Swarm and Evolutionary Computation 28 (2016) 58–77

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

Ageist Spider Monkey Optimization algorithm

Avinash Sharma [a], Akshay Sharma [b], B.K. Panigrahi [c,*], Deep Kiran [c], Rajesh Kumar [a]

[a] Malaviya National Institute of Technology, Jaipur, India
[b] National Institute of Technology, Surathkal, India
[c] Indian Institute of Technology Delhi, New Delhi, India

ARTICLE INFO

Article history:
Received 15 July 2015
Received in revised form
12 December 2015
Accepted 18 January 2016
Available online 18 February 2016

Keywords:
Swarm intelligence
Spider Monkey Optimization
Artificial systems
Numerical optimization
Greedy search

ABSTRACT

Swarm Intelligence (SI) is quite popular in the field of numerical optimization and has enormous scope for research. A number of algorithms based on decentralized and self-organized swarm behavior of natural as well as artificial systems have been proposed and developed in last few years. Spider Monkey Optimization (SMO) algorithm, inspired by the intelligent behavior of spider monkeys, is one such recently proposed algorithm. The algorithm along with some of its variants has proved to be very successful and efficient.

A spider monkey group consists of members from every age group. The agility and swiftness of the spider monkeys differ on the basis of their age groups. This paper proposes a new variant of SMO algorithm termed as Ageist Spider Monkey Optimization (ASMO) algorithm which seems more practical in biological terms and works on the basis of age difference present in spider monkey population. Experiments on different benchmark functions with different parameters and settings have been carried out and the variant with the best suited settings is proposed. This variant of SMO has enhanced the performance of its original version. Also, ASMO has performed better in comparison to some of the recent advanced algorithms.

1. Introduction

A metaheuristic refers to a high level problem independent framework which helps to develop heuristic optimization algorithms [1]. Any approach to problem solving, learning or discovery which focuses on immediate near optimality rather than exact results, using practical methods can be termed as a heuristic. Metaheuristics are developed scientifically to find a solution that is “good enough” in a computing time that is “small enough” [2–4]. The present trend to use heuristic techniques over exact ones is due to fact that many real world problems have been shown to remain forever intractable to exact algorithms, regardless of the ever increasing computational power, simply due to unrealistically large running times [5]. History and various trends related to metaheuristics are mentioned in [5]. One such approach is SI which is a result of collective behavior of different agents present in the population.

SI is a discipline which deals with artificial and natural systems, these systems are composed of swarms of homogeneous individuals and instead of everyone depending on a single central unit, all units are self-organized and they cooperate and share information to carry out their necessary tasks. The collective behavior of the individuals resulted from local interactions with each other and their environment is known as swarm intelligence. It is a metaheuristic approach which makes use of nature inspired techniques to solve optimization problems, the term was introduced by Gerardo Beni in 1989 [6], in the context of cellular robotic systems. A number of natural systems are studied under SI like schools of fish, ant colonies, bird flocks, bee colonies, herds of animals, etc. The engineering application of swarm intelligence is to exploit the understanding of the systems and design systems to solve problems of practical relevance.

The recent advancements in SI have shown its tremendous capability in solving complex problems which otherwise is impossible to solve with other naive approaches and therefore has great application in artificial intelligence. A lot of research has been done and is still

* Corresponding author.
E-mail addresses: avinashmnit30@gmail.com (A. Sharma), ksh.shrm1@gmail.com (A. Sharma), bkpanigrahi@ee.iitd.ac.in (B.K. Panigrahi), dkiran5@gmail.com (D. Kiran), rkumar.ee@gmail.com (R. Kumar).

http://dx.doi.org/10.1016/j.swevo.2016.01.002
2210-6502/

going on to further improve the potential of SI in solving real time optimization problems. A number of nature inspired algorithms like ant colony optimization (ACO) [7] and particle swarm optimization (PSO) [8], artificial bee colony optimization (ABC) [9], bacterial foraging optimization (BFO) [10] has been proposed. These belong to the classes that are based on population, intelligent foraging behavior, social foraging behavior and many more. Early studies [10] of swarm behavior employed mathematical models to simulate and understand the swarm behavior. Three basic rules composing simplest mathematical model are:

- Move in the same direction as your neighbors.
- Remain close to your neighbors.
- Avoid collisions with your neighbors.

Craig Reynolds created programs called *boids* [1] in 1986, these programs simulate the swarm behavior following the above rules. Many current simulation models implement swarm behavior by means of concentric *zones* around each individual like zones of repulsion, alignment and attraction. Researchers, in order to find out as to why animals show swarm behavior, have been developing and studying evolutionary models simulating the population of evolving algorithms. Researchers have developed many algorithms and their improvements in recent years. Among them are various improvements of previously proposed evolutionary and swarm intelligence inspired algorithms.

Yu et al. [11] proposed enhanced comprehensive learning particle swarm optimization (ECLPSO) which improved the performance of CLPSO [12] by introducing perturbation rate and adaptive particle probability to the original algorithm. SP-PSO and SG-PSO [13] consider the effect of second best personal and global position for updating positions of other particles, respectively. Superior solution guided particle swarm optimization (SSG-PSO) [14] maintains and updates a collection of superior positions for updating positions of particles in the swarm. Scatter learning particle swarm optimization (SLPSO) [15] creates a pool of high quality solution scattered throughout search space called exemplar pool that makes particles to select their exemplars from the pool using the roulette wheel rule.

Recent research tries to improve performance of PSO by incorporating various elements of human learning principles within them. Social learning PSO (SL-PSO) [16] introduces a social learning mechanism into PSO such that particle position is updated based on historical information. To empower the searching particles with human like characteristics dynamic mentoring and self-regulation based PSO (DMeSR-PSO) [17] algorithm incorporates a dynamic mentoring scheme along with a self-regulation scheme in the classical PSO algorithm. Competitive and cooperative PSO with ISM (CCPSO-ISM) [18] proposes an information sharing mechanism (ISM) to improve the performance of PSO. Self-regulating particle swarm optimization (SRPSO) [19] algorithm incorporates best human learning strategies within PSO for finding the optimum solution. Adaptive division of labor (ADOL) PSO (ADOLPSO) [20] adopts two new operators, convex operator and reflectance operator to generate new particles from the memory swarm.

Differential Evolutionary (DE) [21] algorithm is an evolutionary search heuristic proposed by Storn and Price in 1995. To improve its performance, Jana et al. proposed Levy distributed DE (LdDE) [22] which control each of its parameters by levy distribution. DE with auto-enhanced population diversity (AEPD-JADE) [23] is proposed to identify the moments when a population becomes converging or stagnating by measuring the distribution of the population in each dimension. Harmony search algorithm [24] is a metaheuristic optimization method developed by Geem et al. imitating the music improvisation process where musicians improvise pitch of their instruments by searching for a perfect state of harmony. Valian et al. proposed IGHS [25] algorithm which presents an improved harmony search algorithm using the swarm intelligence technique.

Gao et al. proposed artificial bee colony algorithm based on information learning (ILABC) [26] which divides the whole population into sub-populations and dynamically adjusts size of sub-population. In enhanced artificial bee colony (EABC) [27] algorithm, two new search equations are presented to generate candidate solutions in the employed bee phase and the onlookers phase, respectively.

Inspired by the behavior of spider monkeys, Bansal et al. proposed an algorithm based on fission–fusion social structure. This algorithm is known as spider monkey optimization (SMO) [28] mimics the social behavior of a south American species of monkeys called spider monkeys, those belong to the class of nature inspired algorithms (NIA) [6]. The necessary principles of intelligent behavior are implemented in the social behavior of monkeys that are *self-organizing* in foraging behavior of monkeys while searching for food or mating and *division of labor* to divide the main group into subgroups for independent foraging. The fitness of the monkey at some particular position refers to its nearness to the global optimum value required, decides the superiority of food and affects behavior of other spider monkeys. The two main parts of an optimization problem, i.e. exploration and exploitation, need to be balanced. While searching for optimum solution the algorithm maintains the balance between deviation and selection processes which ensure exploration and exploitation, respectively.

Recently published modified variants of SMO have shown improvement in its performance, i.e. modified position update in spider monkey optimization (MPU-SMO) [29] that makes use of golden section search (GSS) to enhance performance of SMO. Kumar et al. proposed self-adaptive SMO (Sa-SMO) [30] with algorithm parameters being self-adaptive in nature and tournament selection based spider monkey optimization (TS-SMO) [31] proposed by Gupta et al. replaces the fitness proportionate probability scheme of SMO with tournament selection based probability scheme with an objective.

This paper proposes a new variant of SMO called as Ageist SMO (ASMO) which works on the basis of the fact that not all monkeys in the population are alike; they belong to different age groups and have different levels of activity. Some monkeys are more expeditious than others and, therefore, behave differently from others.

The rest of the paper is organized as follows: introduction is followed by Section 2 that contain details of SMO algorithm, proposed approach of the algorithm is explained in Section 3. A detailed analysis on different benchmark functions for clear understanding and comparison is given in Section 4. Section 5 concludes the paper on the basis of results obtained.

2. Spider monkey Optimization

A new swarm intelligence algorithm is proposed in terms of fission fusion social structure (FFSS) as these monkeys fall in the category of FFSS based animals. This form of social organization occurs in several species of primates (e.g. common chimpanzees and bonobos, hamadryas baboons, geladas, orangutans, spider monkeys, and humans), African elephants, most carnivores and fishes.

2.1. Social behavior of spider monkeys

Spider monkeys follow FFSS in which they form temporary small subgroups, whose members belong to large stable communities. The composition and size of these subgroups changes frequently due to fluid movement between these groups. The members of these subgroups then communicate through barking and other physical activities depending on the availability of food. In this type of society, the parent subgroup can fission into smaller subgroups and can also fuse again into one big group depending on the environmental or social circumstances. These subgroups are led by a *female leader* for searching food which split the subgroups when there is scarcity of food. The main group generally has around 50 members initially and subgroups have at least 3 members. They show territorial behavior after splitting into subgroups to ensure no physical contact.

2.2. Spider Monkey Optimization algorithm

SMO algorithm based of FFSS consists of four basic steps:

1. The group starts foraging and evaluate their distance from the food sources which is termed as the fitness of the monkeys.
2. Based on the fitness of individuals, group members update their positions and then again evaluate the fitness.
3. Local leader (LL) updates its position, i.e. the best position in the group and if the position remains unchanged for a predefined number of times then the group is scattered depending on the perturbation rate (pr).
4. Global leader (GL) updates its position, i.e. the best position among all the monkeys and in case of stagnation; the groups are split into subgroups. If the total number of groups present exceeds the maximum group (MG) limit then all the subgroups are fused into the parent group.

The above steps are continuously executed until the termination criterion is met. Two necessary control parameters in this proposed strategy are *localleaderlimit* and *globalleaderlimit* which are used to avoid stagnation in local and global position updates, respectively. If LL does not update its position in specified number of times then the group is redirected to a different direction for foraging. If GL fails to update its position after a specified number of times then the group is split for independent foraging.

