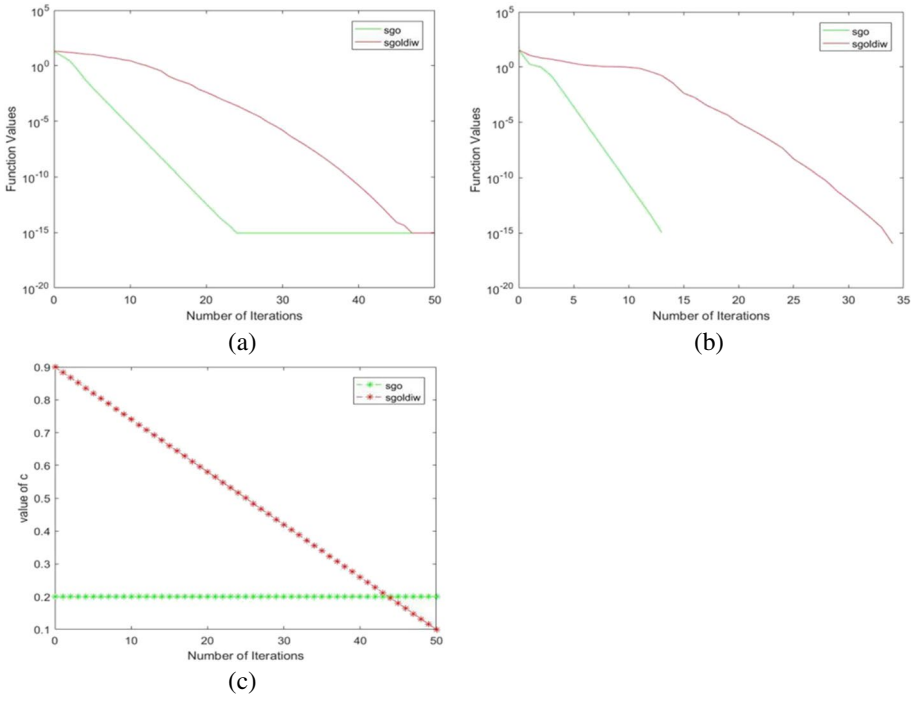


Fig. 1 (**a**) SGO versus SGO with linear decreasing inertia weight for Ackley (**b**) SGO versus SGO with linear decreasing inertia weight for Griewank (**c**) value of 'C' for SGO versus SGO with linear decreasing inertia weight

Figure 2 shows the comparative performance of SGO and SGO with linear increasing inertia weight. It is very clear from the graphs that, SGO with linear increasing inertia weight converges better, but in Ackley function, it could not reach the minimum [refer to Fig. 2a]. In both the functions, convergence graph is either linear or concave in nature, which is desirable and from Fig. 2c, it can be deduced that lower 'C' value in the initial iterations, gives better result. But, as the 'C' value increases, gradually, it may not be suitable for reaching the optimum and hence it gets trapped in local minima for functions like Ackley, whereas for functions like Griewank, it performs better in convergence as well as in reaching the optimum. To study more about the performance of SGO with linear increasing inertia weight (SGOLIIW), it was simulated with other benchmark functions and results are given in the Sect. 4.

$$C(I) = 0.5 \times \frac{I}{\mathrm{Im}ax} + 0.4 \tag{5}$$

3.2.3 Sigmoid increasing and decreasing inertia weight

Sigmoid increasing inertia weight and Sigmoid decreasing inertia weight were proposed by Malik et al. [32] for adjusting the weights in PSO and experimentally they found that, Sigmoid increasing inertia weight performed better. Eqs. (6) and (7) defines Sigmoid

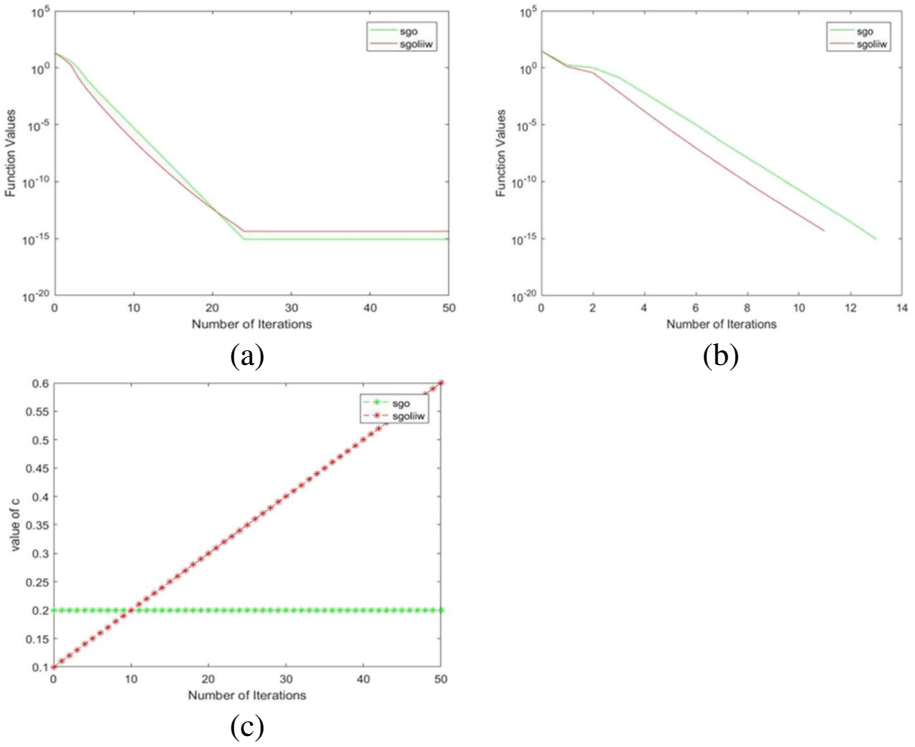


Fig. 2 (**a**) SGO versus SGO with linear increasing inertia weight for Ackley (**b**) SGO versus SGO with linear increasing inertia weight for Griewank (**c**) value of 'C' for SGO versus SGO with linear increasing inertia weight

decreasing inertia weight and Sigmoid increasing inertia weight respectively. They considered the value of 'w' to be ranging between 0.4 to 0.9 and n = 0.25, 0.5 and 0.75. We have considered the value of n=0.25. Value of 'u' is calculated as $u = 10^{\log(\mathrm{Im}\, ax-2)}$, which is used to adjust sharpness of the functions. Figure 3a, b shows convergence graph of SGO versus SGO with sigmoid decreasing inertia weight for Ackley and Griewank function, respectively and Fig. 3c, d shows convergence graph of SGO versus SGO with sigmoid increasing inertia weight for Ackley and Griewank function, respectively. For Ackley

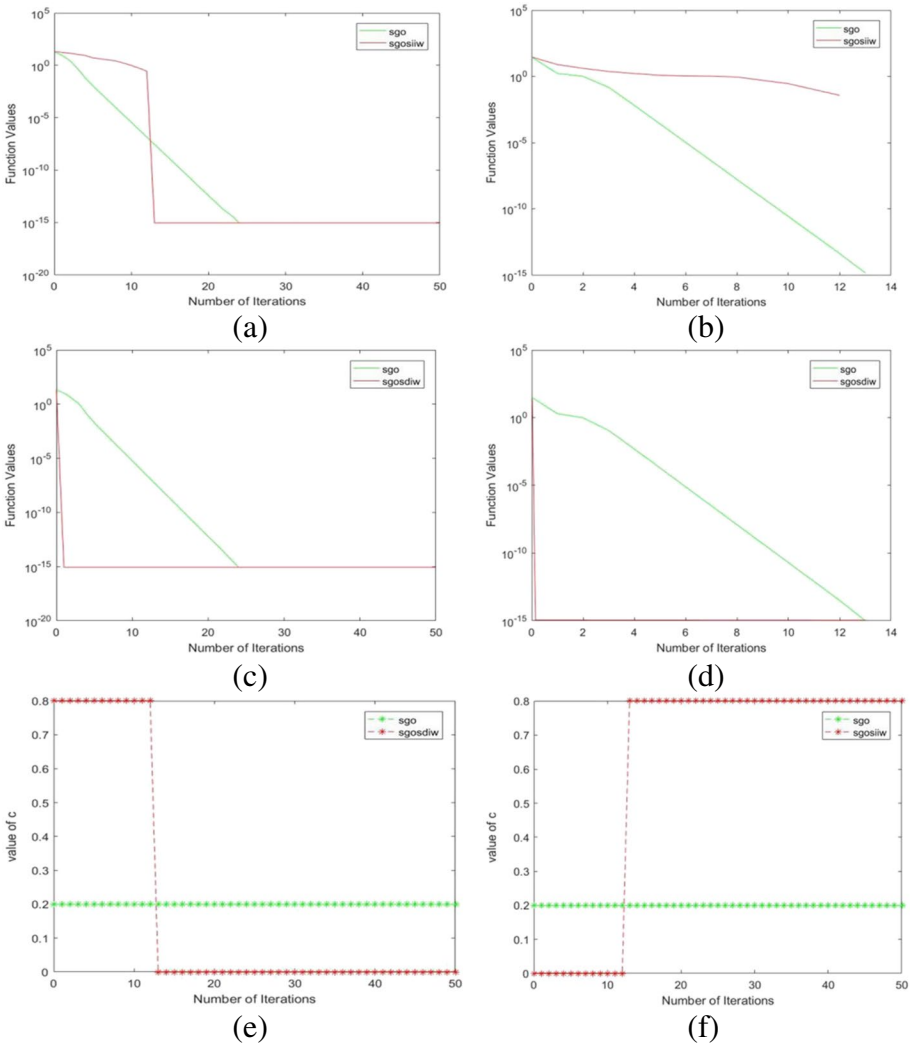


Fig. 3 (**a**) SGO versus SGO with sigmoid decreasing inertia weight for Ackley (**b**) SGO versus SGO with sigmoid decreasing inertia weight for Griewank (**c**) SGO versus SGO with sigmoid increasing inertia weight for Ackley (**d**) SGO versus SGO with sigmoid increasing inertia weight for Griewank (**e**) value of 'C' for SGO versus SGO with sigmoid decreasing inertia weight (**f**) value of 'C' for SGO versus SGO with sigmoid increasing inertia weight

function, though SGO with sigmoid decreasing inertia weight reaches optimal solution faster but for Griewank function, its convergence as well as reaching the optimum, is poor.

$$C(I) = \frac{C\max - C\min}{1 + e^{u(I - n\times \mathrm{Imax})}} \tag{6}$$

$$C(I) = \frac{C\max - C\min}{1 + e^{-u(I - n\times \mathrm{Imax})}} \tag{7}$$

Moreover, in Ackley, SGO's convergence was better for initial number of iterations. Observing Fig. 3e, it may be deduced that for initial iterations, as value of 'C' is greater, so performance was poor, but after some iterations, when value of 'C' lowered down, performance improved. Whereas, SGO with sigmoid increasing inertia weight (SGOSIIW) performs quite well in both Ackley and Griewank function, which can be observed from Fig. 3c, d and converges much faster than SGO. In Fig. 3f, it can be observed that lower 'C' value in initial iterations, may be the reason for faster convergence and higher 'C' value after some iterations, may have helped in reaching the optimal solution faster. Hence, it was applied to other benchmark functions, so that, it's overall performance could be observed and their results have been provided in the Sect. 5.1.

