Sādhanā Vol. 42, No. 7, July 2017, pp. 1103–1121
DOI 10.1007/s12046-017-0674-0

Adaptive switching gravitational search algorithm: an attempt to improve diversity of gravitational search algorithm through its iteration strategy

NOR AZLINA AB AZIZ[1,2,*], ZUWAIRIE IBRAHIM[3], MARIZAN MUBIN[1,*] and SHAHDAN SUDIN[4]

[1]Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia
[2]Faculty of Engineering and Technology, Multimedia University, 75450 Melaka, Malaysia
[3]Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia
[4]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia
e-mail: azlina.aziz@mmu.edu.my; zuwairie@ump.edu.my; marizan@um.edu.my; shahdan@fke.utm.my

MS received 26 November 2015; revised 26 December 2016; accepted 29 December 2016

Abstract. An adaptive gravitational search algorithm (GSA) that switches between synchronous and asynchronous update is presented in this work. The proposed adaptive switching synchronous–asynchronous GSA (ASw-GSA) improves GSA through manipulation of its iteration strategy. The iteration strategy is switched from synchronous to asynchronous update and vice versa. The switching is conducted so that the population is adaptively switched between convergence and divergence. Synchronous update allows convergence, while switching to asynchronous update causes disruption to the population's convergence. The ASw-GSA agents switch their iteration strategy when the best found solution is not improved after a period of time. The period is based on a switching threshold. The threshold determines how soon is the switching, and also the frequency of switching in ASw-GSA. ASw-GSA has been comprehensively evaluated based on CEC2014's benchmark functions. The effect of the switching threshold has been studied and it is found that, in comparison with multiple and early switches, one-time switching towards the end of the search is better and substantially enhances the performance of ASw-GSA. The proposed ASw-GSA is also compared to original GSA, particle swarm optimization (PSO), genetic algorithm (GA), bat-inspired algorithm (BA) and grey wolf optimizer (GWO). The statistical analysis results show that ASw-GSA performs significantly better than GA and BA and as well as PSO, the original GSA and GWO.12

Keywords. Asynchronous; diversity; gravitational search algorithm; iteration strategy; synchronous.

1. Introduction

Nature has inspired many optimization algorithms, including the gravitational search algorithm (GSA). The GSA proposed by Rashedi *et al* (2009) is a memoryless metaheuristic algorithm that is inspired by Newton's law of gravitation and second law of motion. In GSA, the search for optimal solution is conducted by a population of agents. Each agent has its own mass. The agents move and search for the optimal solution guided by attraction force. The attraction force is exerted by the agents themselves towards each other. The strength of the force is proportional to agents masses and inversely proportional to their acceleration. An agent's mass is dependent on the quality of the solution proposed by the agent. The better the solutions, the bigger the agent's mass. Therefore, the highest pulling force in the entire population is exerted by the population's best performer.

GSA is proposed as a single objective optimization algorithm. However, with some modifications GSA has successfully been applied in various types of optimization problems, such as multi-objective [1–3], multimodal problems [4] and binary optimization problems [5–7].

GSA is found to be superior to some well-established optimization algorithms [8], such as central force optimization (CFO) [9], genetic algorithm (GA) [10] and particle swarm optimization (PSO) [11]. The main advantage of GSA is its simplicity; it requires only two parameters tuning compared with other algorithms such as PSO and bat algorithm (BA) [12].

However, GSA algorithm has a tendency to converge prematurely [1, 13, 14]. Premature convergence is often

*For correspondence

caused by unfavourable traps at local optima. The traps cause the population to settle with a non-optimal solution and lead to poor performance. Due to the effect of gravitational force absorption by the agents towards each others, it is difficult for agents of original GSA to escape from premature convergence [15–18]. Premature convergence can be overcome through good control of exploration and exploitation. Exploration is important to ensure that the search space is effectively explored and sub-area with good solution is identified. Exploration is achieved through diversification of agents. On the other hand, agents' convergence allows exploitation. Exploitation helps an algorithm to perform local search and fine-tune the solutions found. Many works had been conducted to find mechanisms to avoid and escape premature convergence in GSA. The mechanisms can be categorized as introduction of new parameters to control the convergence [19–23], redistribution or disruption of the agents' convergence so that diversity is increased [15, 16] and also hybridization of several concepts [13, 14, 17, 18, 24].

For example, new parameters that are derived from Lyapunov stability analysis have been introduced to GSA [19]. These parameters are introduced so that exploration and exploitation can be improved and better performance can be achieved. Fuzzy systems have been used in fine-tuning the parameters of optimization algorithms so that better performance is achieved [25–27]. In GSA, fuzzy systems have been used to tune GSA's gravitational constant, epsilon and alpha [20–23]. The tuning of these parameters allows the exploration and exploitation to be adaptively adjusted based on the condition of the population. Introduction of a new operation to suit the problem to be optimized has also been observed [28].

Redistribution and disruption cause disturbance and diversity in population-based metaheuristics. In PSO and ant colony optimization (ACO), this feature is used to improve their performance [29–34]. A similar method is used in GSA with disruption operation [15]. The disruption preserves diversity and the distance of the masses is used to determine when to disrupt the agents positions. GSA with a momentum operator (GSAM), which was inspired by the concept of elastic collision, was introduced in [16]. The agents in GSAM move in opposite directions when collision occurs, thus providing diversity to the population.

In recent years, quantum mechanics has been adopted as an approach to improve optimization algorithms [35–37]. It has also been hybridized with GSA in several works [13, 24]. Quantum mechanics provides diversity to the population, and prevents premature convergence. The operations of GA, like mutation and crossover, are popular methods used in improving local search and escaping premature convergence in many optimizers including GSA [18, 38–42]. The concept of cooperation, which has been successfully used in GA, is adopted in cooperative PSO for radial basis function network [43]. Meanwhile, hybrid chaotic perturbation and memory of the population have been proposed by Jiang *et al* [14]. The hybridization has been proposed so that premature convergence can be avoided. The concept of memory in PSO has also been adopted for improvement of GA [44]. A GSA that uses social intelligence similar to that in PSO, adaptive *gBest*-guided GSA, was introduced in [17]. This modification helps in exploration and exploitation of the population.

Osman and Laporte, defined metaheuristic as *"an iterative generation process that guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions"* [45]. Hence, iteration strategy is one of the fundamental aspects of a metaheuristic algorithm, especially in population-based metaheuristics like GSA. It determines how the agents are updated with respect to each other. Two types of iteration strategies are available: synchronous and asynchronous updates. The general pseudo-codes of synchronous and asynchronous updates of population-based metaheuristics algorithms are shown in Algorithm 1 and Algorithm 2, respectively. In synchronous update, the agents are updated as a group. Therefore, two separate loops can be seen in Algorithm 1. The first loop is for the population's fitness evaluation, while the second loop is for generation of next solutions by the entire population. In asynchronous update the update process is conducted individually. Thus, evaluation and generation of a new solution is conducted within a single loop, agent by agent.

The original GSA is a synchronous update algorithm, S-GSA. Its agents are found to converge at a rapid rate [46]. In asynchronous GSA (A-GSA) [47], the agents are updated using a mixture of updated and outdated information. The nonuniformity may lead to different sources of highest pulling force, causing divergence to the population. Implementations using both iteration strategies share the same computational cost.

Algorithm 1: General steps of population-based metaheuristics using synchronous update.

1. Random initialization of possible solutions;
   For i = 1:number of agents;
2. Agent i^{th} evaluation;
   End;
   For i = 1:number of agent;
3. Generation next solution of agent i^{th};
   End;
4. Repeat steps 2 and 3 if stopping condition is not met, else end the algorithm

Algorithm 2: General steps of population-based metaheuristics using asynchronous update.

1. Random initialization of possible solutions;
   For i = 1:number of agents;
2. Agent i^{th} evaluation;
3. Generation next solution of agent i^{th};
   End;
4. Repeat steps 2 and 3 if stopping condition is not met, else end the algorithm

In this work, an adaptive switching GSA, ASw-GSA, is introduced. The iteration strategy is switched based on the fitness of the best found solution after a period of time. Thus, the concept of memory is used in ASw-GSA. The memory, fit^*, is used to memorize the best fitness found by the population. A switching threshold value, Δ, is used to determine when to switch. A small Δ encourages switches to be conducted early and frequently, while a bigger Δ delays and reduces the number of switches. The benchmark functions from CEC2014 [48] are used to study the performance of ASw-GSA. The results show that bigger value of Δ provides better performance. Comparison of ASw-GSA with other state-of-the-art optimizers shows that its performance is statistically better than that of some of the optimizers and on par with others. Study performed on the effect of switching towards the diversity shows that as switching is conducted from synchronous to asynchronous, the convergence of the population is disrupted.

This paper is organized as follows. In the next section the original GSA is introduced, together with its implementation using asynchronous update. This is followed with the introduction of the proposed algorithm in section 3. The experiments and results are presented and discussed in section 4 and finally this work is concluded in section 5.

2. The GSA

Newton's law of gravitation and the second law of motion are the roots of GSA. The bodies in universe are represented in GSA by agents. The agents in the original GSA follow the algorithm shown in figure 1.

The algorithm starts with initialization of agents' positions and velocities. An agent i's initial position, $\mathbf{X_i}(t=0)$, is initialized randomly according to the problem's search space, where

$$\mathbf{X_i}(t) = \{x_i^1(t), x_i^2(t), ..., x_i^d(t), ..., x_i^D(t)\} \tag{1}$$

$$x_i^d(t=0) \sim U([x_{min}^d, x_{max}^d]). \tag{2}$$

In Eqs. (1) and (2), d represents the dimension of the position and D is the number of dimensions. The boundary of the search space for the d is represented by $[x_{min}^d, x_{max}^d]$. The velocities are initialized to zero:

$$\mathbf{V_i}(t=0) = 0. \tag{3}$$

Like any other optimization algorithms, GSA is also an iterative process. At the beginning of every GSA's iteration, the fitness of every agent, $fit_i(t)$ is evaluated using

$$fit_i(t) = f(X_i(t-1)) \tag{4}$$

where $f(X_i(t-1))$ is the fitness function. The fitness function depends on the problem to be solved. It evaluates the fitness of the solutions from the previous iteration.

