Memetic Comp. (2014) 6:31–47
DOI 10.1007/s12293-013-0128-0

REGULAR RESEARCH PAPER

Spider Monkey Optimization algorithm for numerical optimization

Jagdish Chand Bansal · Harish Sharma · Shimpi Singh Jadon · Maurice Clerc

Received: 21 June 2012 / Accepted: 7 December 2013 / Published online: 1 January 2014

Abstract Swarm intelligence is one of the most promising area for the researchers in the field of numerical optimization. Researchers have developed many algorithms by simulating the swarming behavior of various creatures like ants, honey bees, fish, birds and the findings are very motivating. In this paper, a new approach for numerical optimization is proposed by modeling the foraging behavior of spider monkeys. Spider monkeys have been categorized as fission–fusion social structure based animals. The animals which follow fission–fusion social systems, split themselves from large to smaller groups and vice-versa based on the scarcity or availability of food. The proposed swarm intelligence approach is named as Spider Monkey Optimization (SMO) algorithm and can broadly be classified as an algorithm inspired by intelligent foraging behavior of fission–fusion social structure based animals.

Keywords Swarm intelligence based algorithm · Optimization · Fission–fusion social system · Spider monkey optimization

J. C. Bansal (✉) · H. Sharma · S. S. Jadon · M. Clerc
ABV-Indian Institute of Information Technology and Management, Gwalior, India
e-mail: jcbansal@gmail.com

J. C. Bansal
South Asian, University, New Delhi, India

H. Sharma
e-mail: harish.sharma0107@gmail.com

S. S. Jadon
e-mail: shimpisingh2k6@gmail.com

M. Clerc
Independent Consultant in Optimization, Groisy, France
e-mail: Maurice.Clerc@WriteMe.com

1 Introduction

The name swarm is used for an accumulation of creatures such as ants, fish, birds, termites and honey bees which behave collectively. The definition given by Bonabeau for the swarm intelligence is "any attempt to design algorithms or distributed problem-solving devices inspired by the collective behaviour of social insect colonies and other animal societies" [3].

Swarm Intelligence is a meta-heuristic approach in the field of nature inspired techniques that is used to solve optimization problems. It is based on the collective behavior of social creatures. Social creatures utilize their ability of social learning and adaptation to solve complex tasks. Researchers have analyzed such behaviors and designed algorithms that can be used to solve nonlinear, non-convex or combinatorial optimization problems in many science and engineering domains. Previous research [7,17,28,39] have shown that algorithms based on Swarm Intelligence have great potential to find a near optimal solution of real world optimization problem. The algorithms that have been emerged in recent years are Ant Colony Optimization (ACO) [7], Particle Swarm Optimization (PSO) [17], Bacterial Foraging Optimization (BFO) [26], Artificial Bee Colony Optimization (ABC) [14] etc.

In order to design a new swarm intelligence based algorithm, it is necessary to understand whether a behavior is swarm intelligent behavior or not. Karaboga et al. mentioned that *Division of Labor* and *Self-Organization* are the necessary and sufficient conditions for obtaining intelligent swarming behaviors.

1. Self-organization: is an important feature of a swarm structure which results in global level response by means of interactions among its low-level components without a

central authority or external element enforcing it through planning. Therefore, the globally coherent pattern appears from the local interaction of the components that build up the structure, thus the organization is achieved in parallel as all the elements act at the same time and distributed as no element is a central coordinator. Bonabeau et al. have defined the following four important characteristics on which self-organization is based [3]:

(i) Positive feedback: is an information extracted from the output of a system and reapplied to the input to promotes the creations of convenient structures. In the field of swarm intelligence positive feedback provides diversity and accelerate the system to new stable state.
(ii) Negative feedback: compensates the effect of positive feedback and helps to stabilize the collective pattern.
(iii) Fluctuations: are the rate or magnitude of random changes in the system. Randomness is often crucial for efflorescent structures since it allows the findings of new solutions. In foraging process, it helps to get-ride of stagnation.
(iv) Multiple interactions: provide the way of learning from the individuals within a society and thus enhance the combined intelligence of the swarm.

2. Division of labour: is a cooperative labour in specific, circumscribed tasks and like roles. In a group, there are various tasks, which are performed simultaneously by specialized individuals. Simultaneous task performance by cooperating specialized individuals is believed to be more efficient than the sequential task performance by unspecialized individuals [5,13,24].

This paper proposes a new swarm intelligence algorithm based on the foraging behavior of spider monkeys. The foraging behavior of spider monkeys shows that these monkeys fall in the category of fission–fusion social structure (FFSS) based animals. Thus the proposed optimization algorithm which is based on foraging behavior of spider monkeys is explained better in terms of FFSS. Further, the proposed strategy is tested on various benchmark and engineering optimization test problems.