2.2.1. Major steps of SMO algorithm

SMO, like other population based algorithms, is also a trial and error based collaborative iterative process where the algorithm tries to reach to the optimum value in minimum number of iterations. The SMO algorithm is divided into six major phases or steps described as follows:

1. Population initialization: A randomly distributed population P of spider monkeys is initialized. Each monkey is a D dimensional vector SM_i ($i = (1, 2, .., P)$), where D represents the number of variables in the optimization problem and SM_i refers to the ith spider monkey in the population. Each SM_i is initialized as:

$$SM_{ij} = SM_{minj} + R_u(0, 1) \times (SM_{maxj} - SM_{minj}) \tag{1}$$

where, SM_{minj} and SM_{maxj} are lower and upper bounds of SM_i in jth ($j = \{1, 2, .., D\}$) dimension respectively and $R_u(0, 1)$ is a uniformly distributed random number in the range [0,1].

2. Local Leader Phase (LLP): In this phase, spider monkeys update their position based on the experience of LL as well as other members of the group. The fitness value of the newly obtained position is calculated and if the fitness value of the new position is more optimum than the old position, then the SM is updated with new position. For ith SM of kth subgroup:

$$SM_{newij} = SM_{ij} + R_u(0, 1) \times (LL_{kj} - SMij) + R_u(-1, 1) \times (SM_{rj} - SM_{ij}) \tag{2}$$

where, SM_{ij} is the ith SM in jth dimension, LL_{kj} represents the jth dimension of the kth local group leader position and SM_{rj} is the rth SM chosen randomly from the kth group such that $r \neq i$.

Algorithm 1. Position update in LLP.

procedure LLP
  for each $k \in \{1, 2\ldots, MG\}$ **do**
    for each member $SM_i \in k$th group **do**
      for each $j \in \{1, 2, \ldots, D\}$ **do**
        if $R_u(0, 1) \geq pr$ **then**
          $SM_{new_{ij}} \leftrightarrow SM_{ij} + R_u(0, 1) \times (LL_{kj} - SM_{ij}) + R_u(-1, 1) \times (SM_{rj} - SM_{ij})$

3. Global Leader Phase (GLP): GLP follows LLP where spider monkeys update their position based on the experience of GL and members of local group using (3).

$$SM_{newij} = SM_{ij} + R_u(0,1) \times (GL_j - SM_{ij}) + R_u(-1,1) \times (SM_{rj} - SM_{ij}) \tag{3}$$

where GL_j is the global leader's position in jth dimension and $j \in 1, 2, 3\ldots, D$ is the randomly chosen index. In this phase, the position update of spider monkeys is constrained by a probability value $prob_i$ which is calculated using their fitness, giving a higher chance to a better candidate to make itself better. Here, $prob_i$ is computed using (4).

$$prob_i = x \times \frac{fitness_i}{max\_fitness} + y \tag{4}$$

where, $fitness_i$ is the fitness of ith monkey. Here, $x + y = 1$ and optimum results are obtained at values $x=0.9$ and $y=0.1$.

Algorithm 2. Position update in GLP.

procedure GLP
  for $k=1$ to MG **do**
    $count \leftarrow 1$
    $GS \leftarrow k$th *group size*
      while $count < GS$ **do**
        for $i=1$ to GS **do**
          if $R_u(0,1) < prob_i$ **then**
            $count \leftarrow count + 1$
            Randomly select $j \in \{1, 2, \ldots D\}$
            Randomly select SM_r from kth group such that $r \neq i$
            $SM_{new_{ij}} \leftarrow SM_{ij} + R_u(0,1) \times (GL_j - SM_{ij}) + R_u(-1,1) \times (SM_{rj} - SM_{ij})$

4. Global Leader Learning Phase (GLL): GL updates its position by applying greedy selection process, *SM* having the best fitness among all the monkeys is selected as the new position of GL, and if the position of GL remains the same, *GlobalLimitCount* is increased by 1.

5. Local Leader Learning Phase (LLL): The position of LL of all the groups are updated by applying greedy selection process and then selecting the monkey *SM* having the best fitness in that group. If the LL's position remains same as before, then the *LocalLimitCount* is increased by 1.

6. Local Leader Decision Phase (LLD): If a LL position is not updated for a predetermined number of iterations i.e. *LocalLeaderLimit*, then the positions of the spider monkeys are updated either by random initialization as in step 1 or by using information from both LL and GL based on *pr* through (5).

$$SM_{newij} = SM_{ij} + R_u(0,1) \times (GL_j - SM_{ij}) + R_u(0,1) \times (SM_{ij} - LL_{kj}) \tag{5}$$

Algorithm 3. Local Leader Decision Phase.

procedure LLDP
  for $k=1$ to MG **do**
    if $locallimitcount_k > localleaderlimit$ **then**
      $locallimitcount_k \leftarrow 0$
      $GS \leftarrow k$th*groupsize*
      for $i=1$ to GS **do**
        for each $j \in \{1, 2, \ldots D\}$ **do**
          if $R_u(0,1) \geq pr$ **then**
            $SM_{new_{ij}} \leftarrow SM_{minj} + R_u(0,1) \times (SM_{maxj} - SM_{minj})$
          else
            $SM_{new_{ij}} \leftarrow SM_{ij} + R_u(0,1) \times (GL_j - SM_{ij}) + R_u(0,1) \times (SM_{ij} - LL_{kj})$

7. Global Leader Decision Phase (GLD): In this phase, the decision about GL position is taken, if the position of GL is not updated in predetermined number of iterations i.e. *globalleaderlimit*, then the population is split into subgroups. The groups are split till the number of groups reaches to maximum allowed groups (MG), then they are combined to form a single group again.

Algorithm 4. Global Leader Decision Phase.

procedure GLDP
  if $globallimitcount > globalleaderlimit$ **then**
    $globallimitcount \leftarrow 0$
    if Number of Groups $< MG$ **then**
      Split Group
    else
      Fuse all groups in one
    Update Local Leader positions

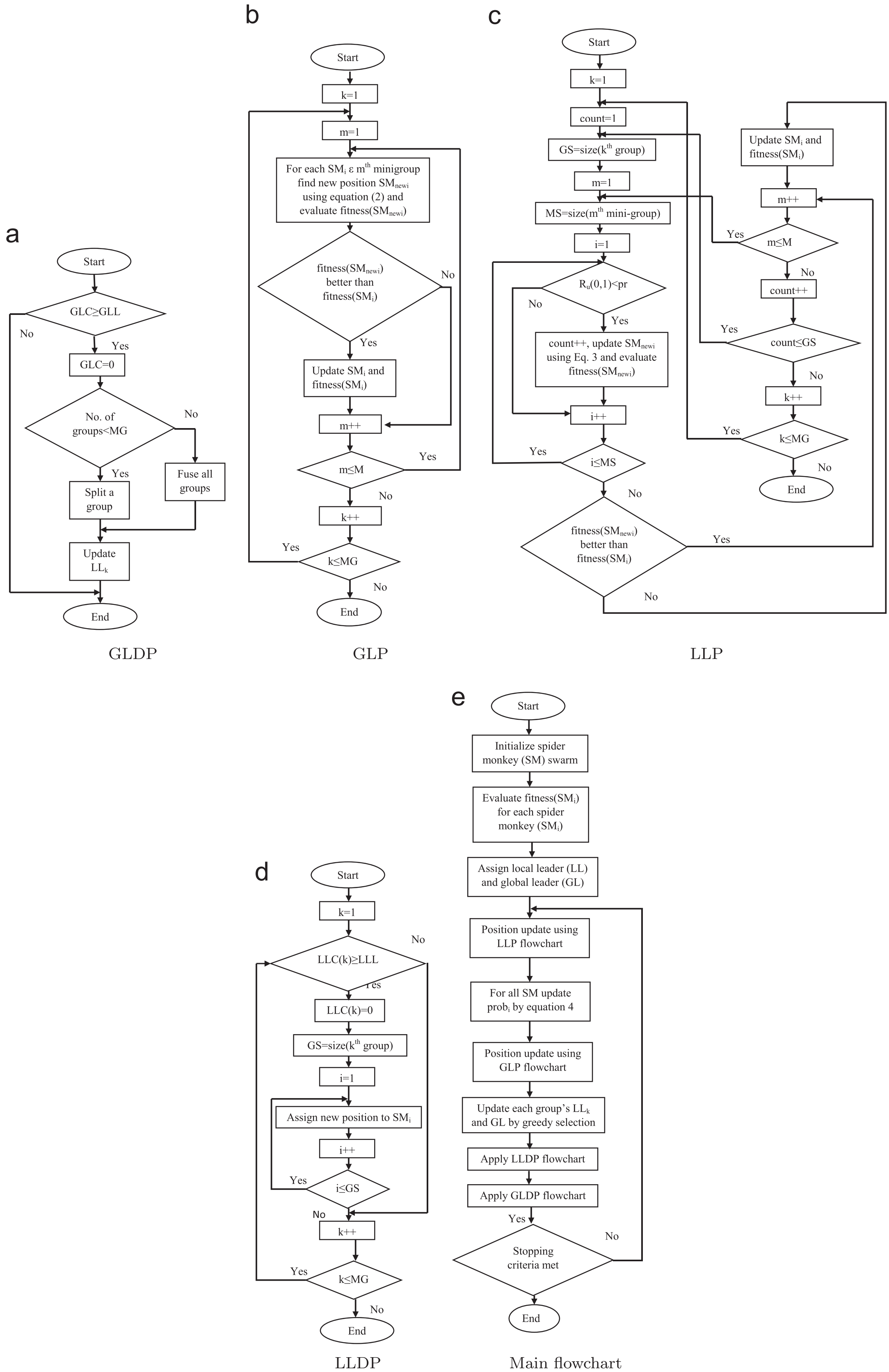

Fig. 1. AMSMO flowcharts.

Table 1
Benchmark function details.