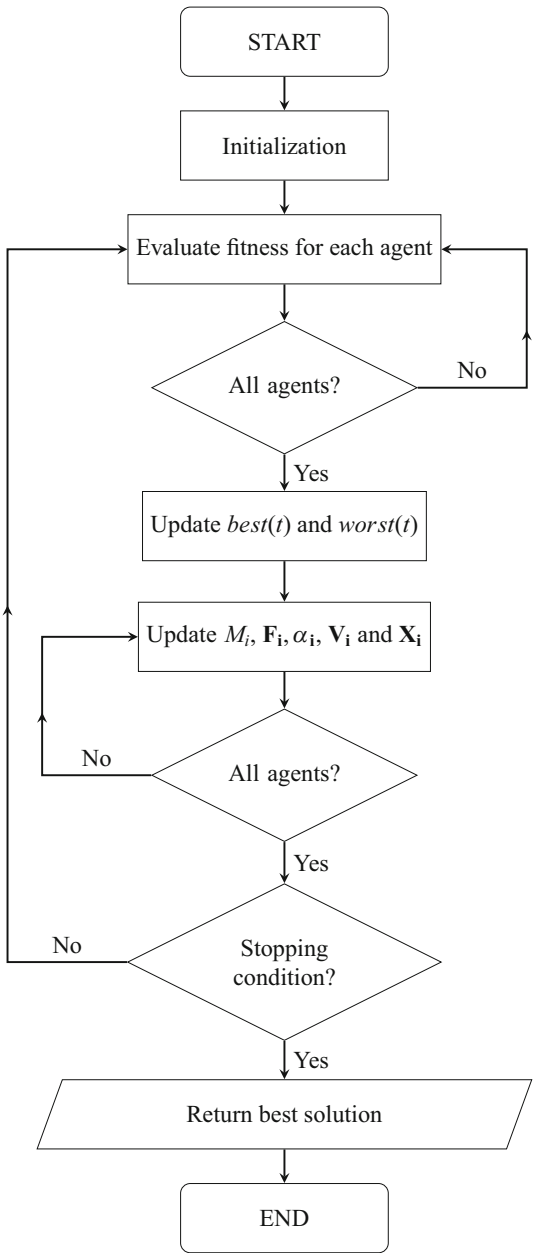

Figure 1. Flowchart of S-GSA algorithm.

After the fitness of the whole population is evaluated, the best and worst fitness within the population at time t are identified. In minimization problems, the best and worst are defined as follows:

$$best(t) = \min\{fit_1(t), fit_2(t), ..., fit_i(t), ..., fit_N(t)\} \quad (5)$$

$$worst(t) = \max\{fit_1(t), fit_2(t), ..., fit_i(t), ..., fit_N(t)\} \quad (6)$$

where N represents the size of the population.

The identification of $best(t)$ and $worst(t)$ is followed by the update phase of the population. The first phases to be updated are the masses of the agents. The mass of agent i, $M_i(t)$, is mapped according to its fitness, $fit_i(t)$:

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \quad (7)$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}. \quad (8)$$

Next, the force acting on each agent is updated. The gravitational force acting on agent i caused by agent j in the d^{th} dimension, $F_{ij}^d(t)$, is calculated using agent j's mass, $M_j(t)$, and agent i's mass, $M_i(t)$:

$$F_{ij}^d(t) = G(t)\frac{M_i(t)M_j(t)}{R_{ij}(t) + \epsilon}(x_j^d(t-1) - x_i^d(t-1)). \quad (9)$$

In Eq. (9), $R_{ij}(t)$ is the Euclidean distance between agent i and j. A small constant ϵ is added to the divisor to avoid division by zero when the agents are overlapping each other. Meanwhile, $G(t)$ is the gravitational constant at time t:

$$G(t) = G_0 \times \exp^{-\frac{\beta t}{T}} \quad (10)$$

where G_0 is the gravitational constant at the start of the search. The β is another constant, and T is the total number of iteration.

The total force acting on agent i in dimension d is

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j^d F_{ij}^d(t) \quad (11)$$

where $rand_j^d$ is a random number in the interval [0, 1].

The agents in GSA move subjected to Newton's law of motion. According to Newton's law, the acceleration of agent i over dimension d, $\alpha_i^d(t)$, can be calculated using Eq. (12):

$$\alpha_i^d(t) = \frac{F_i^d(t)}{M_i(t)}. \quad (12)$$

The agents velocities and positions are then updated using the following equations:

$$v_i^d(t) = rand_i^d \times v_i^d(t-1) + \alpha_i^d(t) \quad (13)$$

$$x_i^d(t) = x_i^d(t-1) + v_i^d(t). \quad (14)$$

These steps are repeated until the stopping condition, i.e. maximum iteration is reached. The update process of the original GSA follows the synchronous update. Hence, the original GSA is also known as S-GSA in this work.

2.1 *Asynchronous GSA*

The flowchart in figure 2 shows A-GSA's algorithm. Like the original GSA, A-GSA is a memoryless algorithm. An agent of A-GSA is updated as soon as its performance is evaluated without waiting for the entire population to be evaluated.

The agents in A-GSA identify their own $best_i(t)$ and $worst_i(t)$ using information available at the time of the evaluation based on the following equation:

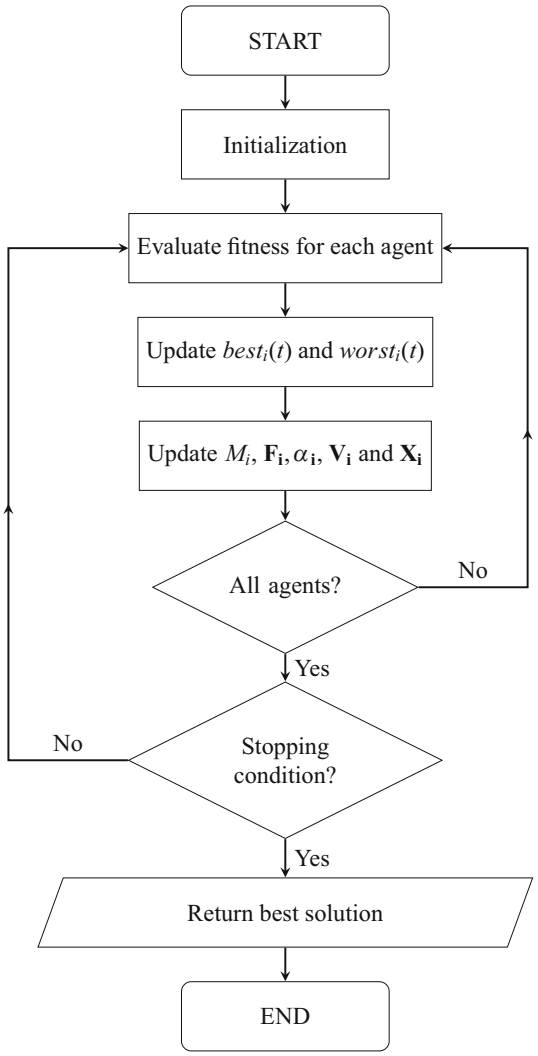

Figure 2. Flowchart of A-GSA algorithm.

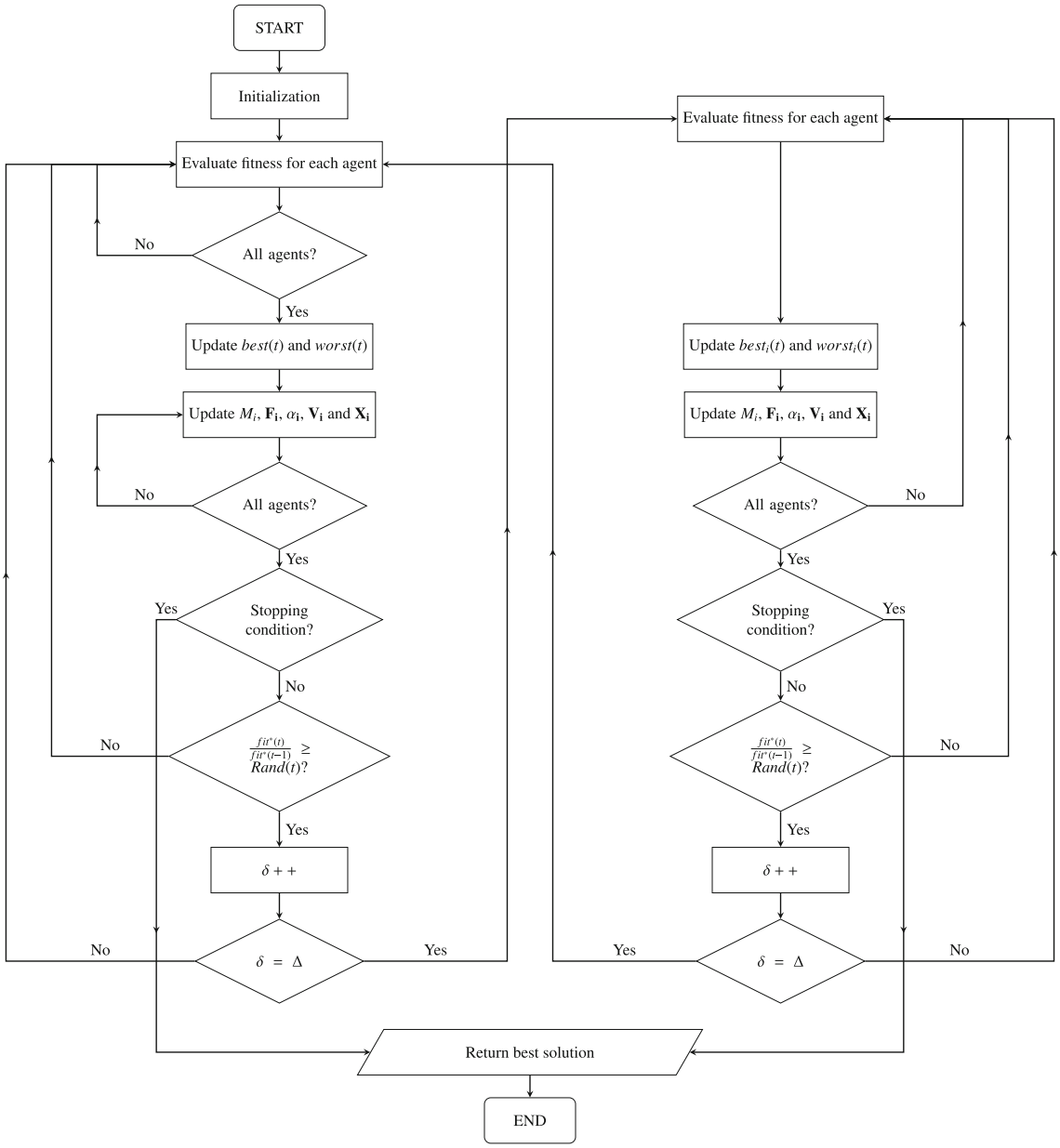

Figure 3. Flowchart of ASw-GSA.

$$best_i(t) = \min\{fit_1(t), fit_2(t), ..., fit_{i-1}(t), fit_i(t), fit_{i+1}(t-1), ..., fit_N(t-1)\}, \tag{15}$$

$$worst_i(t) = \max\{fit_1(t), fit_2(t), ..., fit_{i-1}(t), fit_i(t), fit_{i+1}(t-1), ..., fit_N(t-1)\}. \tag{16}$$

From Eqs. 15 and 16, it can be seen that for agents i's and agent j's (where $i \neq j$) the best and the worst at time t can be different.