3.2.4 Oscillating inertia weight

Kentzoglanakis and Poole [28] proposed an oscillating inertia weight strategy for PSO, which gave better result for some of the function. Its definition is given in Eq. (8).

$$C(I) = \frac{C\min + C\max}{2} + \frac{C\max - C\min}{2}\cos\left(\frac{2\pi I}{T}\right) \tag{8}$$

$$T = (2 \times I)/(3 = 2k) \text{ and } I = 3 * Imax/4 \tag{9}$$

For implementing in SGO, the value of 'k' was chosen to be 7, as in the specified paper. The value of 'T' and 'I' are given in Eq. (9). Figure 4 shows the convergence graph of SGO versus SGO with oscillating inertia weight for Ackley and Griewank, and it is observed that SGO performed better than this variant. From Fig. 3c, it may be deduced that, alternating higher and lower values of 'C' may not be suitable for SGO algorithm.

3.2.5 Chaotic inertia weight

Feng et al. [20] proposed two chaotic strategies to adjust the inertia weight in PSO. They aggregated two popular techniques i.e. Liner decreasing inertia weight and Random inertia weight, with the logistic maps and proposed Chaotic decreasing inertia weight and Chaotic random inertia weight, whose equations are defined in Eqs. (10) and (11) respectively, where z is a random number in the interval of (0,1) and its logistic mapping is defined in Eq. (12). Figure 6a, b shows the convergence graph of SGO versus SGO with chaotic decreasing inertia weight and it can be inferred that SGO performs better. From Fig. 6g, it can be inferred that reason is same as in the case of Linear decreasing inertia weight [Fig. 2a, b]. Figure 6c, d shows the convergence graph of SGO versus SGO with chaotic random inertia weight and it can be observed that SGO performs well in this case too. From Fig. 6h, it can be observed that 'C' values produced are random and always greater than 0.5, hence chaotic maps may not be suitable for SGO. Feng et al. aggregated Logistic maps to linear decreasing inertia weight

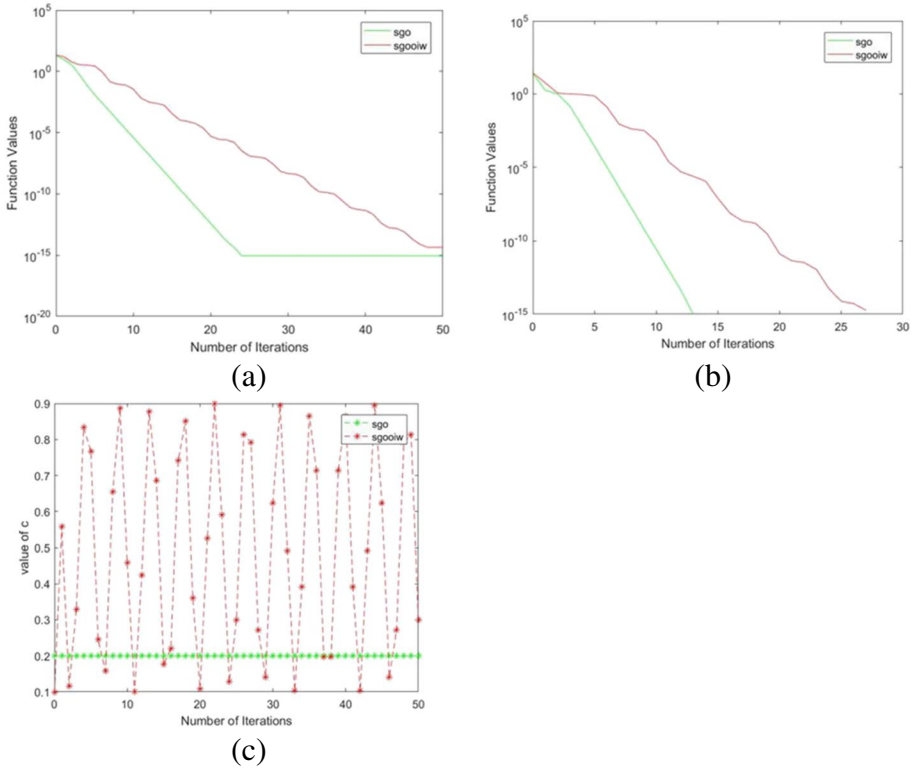


Fig. 4 (**a**) SGO versus SGO with oscillating inertia weight for Ackley (**b**) SGO versus SGO with oscillating inertia weight for Griewank (**c**) value of 'C' for SGO versus SGO with oscillating inertia weight

to enhance its performance, as it performed well for PSO [17]. But from the Fig. 1, it can be clearly deduced that, linear decreasing inertia weight did not perform well for SGO, rather linear increasing inertia weight performed better (Fig. 2). This gave way to the curiosity of finding out the performance of aggregated logistic maps and linear increasing inertia weight (Eq. (5)) for adjusting 'C' of SGO. We defined Chaotic increasing inertia weight, given in Eq. (13), by simply multiplying 'z' to the Eq. (5) and convergence graph of SGO versus SGO with chaotic increasing inertia weight is shown in Fig. 6e, f. From the graphs it can be inferred that, results are very much similar to Linear increasing inertia weight and 'C' values shown in Fig. 6i also depicts the same. So, it may be deduced that, chaotic maps do not affect the convergence of SGO significantly.

$$C(I) = \frac{\mathrm{Im}\, ax - I}{\mathrm{Im}ax}(C\max - C\min) + C\min \times z \tag{10}$$

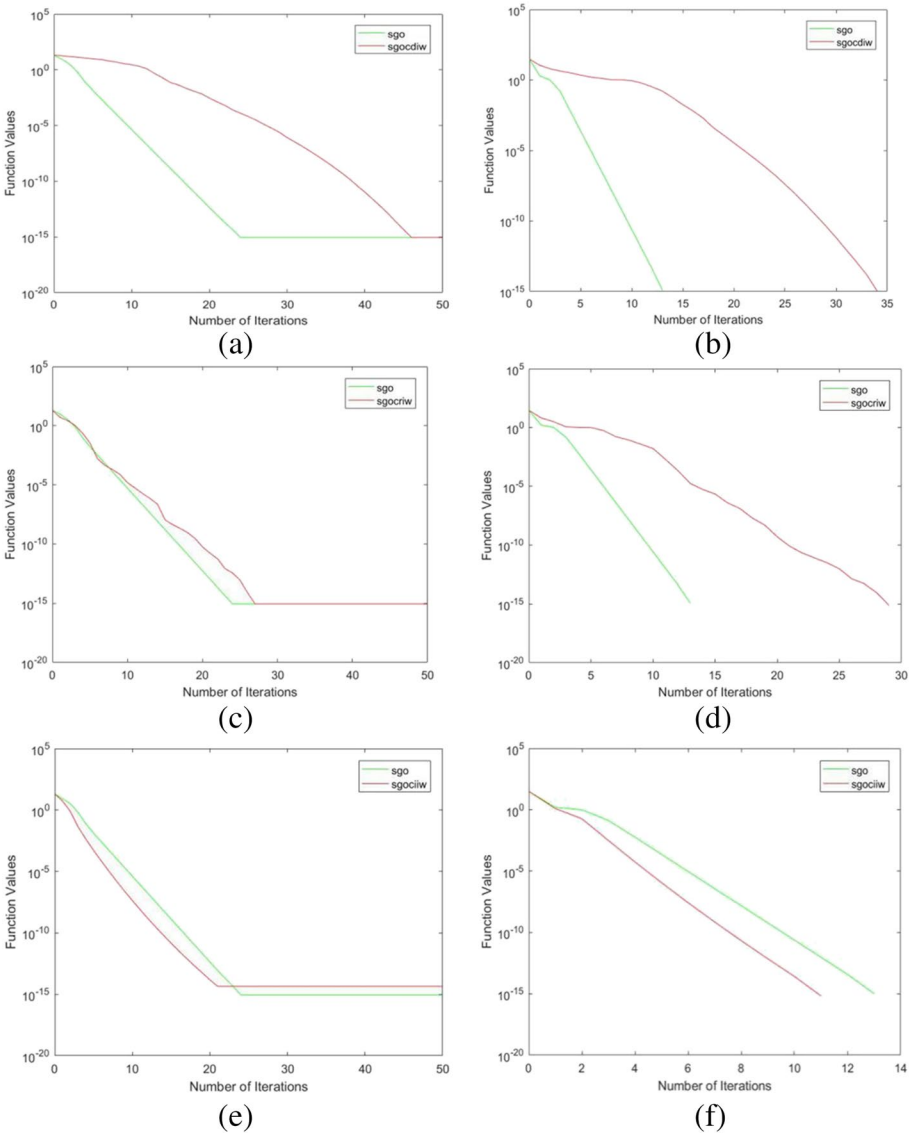
$$C(I) = 0.5 \times rand + 0.5 \times z \tag{11}$$

$$z = 4 \times z \times (1 - z) \tag{12}$$

$$C(I) = 0.5 \times \frac{I}{\mathrm{Im}ax} + 0.2 \times z \tag{13}$$

3.2.6 Adaptive inertia weight

Nickabadi et al. [38] proposed a weight adjustment strategy for PSO, which could adapt according to the success rate of swarms in each iteration. Success for a minimization problem, is defined by ' S ' in Eq. (14), where ' *pbest* ' signifies the local best of each particle 'i'.