Function name	*D*	Range	ME	Type	OV
	30	[−100,100]	1.00E−03	US	0
Elliptic (f_1)	50	[−100,100]	1.00E−03		0
	100	[−100,100]	1.00E−02		0
Ackley(f_2)	10	[−32,32]	1.00E−05	MS	0
	30	[−32,32]	1.00E−03		0
Weierstrass (f_3)	10	[−0.5,0.5]	1.00E−03	MS	0
	30	[−0.5,0.5]	5.00E−02		0
Step (f_4)	30	[−100,100]	0.00E+00	US	0
	30	[−100,100]	1.00E−03	US	0
Axis paralled hyper ellipsoid (f_5)	50	[−100,100]	1.00E−03		0
	100	[−100,100]	1.00E−02		0
Beale (f_6)	2	[−4.5,4.5]	1.00E−05	UN	0
Brain Rcos (f_7)	2	[−5,10], [0,15]	1.00E−06	MN	0
	30	[−10,10]	1.00E−05	US	0
Cigar (f_8)	50	[−10,10]	1.00E−03		0
	100	[−10,10]	1.00E−02		0
Dekkers and Aarts (f_9)	2	[−20,20]	5.00E−01	MN	−24777
Six Hump Camel Back (f_{10})	2	[−5,5]	1.00E−06	MN	−1.0316
Griewank (f_{11})	30	[−600,600]	1.00E−02	MN	0
	50	[−600,600]	1.00E−02		0
Goldstein price (f_{12})	2	[−2,2]	1.00E−06	MN	3
	30	[−100,100]	1.00E−03	US	0
Discus (f_{13})	50	[−100,100]	1.00E−03		0
	100	[−100,100]	1.00E−02		0
Trid (f_{14})	6	[−36,36]	1.00E−05	UN	−50
	10	[−100,100]	1.00E−05		−210
Holder Table (f_{15})	2	[−10,10]	1.00E−20	MN	−19.2085
Drop Wave (f_{16})	2	[−5.12,5.12]	1.00E−05	MN	−1
Hartmann 3D (f_{17})	3	[0,1]	1.00E−06	MN	−3.86218
Levy (f_{18})	10	[−10,10]	1.00E−05	MN	0
Shubert (f_{19})	2	[−10,10]	1.00E−05	MN	−186.731
Shifted Schwefel 1.2 (f_{20})	30	[−100,100]	1.00E+00	UN	0
	50	[−100,100]	5.00E+02		0
Shifted Elliptic (f_{21}	50	[−100,100]	1.00E−03	US	0
	100	[−100,100]	1.00E−02		0
Shifted Rastrigin (f_{22})	50	[−5,5]	1.00E−03	MS	0
Corner Shifted Schwefel 1.2 (f_{23})	50	[−100,100]	1.00E+00	UN	0
	100	[−100,100]	1.00E+01		0
Corner Shifted Ackley (f_{24})	30	[−32,32]	1.00E−02	MS	0
	50	[−32,32]	1.00E−02		0
Corner Shifted Elliptic (f_{25})	50	[−100,100]	1.00E−03	US	0
	100	[−100,100]	1.00E−02		0
Hybrid Sphere Rosenbrock (f_{26})	10	[−5,10]	7.50E−01	MN	0
	30	[−5,10]	7.50E−01		0
Katsuura (f_{27})	30	[−100,100]	1.00E−03	MS	0
	50	[−100,100]	1.00E−03		0
	100	[−100,100]	1.00E−03		0
Treccani (f_{28})	2	[−5,5]	1.00E−20	UN	0
Shifted Rotated Rastrigin (f_{29})	10	[−100,100]	1.00E−03	MN	0
	30	[−100,100]	1.00E−03		0
Hybrid Sphere Rastrigen (f_{30})	30	[−5,5]	0.00E+00	MN	0
	50	[−5,5]	0.00E+00		0

Parameter Settings:

Algorithm	*Algorithm Specifications*
SMO	GLL=20 LLL=500 SS=40 MG=4
ASMO	GLL=20 LLL=500 SS=32 MG=4
AMSMO	GLL=20 LLL=500 SS=32 MG=4

Abbrevations:

SMO	Spider Monkey Optimization
ASMO	Ageist SMO

AMSMO	Ageist modified SMO
AS4	ASMO with 4 mgrp
AM4	AMSMO with 4 mgrp
AS8	ASMO with 8 mgrp
AM8	AMSMO with 8 mgrp
D	Dimensions
M.E.	Max. tolerable error
AI	Average iterations
AFE	Average fun. evaluation
AE	Average error
SR	Success ratio
US	Unimodal seperable
MS	Multimodal seperable
UN	Unimodal nonseperable
MN	Multimodal nonseperable
OV	Optimum value
GLL	Global leader limit
LLL	Local leader limit
SS	Swarm size
MG	Maximum groups
mgrp/M	No. of mini groups

Algorithm—SMO

Step 1: Initialize spider monkey population (Eq. (1)), control parameters (*localleaderlimit* and *globalleaderlimit*), and perturbation rate (pr).

Step 2: Fitness evaluation, calculate the distance of individuals from food sources or the function value at each monkey's position with variables as parameter values in respective dimensions.

Step 3: Update LL and GL by greedy selection process. In greedy selection process best among the given set is chosen (as explained above).

Step 4: While (terminating condition is false) do

Step 4.1: Position update for all the spider monkeys based on LLP (Algorithm 1) i.e. self, LL and group members' experience.

Step 4.2: Selection of better position between the newly generated and the existing one based on fitness and applying greedy selection process.

Step 4.3: Calculate the probability $prob_i$ for all the group members using Eq. (4).

Step 4.4: Position update for all the group members selected by $prob_i$ based on GLP (Algorithm 2) i.e. self, GL and group members' experience.

Step 4.5: Update LL and GL positions by applying greedy selection process on the entire group members.

Step 4.6: If any LL is not updating its position for a predefined number of iterations then redirect all the group members using local leader decision phase as given in Algorithm 3 (foraging algorithm).

Step 4.7: If GL is not updating position for predefined number of iterations then the group is divided, if number of groups present is less than MG else all the subgroups combine to form one single group. This is done by global leader decision phase (Algorithm 4).

end while

2.3. Problems with SMO algorithm

In the original SMO algorithm, the position of each spider monkey is updated depending upon the position of another randomly selected spider monkey in LLP and GLP. This update is irrespective of whether the position of randomly selected monkey is better or not. This leads to low convergence rate further causing high rate group breaking and merging. To tackle problem of low convergence rate, new algorithm is proposed as described in the next section.

3. Modified approach—ASMO

The intelligent behavior of spider monkeys lies behind their fission–fusion based foraging behavior. The spider monkey population shows features like self-organization and division of labor, which are the necessary and sufficient conditions for swarm intelligence behavior. While searching for food, the monkeys interact with their group members, LL as well as GL and update their positions according to the information they get from others.

Now as these monkeys belongs to different age groups, i.e. young, adult and old monkeys. Among which younger monkeys will be faster and more efficient in interacting and updating their positions, than other old and mentally or physically disabled monkeys. These faster monkeys will interact and update their positions (to increase their fitness) before the slower ones and will provide them with better experience with greedily selected positions. Considering this fact and looking at the original SMO algorithm, which updates positions of monkeys assuming they have same interacting and exploring abilities, a variant of SMO algorithm is proposed which is as follows:

This modified algorithm called as ASMO works on the basis of age and dynamical differences between existing monkeys in the group. The strategy is to further divide groups of spider monkeys into mini-groups which can be interpreted as age groups in biological terms. These mini-groups is divided from the group on the basis of different levels of ability to interact and to track changes in the environment and all the monkeys in the mini-group will have the same level of abilities. While updating position of monkeys, the monkeys of best mini-group will update their position first and communicate it to the other monkeys which improve the convergence rate of monkeys towards optimum solution.

3.1. ASMO Algorithm

The position update of monkeys in both GLP and LLP involves using experience of other monkeys in the group along with GL and LL in respective phases.

The idea is to divide groups of spider monkeys into M number of mini-groups, value of M can be set manually and remains constant throughout. Instead of updating positions of all the monkeys of the group and then selecting better position between the previous and the new one by applying greedy selection based on the fitness, the above steps are executed for one mini-group and then it switches to next mini-group in that group (Algorithm 5).

Similar to LLP, Algorithm 5 ageist strategies can also be implemented in GLP as implemented in Algorithm 6. ASMO implements ageist strategy in only LLP. While implementing this in both LLP and GLP gives AMSMO. Stated algorithms (Algorithms 5 and 6) are replacements for Algorithms 1 and 2 of the original SMO respectively. By using Algorithm 5 in place of Algorithm 1 in step 4.1 ASMO algorithm can be implemented. By further replacing Algorithm 2 by Algorithm 6 in step 4.4 we can implement ageist variant of modified spider monkey algorithm called AMSMO algorithm. Modified SMO algorithm involves greed based selection in group leader based position update step of original algorithm.

The main SMO remains the same with the removal of step 4.2 i.e. greedy selection process for choosing a better position as we have already included that part in our modified algorithm ASMO as well as AMSMO. Flowchart of the proposed algorithm is given in Fig. 1.

Algorithm 5. Position update in ASMO.

procedure LLP
  for each $k \in \{1, 2\ldots, MG\}$ **do**
    for each $m \in \{1, 2, \ldots M\}$ **do**
      for each member $SM_i \in$ mth mini-group **do**
        for each $j \in \{1, 2, \ldots, D\}$ **do**
          if $R_u(0, 1) \geq pr$ **then**
            $SM_{new_{ij}} \leftarrow SM_{ij} + R_u(0, 1) \times (LL_{kj} - SM_{ij}) + R_u(-1, 1) \times (SM_{rj} - SM_{ij})$
      for each member $SM_i \in m$th mini-group **do**
        calculate $fitness_{new}$
        if $fitness_{newi}$ is better than $fitness_i$ **then**
          for each $j \in \{1, 2, \ldots, D\}$ **do**
            $SM_{ij} \leftarrow SM_{new_{ij}}$
          $fitness_i \leftarrow fitness_{newi}$

Algorithm 6. Modified position update in AMSMO.

procedure GLP
  for $k=1$ to MG **do**
    $count \leftarrow 1$
    $GS \leftarrow k$th group size
    while $count < GS$ **do**
      for $m=1$ to M **do**
        $MS \leftarrow m$th mini-group size
        for $i=1$ to MS **do**
          if $R_u(0, 1) < prob_i$ **then**
            $count \leftarrow count + 1$
            Randomly select $j \in \{1, 2, \ldots D\}$
            Randomly select SM_r from kth group such that $r \neq i$
            $SM_{new_{ij}} \leftarrow SM_{ij} + R_u(0, 1) \times (GL_j - SM_{ij}) + R_u(-1, 1) \times (SM_{rj} - SM_{ij})$
        for each member $SM_i \in m$th mini-group **do**
          calculate $fitness_{new}$
          if $fitness_{newi}$ is better than $fitness_i$ **then**
            for each $j \in \{1, 2, \ldots, D\}$ **do**
              $SM_{ij} \leftarrow SM_{new_{ij}}$
            $fitness_i \leftarrow fitness_{newi}$

3.2. Algorithm Logic

Position update phases for spider monkeys (Algorithms 1 and 2), while generating new position uses a random spider monkey's experience from that group. In (2) and (3), a random monkey SM_r is selected from the group and its position is used, $R_u(-1, 1) \times (SM_{rj} - SM_{ij})$ is added to the previous position along with LL and GLs experience. If the random number generated by R_u is

Table 2
Comparison between proposed SMO variants and SMO algorithm for function f_1—f_{13}.