The mass of an agent in A-GSA is then updated based on the following equations:

$$m_i(t) = \frac{fit_i(t) - worst_i(t)}{best_i(t) - worst_i(t)}, \tag{17}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{i} m_j(t) + \sum_{j=i+1}^{N} m_j(t-1)}. \tag{18}$$

The asynchronism and non-uniformity as observed from Eqs. (15)–(18) also affect the calculation of gravitational force in A-GSA. The gravitational force acting on agent i by agent j is calculated using the following equation:

Table 1. List of CEC2014's benchmark functions.

Function type	Function ID	Function name	Ideal fitness fit_{ideal}
Unimodal	f1	Rotated High Conditioned Elliptic Function	100
	f2	Rotated Bent Cigar Function	200
	f3	Rotated Discus Function	300
Simple multimodal	f4	Shifted and Rotated Rosenbrock's Function	400
	f5	Shifted and Rotated Ackley's Function	500
	f6	Shifted and Rotated Weierstrass Function	600
	f7	Shifted and Rotated Griewank's Function	700
	f8	Shifted Rastrigins Function	800
	f9	Shifted and Rotated Rastrigin's Function	900
	f10	Shifted Schwefel's Function	1000
	f11	Shifted and Rotated Schwefel's Function	1100
	f12	Shifted and Rotated Katsuura Function	1200
	f13	Shifted and Rotated Happy Cat Function	1300
	f14	Shifted and Rotated HGBat Function	1400
	f15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
	f16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
Hybrid	f17	Hybrid Function 1	1700
	f18	Hybrid Function 2	1800
	f19	Hybrid Function 3	1900
	f20	Hybrid Function 4	2000
	f21	Hybrid Function 5	2100
	f22	Hybrid Function 6	2200
Composite	f23	Composition Function 1	2300
	f24	Composition Function 2	2400
	f25	Composition Function 3	2500
	f26	Composition Function 4	2600
	f27	Composition Function 5	2700
	f28	Composition Function 6	2800
	f29	Composition Function 7	2900
	f30	Composition Function 30	3000
Search range $[-100, 100]^D$			

$$F_{ij}^d(t) = \begin{cases} G(t)\dfrac{M_i(t)M_j(t)}{R_{ij}+\epsilon}(x_j^d(t) - x_i^d(t-1)) & \text{forj} < \text{i} \\ G(t)\dfrac{M_i(t)M_j(t-1)}{R_{ij}+\epsilon}(x_j^d(t-1) - x_i^d(t-1)) & \text{forj} > \text{i}. \end{cases} \quad (19)$$

The remaining A-GSA's update equations, the total force acting on each agent, acceleration, velocity and position are similar to those of S-GSA.

3. The proposed ASw-GSA

The proposed ASw-GSA is a GSA with switching iteration strategy. The switching iteration strategy switches from synchronous to asynchronous update and vice versa.

Table 2. Parameters setting.

Algorithm	ASw-GSA	GSA	GA	PSO	BA	GWO
Dimension D			30			
Population size N			100			
Maximum no. of function evaluation FES			$10000D$			
No. of runs			30			
Algorithm's specific parameters	$G_0 = 100$	$G_0 = 100$	Mutation probability = 0.2	$c_1 = c_2 = 2$	$A = 0.5$	$0 \le a \le 2$
	$\beta = 20$	$\beta = 20$	Crossover probability = 0.5	$0.4 \le \omega \le 0.9$	$r = 0.5$	
	$\Delta = 0.65 \times FES$			$gBest$	$f \in (0, 2)$	

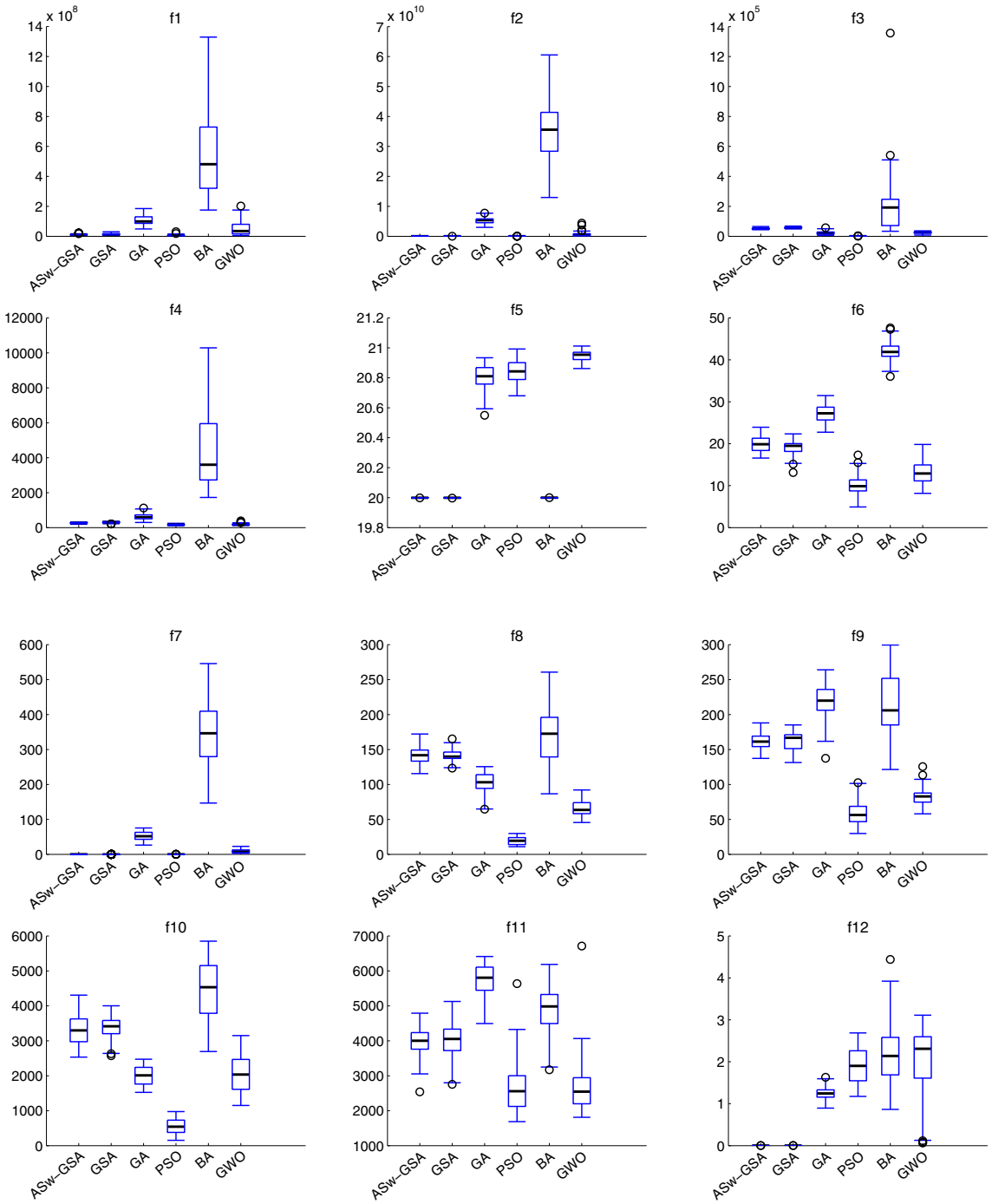

Figure 4. Boxplot of average error, e_{fit}, for f1–f12.

This is conducted to allow the population to adaptively switch between convergence and divergence. Switching from synchronous to asynchronous update causes disruption to the population, which provides diversity to the population and an escape from premature convergence. Switching from asynchronous to synchronous causes the reverse effect. The switching is made based on the condition of the memory that holds the best found solution so far, fit^*. The information hold in fit^* is not used in agents update equation. This information is used only to decide when ASw-GSA needs to switch between the two iteration strategies. If fit^* is found to be static over a period of time, Δ, the iteration strategy is switched. The value of Δ determines the frequency and how soon the switch is conducted.