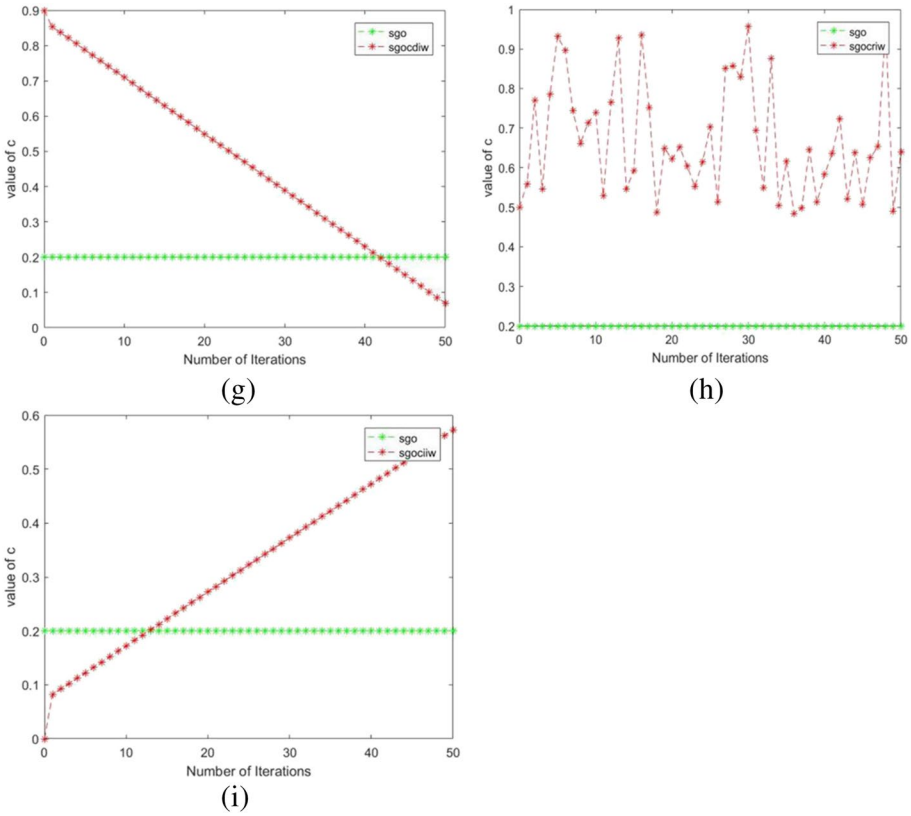


Fig. 5 (**a**) SGO versus SGO with chaotic decreasing inertia weight for Ackley (**b**) SGO versus SGO with chaotic decreasing inertia weight for Griewank (**c**) SGO versus SGO with chaotic random inertia weight for Ackley (**d**) SGO versus SGO with chaotic random inertia weight for Griewank (**e**) SGO versus SGO with chaotic increasing inertia weight for Ackley (**f**) SGO versus SGO with chaotic increasing inertia weight for Griewank (**g**) value of 'C' for SGO versus SGO with chaotic decreasing inertia weight (**h**) value of 'C' for SGO versus SGO with chaotic random inertia weight (**i**) value of 'C' for SGO versus SGO with chaotic increasing inertia weight

$$S(I) = \begin{cases} 1, \ fit(pbest(I) < fit(pbest(I-1) \\ 0, \ fit(pbest(I) < fit(pbest(I-1) \end{cases} \tag{14}$$

$$S(I) = \frac{\sum_{i-1}^{n} S(i)}{n} \tag{15}$$

$$C(I) = (C\max - C\min)R(I) + C\min \tag{16}$$

Success rate is defined by *'R'* in Eq. (15), where *'N'* is the number of particles. Success rate signifies the percentage of population, that gained betterment to their previous iteration. Basing on the success rate, weight would adapt itself, whose definition is given in Eq. (16). Figure 7a, b shows the convergence graph of SGO versus SGO with adaptive inertia weight, but it can be clearly deduced that SGO performance is better. The reason could be

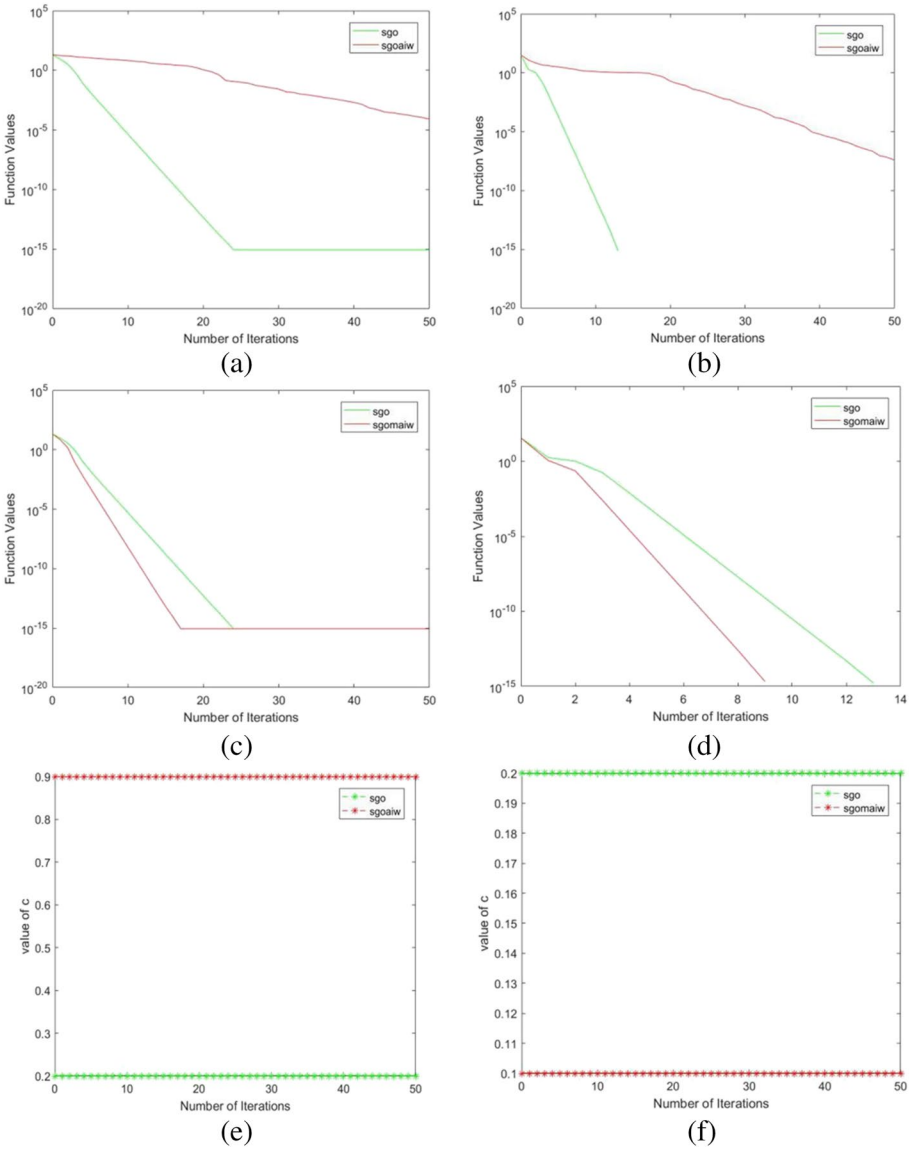


Fig. 6 (**a**) SGO versus SGO with adaptive inertia weight for Ackley (**b**) SGO versus SGO with adaptive inertia weight for Griewank (**c**) SGO versus SGO with modified adaptive inertia weight for Ackley (**d**) SGO versus SGO with modified adaptive inertia weight for Griewank (**e**) value of 'C' for SGO versus SGO with adaptive inertia weight (**f**) value of 'C' for SGO versus SGO with modified adaptive inertia weight

the linear function used in Eq. (16), was suitable for PSO, but may not be suitable for SGO, which is clearly shown in Fig. 7e, where value of 'C' always remained higher. To make it suitable for SGO, we modified the linear function as given in Eq. (17) and the results obtained were better than SGO, which is shown in Fig. 7c, d. The reason may be inferred from Fig. 7f, where 'C' values remain always at minimum. From, Fig. 7e, f, it can also be inferred that, Eqs. (17) and (18) failed to generate adaptive 'C' values for SGO, as the

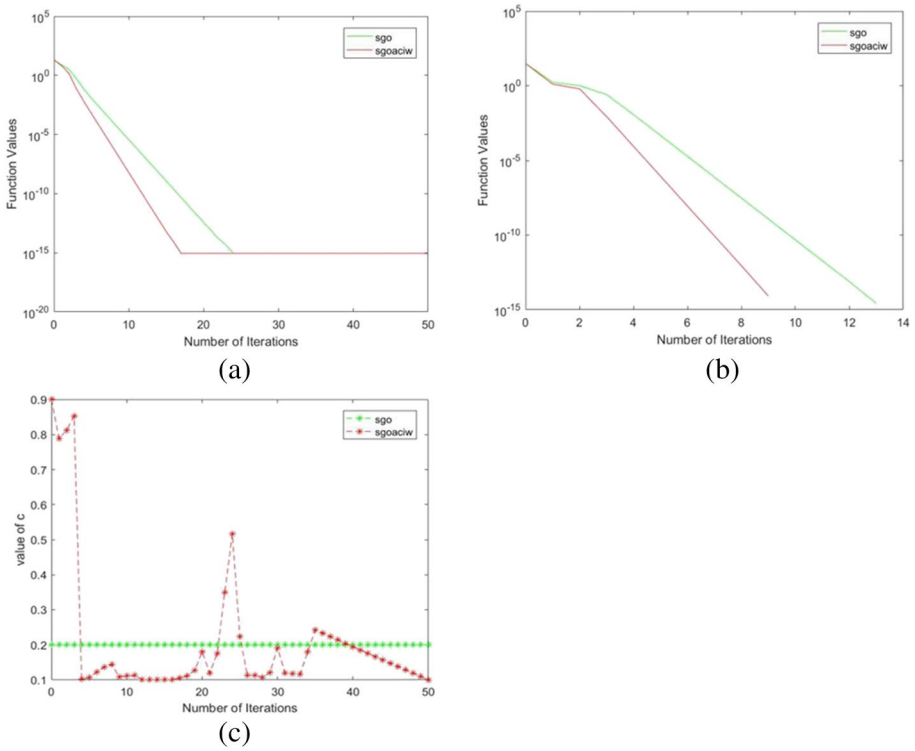


Fig. 7 (**a**) SGO versus SGO with adaptive chaotic inertia weight for Ackley (**b**) SGO versus SGO with adaptive chaotic inertia weight for Griewank (**c**) value of 'C' for SGO versus SGO with adaptive chaotic inertia weight

values generated, may be outside the permitted range and due to normalization, generated values remain constant.

$$C(I) = C\max - (C\max - C\min)R(I) \tag{17}$$

3.2.7 Adaptive chaotic inertia weight

Chen et al. [5], used another adaptive inertia weight to adjust the weight and used logistic maps to introduce chaotic sequence into iterations in Particle swarm optimization for antenna synthesis. Its definition is given in Eq. (18), where value of '*k*' is given in Eq. (19) and value of '$\eth x$' is given in Eq. (20), which is evaluated for every population '*i*'. '*D*' specifies the number of dimensions and '*Gbest*' specifies global best for iteration '*I*'.

$$C(I) = k \times \frac{C\max - C\min}{\max\{\partial X(i)\}} \partial X(i) + C\min \tag{18}$$