		SMO	ASMO (M=4)	AMSMO (M=4)	ASMO (M=8)	AMSMO (M=8)
f_1	AI	432.87	286.8	169	273.1	163.07
$D=30$	AFE	17 315	**9177.6**	**10 816**	**8739.2**	**10 436**
	AE	9.02E−04	8.74E−04	9.30E−04	8.52E−04	9.10E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_1	AI	1215.2	511.07	311.5	515.33	281.5
$D=50$	AFE	48 607	**16 354**	**19 936**	**16 491**	**18 016**
	AE	9.30E−04	9.02E−04	8.98E−04	9.63E−04	9.05E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_1	AI	4136.6	1411.9	617.33	1187.7	639.5
$D=100$	AFE	165 464	**45 182**	**39 509**	**38 005**	**40 928**
	AE	9.45E−03	9.86E−03	9.87E−03	9.76E−03	9.93E−03
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_2	AI	232.8	155.6	105.7	151.67	100
$D=10$	AFE	9312	**4979.2**	6764.8	**4853.4**	6400
	AE	9.12E−06	9.11E−06	9.23E−06	8.95E−06	9.15E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_2	AI	600.5	512.77	202.4	457.4	196.6
$D=30$	AFE	24 020	16 409	**12 954**	14 637	**12 582**
	AE	**9.62E−04**	8.80E−01	**9.22E−04**	4.58E−01	**9.21E−04**
	SR	**100.00**%	40.00%	**100.00**%	60.00%	**100.00**%
f_3	AI	223.17	162.6	99.8	160.6	99
$D=10$	AFE	8926.7	**5203.2**	**6387.2**	**5139.2**	**6336**
	AE	9.62E−04	9.47E−04	9.48E−04	9.56E−04	9.61E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_3	AI	775.2	399	217.6	421.33	193.27
$D=30$	AFE	31 008	**12 768**	**13 926**	**13 483**	**12 369**
	AE	4.89E−02	4.78E−02	4.83E−02	4.88E−02	4.68E−02
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_4	AI	329.8	692.23	264.1	952.47	275.5
	AFE	**13 192**	22 151	**16 902**	30 479	**17 632**
	AE	**0.00E+00**	1.93E+00	9.00E−01	9.30E+00	7.00E−01
	SR	**100.00**%	40.00%	67.00%	3.33%	60.00%
f_5	AI	528.17	364.57	211.33	358	214.27
$D=30$	AFE	21 127	**11 666**	**13 525**	**11 456**	**13 713**
	AE	9.44E−04	9.46E−04	9.23E−04	9.31E−04	9.27E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_5	AI	1487.7	651.5	380.93	648.1	375.33
$D=50$	AFE	59 508	**20 848**	**24 380**	**20 739**	**24 021**
	AE	9.28E−04	9.61E−04	9.66E−04	9.21E−04	9.45E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_5	AI	5129.9	1793	747	1624.1	754.03
$D=100$	AFE	205 196	**57 375**	**47 808**	**51 972**	**48 258**
	AE	9.74E−03	9.68E−03	9.71E−03	9.47E−03	9.68E−03
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_6	AI	60	61.9	26.1	82.233	29.033
	AFE	2400	1980.8	**1670.4**	2631.5	**1858.1**
	AE	8.11E−06	7.47E−06	8.34E−06	8.12E−06	7.79E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_7	AI	74.4	35.167	27	45.9	25.6
	AFE	2976	**1125.3**	1728	1468.8	1638.4
	AE	7.97E−07	7.88E−07	8.78E−07	7.98E−07	8.12E−07
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_8	AI	620.47	395.33	219.4	356.33	210
$D=30$	AFE	24 819	**12 651**	**14 042**	**11 403**	**13 440**
	AE	9.12E−06	8.69E−06	8.78E−06	8.87E−06	8.79E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_8	AI	1474.4	708.33	381.1	670.13	398.77
$D=50$	AFE	58 977	**22 667**	**24 390**	**21 444**	**25 521**
	AE	9.23E−04	9.33E−04	9.45E−04	9.38E−04	9.58E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_8	AI	5606.1	2000.3	819.47	1604	779.37
$D=100$	AFE	224 244	64 010	**52 446**	**51 327**	**49 879**
	AE	9.73E−03	9.68E−03	9.77E−03	9.74E−03	9.71E−03
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_9	AI	30.233	23.667	12.233	23.267	12.7
	AFE	1209.3	**757.33**	**782.93**	**744.53**	**812.8**

Table 2 (*continued*)

		SMO	*ASMO* (*M*=4)	*AMSMO* (*M*=4)	*ASMO* (*M*=8)	*AMSMO* (*M*=8)
	AE	4.97E−01	4.94E−01	4.92E−01	4.96E−01	4.91E−01
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{10}	AI	26.9	18.333	10.2	21.467	12.133
	AFE	1076	**586.67**	**652.8**	**686.93**	**776.53**
	AE	9.61E−07	9.64E−07	9.23E−07	9.31E−07	9.33E−07
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{11}	AI	526.67	245	151.73	249.5	147.67
$D=30$	AFE	21 067	**7840**	**9710.9**	**7984**	**9450.7**
	AE	8.75E−03	9.23E−03	9.07E−03	9.78E−03	9.50E−03
	SR	**100.00**%	75.00%	**100.00**%	70.00%	**100.00**%
f_{11}	AI	1218.3	834.27	249.9	854.77	266.5
$D=50$	AFE	48 732	26 697	**15 994**	27 353	**17 056**
	AE	**9.12E−03**	1.50E−02	**9.24E−03**	2.11E−02	**9.37E−03**
	SR	**100.00**%	70.00%	**100.00**%	70.00%	**100.00**%
f_{12}	AI	43.567	40.67	22.33	34.4	29.767
	AFE	1742.7	**1301.4**	**1429.1**	**1100.8**	1905.1
	AE	9.44E−07	9.23E−07	8.91E−07	9.45E−07	9.07E−07
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{13}	AI	423.6	270.97	169.1	272.57	162.73
$D=30$	*AFE*	16 944	**8670.9**	**10 822**	**8722.1**	**10 415**
	AE	9.11E−04	8.98E−04	8.86E−04	8.87E−04	8.58E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{13}	*AI*	1316.3	533.4	297.83	487.8	287.2
$D=50$	*AFE*	52 653	**17 069**	**19 061**	**15 610**	**18 381**
	AE	9.57E−04	9.28E−04	9.38E−04	9.12E−04	9.37E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{13}	AI	4195.5	1246.5	631.5	1146.4	625.67
$D=100$	*AFE*	167 820	**39 889**	**40 416**	**36 684**	**40 043**
	AE	9.79E−03	9.63E−03	9.78E−03	9.63E−03	9.59E−03
	SR	100.00%	100.00%	100.00%	100.00%	100.00%

positive then it means that the current monkey is going near *r*th monkey and going away if it is negative.

Thus, if the position update of the monkeys are done without breaking them into mini-groups, as in original SMO, the position of the randomly selected spider monkey SM_r may or may not be better than its previous position. There may exist two cases:

Case 1: The randomly selected monkey SM_r has already been updated in current iteration before the SM_i.
Case 2: The randomly selected monkey is not yet updated in current iteration.

In both the cases, the position of the monkey SM_r is not yet chosen from its new or previous position based on fitness, therefore, if the random number generated by R_u is positive then it is not sure that the monkey is going towards better position or not.

In ASMO, the groups are divided into mini-groups and after generating the new positions for all the spider monkeys of that mini-group, the better position is greedily selected for them between the new and the previous one, before switching to the next mini-group for updating positions. Hence, if in the position update, (2) and (3), the randomly selected monkey SM_r has been already updated in the same iteration, then it can be ensured that SM_i will gain better experience and will converge to a better position.

4. Experimental results

4.1. Testing and parameter setting

Three different variants of SMO algorithm have been analyzed, including the original one, with 30 different benchmark functions (f_1–f_{30}). The details of these functions are provided in Table 1 including dimensions (D), range, maximum tolerable error (ME), type and global optimum value (OV). These are continuous, unbiased optimization problems and have different degrees of complexity and multimodality. The set of functions selected have different kinds of properties such as unimodal, multimodal, separable and non-separable. These functions are taken from various sources including CEC2010 [32], CEC2014 [33] and Simon Fraser University [34]. The algorithms are implemented in Python 2.7 and the experiments are done on a system with 2.5 GHz i5 4200 m processor with 4 GB RAM.

A unimodal function has only one extremum (minimum or maximum) in the given range space whereas a multimodal function can have many local extrema. They are used to test if the algorithm is stuck in a local extrema while exploring search space. To analyze different forms of complexities few shifted and rotated functions along with some hybrid functions are also used.

The algorithms involved in experiments are:

1. Original SMO.
2. ASMO with $M=4$ and ASMO with $M=8$.
3. AMSMO with $M=4$ and AMSMO with $M=8$.

Table 3
Comparison between proposed SMO variants and SMO algorithm for function f_{14}—f_{28}.

		SMO	*ASMO* ($M=4$)	*AMSMO* ($M=4$)	*ASMO* ($M=8$)	*AMSMO* ($M=8$)
f_{14}	*AI*	319.87	266.7	140.87	288.7	163.57
$D=6$	*AFE*	12 795	**8534.4**	**9015.5**	**9238.4**	10 468
	AE	9.45E−06	9.39E−06	9.19E−06	9.48E−06	9.23E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{14}	*AI*	1963.3	1371.2	1031.8	1519	1029.9
$D=10$	*AFE*	78 533	**43 879**	66 035	**48 608**	65 914
	AE	9.78E−06	9.73E−06	9.67E−06	9.56E−06	9.55E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{15}	*AI*	45.6	28.667	19	28.7	19.667
	AFE	1824	**917.33**	**1216**	**918.4**	**1258.7**
	AE	5.40E−21	5.94E−21	5.23E−21	6.12E−21	5.77E−21
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{16}	*AI*	263.57	385.1	189.4	248.63	77
	AFE	10 543	12 323	12 122	7956.3	**4928**
	AE	8.42E−06	8.72E−06	8.32E−06	8.56E−06	8.47E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{17}	*AI*	27	23.367	14.833	30.3	15.2
	AFE	1080	**747.73**	949.33	969.6	972.8
	AE	9.12E−07	9.32E−07	9.21E−07	8.87E−07	9.27E−07
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{18}	*AI*	145.7	96.5	56.967	103.43	55.567
	AFE	5828	**3088**	**3645.9**	**3309.9**	**3556.3**
	AE	9.45E−06	9.56E−06	9.41E−06	9.12E−06	9.45E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{19}	*AI*	205.3	128.8	59.9	113.33	58.133
	AFE	8212	**4121.6**	**3833.6**	**3626.7**	**3720.5**
	AE	8.47E−06	8.48E−06	8.33E−06	8.54E−06	8.22E−06
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{20}	*AI*	8000	3141.4	1862	3243.5	2019
$D=30$	*AFE*	320 000	**100 524.8**	**119 168**	**103 792**	129 216
	AE	1.24E+01	9.67E−01	9.82E−01	9.72E−01	9.63E−01
	SR	0.00%	**100.00**%	**100.00**%	**100.00**%	**100.00**%
f_{20}	*AI*	15 000	6095.7	6524	9367	6784.1
$D=50$	*AFE*	600 000	**195 062.4**	417 536	**299 744**	434 182.4
	AE	3.46E+03	4.88E+02	4.91E+02	4.86E+02	4.88E+02
	SR	0.00%	**100.00**%	**100.00**%	**100.00**%	**100.00**%
f_{21}	*AI*	1112.5	496.5	290	465	270.97
$D=50$	*AFE*	44 501	**15 888**	**18 560**	**14 880**	**17 342**
	AE	9.12E−04	8.95E−04	9.01E−04	9.12E−04	8.92E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{21}	*AI*	4011	1302	598.13	1155.3	634
$f=100$	*AFE*	160 440	**41 664**	**38 281**	**36 969**	**40 576**
	AE	9.63E−03	9.77E−03	9.56E−03	9.68E−03	9.71E−03
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{22}	*AI*	8000	10 000	5000	10 000	5000
	AFE	320 000	320 000	320 000	320 000	320 000
	AE	1.36E+02	1.26E+02	**9.70E−02**	6.95E+01	**6.80E**−03
	SR	0.00%	0.00%	0.00%	0.00%	0.00%
f_{23}	*AI*	2014	830.33	499.33	925	450.2
$D=50$	*AFE*	80 560	**26 571**	**31 957**	**29 600**	**28 813**
	AE	9.86E−01	9.78E−01	9.81E−01	9.69E−01	9.77E−01
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{23}	*AI*	1622.1	925.5	401.8	843.7	505.03
$D=100$	*AFE*	64 885	**29 616**	**25 715**	**26 998**	**32 322**
	AE	9.64E+00	9.88E+00	9.78E+00	9.66E+00	9.74E+00
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{24}	*AI*	5000	6250	1681	6250	3125
$D=30$	*AFE*	200 000	200 000	**107 584**	200 000	200 000
	AE	2.11E+01	2.08E+01	**0.00E+00**	2.04E+01	2.00E+01
	SR	0.00%	0.00%	**100.00**%	0.00%	0.00%
f_{24}	*AI*	10 000	12 500	2802.2	12 500	6250
$D=50$	*AFE*	400 000	400 000	**179 340.8**	400 000	400 000
	AE	2.12E+01	2.10E+01	**0.00E+00**	2.07E+01	2.03E+01