A counter, δ, is used to keep track on the change of fit^* value. The counter, δ, is increased if

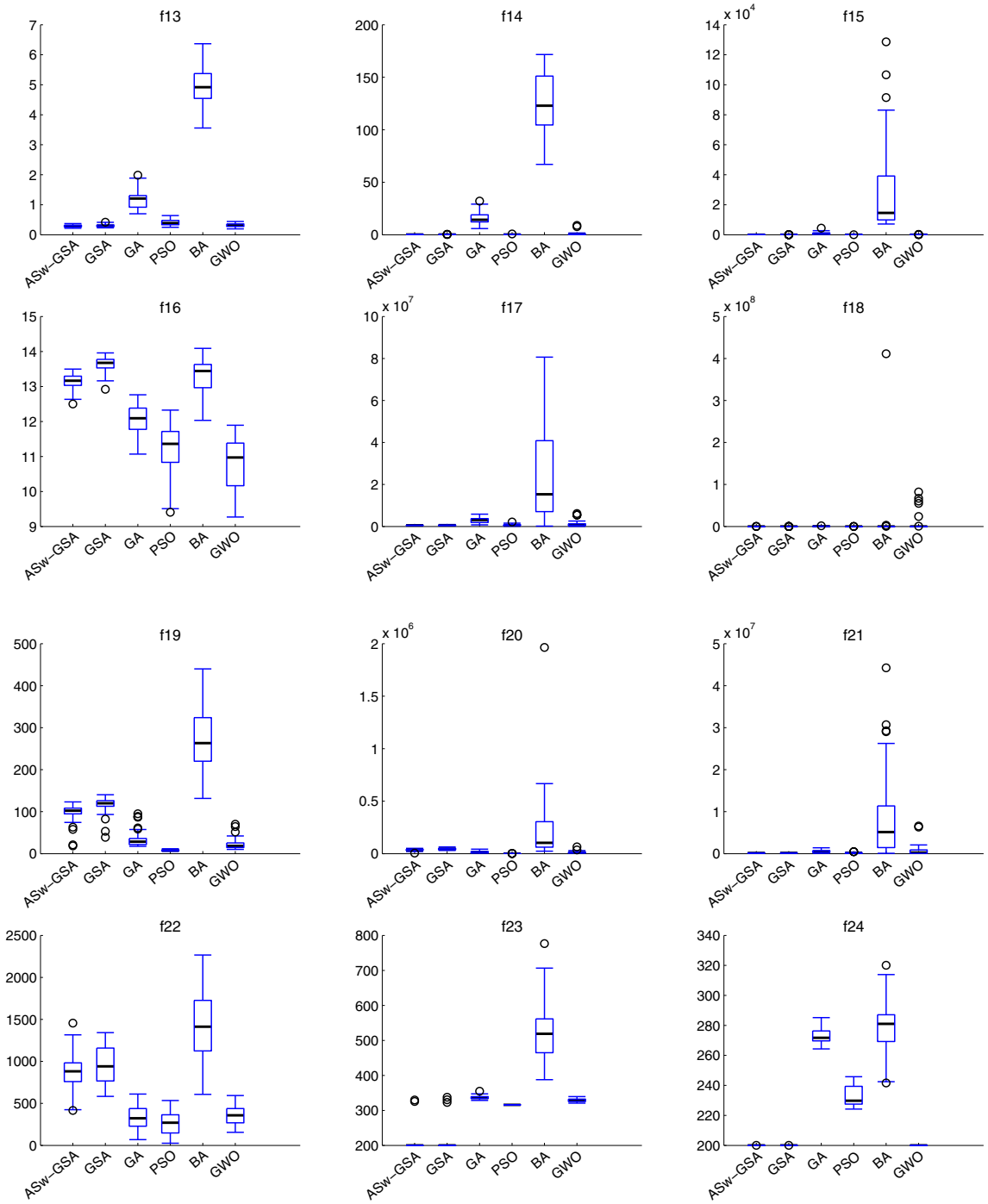

Figure 5. Boxplot of average error, e_{fit}, for f13–f24.

$$\frac{fit^*(t)}{fit^*(t-1)} \geq Rand(t). \tag{20}$$

Once, $\delta = \Delta$, the iteration strategy is switched. If the population is currently in synchronous update then it will be switched to asynchronous update and vice versa.

In Eq. 20, the ratio of fit^* from one iteration to the next iteration is compared with a random value, $Rand(t)$. This random value is ranged from zero to one and randomly drawn from a uniform distribution, $Rand(t) \sim U([0,1])$ in every iteration.

The randomness is introduced to increases the probability of switching. Without the randomness, a marginal improvement will hinder the population from switching and subsequently improving its search. Marginal improvement was reported in [49] as

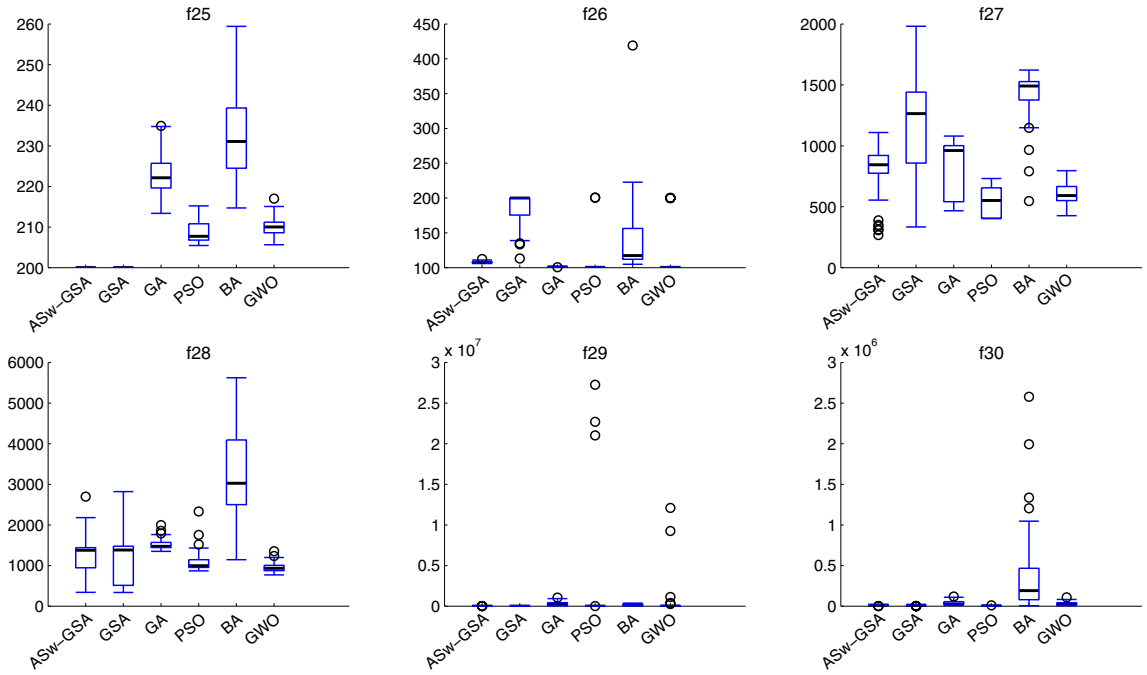

Figure 6. Boxplot of average error, e_{fit}, for f25–f30.

Table 3. Average error, e_{fit}.

Function ID	ASw-GSA	GSA	GA	PSO	BA	GWO
f1	1.16E+07	1.30E+07	1.06E+08	6.67E+06	5.54E+08	5.17E+07
f2	7.92E+03	8.60E+03	5.24E+09	2.88E+02	3.54E+10	8.30E+08
f3	5.18E+04	5.78E+04	2.21E+04	3.66E+02	2.28E+05	2.51E+04
f4	2.72E+02	3.02E+02	6.23E+02	1.75E+02	4.42E+03	1.94E+02
f5	2.00E+01	2.00E+01	2.08E+01	2.09E+01	2.00E+01	2.09E+01
f6	2.00E+01	1.91E+01	2.73E+01	1.03E+01	4.21E+01	1.32E+01
f7	1.19E−10	0.00E+00	5.28E+01	1.06E-02	3.44E+02	9.16E+00
f8	1.42E+02	1.41E+02	1.03E+02	1.92E+01	1.68E+02	6.71E+01
f9	1.62E+02	1.62E+02	2.20E+02	5.87E+01	2.16E+02	8.39E+01
f10	3.34E+03	3.37E+03	2.01E+03	5.58E+02	4.41E+03	2.03E+03
f11	3.96E+03	4.06E+03	5.72E+03	2.64E+03	4.86E+03	2.73E+03
f12	6.27E−04	4.87E−04	1.20E+00	1.89E+00	2.23E+00	1.88E+00
f13	2.95E−01	3.02E−01	1.20E+00	4.09E−01	4.96E+00	3.18E−01
f14	2.30E−01	2.43E−01	1.56E+01	2.85E−01	1.25E+02	1.01E+00
f15	3.67E+00	3.66E+00	9.01E+02	7.40E+00	3.03E+04	2.94E+01
f16	1.32E+01	1.36E+01	1.21E+01	1.13E+01	1.33E+01	1.07E+01
f17	5.42E+05	5.31E+05	2.95E+06	6.78E+05	2.82E+07	1.31E+06
f18	4.35E+02	3.82E+02	5.24E+05	7.47E+03	1.39E+07	9.65E+06
f19	9.50E+01	1.15E+02	3.63E+01	8.05E+00	2.68E+02	2.32E+01
f20	3.62E+04	4.52E+04	1.45E+04	6.02E+02	2.36E+05	1.44E+04
f21	1.50E+05	1.55E+05	5.42E+05	1.36E+05	9.20E+06	8.51E+05
f22	8.71E+02	9.56E+02	3.29E+02	2.56E+02	1.43E+03	3.54E+02
f23	2.09E+02	2.13E+02	3.37E+02	3.16E+02	5.22E+02	3.29E+02
f24	2.00E+02	2.00E+02	2.73E+02	2.33E+02	2.80E+02	2.00E+02
f25	2.00E+02	2.00E+02	2.23E+02	2.09E+02	2.33E+02	2.10E+02
f26	1.08E+02	1.87E+02	1.01E+02	1.07E+02	1.41E+02	1.14E+02
f27	7.76E+02	1.18E+03	8.24E+02	5.51E+02	1.40E+03	6.06E+02
f28	1.30E+03	1.26E+03	1.51E+03	1.10E+03	3.25E+03	9.62E+02
f29	2.01E+02	2.00E+02	3.02E+05	2.37E+06	1.80E+05	7.94E+05
f30	1.29E+04	1.10E+04	3.84E+04	3.97E+03	4.44E+05	2.83E+04

Table 4. Average rankings of the algorithms.

Algorithm	Ranking
PSO	2.18
ASw-GSA	2.72
GSA	3.08
GWO	3.23
GA	4.15
BA	5.63

Table 5. P value table for $\alpha = 0.05$.

i	Algorithms	$z = (R_0 - R_i)/SE$	p	Holm
15	PSO vs. BA	7.142179	0	0.003333
14	ASw-GSA vs. BA	6.038074	0	0.003571
13	GSA vs. BA	5.279002	0	0.003846
12	BA vs. GWO	4.968472	0.000001	0.004167
11	GA vs. PSO	4.071387	0.000047	0.004545
10	GA vs. BA	3.070792	0.002135	0.005
9	ASw-GSA vs. GA	2.967282	0.003004	0.005556
8	GSA vs. GA	2.20821	0.02723	0.00625
7	PSO vs. GWO	2.173707	0.029727	0.007143
6	GA vs. GWO	1.89768	0.057738	0.008333
5	GSA vs. PSO	1.863177	0.062437	0.01
4	ASw-GSA vs. PSO	1.104105	0.269548	0.0125
3	ASw-GSA vs. GWO	1.069602	0.284799	0.016667
2	ASw-GSA vs. GSA	0.759072	0.447809	0.025
1	GSA vs. GWO	0.31053	0.756158	0.05

a factor that hinders PSO from improving its performance.