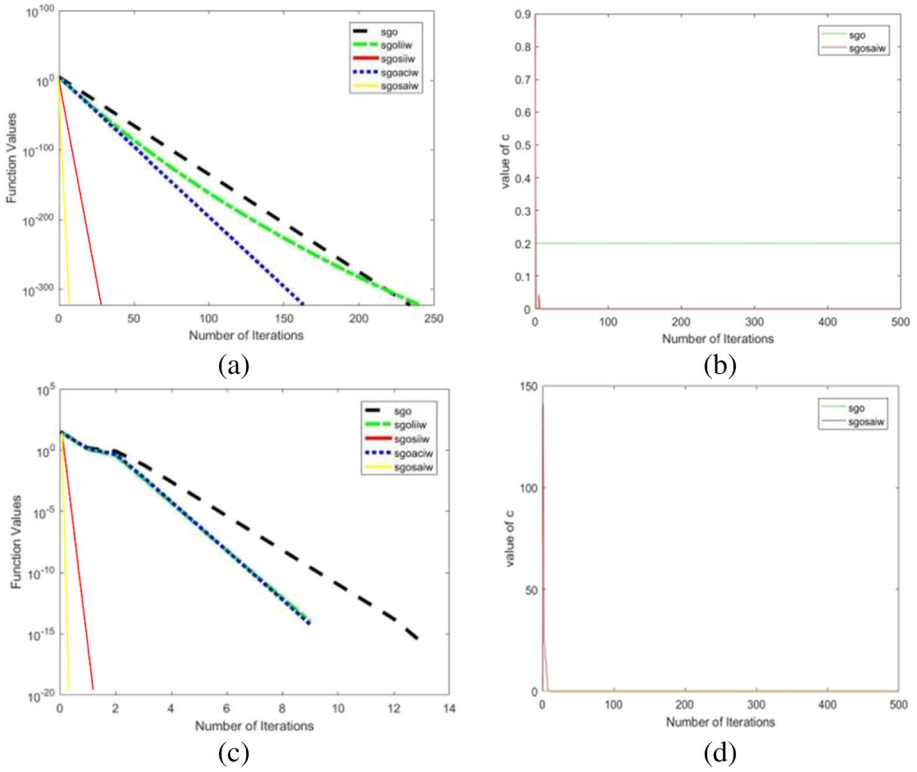
$$k = \frac{\mathrm{Im}ax - 1}{\mathrm{Im}ax} \tag{19}$$

$$\partial X(i) = \sqrt{\sum_{d-1}^{D} X_{id} - Gbest_d} \tag{20}$$

Authors have used logistic maps to alter the optimal solution in each iteration and replace it with one of the population. Figure 8a, b shows SGO versus SGO with adaptive chaotic inertia weight for Ackley and Griewank function respectively, from which it can be deduced that SGO with adaptive chaotic inertia weight performs better. Figure 8c shows adaptive 'C' values generated with each iterations and it can be inferred that Eq. (18) could successfully generate adaptive values for SGO. For the experimentation purpose, we removed the chaotic maps and simulated only adaptive inertia weight with SGO and it was found that, without chaotic maps also performance was same. It is the adaptive function which has significant effect on the algorithm. To study more about the performance of SGO with adaptive chaotic inertia weight (SGOACIW), it was simulated with other benchmark functions whose results are given in Sect. 5.1.

4 Sigmoid-adaptive inertia weight strategy and its effect on SGO

In Sect. 3.2, we have presented the simulation graphs of only two benchmark functions i.e. Ackley and Griewank, to simply segregate those inertia weight strategies, which performed better for SGO. In Sect. 5.1, the simulation results of all the better performing algorithms



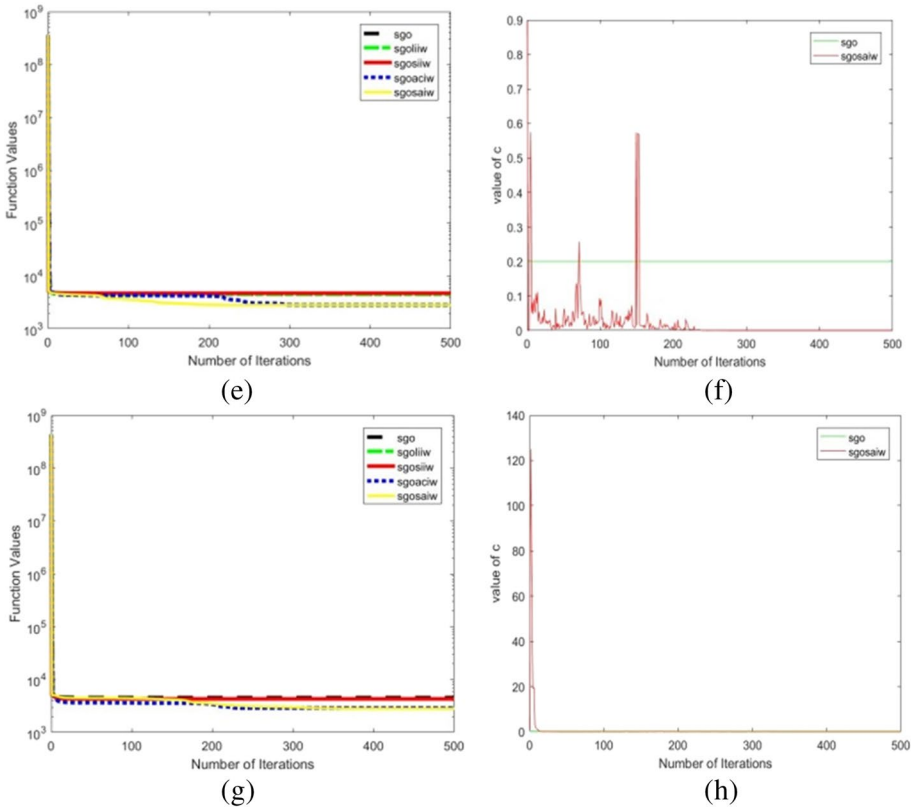


Fig. 8 Convergence graph of SGO, SGOLIIW, SGOSIIW, SGOACIW and SGOSAIW for (**a**) Sphere (**c**) Griewank (**e**) Rosebrock (**g**) Schwefel 2.26 and values of 'c' of SGO and SGOSAIW for (**b**) Sphere (**d**) Griewank (**f**) Rosebrock (**h**) Schwefel 2.26

has been shown, from which a very peculiar behavior can be noticed. For some algorithms, SGO with sigmoid increasing inertia weight (SGOSIIW) converges much faster whereas for some problems SGO with adaptive chaotic inertia weight (SGOACIW) performs better. To know the reason, we analyzed the nature of the functions and found that, for simple functions such as Sphere, Griewank etc., which are uni-modal and separable, SGOSIIW converged faster, though all the algorithms reached the minima, and for complex functions such as Rosenbrock and Schwefel, which are multimodal and non-separable, all algorithms got trapped in local minima, but SGOACIW gave more optimum value. Our proposed technique i.e. Sigmoid-adaptive inertia weight, carries properties of both *Sigmoid increasing inertia weight* (refer to Sect. 3.2.3) and *Adaptive inertia weight* (refer to Sect. 3.2.7), whose equation is given in Eq. (21), where value of 'k' is given in Eq. (19), value of '$\eth x$' is given in Eq. (20) and value of 'n' has been considered as 0.25. Sigmoid adaptive inertia weight has been applied to SGO (SGOSAIW), to vary the values of 'C' and study the effects on the performance of the algorithm. We are not using chaotic maps (as used in Sect. 3.2.7), as we found that, it is not that much effective for SGO. SGOSAIW, along with other SGO variants discussed in Sect. 3.2, were simulated upon two simple functions Sphere and Griewank as well as two complex functions Rosenbrock and Schwefe l2.26. Their comparative convergence graph and the generation of 'C' value for every iteration, has been shown in

Fig. 8. From Fig. 8a, c, e and g, it can be observed that SGOSAIW performs well for simple functions like Sphere and Griewank as well as for complex functions like Rosenbrock and Schwefel2.26. Thus, this inertia weight technique seems suitable for variety of real word problems. From Fig. 8b, d and f, it can be inferred that, for simple functions, values of 'C' need not have to adapt much, hence curve is smooth, but for complex functions more adaptations are required, so the corresponding graph is uneven.

$$C(I) = k \times \frac{Cmax - Cmin}{1 + e^{(\min\{\partial x\} - n \times \max\{\partial x\})}} \times \partial x(i) \tag{21}$$

Table 3 provides the summary of the performance of the discussed inertia weight strategies when simulated with SGO, on some of the numerical benchmark functions with the inferences drawn from each simulation.

5 Simulation and results

5.1 Performance evaluation using benchmark functions

All the simulations were carried out in Matlab 2016a on the system having Intel Core i7 2.67 GHz processor and 8 GB RAM. To know the comparative performance of SGOSAIW to other algorithms, SGOACIW, was compared to Adaptive inertia weight based teaching-learning-based optimization (ATLBO) proposed by Shukla et al. [51] and Adaptive inertia weight Bat algorithm with Sugeno-function fuzzy search (ASF-BA), proposed by Rauf et al. [48]. Shukla et al. have used an eight benchmark functions suite, given in Table 4, to compare ATLBO with GA, various variants of PSO, various variants of ABC, DE, TLBO and Rank Inertia Weight TLBO. Similarly, Rauf et al. have used a nineteen benchmark function suite, given in Table 6, to compare the performance of ASF-BA against BA and PSO.

In Sect. 4, SGOSAIW was simulated on only four benchmark functions. To get a clearer picture, it was simulated upon the eight benchmark function suite [51] and nineteen benchmark function suite [48], along with other inertia weights based SGO discussed in Sect. 3.2. Table 5 provides the simulation results of the algorithms, upon eight benchmark function suite and its convergence graph is shown in Fig. 9. From Fig. 9a, c, e and g, it can be observed that, SGOSIIW performs better than SGOACIW, because it has better convergence capability, but, in Fig. 9b, d, f and h, SGOACIW performs better than SGOSIIW, because it is capable of finding more optimal solution by adaptive approach. Both of the scenarios are different and demands different approach to find an optimal solution. By observing the convergence graphs and the obtained results, it can be clearly deduced that SGOSAIW performs as better as SGOSIIW and SGOACIW in all the cases. It means, SGOSAIW excels in convergence ability as well as in finding optimal solution using adaptive approach, and hence could be used for variety of problems.

Table 6 provides the comparative performance of PSO, ABC, DE, TLBO, RaIWTLBO, ATLBO and SGOSAIW. For fair evaluation, SGOSAIW was simulated using same basic input parameters as mentioned in [51] for eight benchmark mark function suite i.e. number of population was taken as 20, dimension was set to 30 and iterations was set to 1000. From Table 6, it can be observed that, SGOSAIW performed better than the other algorithms, for all the cases except F2, where ATLBO performed better.