The rest of the paper is organized as follows: Sect. 2 describes the foraging behavior and social structure of spider monkeys. In Sect. 3, first, the foraging behavior is critically evaluated to be a swarm intelligent behavior over the necessary and sufficient conditions of swarm intelligence and then Spider Monkey Optimization algorithm is proposed. A detail discussion about the proposed strategy is presented in Sect. 4. In Sect. 5, performance of the proposed strategy is analyzed and compared with four state-of-the-art algorithms, namely DE, PSO, ABC and CMA-ES. Finally, in Sect. 6, paper is concluded.

2 Foraging and social behavior of spider monkeys

Fission–fusion swarm is a social grouping pattern in which individuals form temporary small parties (also called subgroups) whose members belong to a larger community (or unit-group) of stable membership, there can be fluid movement between subgroups and unit-groups such that group composition and size changes frequently [37].

The fission–fusion social system of swarm can minimize direct foraging competition among group members, so they divide themselves into sub-groups in order to search food. The members of these subgroups then communicate (through barking and other physical activities) within and outside the subgroup depending upon the availability of food. In this society, social group sleep in one habitat together but forage in small sub-groups going off in different directions during the day. This form of social formation occurs in several species of primates like hamadryas, bonobo, chimpanzees, gelada baboons and spider monkeys. These societies change frequently in their size and composition, making up a strong social group called the 'parent group'. All the individual members of a faunal community comprise of permanent social networks and their capability to track changes in the environment varies according to their individual animal dynamics. In a fission–fusion society, the main parent group can fission into smaller subgroups or individuals to adapt to the environmental or social circumstances. For example, members of a group are separated from the main group in order to hunt or forage for food during the day, but at night they return to join (fusion) the primary group to share food and to take part in other activities [37].

The society of spider monkeys is one of the example of fission–fusion social structure. In subsequent subsections, a brief overview on swarming of spider monkeys is presented.

2.1 Social organization and behavior

The social organization of spider monkeys is related to fission–fusion social system. Figure 1 provide some pictures of spider monkeys [31]. They are social animals and live in group of up to 50 individuals [34]. Spider monkeys break up into small foraging groups that travel together and forage throughout the day within a core area of the larger group's home range [34]. Spider monkeys find their foods in a very different way: a female leads the group and is responsible for finding food sources. In case if she doesn't find sufficient food for the group, she divides the group into smaller subgroups that forage separately [23]. The subgroups within the band are temporary and may vary in formation frequently throughout the day, but on average 3 members can be found in any group at any time [23,38]. When two different bands of spider monkeys come closer, the males in each band display aggressiveness and territorial behavior such as calling

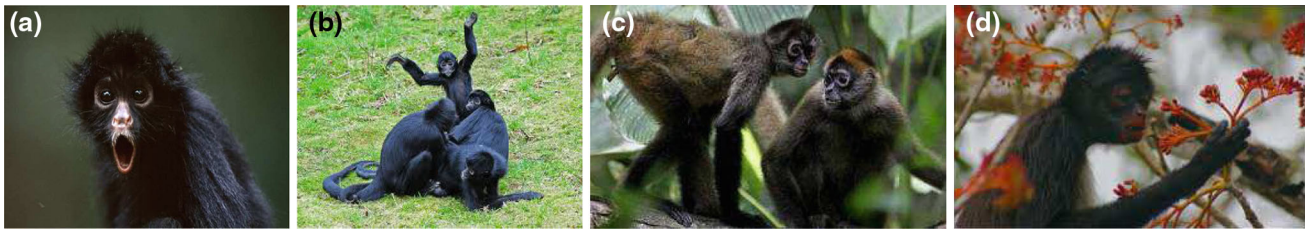

Fig. 1 Social Organization and Behavior **a** Spider-Monkey, **b** Spider Monkey Group, **c** Spider Monkey sub-group, **d** foods foraging [31]

and barking. These communications occur with much distance between the two subgroups and do not involve any physical contacts, showing that groups respect distinct territory boundaries [38]. Members of a society might not ever be noticed closer at one place, but their mutual tolerance of each other when they come into contact reflects that they are component of the larger group [38]. The main reason behind emerging of fission–fusion social system is the competition for food among the group members when there is a shortage in food availability due to seasonal reasons [23]. When a big group gets food at particular location, there is likely to be less food per group member compared to a small group. After some time, when food scarcity is at its peak, average subgroup size is the smallest and during period of highest food availability, subgroup size is the largest, indicating that competition for scarce resources necessitates breaking into smaller foraging groups [22,38]. One reason spider monkeys break into smaller foraging groups but still remain part of a larger social unit is the advantage to individual group members in terms of increased mating chances and security from predators.

2.2 Communication

Spider monkeys share their intentions and observations using postures and positions, such as postures of sexual receptivity and of attack. During traveling, they interact with each other over long distances using a particular call which sounds like a horse's whinny. Each individual has its own discernible sound so that other members of the group can easily identify who is calling. This long-distance communication permits spider monkeys to get-together, stay away from enemies, share food and gossip. In order to interact to other group members, they generally use visual and vocal communication [30].