A flowchart of ASw-GSA is shown in figure 3. The algorithm starts as a synchronous update population. S-GSA is the original GSA. Switching is proposed in ASw-GSA to improve the performance of GSA.

4. Experiments, results and discussion

The proposed ASw-GSA is tested using CEC2014's single objective real-parameter numerical optimization test suite [48]. There are 30 functions in the CEC2014's test suite. The functions are listed in table 1. The ideal fitness of each function is shown in the last column of the table. The functions are all minimization functions. They consist of three rotated unimodal functions, 13 simple multimodal functions that are either shifted only or shifted and rotated problems, six hybrid functions and eight composition functions. The rotation and shifting of the functions change the location of the optimal solution so that it is not located at the centre of the search space; this made finding the best solution more challenging. The hybrid functions are a combination of more than one function, while the composition functions consist of unimodal, multimodal and hybrid functions with local optima trap set at the origin, which is at the centre of the search area. These functions are derived from the following 14 basic functions:

1. High Conditioned Elliptic Function

$$f_1(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2. \tag{21}$$

2. Bent Cigar Function

$$f_2(x) = x_1^2 + (10^6) \sum_{i=2}^{D} x_i^2. \tag{22}$$

3. Discus Function

$$f_3(x) = (10^6) x_1^2 + \sum_{i=2}^{D} x_i^2. \tag{23}$$

4. Rosenbrock's Function

$$f_4(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_i)^2 + (x_i - 1)^2). \tag{24}$$

5. Ackley's Function %

$$f_5(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + \exp(1) \tag{25}$$

6. Weierstrass Function

$$f_6(x) = \sum_{i=1}^{D} \left(\sum_{k=0}^{kmax} [a^k \cos(2\pi b^k (x_i + 0.5))] \right) - D \sum_{k=0}^{kmax} [a^k \cos(2\pi b^k \times 0.5))] \tag{26}$$

where $a = 0.5, b = 3$, and $kmax = 20$.

7. Griewank's Function

$$f_7(x) = \sum_{i=1} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \tag{27}$$

8. Rastrigin's Function

$$f_8(x) = \sum_{i=1}^{D} (x_i^2 - 10 \cos(2\pi x_i) + 10). \tag{28}$$

Table 6. Average number of switches.

Function ID	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%
f1	19	9	6	4	3	3	2	2	2	1	0
f2	18.07	8.97	5.43	4	3	2.47	2	2	1.4	1	0
f3	19	9	6	4	3	3	2	2	2	1	0
f4	19	9	6	4	3	3	2	2	2	1	0
f5	19	9	6	4	3	3	2	2	2	1	0
f6	19	9	6	4	3	3	2	2	2	1	0
f7	19	9	6	4	3	3	2	2	2	1	0
f8	19	9	6	4	3	3	2	2	2	1	0
f9	19	9	6	4	3	3	2	2	2	1	0
f10	19	9	6	4	3	3	2	2	2	1	0
f11	19	9	6	4	3	3	2	2	2	1	0
f12	19	9	6	4	3	3	2	2	2	1	0
f13	19	9	6	4	3	3	2	2	2	1	0
f14	19	9	6	4	3	3	2	2	2	1	0
f15	19	9	6	4	3	3	2	2	2	1	0
f16	19	9	6	4	3	3	2	2	2	1	0
f17	19	9	6	4	3	3	2	2	2	1	0
f18	19	9	6	4	3	3	2	2	2	1	0
f19	19	9	6	4	3	3	2	2	2	1	0
f20	19	9	6	4	3	3	2	2	2	1	0
f21	19	9	6	4	3	3	2	2	2	1	0
f22	19	9	6	4	3	3	2	2	2	1	0
f23	19	9	6	4	3	3	2	2	2	1	0
f24	19	9	6	4	3	3	2	2	2	1	0
f25	19	9	6	4	3	3	2	2	2	1	0
f26	19	9	6	4	3	3	2	2	2	1	0
f27	19	9	6	4	3	3	2	2	2	1	0
f28	19	9	6	4	3	3	2	2	2	1	0
f29	19	9	6	4	3	2.37	1.77	1.37	1	1	0
f30	19	9	6	4	3	2.93	2	1.97	1.9	1	0

9. Modified Schwefel's Function

$$f_9(x) = 418.9829D - \sum_{i=1}^{D} g(z_i) \tag{29}$$

where

$$z_i = x_i + 4.209687462275036e + 002$$

$$g(z_i) = \begin{cases} z_i \sin(|z_i|^{1/2}), & \text{if } |z_i| \le 500 \\ (500 - mod(z_i, 500)) \\ \times \sin(\sqrt{|500 - mod(z_i, 500)|}) \\ -\dfrac{(z_i - 500)^2}{10000D}, & \text{if } z_i > 500 \\ (mod(z_i, 500) - 500) \\ \times \sin(\sqrt{|mod(z_i, 500) - 500|}) \\ -\dfrac{(z_i + 500)^2}{10000D}, & \text{if } z_i < -500 \end{cases}.$$

10. Katsuura Function

$$f_{10}(x) = \frac{10}{D^2} \prod_{i=1}^{D} \left(1 + i \sum_{j=1}^{32} \frac{|2^j x_i - round(2^j x_i)|}{2^j}\right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2}. \tag{30}$$

11. Happy Cat Function

$$f_{11}(x) = \left|\sum_{i=1}^{D} x_i^2 - D\right|^{1/4} + \left(0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i\right)/D + 0.5. \tag{31}$$

12. HGB at Function

$$f_{12}(x) = \left|\left(\sum_{i=1}^{D} x_i^2\right)^2 - \left(\sum_{i=1}^{D} x_i\right)^2\right|^{1/2} + \left(0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i\right)/D + 0.5. \tag{32}$$

Table 7. Average number of switches (continued).

Function ID	60%	65%	70%	75%	80%	85%	90%	95%
f1	1	1	1	1	1	1	1	0.93
f2	1	1	1	1	1	0.97	0.47	0
f3	1	1	1	1	1	1	1	1
f4	1	1	1	1	1	1	1	1
f5	1	1	1	1	1	1	1	1
f6	1	1	1	1	1	1	1	1
f7	1	1	1	1	1	1	1	1
f8	1	1	1	1	1	1	1	1
f9	1	1	1	1	1	1	1	1
f10	1	1	1	1	1	1	1	1
f11	1	1	1	1	1	1	1	1
f12	1	1	1	1	1	1	1	1
f13	1	1	1	1	1	1	1	1
f14	1	1	1	1	1	1	1	1
f15	1	1	1	1	1	1	1	1
f16	1	1	1	1	1	1	1	1
f17	1	1	1	1	1	1	1	0.83
f18	1	1	1	1	1	1	1	0.5
f19	1	1	1	1	1	1	1	1
f20	1	1	1	1	1	1	1	1
f21	1	1	1	1	1	1	1	0.63
f22	1	1	1	1	1	1	1	1
f23	1	1	1	1	1	1	1	1
f24	1	1	1	1	1	1	1	1
f25	1	1	1	1	1	1	1	1
f26	1	1	1	1	1	1	1	1
f27	1	1	1	1	1	1	1	1
f28	1	1	1	1	1	1	1	1
f29	1	0.97	0.8	0.37	0.1	0.03	0	0
f30	1	1	1	1	0.97	0.93	0.97	0.53

13. Expanded Griewank's plus Rosenbrock's Function

$$f_{13}(x) = f_7(f_4(x_1, x_2)) + f_7(f_4(x_2, x_3)) + \cdots + f_7(f_4(x_{D-1}, x_D)) + f_7(f_4(x_D, x_1)). \quad (33)$$

14. Expanded Scaffer's F6 Function

$$f_{14}(x) = g(x_1, x_2) + g(x_2, x_3) + \cdots + g(x_{D-1}, x_D) + g(x_D, x_1). \quad (34)$$

Scaffer F6 Function: $g(x, y) = 0.5 + \frac{\sin^2\left(\sqrt{x^2+y^2}\right)-0.5}{(1+0.001(x^2+y^2))^2}$.

The performance of ASw-GSA is compared to that of the original GSA and four state-of-the-art metaheuristics algorithms: GA, PSO, BA and grey wolf optimization (GWO) [50]. The performance of the algorithms is measured using the average error, e_{fit}. The average error is the difference between the average fitness of the best solutions found, fit_{ave}, with the ideal fitness, fit_{ideal}:

$$e_{fit} = fit_{ave} - fit_{ideal}. \quad (35)$$

The parameters settings for the performance measurement are listed in table 2. The size of test function's dimension, D, is set to 30. The population size for all algorithms is set to 100 agents and their stopping condition is based on CEC2014's setting, which is that the maximum number of function evaluations, FES, does not exceed $10000D$. Each of the test functions is run for 30 times. The G_0 of ASw-GSA and original GSA is set to 100, while β is set to 20. The value Δ for ASw-GSA is set to be equivalent to 65% of FES. The GA is adapted from [10], where its mutation probability is 0.2 and crossover probability is 0.5. Global neighbourhood PSO with $c_1 = c_2 = 2$ and linearly decreasing inertia weight $0.4 \leq \omega \leq 0.9$ is used here. The parameters for BA and GWO are set according to their respective original works. The loudness of BA, A, is set to 0.5, pulse rate $r = 0.5$ and the frequency $f \in (0, 2)$. The GWO's a is linearly decreased from 2 to 0.

Other aspects studied in this work are the effect of Δ towards the performance of ASw-GSA and how adaptive switching changes the agents search behaviour.

4.1 *ASw-GSA's performance*

The ASw-GSA with $\Delta = 65\%$ is compared to the original GSA, GA, PSO, BA and GWO. The distribution of average errors of the algorithms tested is shown in the boxplots of figures 4–6. The horizontal line in each of the boxes shows median of e_{fit}, while the outliers are presented using $\circ$. The outliers to be out of norm values, a lesser number of outliers are desired. Lower boxplot indicates ability of an algorithm to reduce the error while a smaller box shows stability of the algorithm.