Table 4 Eight benchmark function suite used for performance analysis of ATLBO and SGOSAIW

Function Name	Function Equation	min	Input range
Sphere (F1)	$F(x) = \sum_{i-1}^{D} x(i)^2$	0	[-100,100]
Rosenbrock (F2)	$F(x) = \sum_{i=1}^{D} \left[100\left(x(i)^2 - x(i) + 1\right)^2 + (1 - x(i)^2\right]$	0	[-30,30]
Griewank (F3)	$F(x) = \frac{1}{4000}\sum_{i=1}^{D} x(i)^2 - \prod_{i=1}^{D}\cos\left(\frac{x(i)}{\sqrt{i}}\right) + 1$	0	[-600,600]
Weierstrass (F4)	$F(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{kmax}\left[a^k\cos\left(2\pi b^k(x(i)+0.5)\right)\right]\right) - D\sum_{i=1}^{D}\left(\sum_{k=0}^{kmax}\left[a^k\cos\left(2\pi b^k 0.5\right)\right]\right)$ $a = 0.3, b = 3, kmax = 20$	0	[-0.5,0.5]
Rastrigin (F5)	$F(x) = \sum_{i=1}^{D}\left(x(i)^2 - 10\cos(2\pi x(i)) + 10\right)$	0	[-5.12.5.12]
Ackley (F6)	$F(x) = 20 + e - 20\exp\left(-0.2\sqrt{\sum_{i=1}^{D} x(i)^2/D}\right) - \exp\left(\sum_{i=1}^{D}\cos(2\pi x(i))/n\right)$	0	[-32,32]
NCRastrigin (F7)	$F(x) = \sum_{i=1}^{D}\left(y_i^2 - 10\cos\left(2\pi y_i\right) + 10\right)$ and $y_i = \{\begin{matrix} x(i), \lvert x(i)\rvert < 0.5 \\ round(2x(i)/2, \lvert x(i)\rvert \geq 0.5)\end{matrix}\}$	0	[-5.12,5.12]
Schwefel2.26 (F8)	$\mathrm{F(x)} = 418.9829d - \sum_{i=1}^{n} x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	0	[-500,500]

SGOSAIW along with other SGO variants discussed in Sect. 3.2, were also simulated upon nineteen benchmark function suite, used by Rauf et al, which is given in Table 7. Result of the simulation is provided in Table 8 and convergence graphs of all the functions are shown in Fig. 10. A similar observation can be made from these functions too as it was in Fig. 9. From Fig. 10a–s, it can be observed that SGOSIIW performs much better than SGO, SGOLIIW and SGOACIW. Even in some cases, as shown in Fig. 10c, i, SGO performs better than SGOLIIW and SGOACIW. It means one technique may show different behavior for different types of problems. But, SGOACIW also performs better in some cases too, as shown in Fig. 10f, k and q, where reaching minima is difficult, as functions are quite complex. But in all the cases, SGOSAIW, performs better, irrespective of the nature of the function and same can be deduced from Table 8 also. So, it again proves that SGOSAIW can perform better in varied conditions and is suitable for application to different types of problem.

Table 9 provides the comparative performance of PSO, BA, ASF-BA and SGOSAIW. The values are taken from [48] and for fair evaluation SGOSAIW was simulated with same parameters as mentioned by Rauf et al. Number of population and number of iterations was set to 50 and 5000 respectively and dimension was set to 50 and each process was repeated for 100 times. The algorithms were run for 30 times and their mean and standard deviation were considered as the final results. From Table 9, it can be observed that, SGOACIW provides better result for all function except G17, where ASF-BA performs better.

Overall performance of SGOSAIW was found to be better than other algorithms in simulation of both the benchmark functions suite. In many cases zero or nearer to zero

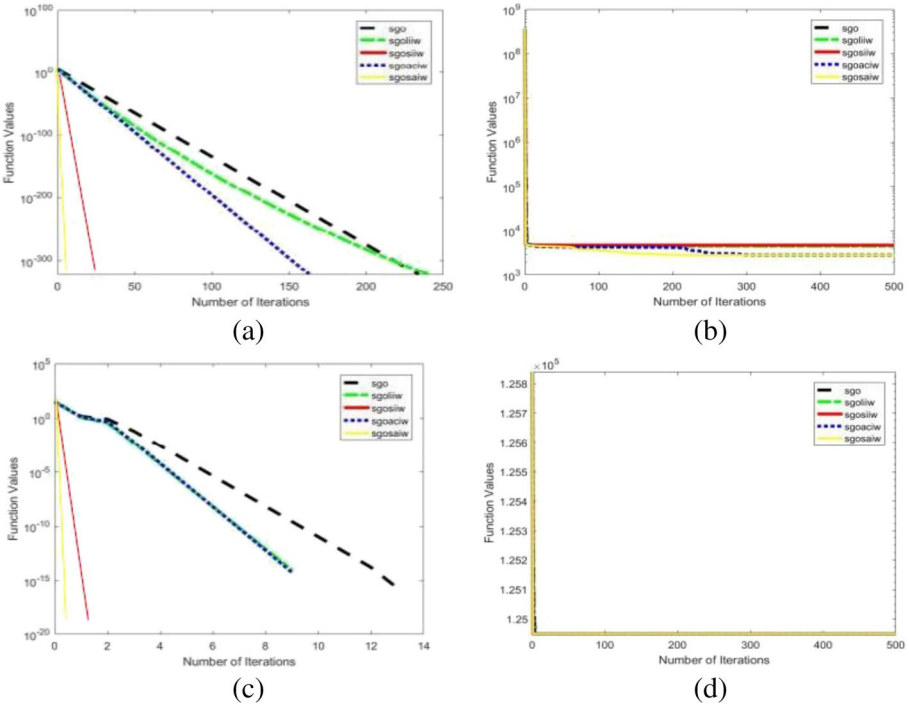
Table 5 Performance analysis of SGO, SGOLIIW, SGOSIIW, SGOACIW and SGOSAIW with eight benchmark functions suite given in Table 4

Functions	SGO (Mean,SD)	SGOLIIW (Mean,SD)	SGOSIIW (Mean,SD)	SGOACIW (Mean,SD)	SGOSAIW (Mean,SD)
F1	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
F2	2.8318E-01, 2.12E-01	2.7613E-01, 1.414E-01	2.8221E-01, 7.07E-01	2.1132E-01, 7.78E-01	**2.0037E-01, 7.23E-01**
F3	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
F4	3.91E-05, 1.23E-03	4.19E-05, 1.31E-03	4.71E-05, 1.22E-03	1.036E-05, 1.15E-04	**0,0**
F5	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
F6	2.28E-16, 6.44E-19	2.07E-16, 5.74E-19	1.58E-16, 5.21E-19	1.4878E-16, 5.51E-19	**1.0353E-16, 5.12E-19**
F7	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
F8	0.08, 3.5E-03	0.08, 2.3E-03	0.06, 2.87E-03	0.04, 3.43E-03	**0.02, 3.9E-03**

standard deviation has been obtained which shows that the proposed algorithm has good stability i.e. same results are obtained in most of the runs. From the analysis, it is very clear that, irrespective of any inertia weight adjustment technique, SGO performs better with lower 'C' value or gradually increasing 'C' value. The reason may be found from the basic principle of SGO which is based on social behavior of individuals and how they put impact on each other or we can say how their traits change by interacting among themselves, which is already discussed in Sect. 3.1. Self-introspection parameter 'C' signifies the amount of self-introspection a person does to know his inner traits, to judge himself, so that he can improve. Learning cannot happen, if an individual don't testify himself, to realize his own merits and demerits. So, lower 'C' value or gradually increasing 'C' value indicates that, at the beginning of the process, a person's self-introspection percentage is very less, but gradually it increases as a person improves himself. Thus, at later stage of the process, a person improves himself as well as his ability to judge himself also increases.

Sigmoid adaptive inertia weight means, the value increases in sigmoidal nature as well as it takes into account, its own previous value, current value and the best value achieved so far, and changes adaptively. It signifies, not all the problems can be treated same. So for different types of problems, problem solving approach must be different. Thus, a person, along with gradually increasing his self-introspection, should also change himself according to the surrounding conditions or adapt himself, and then only optimal solution can be ensured, in most of the situations.

For the functions, on which SGOSAIW, did not perform well, it can be deduced that, tuning the parameters, may perform well for some type of problems, but not all. An



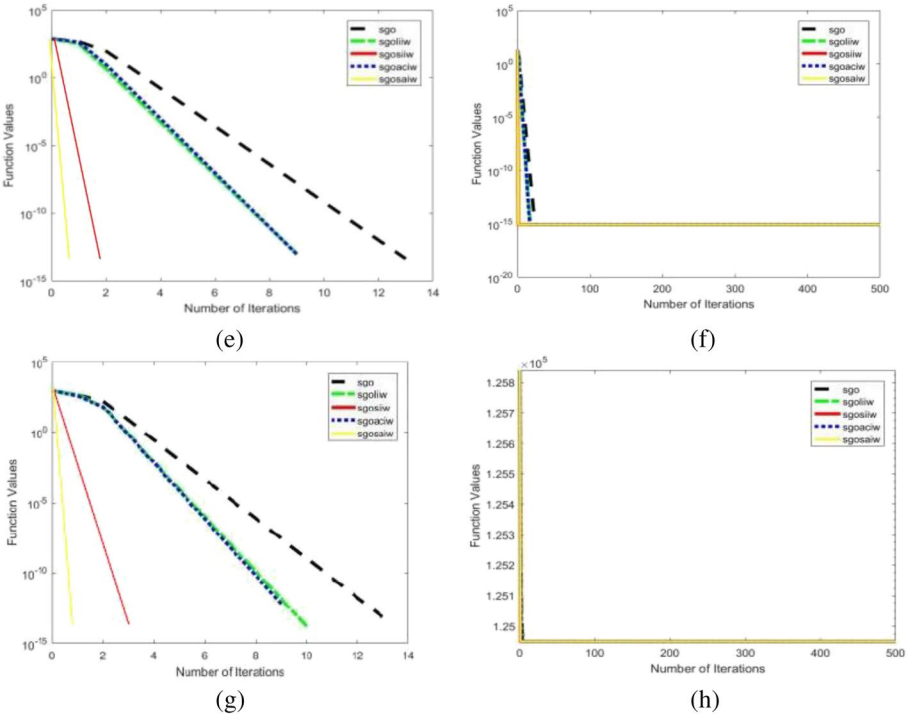


Fig. 9 Convergence graph of SGO, SGOLIIW, SGOSIIW, SGOACIW and SGOSAIW for (**a**) F1 (**b**) F2 (**c**) F3 (**d**) F4 (**e**) F5 (**f**) F6 (**g**) F7 (**h**) F8

evolutionary algorithm has many more aspects, on which developmental works can be done such as population initialization, operator hybridization etc., which can improve the performance of the algorithm. In SGOSAIW, we have not taken any such aspects into the consideration, which can apparently be treated as a limitation of the algorithm.

For validation purpose, number of population, number of iterations and dimension of the benchmark functions was kept same. Simulating SGOSAIW by varying these parameters will help in analyzing impact of these algorithm independent parameters on the proposed methodology. Table 10 shows the mean best values got by simulation of SGOSAIW with varied parameter values for Sphere, Griewank, Rosenbrock and Schwefel 2.26 numerical benchmark functions, whose comparative performance analysis is given in Sect. 4. Simulation has been carried out for dimension –20 and 50, for population size –20, 50 and 100 and for number of iterations –50 and 100. It is observed that, for simple functions such as Sphere and Griewank, the optimal value is reached with a few numbers of population and iterations. For complex functions such as Rosenbrock and Schwefel 2.26, optimality is directly proportional to the increasing number of population and iterations, but this condition holds true up to a certain value of fitness function. After a certain point no change in the function value will happen irrespective of population size and number of iterations. So, it can be inferred that algorithm independent parameters like population size and number iterations helps in convergence but for reaching the optimal solution, algorithm dependent parameters and algorithmic operations play a major role.