Table 3 (*continued*)

		SMO	ASMO (M=4)	AMSMO (M=4)	ASMO (M=8)	AMSMO (M=8)
	SR	0.00%	0.00%	**100.00%**	0.00%	0.00%
f_{25} D=50	AI	725.33	411.5	274	382.57	253.8
	AFE	29 013	**13 168**	**17 536**	**12 242**	**16 243**
	AE	9.45E−04	9.31E−04	8.71E−04	8.77E−04	8.85E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{25} D=100	AI	2750.5	1102.7	904.67	1142.2	605.67
	AFE	110 020	**35 285**	57 899	**36 551**	**38 763**
	AE	9.87E−03	9.54E−03	9.44E−03	9.38E−03	9.61E−03
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{26} D=10	AI	173.8	103.6	56.4	181.47	57.3
	AFE	6952	**3315.2**	**3609.6**	5806.9	**3667.2**
	AE	7.32E−01	7.28E−01	7.33E−01	7.39E−01	7.26E−01
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{26} D=30	AI	5000	6250	3125	6250	3125
	AFE	200 000	200 000	200 000	200 000	200 000
	AE	4.78E+00	**8.51E−01**	**8.93E−01**	**1.02E+00**	**8.70E−01**
	SR	0.00%	0.00%	0.00%	0.00%	0.00%
f_{27} D=30	AI	595.93	329.23	190.1	302.2	210.2
	AFE	23 837	**10 535**	**12 166**	**9670.4**	**13 453**
	AE	9.64E−04	9.55E−04	9.71E−04	9.66E−04	9.59E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{27} D=50	AI	1316.3	533.4	297.83	487.8	287.2
	AFE	52 653	**17 069**	**19 061**	**15 610**	**18 381**
	AE	9.68E−04	9.57E−04	9.68E−04	9.63E−04	9.62E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{27} D=100	AI	11 201	2069.8	902	2020.5	970.23
	AFE	448 040	**66 234**	**57 728**	**64 656**	**62 095**
	AE	9.67E−04	9.68E−04	9.78E−04	9.69E−04	9.77E−04
	SR	100.00%	100.00%	100.00%	100.00%	100.00%
f_{28}	AI	99.2	72.533	54.833	72.433	48.2
	AFE	3968	**2321.1**	3509.3	**2317.9**	3084.8
	AE	5.55E−21	4.51E−21	5.84E−21	3.91E−21	4.82E−21
	SR	100.00%	100.00%	100.00%	100.00%	100.00%

Table 4
Comparison between proposed SMO variants and SMO algorithm for function f_{29} and f_{30}.

		SMO	ASMO (M=4)	AMSMO (M=4)	ASMO (M=8)	AMSMO (M=8)
f_{29} D=10	AI	5000	6250	3125	6250	3125
	AFE	200 000	200 000	200 000	200 000	200 000
	AE	1.93E+01	**4.63E+00**	**1.80E+00**	**3.65E+00**	**1.35E+00**
	SR	0.00%	0.00%	0.00%	0.00%	0.00%
f_{29} D=30	AI	5000	6250	3125	6250	3125
	AFE	200 000	200 000	200 000	200 000	200 000
	AE	1.98E+02	1.81E+02	**6.45E+01**	1.84E+02	**6.15E+01**
	SR	0.00%	0.00%	0.00%	0.00%	0.00%
f_{30} D=30	AI	3000	3750	1875	3750	1875
	AFE	120 000	120 000	120 000	120 000	120 000
	AE	2.38E+01	2.24E+00	**3.85E−23**	5.57E+00	**8.66E−31**
	SR	0.00%	0.00%	0.00%	0.00%	0.00%
f_{30} D=50	AI	5000	6250	3125	6250	3125
	AFE	200 000	200 000	200 000	200 000	200 000
	AE	6.74E+01	2.05E+01	**3.56E−07**	2.36E+01	**6.19E−13**
	SR	0.00%	0.00%	0.00%	0.00%	0.00%

where, M is number of mini-groups in each group. The parameter settings for these algorithms are provided in Table 1 along with benchmark functions. The perturbation rate (pr) is varied linearly from 0.1 to 0.4 based on the equation $pr = 0.1 + (0.4 - 0.1) * \frac{iter}{max\_iter}$ where *iter* is the current iteration and *max_iter* are maximum iterations given.

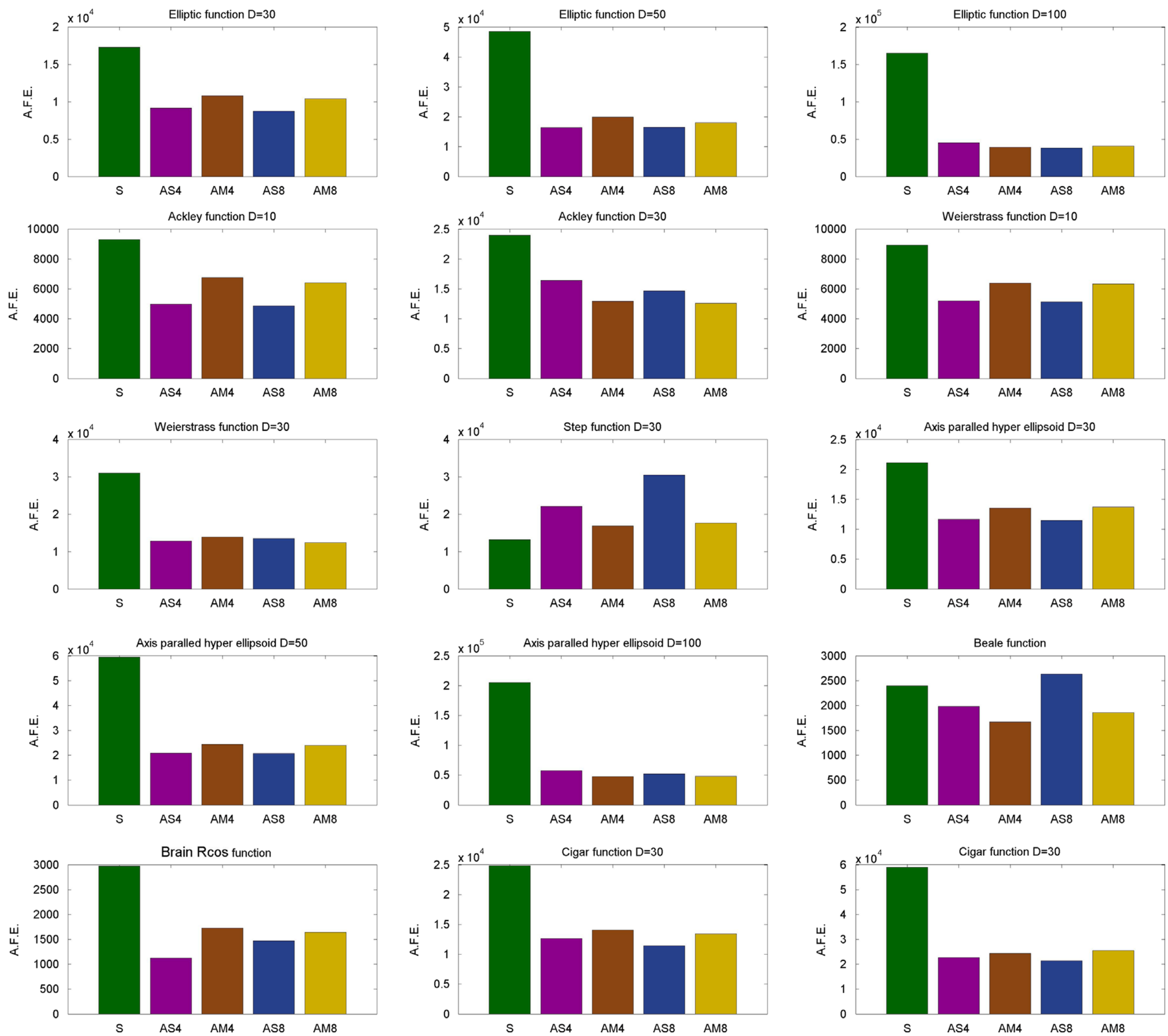

Fig. 2. Comparison graphs for functions f_1–f_8.

4.2. Comparison between different variants of SMO

Numerical results for benchmark problems (f_1–f_{30}) listed in Table 1 are provided in Tables 2–4. In these tables, the algorithm variants are shown as column headers and average iterations (AI), average function evaluations (AFE), average error (AE) and success ratio (SR), are shown as rows in front of respective functions. The AFE is the average of the function evaluations that are required to reach to terminating condition in 60 runs. It can be shown mathematically as $\frac{\sum_{i=1}^{60} FE_i}{60}$ where FE_i is the number of evaluations required in the *ith* trail to reach the terminating criteria. To compare algorithms bar-graphs of the functions (Figs. 2–4) with different dimensions are shown. Also, for proper analysis and comparison convergence plots (Fig. 5) are shown for some functions. AFE and AE comparison with SMO for different functions is shown in Tables 5 and 6.

For comparison between various variants of SMO (for results given from Tables 2 to 6) ME has been used as the primary stopping criteria. Thus, if the fitness value reaches below ME as given in Table 1 the function evaluation is stopped. This has been done to compare the convergence rate of different variants of SMO. Further, maximum function evaluation (MFE) has been used as the secondary stopping criteria if the function is not able to converge within the given MFE (as given in Table 1).

4.2.1. AFE comparison between variants of SMO

Table 2 shows comparison between SMO, ASMO and AMSMO for function f_1–f_{13}. For almost all of these functions SMO and all its ageist variants converged below ME within MFE. It is quite clear from these functions that ageist variants got converged much faster than the SMO algorithm for these functions with an exception being step function (f_4). Also the convergence rate of ageist variants for most functions is almost similar with a few exceptions. For Ackley function (f_2) at 30 dimensions, SR for ASMO was much lesser in comparison to SMO and AMSMO (which showed 100% SR) and it got stuck in local minima at many occasions leading to smaller AE in comparison to SMO and AMSMO. Similar to Ackley function (f_2), Greiwank function (f_{11}) also showed lower SR in the case of ASMO as compared to SMO and AMSMO (showing 100% SR).