- Original GSA: Generally, for most functions, the boxplots of both ASw-GSA and original GSA are located at almost similar locations. However, upon closer examination it can be seen for several functions such as f1, f11, f13, f16, f19, f20, f22, f26 and f27 that ASw-GSA has lower boxplots. Additionally, GSA is observed to have more number of outliers than ASw-GSA for f2, f4, f6, f7, f8, f10, f14, f15 and f23. They show that implementation of adaptive switching by ASw-GSA is able to improve GSA.
- GA: GA has more number of higher boxes compared with ASw-GSA. GA has lower boxes for nine out of the 30 test functions.
- PSO: Although PSO is seen to have better boxes in more number of test functions compared with ASw-GSA, ASw-GSA is observed to perform better for f2, f5, f12, f13, f17, f23, f24 and f25.
- BA: Based on the boxplots, ASw-GSA has better performance than BA for all test functions.
- GWO: ASw-GSA has lower and smaller boxes than GWO for at least 15 out of the 30 test functions.

Table 8. Average e_{fit} of various Δ.

Function ID	S-GSA	A-GSA	5%	10%	15%	20%	25%	30%
f1	1.30E+07	7.11E+08	1.66E+08	5.51E+07	2.20E+07	1.63E+07	1.36E+07	1.17E+07
f2	8.60E+03	5.94E+10	1.22E+06	8.59E+04	1.24E+04	8.51E+03	8.93E+03	8.44E+03
f3	5.78E+04	9.77E+04	7.36E+04	7.78E+04	7.05E+04	5.71E+04	5.16E+04	5.31E+04
f4	3.02E+02	1.01E+04	4.49E+02	3.22E+02	2.92E+02	2.86E+02	2.75E+02	2.78E+02
f5	2.00E+01	2.10E+01	2.02E+01	2.09E+01	2.02E+01	2.00E+01	2.00E+01	2.00E+01
f6	1.91E+01	3.90E+01	2.38E+01	2.12E+01	2.03E+01	2.02E+01	1.97E+01	1.99E+01
f7	0.00E+00	5.44E+02	1.17E+00	8.28E−01	2.00E−01	3.04E−02	3.80E−03	4.33E-04
f8	1.41E+02	3.29E+02	1.40E+02	1.41E+02	1.40E+02	1.42E+02	1.40E+02	1.38E+02
f9	1.62E+02	3.78E+02	1.59E+02	1.65E+02	1.58E+02	1.62E+02	1.62E+02	1.58E+02
f10	3.37E+03	7.02E+03	3.27E+03	3.27E+03	3.36E+03	3.20E+03	3.32E+03	3.31E+03
f11	4.06E+03	7.16E+03	4.09E+03	3.83E+03	4.01E+03	4.15E+03	4.04E+03	4.07E+03
f12	4.87E−04	2.45E+00	3.62E−01	9.27E−02	2.61E−02	8.69E−03	3.92E−03	1.69E−03
f13	3.02E−01	6.15E+00	4.44E−01	3.75E−01	3.34E−01	3.34E−01	3.22E−01	3.24E−01
f14	2.43E−01	1.75E+02	2.93E−01	2.60E−01	2.50E−01	2.45E−01	2.43E−01	2.40E−01
f15	3.66E+00	3.47E+05	2.23E+01	9.08E+00	4.62E+00	3.59E+00	4.12E+00	3.89E+00
f16	1.36E+01	1.31E+01	1.31E+01	1.32E+01	1.31E+01	1.31E+01	1.31E+01	1.31E+01
f17	5.31E+05	1.84E+07	1.93E+07	5.29E+06	1.94E+06	1.05E+06	8.15E+05	6.30E+05
f18	3.82E+02	9.81E+08	4.70E+04	2.32E+03	4.97E+02	4.40E+02	4.15E+02	3.67E+02
f19	1.15E+02	2.92E+02	1.32E+02	1.09E+02	1.03E+02	1.14E+02	1.02E+02	9.27E+01
f20	4.52E+04	7.10E+04	6.70E+04	7.73E+04	7.84E+04	5.46E+04	4.25E+04	4.36E+04
f21	1.55E+05	4.76E+06	4.93E+06	1.75E+06	3.77E+05	2.06E+05	1.83E+05	1.74E+05
f22	9.56E+02	1.30E+03	9.99E+02	8.98E+02	8.65E+02	9.08E+02	9.10E+02	8.74E+02
f23	2.13E+02	6.70E+02	2.36E+02	2.20E+02	2.07E+02	2.07E+02	2.10E+02	2.00E+02
f24	2.00E+02	2.73E+02	2.06E+02	2.02E+02	2.01E+02	2.00E+02	2.00E+02	2.00E+02
f25	2.00E+02	2.25E+02	2.01E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
f26	1.87E+02	1.06E+02	1.07E+02	1.07E+02	1.07E+02	1.07E+02	1.07E+02	1.07E+02
f27	1.18E+03	8.29E+02	8.32E+02	7.97E+02	8.33E+02	7.93E+02	7.30E+02	7.78E+02
f28	1.26E+03	4.70E+03	1.49E+03	1.33E+03	1.41E+03	1.22E+03	1.23E+03	1.10E+03
f29	2.00E+02	1.17E+08	5.38E+07	2.05E+07	6.82E+06	1.68E+06	4.17E+05	6.40E+04
f30	1.10E+04	7.47E+05	6.50E+05	1.83E+05	1.07E+05	5.14E+04	2.12E+04	1.41E+04

Table 9. Average e_{fit} of various Δ (continued).

Function ID	S-GSA	A-GSA	35%	40%	45%	50%	60%	65%
f1	1.30E+07	7.11E+08	1.14E+07	1.13E+07	1.17E+07	1.18E+07	1.48E+07	1.16E+07
f2	8.60E+03	5.94E+10	9.36E+03	8.51E+03	8.11E+03	8.70E+03	8.85E+03	7.92E+03
f3	5.78E+04	9.77E+04	5.17E+04	5.27E+04	5.21E+04	5.41E+04	5.75E+04	5.18E+04
f4	3.02E+02	1.01E+04	2.72E+02	2.58E+02	2.53E+02	2.62E+02	2.97E+02	2.72E+02
f5	2.00E+01	2.10E+01	2.00E+01	2.00E+01	2.00E+01	2.00E+01	2.00E+01	2.00E+01
f6	1.91E+01	3.90E+01	1.98E+01	1.91E+01	1.96E+01	1.96E+01	1.92E+01	2.00E+01
f7	0.00E+00	5.44E+02	5.07E−05	5.88E−06	6.27E−07	7.54E−08	0.00E+00	1.19E−10
f8	1.41E+02	3.29E+02	1.39E+02	1.40E+02	1.46E+02	1.39E+02	1.40E+02	1.42E+02
f9	1.62E+02	3.78E+02	1.63E+02	1.64E+02	1.64E+02	1.64E+02	1.65E+02	1.62E+02
f10	3.37E+03	7.02E+03	3.23E+03	3.22E+03	3.25E+03	3.17E+03	3.22E+03	3.34E+03
f11	4.06E+03	7.16E+03	4.04E+03	3.93E+03	4.19E+03	4.15E+03	4.18E+03	3.96E+03
f12	4.87E−04	2.45E+00	1.73E−03	1.11E−03	8.77E−04	6.08E−04	6.38E−04	6.27E−04
f13	3.02E−01	6.15E+00	3.06E−01	3.11E−01	3.07E−01	3.00E−01	3.05E−01	2.95E−01
f14	2.43E−01	1.75E+02	2.47E−01	2.50E−01	2.47E−01	2.31E−01	2.38E−01	2.30E−01
f15	3.66E+00	3.47E+05	3.58E+00	3.74E+00	3.79E+00	3.81E+00	3.61E+00	3.67E+00
f16	1.36E+01	1.31E+01	1.32E+01	1.32E+01	1.32E+01	1.32E+01	1.36E+01	1.31E+01
f17	5.31E+05	1.84E+07	5.88E+05	5.82E+05	5.68E+05	5.39E+05	5.40E+05	5.42E+05
f18	3.82E+02	9.81E+08	4.11E+02	3.46E+02	3.50E+02	4.09E+02	3.86E+02	4.34E+02
f19	1.15E+02	2.92E+02	8.70E+01	9.52E+01	9.34E+01	9.55E+01	1.11E+02	9.50E+01
f20	4.52E+04	7.10E+04	3.95E+04	3.79E+04	3.95E+04	3.90E+04	4.28E+04	3.62E+04
f21	1.55E+05	4.76E+06	1.68E+05	1.59E+05	1.57E+05	1.50E+05	1.63E+05	1.50E+05
f22	9.56E+02	1.30E+03	9.46E+02	9.10E+02	9.45E+02	8.66E+02	9.24E+02	8.71E+02
f23	2.13E+02	6.70E+02	2.00E+02	2.09E+02	2.00E+02	2.12E+02	2.13E+02	2.09E+02
f24	2.00E+02	2.73E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
f25	2.00E+02	2.25E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
f26	1.87E+02	1.06E+02	1.07E+02	1.08E+02	1.08E+02	1.07E+02	1.96E+02	1.08E+02
f27	1.18E+03	8.29E+02	7.56E+02	7.78E+02	8.80E+02	8.22E+02	1.15E+03	7.76E+02
f28	1.26E+03	4.70E+03	1.13E+03	1.39E+03	1.33E+03	1.26E+03	1.19E+03	1.30E+03
f29	2.00E+02	1.17E+08	1.05E+04	3.18E+03	2.92E+02	2.35E+02	2.00E+02	2.01E+02
f30	1.10E+04	7.47E+05	1.27E+04	1.08E+04	1.32E+04	1.41E+04	1.31E+04	1.29E+04

Table 10. Average e_{fit} of various Δ (continued).