Table 6 Performance analysis of PSO, ABC, DE, TLBO, RaIWTLBO, ATLBO and SGOSAIW on eight benchmark functions suite given in Table 4

Function	PSO (Mean,SD)	ABC (Mean,SD)	DE (Mean,SD)	TLBO (Mean,SD)	RWTLBO(Mean,SD)	ATLBO (Mean,SD)	SGOSAIW (Mean,SD)
F1	6.12E-96, 1.32E-95	1.02E-45, 6.346E-46	8.52E-139, 4.66E-138	**0,0**	**0,0**	**0,0**	**0,0**
F2	1.90,0.16	6.86,0.25	1.65,1.34	0.18,0.74	0.81,1.54	**8.59E-02, 1.22E-01**	2.0037E-01, 7.23E-01
F3	4.24E-04, 5.90E-04	0.07, 0.0020	5.31E-04, 9.14E-04	4.46E-04, 4.96E-04	5.77E-06, 7.01E-07	1.30E-06, 1.58E-08	**0,0**
F4	**0,0**	5.58E-20, 6.16E-20	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
F5	2.96E-264,0	2.41E-04, 1.13E-04	2.81E-26, 1.54E-25	4.89E-197,0	1.40E-219,0	**0,0**	**0,0**
F6	0.1305, 8.08E-17	0.4301,0.06	**0,0**	0.13, 3.81E-08	2.35E-15, 2.53E-15	1.12E-16, 1.71E-20	**1.0353E-16, 5.12E-19**
F7	0.77,0	4.17,0.17	**0,0**	1.55, 1.37E-11	**0,0**	**0,0**	**0,0**
F8	16.91,0	22.44,1.51	**0,0**	0.09,0.54	**0,0**	**0,0**	0.02, 3.9E-03

Table 7 Nineteen benchmark function used for performance analysis of ASF-BA and SGOSAIW

Function Name	Function Equation	min	Input range
Sphere (G1)	$F(x) = \sum_{i-1}^{D} x(i)^2$	0	[-100,100]
Axis parallel hyper- Ellipsoid (G2)	$F(x) = \sum_{i-1}^{D} i.x(i)^2$	0	[-5.12,5.12]
Schwefel 2.21 (G3)	$F(x) = \frac{\max\lvert x_i\rvert}{0pt1<i<D}$	0	[-100,100]
Powell Singular 2 (G4)	$F(x) = \sum_{i-1}^{D}(x_{i-1} + 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_i - 2x_{i+1})^4$	0	[-4, 5]
Rastrigin (G5)	$F(x) = \sum_{i-1}^{D}(x(i)^2 - 10\cos(2\pi x(i)) + 10)$	0	[-5.12.5.12]
Ackley (G6)	$F(x) = 20 + e - 20\exp(-0.2\sum_{i-1}^{D} x(i)^2/D) - \exp(\sum_{i-1}^{D}(\cos 2\pi x(i))/n)$	0	[-32,32]
Sum of Different Power (G7)	$F(x) = \sum_{i-1}^{D}\lvert x_i\rvert^{i+1}$	0	[-1, 1]
Moved axis parallel hyper- Ellipsoid (G8)	$F(x) = \sum_{i-1}^{D} 5i.x_i^2$	0	[-5.12,5.12]
Schwefel 2.22 (G9)	$F(x) = \sum_{i-1}^{D}\lvert x_i\rvert + \prod_{i-1}^{D}\lvert x_i\rvert$	0	[-100, 100]
Salomon Function (G10)	$F(x) = 1 - \cos\left(2\pi\sqrt{\sum_{i-1}^{D} x_i^2}\right) + 0.1\left(\sqrt{\sum_{i-1}^{D} x_1^2}\right)$	0	[-100,100]
Dixon & Price Function (G11)	$F(x) = (x_i-)^2 + \sum_{i-1}^{D} i.(2x_1^2 - x_{i-1})^2$	$\left(f\left(2\left(\frac{2^i-2}{2^i}\right)\right)\right)$	[-10, 10]
Zakharov Function (G12)	$F(x) = \sum_{i-1}^{D} x_i^2 + \left(\frac{1}{2}\sum_{i-1}^{D} i.x_i\right)^2 + \left(\frac{1}{2}\sum_{i-1}^{D} i.x_i\right)^4$	0	[-5, 10]

Table 7 (continued)

Function Name	Function Equation	min	Input range
Schumar Steiglitz (G13)	$F(x) = \sum_{i-1}^{D} x(i)^4$	0	[-5.12,5.12]
Schwefel 2.23 (G14)	$F(x) = \sum_{i-1}^{D} x_i^{10}$	0	[-10, 10]
Schwefel 1.2 (G15)	$F(x) = \sum_{i-1}^{D} \left(\sum_{j-1}^{i} x_j \right)^2$	0	[-100,100]
Rotated hyper- Ellipsoid (G16)	$F(x) = \sum_{i-1}^{D} \left(\sum_{j-1}^{i} x_j \right)^2$	0	[-65.536, 65.536]
Noisy Function (G17)	$F(x) = \sum_{i-1}^{D} (i+1) x_i^2 + rand[0,1]$	0	[-1.28, 1.28]
Sum Squares Function (G18)	$F(x) = \sum_{i-1}^{D} i.x(i)^2$	0	[-10,10]
Cigar (G19)	$F(x) = x_i^2 + 10^6 \sum_{i-2}^{D} x_i^2$	0	[-100,100]

Table 8 Performance analysis of SGO, SGOLIIW, SGOSIIW, SGOACIW and SGOSAIW with nineteen benchmark functions suite given in Table 7

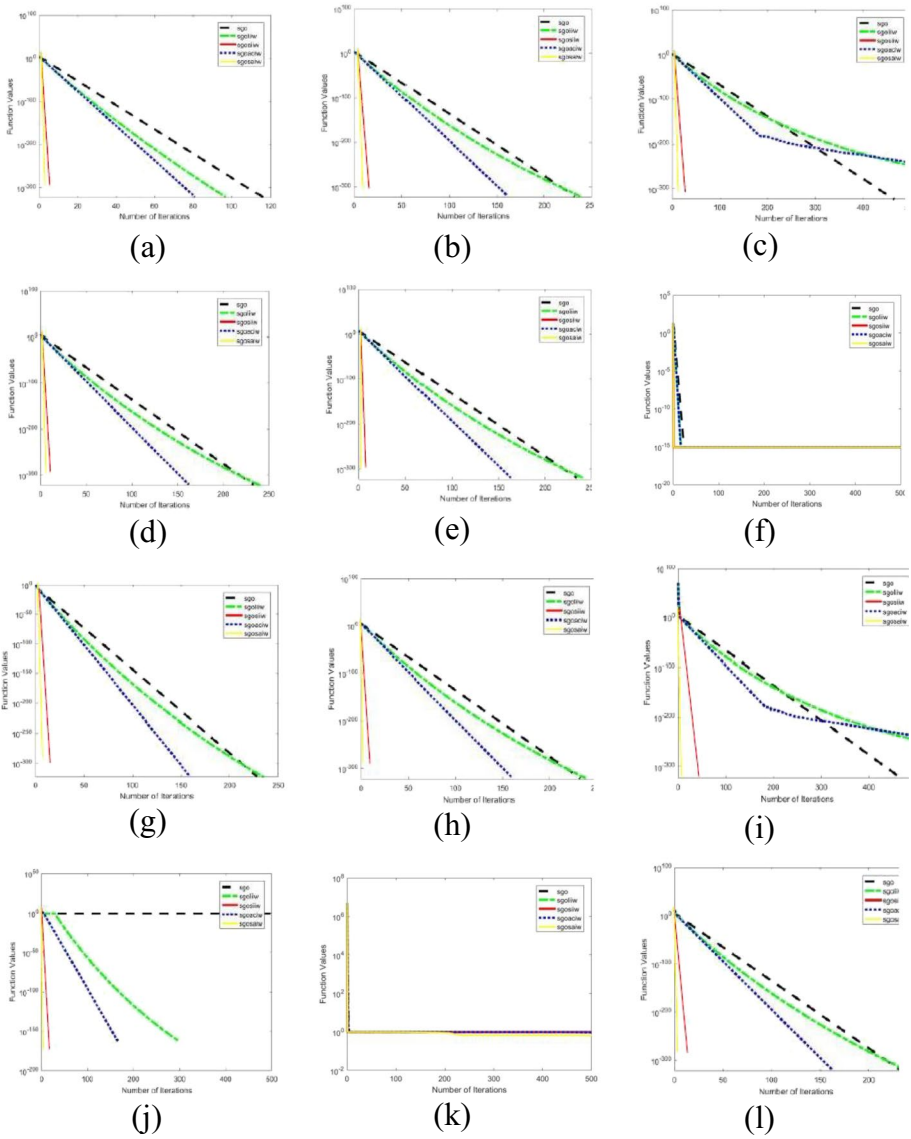
Functions	SGO (Mean,SD)	SGOLIIW (Mean,SD)	SGOSIIW (Mean,SD)	SGOACIW (Mean,SD)	SGOSAIW (Mean,SD)
G1	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G2	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G3	**0,0**	1.1653E-248, 7.43E-12	**0,0**	2.1771e-242, 6.53E-12	**0,0**
G4	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G5	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G6	2.28E-16, 6.44E-19	2.07E-16, 5.74E-19	1.58E-16, 5.21E-19	1.4878E-16, 5.13E-19	**1.0353E-16, 4.12E-19**
G7	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G8	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G9	**0,0**	2.8763E-247, 5.82E-18	**0,0**	1.4398E-239, 2.33E-20	**0,0**
G10	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G11	0.9798 2.78E-15	0.9889, 3.21E-15	0.9853, 3.04E-15	0.9470, 2.54E-17	**0.6667, 6.48E-20**
G12	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G13	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G14	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G15	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G16	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G17	17.2823, 2.76E-14	16.4012, 2.36E-14	17.0706. 2.52E-14	16.7371, 2.41E-14	**15.9599, 1.03E-15**
G18	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
G19	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**

Comparing 'C' of SGO to 'w' of PSO, it can be said that 'w' is one of the parameter in PSO which added weight to the initial velocity whereas 'C' is the self- introspection parameter in SGO which impacts its improving phase. Both are not same, but change in both parameters affect the performance of the respective algorithms. Inertia weight techniques are parameter tuning or parameter changing techniques according to a specific function. That's why it can be observed that, while one technique works for 'w' in PSO, it fails for 'C' in SGO. It is completely algorithm dependent. As the nature of algorithms is different, so they behave differently and hence it requires extensive analysis to draw some conclusions.