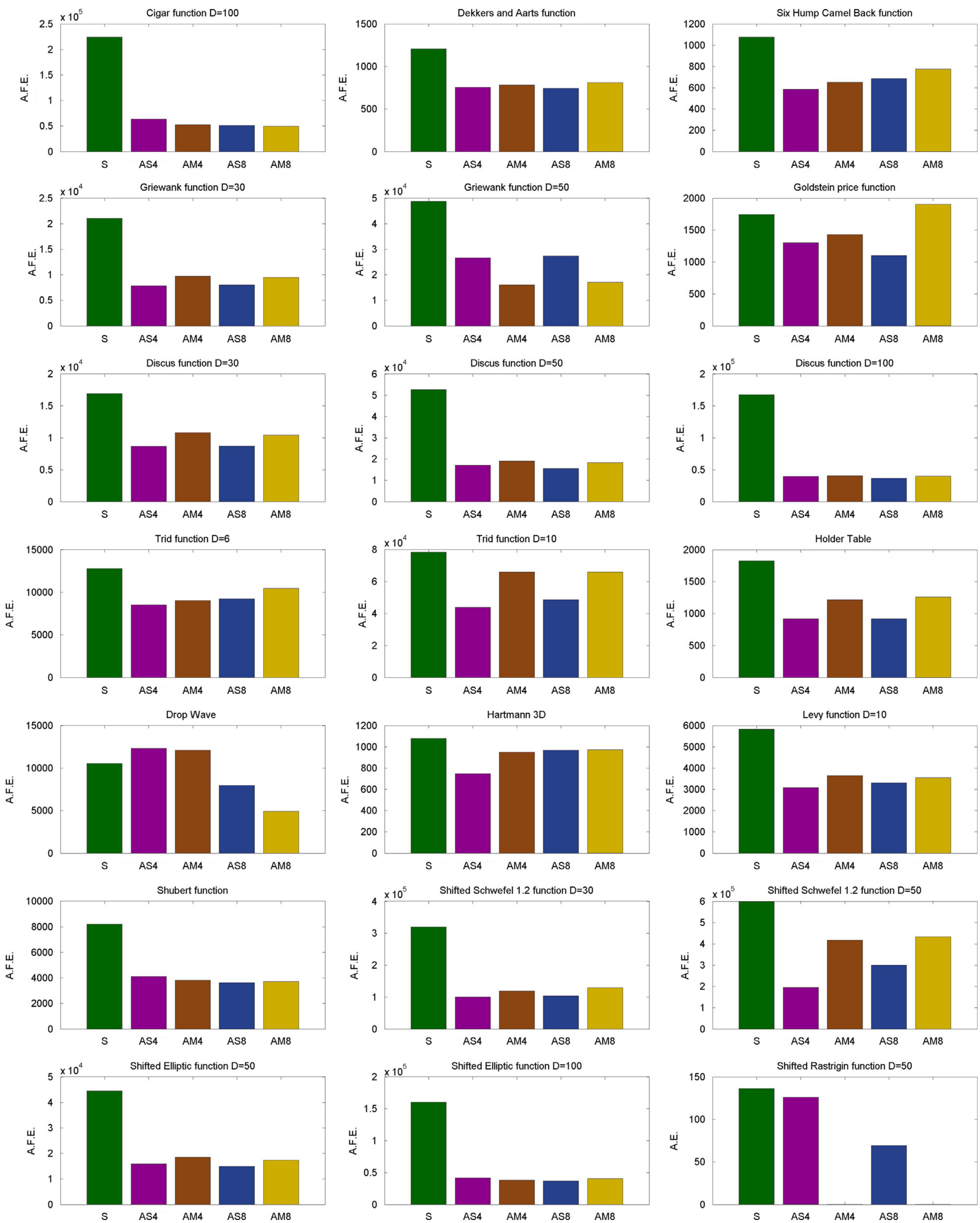

Fig. 3. Comparison graphs for functions f_8–f_{22}.

Table 3 shows comparison between SMO, ASMO and AMSMO for function f_{14}–f_{28}. Similar to functions in Table 2 all concerned algorithms converged below ME in given MFE and convergence of ageist variants was much better in comparison to SMO. With few exceptions performance of all ageist variants was quite close to each other. Also, similar to Table 2, there is not much performance difference in variants with 4 and 8 mini-groups. SMO algorithm was not able to converge below ME within MFE for shifted Schwefel 1.2 function (f_{20}). Compared to this all ageist variants easily converged below ME for both 30 and 50 dimensions. For shifted Rastrigen function (f_{22}) only

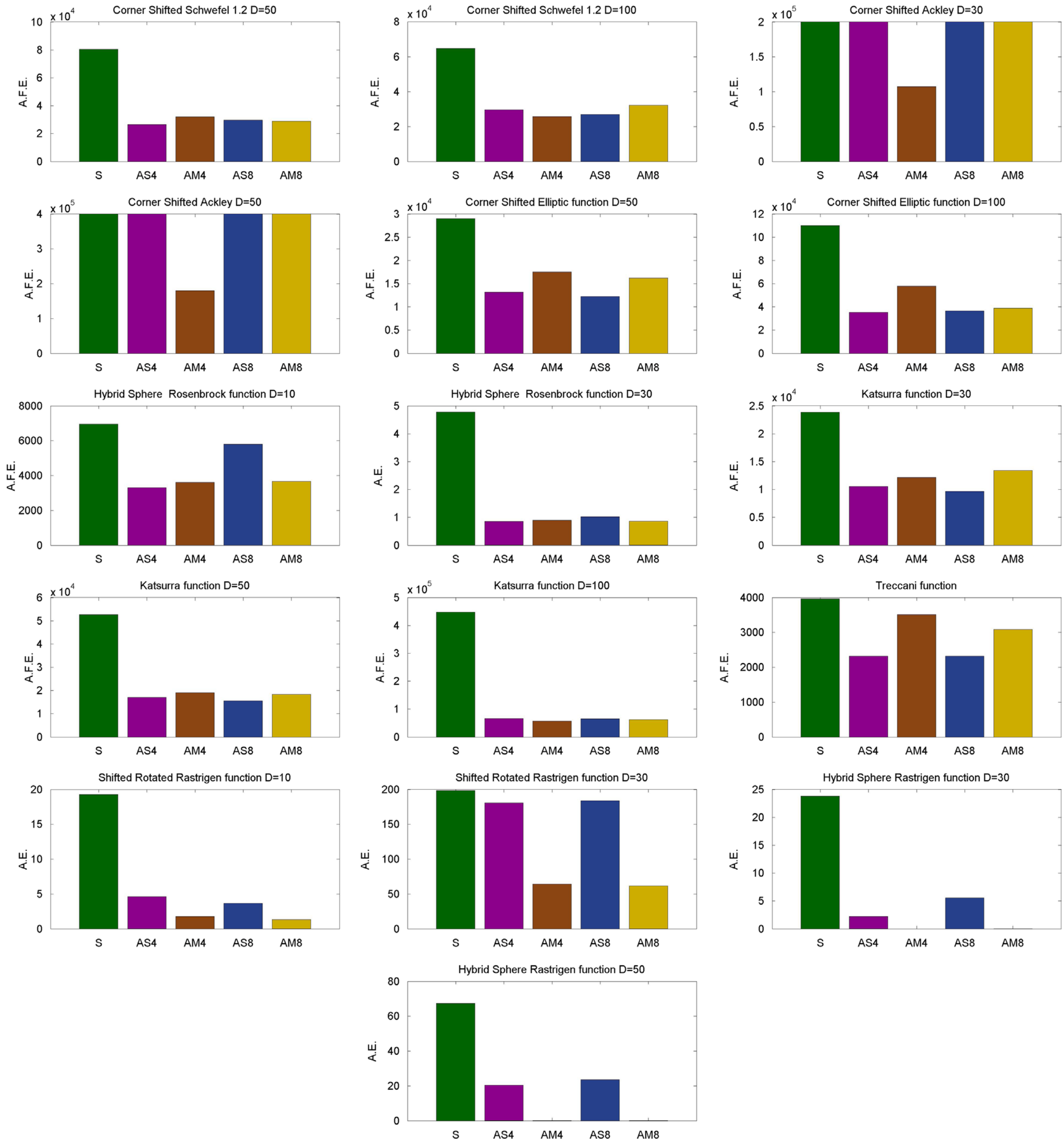

Fig. 4. Comparison graphs for functions f_{23}–f_{30}.

AMSMO variants were able to reach global minima with SMO and ASMO getting struck at local minima. For corner shifted Ackley function (f_{24}) only AMSMO with 4 mini-groups ($M=4$) was able to converge to global minima with other algorithms showing absolutely no convergence as shown by their AE and SR.

Table 4 shows comparison between SMO variants for functions f_{29} and f_{30}. The table clearly shows that AMSMO performed much better in terms of AE in comparison to SMO and ASMO for both shifted rotated Rastrigen (f_{29}) and hybrid sphere Rastrigen functions (f_{30}).

Figs. 2–4 show comparison in bar graphs between various SMO variants for functions f_1–f_{30}. In these graphs, S is for SMO, AS4 is for ASMO with $M=4$, AM4 is for AMSMO with $M=4$, AS8 is for ASMO for $M=8$ and AM8 is for AMSMO with $M=8$. These have been plotted to clearly visualize Tables 2–4 data. Fig. 2 includes AFE comparison bar graphs for functions f_1–f_8. In all these y-axis represents AFE taken for convergence. Fig. 3 includes AFE comparison bar graphs for functions f_8–f_{21} and AE comparison bar graph for function f_{22}. Fig. 4 includes AFE comparison bar graphs for functions f_{23}–f_{28} and AE comparison bar graph for functions f_{29} and f_{30}. These bar graphs clearly confirm the above mentioned observations.

Fig. 5 shows convergence curves for hybrid sphere Rastrigen function ($D=50$), hybrid sphere Rosenbrock function ($D=30$), Weierstrass function ($D=10$) and elliptic function ($D=100$). The convergence curve for hybrid sphere Rastrigen function shows convergence only in the case of AMSMO. Least convergence is shown by SMO which is along with ASMO got struck at local minima while AMSMO got fully

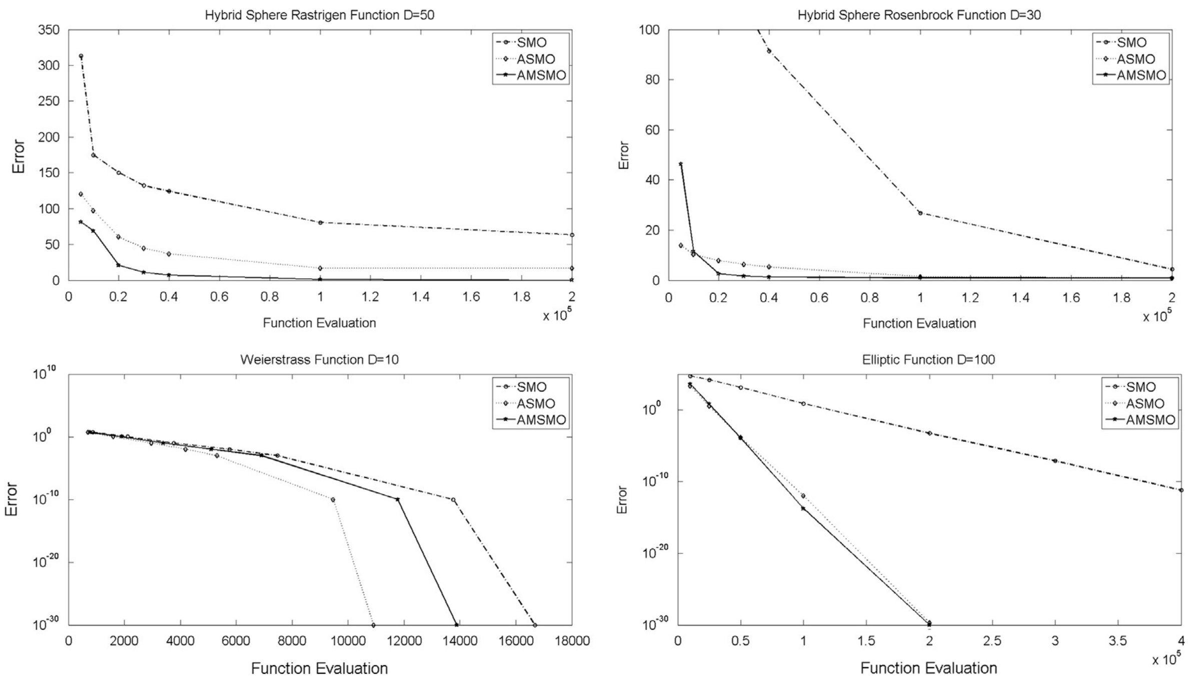

Fig. 5. Convergence plots.

converged to global minima. Convergence plots of elliptic and hybrid sphere Rosenbrock function clearly shows that how easily ASMO and AMSMO outperform SMO with AMSMO performing marginally better. For Weierstrass function, convergence of ASMO was better than that of AMSMO with both of them outperforming SMO.