Function ID	S-GSA	A-GSA	70%	75%	80%	85%	90%	95%
f1	1.30E+07	7.11E+08	1.18E+07	1.14E+07	1.07E+07	1.18E+07	1.18E+07	1.21E+07
f2	8.60E+03	5.94E+10	8.39E+03	8.22E+03	8.08E+03	8.47E+03	8.20E+03	8.16E+03
f3	5.78E+04	9.77E+04	5.09E+04	4.95E+04	5.11E+04	5.08E+04	4.85E+04	5.24E+04
f4	3.02E+02	1.01E+04	2.63E+02	2.64E+02	2.67E+02	2.54E+02	2.66E+02	2.74E+02
f5	2.00E+01	2.10E+01	2.00E+01	2.00E+01	2.00E+01	2.00E+01	2.00E+01	2.00E+01
f6	1.91E+01	3.90E+01	1.96E+01	1.92E+01	1.94E+01	1.98E+01	1.89E+01	1.96E+01
f7	0.00E+00	5.44E+02	1.30E−11	1.48E−12	1.14E−13	0.00E+00	0.00E+00	0.00E+00
f8	1.41E+02	3.29E+02	1.39E+02	1.40E+02	1.38E+02	1.42E+02	1.39E+02	1.42E+02
f9	1.62E+02	3.78E+02	1.63E+02	1.63E+02	1.65E+02	1.62E+02	1.59E+02	1.64E+02
f10	3.37E+03	7.02E+03	3.34E+03	3.30E+03	3.24E+03	3.28E+03	3.25E+03	3.14E+03
f11	4.06E+03	7.16E+03	4.02E+03	4.06E+03	4.02E+03	4.06E+03	3.95E+03	4.10E+03
f12	4.87E−04	2.45E+00	1.13E−03	1.17E−03	1.07E−03	1.07E−03	6.77E−04	5.33E−04
f13	3.02E−01	6.15E+00	3.00E−01	2.97E−01	3.10E−01	2.91E−01	3.00E−01	2.93E−01
f14	2.43E−01	1.75E+02	2.47E−01	2.45E−01	2.45E−01	2.35E−01	2.35E−01	2.39E−01
f15	3.66E+00	3.47E+05	3.77E+00	3.80E+00	3.63E+00	3.68E+00	3.80E+00	3.67E+00
f16	1.36E+01	1.31E+01	1.32E+01	1.32E+01	1.33E+01	1.33E+01	1.32E+01	1.33E+01
f17	5.31E+05	1.84E+07	5.58E+05	5.70E+05	5.61E+05	5.65E+05	5.53E+05	6.06E+05
f18	3.82E+02	9.81E+08	4.64E+02	4.50E+02	3.87E+02	4.02E+02	4.36E+02	3.69E+02
f19	1.15E+02	2.92E+02	8.74E+01	9.51E+01	8.38E+01	8.59E+01	9.64E+01	8.55E+01
f20	4.52E+04	7.10E+04	3.93E+04	3.54E+04	3.79E+04	3.69E+04	3.40E+04	3.57E+04
f21	1.55E+05	4.76E+06	1.65E+05	1.63E+05	1.60E+05	1.68E+05	1.63E+05	1.66E+05
f22	9.56E+02	1.30E+03	8.81E+02	8.59E+02	9.22E+02	9.11E+02	8.84E+02	8.23E+02
f23	2.13E+02	6.70E+02	2.00E+02	2.00E+02	2.09E+02	2.08E+02	2.04E+02	2.08E+02
f24	2.00E+02	2.73E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
f25	2.00E+02	2.25E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
f26	1.87E+02	1.06E+02	1.08E+02	1.09E+02	1.09E+02	1.08E+02	1.09E+02	1.17E+02
f27	1.18E+03	8.29E+02	8.18E+02	8.70E+02	8.71E+02	8.65E+02	9.32E+02	8.98E+02
f28	1.26E+03	4.70E+03	1.13E+03	1.28E+03	1.17E+03	1.16E+03	1.18E+03	1.14E+03
f29	2.00E+02	1.17E+08	2.01E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
f30	1.10E+04	7.47E+05	1.03E+04	1.17E+04	1.12E+04	1.22E+04	1.27E+04	1.11E+04

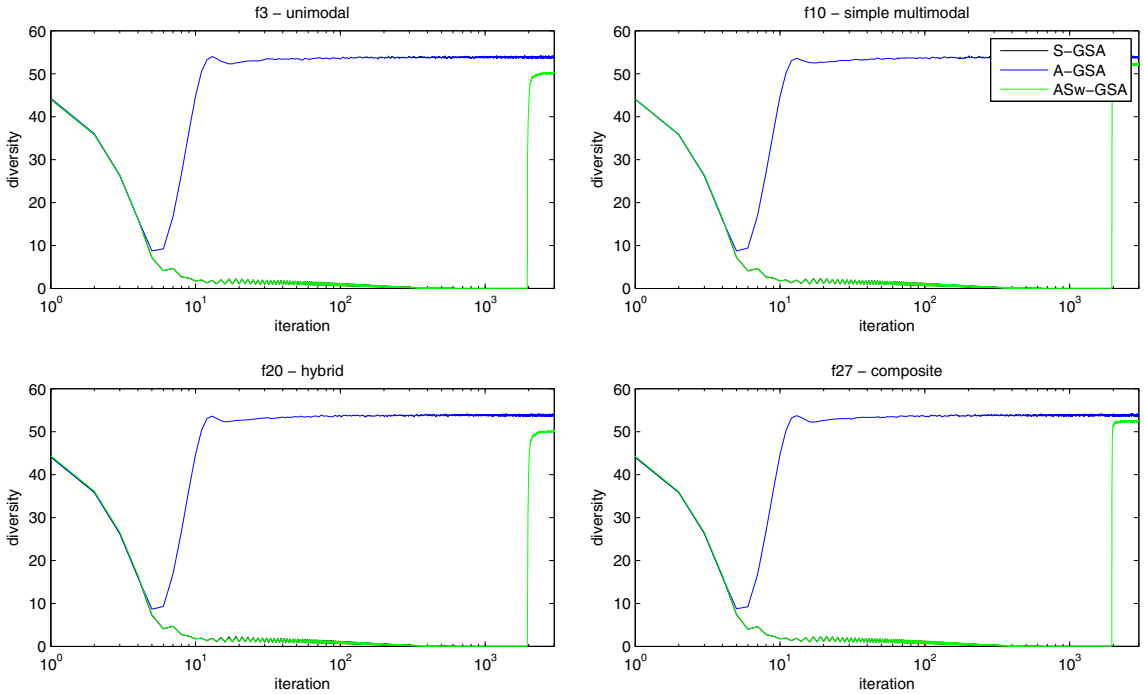

Figure 7. Rate of position diversity.

Table 3 presents the average error of the algorithms. These results are used for statistical analysis using the Friedman test. The average ranks of the algorithms using Friedman $N \times N$ test are tabulated in table 4. It is observed that the proposed PSO is ranked the best while the proposed algorithm, ASw-GSA, is ranked the second best; this is followed by GSA, GWO, GA and BA.

Based on the statistical value of Friedman $N \times N$ test, it is observed that significant difference exists between the algorithms tested. The results are further analysed using the Holm post hoc procedure, and the findings are presented in table 5. With significance level of $\alpha = 0.05$, the Holm's procedure rejects the null hypotheses of on par performance for the algorithms that have $i \geq 9$. Therefore, it is found that ASw-GSA is significantly better than GA and BA and performs as well as PSO, the original GSA and GWO. The Holm procedure ranked GSA and GWO as on par with GA.

4.2 *Effect of* Δ

The effect of Δ is investigated here. It is tested with value ranging from 5% to 95% of the *FES*. Its effect on number of switches is shown in tables 6 and 7. Smaller Δ allows more number of switches. However, due to the adaptiveness of the proposed algorithm, the maximum number of switching is not guaranteed. For $\Delta = 55\%$ no switching is observed.

The average errors of the test carried using different values of Δ (excluding $\Delta = 55\%$) are tabulated in tables 8, 9 and 10. Pairwise statistical analysis using the Wilcoxon sign ranked test was conducted using these results. Table 11 presents the statistical values of the Wilcoxon sign ranked test.

It is found that ASw-GSA with $\Delta = 40\%$, 60% and 80% are significantly better than S-GSA with significance level of 10% and ASw-GSA with Δ=65%, 70% and 90% are better than S-GSA with level of significance of 5%. On the other hand, ASw-GSA with $\Delta = 5\%$ and 10% are worse than S-GSA with significance level of 1% and 2%, respectively.

The results of Wilcoxon sign rank test against A-GSA show that ASw-GSA with all values of Δ performed significantly better with 1% significance level.

4.3 *Population's diversity*

In order to understand how adaptive switching changes the population's search behaviour and helps in improving the performance of GSA, the population's position diversity, D^p, of ASw-GSA is measured and compared against S-GSA's and A-GSA's. The position diversity shows the distribution of the agents in the search area and gives a picture on the state of exploration and exploitation of the

agents. It is calculated following the L_1 normalized diversity measurements as discussed in [51]. The calculation starts with the computation of the mean position for each d^{th} dimension of the population, $\overline{x}^d$:

$$\overline{x}^d = \frac{1}{N}\sum_{i=1}^{N} x_i^d. \tag{36}$$

Next is the diversity of the agents' position with respect to mean position for every dimension D^{pd}:

$$D^{pd} = \frac{1}{N}\sum_{i=1}^{N} |x_i^d - \overline{x}^d|. \tag{37}$$

Finally the population's position diversity, D^p, is calculated as shown in Eq. (38):

$$D^p = \frac{1}{D}\sum_{d=1}^{D} D^{pd}. \tag{38}$$

Figure 7 shows the change of position diversity of ASw-GSA, S-GSA and A-GSA. Four functions are selected here, one from each category. Similar to [46], S-GSA is observed to converge rapidly. The population of A-GSA exhibits only converging trend at early state of the search; as the search progresses the population diverged and this is maintained until the end. For ASw-GSA, it can be seen that switching provides disruption to the population's diversity. The disruption caused the diversity to increase and the agents to escape convergence. However, the increased diversity of ASw-GSA is not as high as that of A-GSA. The increased diversity may result in discovery of better solution in neighbouring area within the search space.

Table 11. Wilcoxon signed rank test.

	ASw-GSA vs. S-GSA		ASw-GSA vs. A-GSA	
Δ (%)	R^+	R^-	R^+	R^-
5	433	32	55	410
10	351	114	25	440
15	303	162	28	437
20	266	199	3	462
25	225	240	3	462
30	165	300	2	463
35	195.5	269.5	3	462
40	151.5	313.5	4	461
45	207.5	257.5	12	453
50	172.5	262.5	3	462
60	140.5	294.5	12	453
65	133.5	301.5	4	461
70	127.5	307.5	4	461
75	162.5	272.5	12	453
80	137.5	297.5	12	453
85	159	306	12	453
90	124	341	13	452
95	158	307	14	451

5. Conclusion

A variant of GSA with adaptive switching iteration strategy, ASw-GSA, is proposed here. The proposed algorithm combines both synchronous and asynchronous updates. The integration of both iteration strategies changes the behaviour of the agents. From the experiments conducted, switching from synchronous update to asynchronous update at a later stage of the search gives a better performance. The switching provides disturbance of the population's diversity and allows the agents to escape from convergence. Based on the results obtained, the ASw-PSO is found to perform significantly better than state-of-the-art optimizers, GA and BA, while performing as well as PSO and GWO.