5.2 Performance evaluation using engineering design problems

In this section, we simulated SGO and SGOSAIW on some of the engineering design problems because these problems integrate constraints satisfiability along with searching the optimal result. Most of the real world problems can be mapped to these Constrained Optimization (CO) problems. In most of the papers, they use several design problems such as mechanical design problems, but from the single domain only. In this

section, we have used Three bar truss problem [46] and Cantilever beam problem [46] from mechanical engineering domain and two Process synthesis and design problems [30] from chemical engineering domain, whose problem formulation and description has been provided in Appendix. Every problem domain has its unique constraints, thus using wide variety of problem domains helps in studying the nature of the algorithms better. All the four problems considered here are minimization problems. Three bar truss problem has two variables to optimize constrained to three conditions. Cantilever beam problem has five variables to optimize constrained to one condition. Process synthesis problem has two variables to optimize constrained to two conditions and



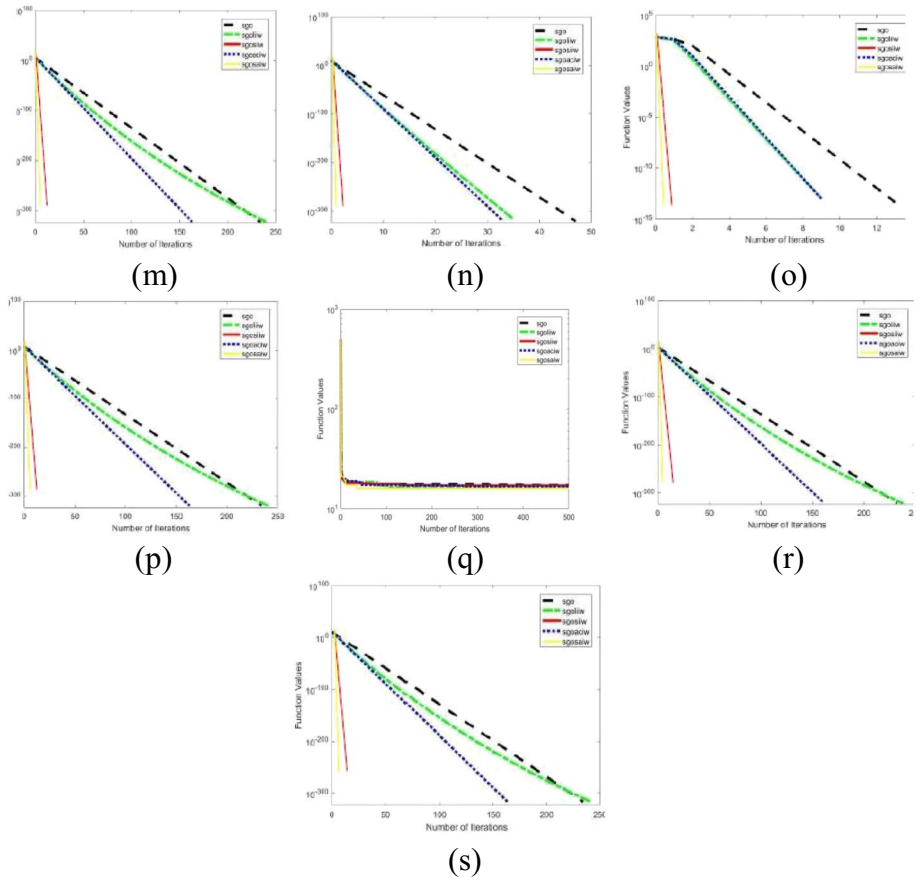


Fig. 10 Convergence graph of SGO, SGOLIIW, SGOSIIW, SGOACIW and SGOSAIW for (**a**) G1 (**b**) G2 (**c**) G3 (**d**) G4 (**e**) G5 (f) G6 (**g**) G7 (**h**) G8 (**i**) G9 (**j**) G10 (**k**) G11 (**l**) G12 (**m**) G13 (**n**) G14 (**o**) G15 (**p**) G16 (**q**) G17 (**r**) G18 (**s**) G19

Process synthesis and design problem has three variables to optimize constrained to two conditions. Results obtained for Three bar truss and Cantilever beam have been compared to SAMP Rao algorithms [46] and results obtained for the two Process synthesis design problems have been compared to Improved Unified Differential Algorithms (IUDE) [52], ὲ - constraint Matrix adaptation gradient approximation evolution strategy(ὲ-MAgES) [23] and improved LSHADE with ὲ- constraint (iLSHADE ὲ) [18], whose results are taken from Kumar et al. [30]. Population size was considered to be 500 and run for 5000 iterations. Each algorithm was simulated for 50 individual runs and the best value and standard deviation obtained have been considered as the result. Comparative performance analysis for the engineering design problems have been given in Tables 11 and 12, from which it can be observed that SGOSAIW gives satisfactory results. In Table 11, it can be observed that, for Three bar truss problem and Cantilever beam problem, proposed method was able to find the optimal value but with better

Table 9 Performance analysis of PSO, BA, ASF-BA and SGOACIW on nineteen benchmark functions suite given in Table 7

Functions	PSO	BA	ASF-BA	SGOSAIW
G1	4.68E-28, 2.52E-27	1.51E-15, 5.55E-16	**0,0**	**0,0**
G2	3.67E+01,1.98E+02	1.77E-05, 3.50E-06	**0,0**	**0,0**
G3	4.23E-02, 2.26E-01	6.08E+01, 5.39E+00	**0,0**	**0,0**
G4	1.00E+05, 2.91E-11	1.20E+04, 8.30E+03	**0,0**	**0,0**
G5	3.74E+00, 2.01E+01	1.54E+06, 1.67E+06	**0,0**	**0,0**
G6	2.72E+00, 8.88E-16	3.56E+00, 3.74E-01	4.44E-16,3.93E-16	**1.0353E-16, 5.12E-19**
G7	3.77E-144, 2.03E-143	1.16E-03, 3.35E-04	**0,0**	**0,0**
G8	1.84E+02, 9.88E+02	8.85E-05, 1.75E-05	**0,0**	**0,0**
G9	4.23E-02, 2.26E-01	1.44E+63, 7.15E+63	**0,0**	**0,0**
G10	9.66E-02, 2.55E-01	2.20E+01, 2.50E+00	**0,0**	**0,0**
G11	1.05E+00, 2.07E+00	3.25E+06, 8.40E+05	9.09E-01, 1.08E-01	**6.667E-01, 6.48E-20**
G12	3.74E-08, 2.02E-07	7.82E+04, 2.70E+05	**0,0**	**0,0**
G13	8.74E-01, 4.71E+00,	1.28E-07, 2.19E-08	**0,0**	**0,0**
G14	9.83E-143, 5.29E-142	3.89E+02, 1.61E+03	**0,0**	**0,0**
G15	-2.41E+05, 2.09E+04	3.37E+02, 4.63E+01	**0,0**	**0,0**
G16	2.25E-40, 1.21E-39	1.47E+09, 6.73E+08	**0,0**	**0,0**
G17	7.21E-01, 3.86E+00	4.33E-01, 1.24E-01	**7.89E-05, 1.52E-04**	15.9599, 1.03E-15
G18	1.62E-04, 8.71E-04	9.84E+00,2.27E+01	**0,0**	**0,0**
G19	2.34E-07, 1.26E-06	3.19E+05, 9.74E+04	1.37E-23, 7.14E-23	**0,0**

standard deviation value. In Table 12, for both the problems though SGOSAIW performs better than SGO, but not better than the other algorithms. The reason is, some large scale optimization problems require inclusion of extra parameters as in [23, 52] and [18] to get the most optimal value. As in the proposed methodology no such parameters have been included, so it lacks in reaching the optimum.

Table 10 Mean best values of SGOSAIW with different values of Dimensions, Iterations and Population size for Sphere, Griewank, Rosenbrock and Schwefel 2.26 numerical benchmark functions

Functions	*Dimensions*	*Iterations-50 Pop-20,50,100*			*Iterations-100 Pop-20,50,100*		
Sphere	20	0	0	0	0	0	0
	50	0	0	0	0	0	0
Griewank	20	0	0	0	0	0	0
	50	0	0	0	0	0	0
Rosenbrock	20	3.8E-01	3.7E-01	3.5E-01	3.6E-01	3.4E-01	3.2E-01
	50	4.9E-01	4.7E-01	4.4E-01	4.7E-01	4.3E-01	4.1E-01
Schwefel 2.26	20	0.08	0.09	0.085	0.065	0.045	0.03
	50	0.06	0.05	0.05	0.04	0.033	0.02

Table 11 Performance analysis of SAMPRao algorithms, SGO and SGOACIW on Three bar truss and Cantilever beam problem

Problem	SAMPRao1 (Best)	SAMPRao2 (Best)	SAMPRao3 (Best)	SGO (Best, SD)	SGOSAIW(Best, SD)
Three bar truss	263.896	263.896	263.896	263.67E+00, 1.6753E-06	263.45E+00, 2.8725E-08
Cantilever Beam	1.339957	1.339957	1.339957	1.339E+00, 4.56E-12	1.339E+00, 6.78E-14

Table 12 Performance analysis of IUDE, ϵ MAgES, iLSHADEϵ, SGO and SGOACIW on Process synthesis design problems

Problem	IUDE (Best,SD)	ϵMAgES (Best,SD)	iLSHADEϵ (Best,SD)	SGO (Best,SD)	SGOSAIW (Best,SD)
Process Synthesis	2.00E+00, 6.41E-17	2.00E+00, 1.52E-01	2.00E+00, 00E+00	2.00E+00, 2.54E-01	2.00E+00, 1.86E-01
Process Synthesis and Design	2.56E+00, 1.36E-15	2.56E+00, 2.70E-01	2.56E+00, 1.46E-07	2.6246E+00, 2.22E-02	2.5834E+00, 1.73E-02

6 Conclusion

Sigmoid adaptive inertia weight based SGO was tested with several benchmark functions and it was found performing better than other algorithms, for most of the functions, but not all such as 'Noisy' benchmark function and 'Schwefel 2.26' benchmark function. Mainly, the proposed inertia weight strategy was found suitably adapting to most of the problems, irrespective of their complexities, unlike other inertia weight strategies. Though parameter tuning can affect the performance of an evolutionary algorithm greatly, but there are other aspects as well, which affects the performance too. In this paper, we have only focused on parameter tuning using different inertia weights. Modifying other aspects and analyzing their effects on the performance of the algorithm, may be taken as one of the future works. Moreover, application of inertia weight strategies have been clearly explained with detailed analysis and insights, which are very easy to follow and can be helpful to the researchers who are new to this field. A thorough analysis on changing behavior of algorithms, on application of different inertia weight strategies, was done and reasonable explanations were provided. Proposed inertia weight strategy can be applied to several recent evolutionary algorithms for adaptation to various real-world problems and new paradigms may be explored.

For engineering design problems of mechanical engineering domain and chemical engineering domain, performance of SGOSAIW was satisfactory, but here also a scope of improvement could be found. SGOSAIW is a generalized algorithm and it has not been designed specifically for constrained optimization problems. So, a specialized version could also be designed with SGOSAIW for dealing especially with complex constrained optimization problems.