Table 5 gives percentage improvement in terms of amount of AFE required by concerned algorithm for convergence to ME in comparison to SMO. Algorithms compared are ASMO and AMSMO with 4 and 8 mini-groups. If value in this table is negative then the concerned algorithm takes that percent less AFE for convergence to ME while if it is positive then AFE taken by concerned algorithm is more than that taken by SMO. Table 6 gives the percentage improvement in terms of AE given by concerned algorithm with respect to SMO algorithm.

It is clear from these tables and graphs that ageist variants of SMO (i.e. ASMO and AMSMO) performed much better than SMO in terms of AFE and AE except function f_4 (step function). Among the ageist variants AMSMO with 4 mini-groups turned out to be most stable of the lot.

4.3. Parameteric and Non-parametric tests between SMO, ASMO and AMSMO

For the parametric and non parametric tests, AMSMO (4 mini-groups) has been used as the base algorithm with which SMO and ASMO have been compared. Table 7 shows the *p-value*, *h-value* along with the corresponding *t-value* of SMO and ASMO in comparison to AMSMO for the *t*-test. *p*-Value represents probability of rejection of null hypothesis. Its value is between 0 and 1. Lesser the *p*-value more is the difference between the compared algorithms. Hypothesis test or *h*-value also indicates the rejection of null hypothesis. $h=1$ represents confirmation on rejection of null hypothesis and thus represents that compared algorithms are different. For hypothesis test significance level of 5% is taken. The *t*-test assesses whether the means of two groups of results are statistically different from each other. For purpose of testing two-tailed *t*-tests was adopted with 5% significance level and 118 degrees of freedom. The negative *t*-value indicates that AMSMO is better than the concerned algorithm. Further comparison has been done using the *Wilcoxon signed rank test* [35] on AFE and AE given in Tables 2–4. For this test the comparison data (Tables 2–4) was taken in normalized form with a significance level of 5%. For most of the functions, *t*-test has given a negative value with the *p*-value being small and *h* being 1 for both SMO and ASMO algorithms in comparison to AMSMO with the exception being step function in SMO and elliptic function and Goldstien function in ASMO. But in these functions, the performance of AMSMO was comparable to the concerned function. The highly negative *t*-value of SMO and ASMO for corner shifted Ackley in comparison to AMSMO is due to lack of convergence in the case of ASMO and SMO for this function. Compared to this AMSMO was easily able to converge to global minima. The high performance of AMSMO in the case of corner shifted Ackley, shifted Rastrigen and shifted rotated Rastrigen function in comparison to SMO and ASMO again proves the high stability of AMSMO.

4.4. Complexity comparison of SMO, ASMO and AMSMO

For calculation of complexity, formula given in CEC 2014 benchmark function report has been used. Complexity value for SMO, ASMO and AMSMO are found to be 23.19, 15.61 and 13.42, respectively. The reduction in complexity is due to increased convergence rate for AMSMO and ASMO in comparison to SMO. Due to low convergence of original SMO algorithm, rate of group breaking and merging is much more as compared to AMSMO and ASMO algorithms.

Due to lower complexity of AMSMO, in comparison to SMO and ASMO, for the same amount of function evaluations, AMSMO algorithm takes much lesser computational time in comparison to SMO and ASMO.

Table 5
Percentage AFE required in comparison to original SMO.

Function name	*Dimension*	*ASMO (M=4) (%)*	*ASMO (M=8) (%)*	*AMSMO (M=4) (%)*	*AMSMO (M=8) (%)*
Elliptic	30	−47.00	**−49.53**	−37.53	−39.73
	50	−66.35	−66.07	−58.99	−62.94
	100	−72.69	−77.03	−76.12	−75.26
Ackley	10	**−46.53**	**−47.88**	−27.35	−31.27
Weierstrass	10	**−41.71**	**−42.43**	−28.45	−29.02
	30	−58.82	−56.52	−55.09	−60.11
Step function	30	67.92	131.04	**28.13**	**33.66**
Axis paralled – hyper ellipsoid	30	−44.78	−45.77	−35.98	−35.09
	50	−64.97	−65.15	−59.03	−59.63
	100	−72.02	−74.66	−76.69	−76.47
Beale	2	−17.47	9.64	**−30.40**	−22.58
Brain Rcos	2	**−62.19**	−50.65	−41.94	−44.95
Cigar	30	−49.03	−54.06	−43.42	−45.85
	50	−61.57	−63.64	−58.64	−56.73
	100	−71.46	−77.11	−76.61	−77.76
Dekkers and Aarts	2	−37.38	−38.43	−35.26	−32.79
Six Hump Camel Back	2	**−45.48**	**−36.16**	**−39.33**	−27.83
Griewank	30	−62.78	−62.10	−53.90	−55.14
	50	−45.22	−43.87	**−67.18**	**−65.00**
Goldstein price	2	−25.33	**−36.83**	−17.98	9.32
Discus	30	−48.83	−48.52	−36.13	−38.53
	50	−67.58	−70.35	−63.80	−65.09
	100	−76.23	−78.14	−75.92	−76.14
Trid	6	−33.30	−27.80	−29.52	−18.17
	10	**−44.13**	**−38.11**	−15.91	−16.06
Holder Table	2	−49.65	−49.65	−33.33	−30.88
Drop Wave	2	16.88	**−24.55**	14.72	**−53.26**
Hartmann 3D	3	**−30.68**	−10.22	−11.70	−9.93
Levy	10	−47.01	−43.22	−37.41	−38.97
Shubert	2	−49.81	−55.85	−53.30	−52.57
Shifted Schwefel 1.2	30	−68.59	−67.57	−62.76	−59.62
	50	**−67.49**	**−50.04**	−30.41	−27.64
Shifted Elliptic	50	−64.30	−66.56	−58.29	−61.03
	100	−74.03	−76.96	−76.15	−74.71
Corner Shifted – Schwefel 1.2	50	−67.02	−63.26	−60.33	−64.23
	100	−54.36	−58.38	−60.17	−50.19
Corner Shifted – Elliptic	50	−54.61	−57.80	−39.56	−44.01
	100	−67.93	−66.77	−70.39	−64.80
Hybrid Sphere Rosenbrock	10	**−52.36**	−16.04	**−48.08**	**−47.25**
Katsurra	30	−55.81	−59.43	−48.99	−43.55
	50	−67.58	−70.37	−63.80	−65.09
	100	−85.22	−85.57	−87.12	−86.14
Treccani	2	**−37.91**	**−36.53**	−5.34	−15.92

4.5. Comparison of AMSMO with various newly proposed algorithms

Table 8 compares AMSMO with five recently proposed state-of-the-art algorithms. Ten functions have been used to compare our proposed modified variant of SMO (AMSMO). All functions are allowed to evaluate for 2×10^5 evaluations. Average of 20 runs has been taken for comparison purpose. For the convenience error value of 1×10^{-100} has been taken as 0. Table 8 clearly shows that the performance of AMSMO algorithm is comparable to newly proposed algorithms even outperforming other algorithms as in the case of Schwefel 2.22 function. Further Wilcoxon test confirmed the comparative performance of AMSMO algorithm in comparison to these current state-of-the-art algorithms. It can also be stated from p- and h-value (wilcoxon test) for LdDE and ECLPSO that AMSMO has outperformed for the compared functions.

Table 6
Percent AE in comparison to original SMO.

Function name	Dimensions	ASMO (M=4) (%)	ASMO (M=8) (%)	AMSMO (M=4) (%)	AMSMO (M=8) (%)
Shifted Rastrigin	50	−7.61	−49.12	**−99.93**	**−100.00**
Hybrid Sphere Rosenbrock	30	−82.17	−78.68	−81.31	−81.79
Shifted Rotated Rastrigin	10	−75.98	−81.07	−90.68	−92.97
	30	−8.94	−7.42	−67.51	−69.01
Hybrid Sphere Rastrigin	30	−90.59	−76.57	**−100.00**	**−100.00**
	50	−69.59	−64.96	**−100.00**	**−100.00**

Table 7
Non parametric tests for comparison of SMO and ASMO with AMSMO.

Function	D	SMO			ASMO		
		p-value	**h**	**t-value**	**p-value**	**h**	**t-test**
Elliptic	50	3.84E−08	1	−14.8556	0.001	1	4.5582
Ackley	30	7.87E−06	1	−8.3751	0.0027	1	−3.4802
Step	30	0.4932	0	0.6994	0.1692	0	−1.4325
Corner Shifted Ackley	30	1.09E−17	1	−317.2561	6.54E−24	1	−1901.8
Greiwank	50	8.95E−04	1	−9.2630	0.03177	1	−3.0895
Goldstein Price	12	0.0053	1	−3.3948	0.6625	0	0.4497
Shifted Rastrigen	50	0.0037	1	−4.5997	0.0032	1	−4.3614
Shifted Rotated Rastrigen	30	1.62E−11	1	−32.8377	1.02E−08	1	−17.0384
Wilcoxon test		2.41E−10	1		0.4882	0	

Table 8
Comparison of AMSMO with various newly proposed algorithms.

	D	AMSMO	LdDE [22]	ILABC [26]	SSG-PSO [14]	ECLPSO [11]	EABC [27]
Sphere	30	**0.00E+00**	5.68E−14	7.54E−43	**0.00E+00**	**1.00E−96**	9.26E−67
Elliptic	30	**0.00E+00**	6.23E−14	8.61E−39	**0.00E+00**	**8.41E−92**	2.76E−64
Ackley	30	**2.18E−14**	3.26E−11	**2.77E−14**	**1.25E−14**	**3.55E−15**	**1.36E−14**
Rosenbrock	30	8.27E+00	1.87E+00	**1.01E−01**	6.90E+00	2.75E+01	**9.06E−02**
Rastrigen	30	**0.00E+00**	3.21E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Griewank	30	**0.00E+00**	2.11E−02	3.64E−13	**0.00E+00**	**0.00E+00**	**0.00E+00**
Schwefel 2.22	30	**2.25E−86**	4.34E−08	6.02E−23	9.33E−22	2.02E−31	5.85E−35
Schwefel 1.2	30	1.29E−01	**3.74E−09**	8.92E+01	4.16E+01	5.62E+01	1.14E+02
Shifted Rosenbrock	30	1.26E+01	3.27E+00	8.34E−01	**2.64E−13**	3.42E+01	2.17E−01
Shifted Rastrigen	30	**0.00E+00**	4.91E+00	0.00E+00	1.22E+01	**0.00E+00**	**0.00E+00**
Wilcoxon test	p		0.0273	0.0781	0.4375	0.0313	0.2188
	h		1	0	0	1	0

4.6. Comparison of AMSMO with newly proposed SMO variants

Table 9 compares AMSMO with newly proposed SMO variants. Comparison has been done in terms of the average number of function evaluations taken by the algorithm to reach the ME as given in the table.