Acknowledgements

This research is funded by the Ministry of Higher Education, Malaysia, under the Fundamental Research Grant Scheme (FRGS/1/2015/ICT02/MMU/03/1), which is awarded to Multimedia University and the University of Malaya's Postgraduate Research Grant (PG097-2013A). We are grateful to Dr Sophan Wahyudi Nawawi of Universiti Teknologi Malaysia for his input and support. The authors would also like to acknowledge the anonymous reviewers for their valuable comments and insights.

References

[1] Nobahari H, Nikusokhan M and Siarry P 2011 Non-dominated sorting gravitational search algorithm. In: *Proceedings of the International Conference on Swarm Intelligence*

[2] Hassanzadeh H R and Rouhani M. A multi-objective gravitational search algorithm. In: *Proceedings of the International Conference on Computational Intelligence, Communication Systems and Networks, CICSyN 2010*, pp. 7–12

[3] Ibrahim Z, Muhammad B, Ghazali K H, Lim K S, Nawawi S W and Yusof Z M 2012 Vector evaluated gravitational search algorithm (VEGSA) for multi-objective optimization Problems. In: *Proceedings of the Computational Intelligence, Modelling and Simulation (CIMSiM)*, Fourth International Conference on, pp. 13–17

[4] Yazdani S, Nezamabadi-pour H and Kamyab S 2014 A gravitational search algorithm for multimodal optimization. *Swarm Evolutionary Comput.* 14: 1–14

[5] Rashedi E, Nezamabadi-Pour H and Saryazdi S 2010 BGSA: binary gravitational search algorithm. *Nat. Comput.* 9: 727–745

[6] Mirjalili S, Wang GG and Coelho LS 2014 Binary optimization using hybrid particle swarm optimization and gravitational search algorithm. *Neural Comput. Appl.* 25: 1423–1435

[7] Ibrahim I, Ibrahim Z, Ahmad H et al 2015 An assembly sequence planning approach with a rule-based multi-state

gravitational search algorithm. *Int J. Adv. Manuf. Technol.* 79(5): 1363–1376

[8] Rashedi E, Nezamabadi-pour H and Saryazdi S 2009 GSA: a gravitational search algorithm. *Inf. Sci.* 179: 2232–2248

[9] Formato R A 2007 Central force optimization: a new metaheuristic with applications in applied electromagnetics. *Prog. Electromagn. Res.* 77: 425–491 2007

[10] Haupt R L and Haupt S E 2004 *Practical genetic algorithms*, 2nd ed. Hoboken, N.J.: Wiley

[11] Kennedy J and Eberhart R 1995 Particle swarm optimization. In: *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948

[12] Yang X S 2010 *Nature-inspired metaheuristic algorithms*, 2nd edn. UK: Luniver Press

[13] Moghadam M S, Nezamabadi-Pour H and Farsangi M M 2014 A quantum inspired gravitational search algorithm for numerical function optimization. *Inf. Sci.* 267: 83–100

[14] Jiang S, Wang Y and Ji Z 2014 Convergence analysis and performance of an improved gravitational search algorithm. *Appl. Soft Comput.* 24: 363–384

[15] Sarafrazi S, Nezamabadi-Pour H and Saryazdi S 2011 Disruption: a new operator in gravitational search algorithm. *Sci. Iran.* 18(3): 539–548

[16] Hari Ginardi R V and Izzah A 2014 A new operator in gravitational search algorithm based on the law of momentum. In: *Proceedings of the International Conference on Information, Communication Technology and System*, pp. 105–110

[17] Mirjalili S and Lewis A 2014 Adaptive gbest-guided gravitational search algorithm. *Neural Comput. Appl.* 25: 1569–1584

[18] Shang Z 2013 Neighborhood crossover operator: a new operator in gravitational search algorithm. *Int. J. Comput. Sci. Issues* 10(5): 116–126

[19] Farivar F and Shoorehdeli M A 2016 Stability analysis of particle dynamics in gravitational search optimization algorithm. *Inf. Sci.* 337–338: 25–43

[20] Saeidi-Khabisi F S and Rashedi E 2012 Fuzzy gravitational search algorithm. In: *Proceedings of the International e-Conference on Computer and Knowledge Engineering*, pp. 156–160

[21] Olivas F, Valdez F and Castillo O 2016 A fuzzy system for dynamic parameter adaptation in gravitational search algorithm. In: *Proceedings of the 2016 IEEE 8th International Conference on Intelligent Systems*, pp. 146–151

[22] Precup R E, David R C, Petriu E M, Preitl S and Radac M B 2013 Fuzzy logic-based adaptive gravitational search algorithm for optimal tuning of fuzzy-controlled servo systems. *IET Control Theory Appl.* 7(1): 99–107

[23] Sombra A, Valdez F, Melin P and Castillo O 2013 A new gravitational search algorithm using fuzzy logic to parameter adaptation. In: *Proceedings of the 2013 IEEE Congress on Evolutionary Computation, CEC 2013*, pp. 1068–1074

[24] Moghadam M S, Nezamabadi-Pour H and Farsangi M M 2012 A quantum behaved gravitational search algorithm. *Intell. Inf. Manage.* 4: 711–714

[25] Liu C and Ouyang C 2010 An adaptive fuzzy weight PSO algorithm. In: *Proceedings of the Fourth International Conference on Genetic and Evolutionary Computing*, pp. 8–10

[26] Rodriguez L, Castillo O and Soria J 2016 Grey wolf optimizer with dynamic adaptation of parameters using fuzzy logic. In: *Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 3116–3123

[27] Perez J, Valdez F and Castillo O 2015 Modification of the bat algorithm using fuzzy logic for dynamical parameter adaptation. In: *Proceedings of the 2015 IEEE Congress on Evolutionary Computation, CEC 2015*, pp. 464–471

[28] De A, Mamanduru V K R, Gunasekaran A, Subramanian N and Tiwari M K 2016 Composite particle algorithm for sustainable integrated dynamic ship routing and scheduling optimization. *Comput. Ind. Eng.* 96: 201–215

[29] Binkley K J and Hagiwara M 2008 Balancing exploitation and exploration in particle swarm optimization: velocity-based reinitialization. *Trans. J. Soc. Artif. Intell.* 23(1): 27–35

[30] Budhraja K K, Singh A, Dubey G and Khosla A 2013 Exploration enhanced particle swarm optimization using guided re-initialization. *Adv. Intell. Syst. Comput.* vol. 20 pp. 277–288

[31] Guo J and Tang S J 2009 An improved particle swarm optimization with re-initialization mechanism. In: *Proceedings of the International Conference on Intelligent Human–Machine Systems and Cybernetics*, pp. 437–441

[32] Mavrovouniotis M and Yang S 2013 Ant colony optimization with re-initialization. *Autom. Control Intell. Syst.* 1(3): 371–380

[33] Kaucic M 2013 A multi-start opposition-based particle swarm optimization algorithm with adaptive velocity for bound constrained global optimization. *J. Global Optim.* 55(1): 165–188

[34] Riget J and Vesterstrøm J S 2002 A diversity-guided particle swarm optimizer—the ARPSO. Technical Report

[35] Coelho L S and Mariani V C 2008 Particle swarm approach based on quantum mechanics and harmonic oscillator potential well for economic load dispatch with valve-point effects. *Energy Convers. Manage.* 49(11): 3080–3085

[36] Huang Z, Wang Y, Yang C and Wu C 2009 A new improved quantum-behaved particle swarm optimization model. In: *Proceedings of the 2009 4th IEEE Conference on Industrial Electronics and Applications*, pp. 1560–1564

[37] Daoud E A 2015 Quantum meta-heuristic algorithm based on harmony search. *Int. J. Eng. Sci. Invent.* 4(10): 13–18

[38] Engelbrecht A P 2013 Particle swarm optimization with discrete crossover. In: *Proceedings of the IEEE Congress on Evolutionary Computation*, number 2, pp. 2457–2464

[39] Engelbrecht A P 2014 Asynchronous particle swarm optimization with discrete crossover. In: *Proceedings of the 2014 IEEE Symposium on Swarm Intelligence*, pp. 1–8

[40] Engelbrecht A P 2015 Particle swarm optimization with crossover: a review and empirical analysis. *Artif. Intell. Rev.* 45(2): 131–165

[41] Higashi N and Iba H 2003 Particle swarm optimization with Gaussian mutation. In: *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, SIS'03 (Cat. No.03EX706)*, pp. 72–79

[42] Zhao N, Wu Z, Zhao Y and Quan T 2010 Ant colony optimization algorithm with mutation mechanism and its applications. *Expert Syst. Appl.* 37(7): 4805 – 4810

[43] Alexandridis A, Chondrodima E and Sarimveis H 2016 Cooperative learning for radial basis function networks using particle swarm optimization. *Appl. Soft Comput.* 49: 485–497

[44] Soleimani H and Kannan G 2014 A hybrid particle swarm optimization and genetic algorithm for closed-loop supply chain network design in large-scale networks. *Appl. Math. Model.* 39(14): 3990–4012

[45] Osman I H and Laporte G 1996 Metaheuristics: a bibliography. *Ann. Oper. Res.* 63(5): 513–628

[46] Aziz N A A, Mubin M, Ibrahim Z and Nawawi S W 2014 Performance and diversity of gravitational search algorithm. *Adv. Appl. Converg. Lett.* 3(1): 232–235

[47] Aziz N A A, Ibrahim Z, Nawawi S W, Ibrahim I, Tumari M Z M and Mubin M 2013 Synchronous vs asynchronous gravitational search algorithm. In: *Proceedings of the First International Conference on Artificial Intelligence, Modelling and Simulation*, pp. 29–34

[48] Liang J J, Qu B Y and Suganthan P N 2013 *Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization*. Technical Report

[49] Voglis C A, Parsopoulos K E and Lagaris I E 2012 Particle swarm optimization with deliberate loss of information. *Soft Comput.* 16(8): 1373–1392

[50] Mirjalili S, Mirjalili S M and Lewis A 2014 Grey wolf optimizer. *Adv. Eng. Softw.* 69: 46–61

[51] Cheng S and Shi Y 2011 Diversity control in particle swarm optimization. In: *Proceedings of the IEEE Symposium on Swarm Intelligence*, pp. 1–9