Appendix

Problem-1 Three Bar Truss

Minimize: $f(x, y) = \left(2\sqrt{2}x + y\right)L$

Subject to :

$$g1(x, y) = \left(\sqrt{2}x + y\right)P - \sigma(\sqrt{2}x^2 + 2xy) \leq 0$$

$$g2(x, y) = yP - \sigma(\sqrt{2}x^2 + 2xy) \leq 0$$

$$g3(x, y) = P - \sigma(\sqrt{2}y + x) \leq 0$$

Where :

$$0 \leq x, y \leq 1,$$

$$L = 100\text{cm},\ P = 2\text{KN/cm}^2, \sigma = 2\text{KN/cm}^2$$

Problem-2 Cantilever Beam

Minimize: $f(a, b, c, d, e) = 0.0624(a + b + c + d + e)$

Subject to : $g(a, b, c, d, e) = \frac{61}{a^3} + \frac{37}{b^3} + \frac{19}{c^3} + \frac{7}{d^3} + \frac{1}{e^3} \leq 1$

Where : $0.01 \leq a, b, c, d, e \leq 100$

Problem-3 Process Synthesis Problem

Minimize : $f(x, y) = x + 2xy$

Subject to : $g1(x, y) = 1.25 - x^2 - y \leq 0$

$$g2(x, y) = x + y - 1.6 \leq 0$$

Where : $0 \leq x \leq 1.6,\ y \in \{0,1\}$

Problem-4 Process Synthesis and Design Problem

Minimize : $f(a, b, c) = 2a + b - c$

Subject to : $g1(a, b) = a - 2e^{-b} = 0$

$$g2(a, b, c) = c + b - a \leq 0$$

Where : $0.5 \leq a, b \leq 1.4,\ c \in \{0,1\}$

Declarations

Conflict of interest Authors have no conflict of interest

References

1. Azqandi MS, Delavar M, Arjmand M (2020) An enhanced time evolutionary optimization for solving engineering design problems. Eng Comput 36(2):763–81
2. Bansal JC, Singh PK, Saraswat M, Verma A, Jadon SS, Abraham A (2011) Inertia weight strategies in particle swarm optimization. In 2011 Third world congress on nature and biologically inspired computing 633–640. IEEE
3. Chauhan P, Deep K, Pant M (2013) Novel inertia weight strategies for particle swarm optimization. Memetic Comput 5(3):229–51
4. Chen H, Xu Y, Wang M, Zhao X (2019) A balanced whale optimization algorithm for constrained engineering design problems. Appl Math Model 1(71):45–59
5. Chen ZR, Guan KK, Tong MS (2019) An Improved Adaptive Chaotic Particle Swarm Optimization Algorithm for Antenna Synthesis. In 2019 Photonics & Electromagnetics Research Symposium-Fall (PIERS-Fall) 207–210. IEEE
6. Chen M, Zhong Y, Wang L (2019) An Improved Pigeon-Inspired Optimization Combining Adaptive Inertia Weight with a One-Dimension Modification Mechanism. In International Conference on Bio-Inspired Computing: Theories and Applications 177–192. Springer, Singapore
7. Chou JS, Ngo NT (2017) Modified firefly algorithm for multidimensional optimization in structural design problems. Struct Multidiscip Optim 55(6):2013–28
8. Coello CA, Lamont GB, Van Veldhuizen DA (2007) Evolutionary algorithms for solving multi-objective problems. Springer, New York
9. Das S, Saha P, Satapathy SC, Jena JJ (2020) Social group optimization algorithm for civil engineering structural health monitoring. Eng Optim 3:1–20
10. de Paula Garcia R, de Lima BS, de Castro Lemonge AC, Jacob BP (2017) A rank-based constraint handling technique for engineering design optimization problems solved by genetic algorithms. Comput Struct 15(187):77–87
11. Dey N, Rajinikanth V, Ashour AS, Tavares JM (2018) Social group optimization supported segmentation and evaluation of skin melanoma images. Symmetry 10(2):51
12. Dey N, Rajinikanth V, Shi F, Tavares JM, Moraru L, Karthik KA, Lin H, Kamalanand K, Emmanuel C (2019) Social-Group-Optimization based tumor evaluation tool for clinical brain MRI of Flair/diffusion-weighted modality. Biocybern Biomed Eng 39(3):843–56
13. Dhiman G, Kaur A (2017) Spotted hyena optimizer for solving engineering design problems. In 2017 International Conference on Machine Learning and Data Science (MLDS) 114–119. IEEE
14. Dorigo M, Di Caro G (1999) Ant colony optimization: a new meta-heuristic. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406) 2:1470–1477. IEEE
15. Duan H, Qiao P (2014) Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. International Journal of Intelligent Computing and Cybernetics
16. Eberhart R, Kennedy J (1995) Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks 4:1942–1948). Citeseer
17. Eberhart RC, Shi Y (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512) 1:84–88. IEEE
18. Fan Z, Fang Y, Li W, Yuan Y, Wang Z, Bian X (2018) LSHADE44 with an Improved ϵ - Constraint-Handling Method for Solving Constrained Single-Objective Optimization Problems. In 2018 IEEE Congress on Evolutionary Computation (CEC) 1–8. IEEE
19. Fang J, Zheng H, Liu J, Zhao J, Zhang Y, Wang K (2018) A transformer fault diagnosis model using an optimal hybrid dissolved gas analysis features subset with improved social group optimization-support vector machine classifier. Energies 11(8):1922
20. Feng Y, Teng GF, Wang AX, Yao YM (2007) Chaotic inertia weight in particle swarm optimization. In 2nd International Conference on Innovative Computing, Informatio and Control (ICICIC 2007) 475–475. IEEE
21. Gandomi AH, Yang XS, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29(1):17–35
22. Harrison KR, Engelbrecht AP, Ombuki-Berman BM (2016) Inertia weight control strategies for particle swarm optimization. Swarm Intell 10(4):267–305
23. Hellwig M, Beyer HG (2018) A matrix adaptation evolution strategy for constrained real-parameter optimization. In 2018 IEEE Congress on Evolutionary Computation (CEC) 1–8. IEEE
24. Hu H, Bai Y, Xu T (2017) Improved whale optimization algorithms based on inertia weights and theirs applications. International Journal of Circuits, Systems and Signal Processing 11:12–26

25. Huang X, Li C, Chen H, An D (2019) Task scheduling in cloud computing using particle swarm optimization with time varying inertia weight strategies. Cluster Comput 9:1–1
26. Imrana M, Hashima R, Abd Khalidb NE (2013) An overview of particle swarm optimization variants. Procedia Eng 53:491–6
27. Karaboga D (2010) Artificial bee colony algorithm. Scholarpedia 5(3):6915
28. Kentzoglanakis K, Poole M (2009) Particle swarm optimization with an oscillating inertia weight. In Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation 1749–1750
29. Kiani AT, Nadeem MF, Ahmed A, Sajjad IA, Raza A, Khan IA (2020) Chaotic Inertia Weight Particle Swarm Optimization (CIWPSO): An Efficient Technique for Solar Cell Parameter Estimation. In 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET) 1–6. IEEE
30. Kumar A, Wu G, Ali MZ, Mallipeddi R, Suganthan PN, Das S (2020) A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. Swarm and Evol Comput 12:100693
31. Liu H, Zhang XW, Tu LP (2020) A modified particle swarm optimization using adaptive strategy. Exp Syst Appl 4:113353
32. Malik RF, Rahman TA, Hashim SZ, Ngah R (2007) New particle swarm optimizer with sigmoid increasing inertia weight. Int J Comput Sci Secur 1(2):35–44
33. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 1(95):51–67
34. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 1(69):46–61
35. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Adv Eng Softw 1(114):163–91
36. Naik A, Satapathy SC, Ashour AS, Dey N (2018) Social group optimization for global optimization of multimodal functions and data clustering problems. Neural Comput Appl 30(1):271–87
37. Naik A, Satapathy SC, Abraham A (2020) Modified Social Group Optimization-a meta-heuristic algorithm to solve short-term hydrothermal scheduling. Appl Soft Comput 16:106524
38. Nickabadi A, Ebadzadeh MM, Safabakhsh R (2011) A novel particle swarm optimization algorithm with adaptive inertia weight. Appl Soft Comput 11(4):3658–70
39. Olivas F, Valdez F, Castillo O, Gonzalez CI, Martinez G, Melin P (2017) Ant colony optimization with dynamic parameter adaptation based on interval type-2 fuzzy logic systems. Appl Soft Comput 1(53):74–87
40. Orouskhani M, Mansouri M, Teshnehlab M (2011) Average-inertia weighted cat swarm optimization. Int Conf Swarm Intell 12. Springer, Berlin, Heidelberg 321–328
41. Pawan YV, Prakash KB (2020) Impact of Inertia Weight and Cognitive and Social Constants in Obtaining Best Mean Fitness Value for PSO. In Soft Computing for Problem Solving 197–206. Springer, Singapore
42. Praveen SP, Rao KT, Janakiramaiah B (2018) Effective allocation of resources and task scheduling in cloud environment using social group optimization. Arab J Sci Eng 43(8):4265–72
43. Rajinikanth V, Satapathy SC (2018) Segmentation of ischemic stroke lesion in brain MRI based on social group optimization and Fuzzy-Tsallis entropy. Arab J Sci Eng 43(8):4365–78
44. Rani KA, Hoon WF, Abd Malek MF, Affendi NA, Mohamed L, Saudin N, Ali A, Neoh SC (2012) Modified cuckoo search algorithm in weighted sum optimization for linear antenna array synthesis. In 2012 IEEE symposium on wireless technology and applications (ISWTA) 210–215. IEEE
45. Rao RV, Kalyankar VD (2011) Parameters optimization of advanced machining processes using TLBO algorithm. EPPM, Singapore 20(20):21–31
46. Rao RV, Pawar RB (2020) Self-adaptive multi-population Rao algorithms for engineering design optimization. Appl Artif Intell 34(3):187–250
47. Rathore A, Sharma H (2017) Review on inertia weight strategies for particle swarm optimization. In Proceedings of 6th International Conference on Soft Computing for Problem Solving 76–86. Springer, Singapore
48. Rauf HT, Malik S, Shoaib U, Irfan MN, Lali MI (2020) Adaptive inertia weight Bat algorithm with Sugeno-Function fuzzy search. Appl Soft Comput 90:106159
49. Satapathy S, Naik A (2016) Social group optimization (SGO): a new population evolutionary optimization technique. Complex Intell Syst 2(3):173–203
50. Singh SB, Singh N, Hachimi H (2019) Inertia Constant strategy on Mean Grey Wolf Optimizer Algorithm for Optimization Functions. In 2019 5th International Conference on Optimization and Applications (ICOA) 1–7. IEEE
51. Shukla AK, Singh P, Vardhan M (2020) An adaptive inertia weight teaching-learning-based optimization algorithm and its applications. Appli Math Model 1(77):309–26

52. Trivedi A, Srinivasan D, Biswas N (2018) An improved unified differential evolution algorithm for constrained optimization problems. In Proceedings of 2018 IEEE Congress on Evolutionary Computation 1–10. IEEE
53. Yang XS (2010) A new metaheuristic bat-inspired algorithm. InNature inspired cooperative strategies for optimization (NICSO 2010). Springer, Berlin, Heidelberg 65–74
54. Yang XS (2008) Firefly algorithm: Nature-Inspired Metaheuristic Algorithms. Luniver Press
55. Yue X, Zhang H (2020) Modified hybrid bat algorithm with genetic crossover operation and smart inertia weight for multilevel image segmentation. Appl Soft Comput 90:106157
56. Zheng YL, Ma LH, Zhang LY, Qian JX (2003) On the convergence analysis and parameter selection in particle swarm optimization. In Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 03EX693) 3:1802–1807. IEEE