Table 9 clearly shows that AMSMO outperforms MPU-SMO and Sa-SMO in most of the tested functions which is further proved by the wilcoxon test which gave low p-values (lower than 0.05) and h-value of 1 for both MPU-SMO and Sa-SMO algorithms.

4.7. Comparison of AMSMO with MVMO

Table 10 shows AE comparison for 2×10^5 function evaluations for 9 different functions between AMSMO and CEC2014 winner Mean Variance Mapping Optimization (MVMO) [36]. For the purpose of testing, the rotation and shifting data as in CEC2014 is used. An error of 1×10^{-8} has been taken as zero error. The above table clearly shows the better performance of AMSMO in terms of AE as compared to

Table 9
Comparison of AMSMO with newly proposed SMO variants.

	D	ME	AMSMO	MPU-SMO [29]	Sa-SMO [30]
Sphere	30	1.00E−05	**13 120.2**	44 435.12	**14 597.25**
Elliptic	30	1.00E−05	**13 760.133**	65 693.17	**17 563.39**
Griewank	30	1.00E−05	**12 864.1**	87 401.67	28 207.11
Rosenbrock	30	5.00E+01	**33 088**	201 808.6	67 433
Rastrigen	30	1.00E−05	144 680.73	**91 623.6**	**81 293.64**
Beale	2	1.00E−05	**1670.4**	2898.423	4414.41
Branin Rcos	2	1.00E−06	**1728**	18 496.32	31 362.01
Ackley	30	1.00E−05	18 624	**10 824.76**	24 075.81
Shifted Rastrigen	30	1.00E−05	**141 160.5666**	Not Converged	Not Converged
Goldstien	2	1.00E−14	**3392.23**	8595.18	4885.353
Six Hump Camel Back	2	1.00E−06	**652.8**	Not Converged	Not Converged
Dekker's and Aarts	2	5.00E−01	**782.93**	2181.96	1407.78
Wilcoxon test	p			0.0034	0.0049
	h			1	1

Table 10
Comparison of AMSMO with MVMO.

	D	AMSMO	MVMO [36]
Shifted sphere	10	0.000E+00	0.000E+00
	20	0.000E+00	0.000E+00
	30	0.000E+00	0.000E+00
Shifted ellipsoid	10	0.000E+00	0.000E+00
	20	0.000E+00	0.000E+00
	30	0.000E+00	0.000E+00
Shifted rotated ellipsoid	10	0.000E+00	0.000E+00
	20	0.000E+00	0.000E+00
	30	**0.000E+00**	8.849E−01
Shifted step function	10	**0.000E+00**	2.650E+00
	20	**8.333E−02**	6.550E+00
	30	**9.600E−01**	1.270E+01
Shifted rotated Rastrigin	10	**1.795E+00**	2.617E+01
	20	**6.943E+00**	4.253E+01
	30	6.446E+01	8.493E+01
Shifted Griewank	10	2.027E−02	4.397E−01
	20	0.000E+00	0.000E+00
	30	0.000E+00	0.000E+00
Shifted Rosenbrock	10	2.787E+00	9.546E+00
Hybrid function (F18−CEC14)	30	2.786E+01	2.894E+01
Composition function(F23−CEC−14)	30	3.20E+02	3.15E+02
Wilcoxon test	p		0.0020
			1

MVMO for the same number of AFE for the shifted step and shifted rotated Rastrigen functions. For other functions performance of AMSMO and MVMO was comparable. Further Wilcoxon test on these two algorithms gave a *p*-value lower than the significance level and the *h*-value of 1 thus indicating better performance of AMSMO in comparison to MVMO.

5. Conclusion

The paper comprises newly proposed variants of SMO, known as ASMO and AMSMO respectively. These algorithms are based upon difference in age and other dynamic abilities of spider monkeys like interaction, speed of communication and adapting to the changes in the environment. These algorithms are compared with the original SMO algorithm and results are recorded. The graph and tables proves the importance of adding this feature in terms of convergence rate. In all the above variants of SMO tested and compared it is found that the modified version of ASMO i.e. AMSMO with 4 mini-groups is most stable and has shown thehighest convergence rate in many of the tested benchmark functions. To further compare the performance various non-parametric tests were done which again showed the significance of AMSMO algorithm compared to SMO and ASMO. For better analysis of convergence rate in terms of time, complexity calculations were done. Lower complexity and better convergence of AMSMO proves it to have a better convergence rate in terms of time in comparison to SMO and ASMO algorithms. Further comparisons of AMSMO algorithm was done with various state-of-the-art algorithms like LdDE, ILABC, SSG-PSO, ECLPSO, EABC, MPU-SMO, Sa-SMO and MVMO proves the significance of AMSMO in comparison to modern optimization techniques.

Future prospect would be to extend the use of AMSMO algorithm in solving multiobjective optimization problems. The proposed algorithm can be used in various complex real world optimization problems like design of wireless telecommunications networks, hydro-thermal coordination, clustering and data mining.

References

[1] Craig Reynolds, Flocks, herds and schools: a distributed behavioral model. SIGGRAPH '87: in: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (Association for Computing Machinery), 1987. pp. 25–34.
[2] Bianchi Leonora, Marco Dorigo, Luca Maria Gambardella, Walter J. Gutjahr, A survey on metaheuristics for stochastic combinatorial optimization, Natural Comput. Int. J. 8 (2) (2009) 239–287.
[3] C. Blum, A. Roli, Metaheuristics in combinatorial optimization: overview and conceptual comparison, ACM Comput. Surv. 35 (3) (2003) 268–308.
[4] E.-G. Talbi, Metaheuristics: From Design to Implementation, vol. 74, John Wiley & Sons, 2009.
[5] Kenneth Sorensen, Metaheuristic—the metaphor exposed, Int. Trans. Oper. Res. 22 (2012) 3–18.
[6] G. Beni, J. Wang, Swarm intelligence in cellular robotic systems, in: Proceedings of the NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, June 26–30, 1989.
[7] M. Dorigo, T. Stützle, Ant colony optimization: overview and recent advances, Techreport, IRIDIA, Universite Libre de Bruxelles, 2009.
[8] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, IEEE, 1995, pp. 1942-1948.
[9] K.M. Passino, Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Syst. Mag. 22 (3) (2002) 52–67.
[10] D. Karaboga, et al., A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 214 (1) (2009) 108–132.
[11] X. Yu, X. Zhang, Enhanced comprehensive learning particle swarm optimization, Appl. Math. Comput. 242 (2014) 265–276.
[12] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (2006) 281–295.
[13] Y.-B. Shin, E. Kita, Search performance improvement of particle swarm optimization by second best particle information, Appl. Math. Comput. 246 (2014) 346–354.
[14] G. Wu, D. Qiu, Y. Yu, W. Pedrycz, M. Ma, H. Li, Superior solution guided particle swarm optimization combined with local search techniques, Expert Syst. Appl. 41 (16) (2014) 7536–7548.
[15] Z. Ren, A. Zhang, C. Wen, Z. Feng, A scatter learning particle swarm optimization algorithm for multimodal problems, IEEE Trans. Cybern. 44 (2014) 1127–1140.
[16] Ran Cheng, Yaochu Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inf. Sci. 291 (2015) 43–60.
[17] M.R. Tanweer, S. Suresh, N. Sundararajan, Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems, Inf. Sci. 326 (2016) 1–24.
[18] Yuhua Li, et al., Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems, Inf. Sci. 293 (2015) 370–382.
[19] M.R. Tanweer, S. Suresh, N. Sundararajan, Self regulating particle swarm optimization algorithm, Inf. Sci. 294 (2015) 182–202.
[20] Wei Hong Lim, Nor Ashidi Mat Isa, Adaptive division of labor particle swarm optimization, Expert Syst. Appl. 42.14 (2015) 5887–5903.
[21] R. Storn, K.V. Price, Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, ICSI, USA, Technical Report TR-95-012, March, 1995.
[22] N.D. Jana, J. Sil, Levy distributed parameter control in differential evolution for numerical optimization, Nat. Comput. (2015) 1–14.
[23] M. Yang, C. Li, Z. Cai, J. Guan, Differential evolution with auto-enhanced population diversity, IEEE Trans. Cybern. 45 (2) (2015) 302–315, art. no. 6868218.
[24] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search, Simul. Soc. Comput. Simul. 76 (2001) 60–68.
[25] E. Valian, S. Tavakoli, S. Mohanna, An intelligent global harmony search approach to continuous optimization problems, Appl. Math. Comput. 232 (2014) 670–684.
[26] W.F. Gao, L.L. Huang, S.Y. Liu, C. Dai, Artificial bee colony algorithm based on information learning, IEEE Trans. Cybern. 45 (12) (2015) 2827–2839.
[27] Wei-feng Gao, San-yang Liu, Ling-ling Huang, Enhancing artificial bee colony algorithm using more information-based search equations, Inf. Sci. 270 (2014) 112–133.
[28] J.C. Bansal, H. Sharma, S.S. Jadon, M. Clerc, Spider Monkey Optimization algorithm for numerical optimization, Memetic Comput. (2013) 1–17.
[29] S. Kumar, V.K. Sharma, R. Kumari, Modified position update in Spider Monkey Optimization Algorithm, Int. J. Emerg. Technol. Comput. Appl. Sci. 7 (2) (2014) 198–204.
[30] S. Kumar, V. Kumar Sharma, R. Kumari, Self-adaptive spider monkey optimization algorithm for engineering optimization problems, Int. J. Inf. Commun. Comput. Technol. II (2014) 96–107.
[31] K. Gupta, K. Deep Tournament Selection Based Probability Scheme in Spider Monkey Optimization Algorithm: In Harmony Search Algorithm, Springer, Berlin, Heidelberg, 2016, pp. 239–250.
[32] Ke Tang, Xiaodong Li, P.N. Suganthan, Zhenyu Yang, Thomas Weise, Benchmark functions for the CEC 2010 special session and competition on large-scale global optimization, Nature Inspired Computation and Applications Laboratory (NICAL), School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui, China, School of Computer Science and Information Technology, RMIT University, Australia, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, January 8, 2010.
[33] J.J. Liang1, B.Y. Qu2, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization, School of Electrical Engineering, Zhengzhou University, Zhengzhou, China, School of Electric and Information Engineering, Zhongyuan University of Technology, Zhengzhou, China, School of EEE, Nanyang Technological University, Singapore, December 2013.
[34] S. Surjanovic, D. Bingham. Virtual Library of Simulation Experiments: Test Functions and Datasets, Retrieved July 15, 2015, from ⟨http://www.sfu.ca/∼ssurjano⟩, 2013.
[35] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.
[36] T.H. Khoa, P.M. Vasant, M.S.B. Singh, V.N. Dieu, Swarm based mean-variance mapping optimization (MVMOs) for economic dispatch problem with valvepoint effects, in: IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2014, pp. 59–63.