3 Spider Monkey Optimization algorithm

Social behavior of spider monkeys inspires authors to develop an stochastic optimization technique that mimics fission–fusion social structure (FFSS) based foraging behavior of spider monkeys. Following are the key features of the FFSS.

1. The fission–fusion social structure based animals are social and live in groups of 40–50 individuals. The FFSS of swarm may reduce the foraging competition among group members by dividing them into sub-groups in order to search food.
2. A female (global Leader) generally leads the group and is responsible for searching food sources. If she is not able to get enough food for the group, she divides the group into smaller subgroups (size varies from 3 to 8 members) that forage independently.
3. Sub-groups are also supposed to be lead by a female (local leader) who becomes decision-maker for planning an efficient foraging route each day.
4. The members of these subgroups then communicate within and outside the subgroup depending upon the availability of food and to maintain territorial boundaries.

In the developed strategy, the foraging behavior of FFSS based animals (e.g. spider monkeys) is divided into four steps. First, the group starts food foraging and evaluates their distance from the food. In the second step, based on the distance from the foods, group members update their positions and again evaluate distance from the food sources. Furthermore, in the third step, the local leader updates its best position within the group and if the position is not updated for a specified number of times then all members of that group start searching of the food in different directions. Next, in the fourth step, the global leader, updates its ever best position and in case of stagnation, it splits the group into smaller size subgroups. All the four steps mentioned before, are continuously executed until the desired output is achieved. There are two important control parameters necessary to introduce in the proposed strategy, one is '*GlobalLeaderLimit*' and another is '*LocalLeaderLimit*' which helps local and global leaders to take appropriate decisions.

The control parameter *LocalLeaderLimit* is used to avoid stagnation i.e., if a local group leader does not update herself in a specified number of times then that group is re-directed to a different direction for foraging. Here, the term 'specified number of times' is referred as *LocalLeaderLimit*. Another

control parameter, *GlobalLeaderLimit* is used for the same purpose for global leader. The global leader breaks the group into smaller sub-groups if she does not update in a specified number of times.

The proposed strategy follows self-organization and division of labour properties for obtaining intelligent swarming behaviors of animals. As animals updating their positions by learning from local leader, global leader and self experience in the first and second steps of algorithm, it shows positive feedback mechanisms of self-organization. The third step, in which the stagnated group members are re-directed to different directions for food searching, is responsible for fluctuations in the food foraging process. In the fourth step, when the global leader is get stuck, it divides the groups into smaller subgroups for foraging of foods. This phenomena presents division of labour property. 'Local leader limit' and 'Global leader limit' provides negative feedback to help local and global leader's for their decisions.

However, the proposed strategy is inspired from the foraging behavior of spider monkeys, it is different from the natural foraging behavior of spider monkeys. In In the proposed strategy, the post of leader (local or global) is not permanent but depends upon the ability of leader to search of food. Further, the spider monkeys use different type of communication tactics which are not simulated by the proposed strategy. In this way, the proposed strategy is different from the real foraging behavior of spider monkeys.

3.1 Main steps of Spider Monkey Optimization algorithm (SMO)

Similar to the other population-based algorithms, SMO is a trial and error based collaborative iterative process. The SMO process consists of six phases: Local Leader phase, Global Leader phase, Local Leader Learning phase, Global Leader Learning phase, Local Leader Decision phase and Global Leader Decision phase. The position update process in Global Leader phase is inspired from the Gbest-guided ABC [42] and modified version of ABC [16]. The details of each step of SMO implementation are explained below:

3.1.1 Initialization of the population

Initially, SMO generates a uniformly distributed initial population of N spider monkeys where each monkey $SM_i (i = 1, 2, ..., N)$ is a D-dimensional vector. Here D is the number of variables in the optimization problem and SM_i represent the i^{th} Spider Monkey (SM) in the population. Each spider monkey SM corresponds to the potential solution of the problem under consideration. Each SM_i is initialized as follows:

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

where SM_{minj} and SM_{maxj} are bounds of SM_i in j^{th} direction and $U(0, 1)$ is a uniformly distributed random number in the range [0, 1].

3.1.2 Local Leader Phase (LLP)

In the Local Leader phase, each Spider Monkey SM modifies its current position based on the information of the local leader experience as well as local group members experience. The fitness value of so obtained new position is calculated. If the fitness value of the new position is higher than that of the old position, then the SM updates his position with the new one. The position update equation for ith SM (which is a member of kth local group) in this phase is

$$SMnew_{ij} = SM_{ij} + U(0,1) \times (LL_{kj} - SM_{ij}) + U(-1,1) \times (SM_{rj} - SM_{ij}) \quad (2)$$

where SM_{ij} is the jth dimension of the ith SM, LL_{kj} represents the jth dimension of the kth local group leader position. SM_{rj} is the jth dimension of the rth SM which is chosen randomly within kth group such that $r \neq i$, $U(0, 1)$ is a uniformly distributed random number between 0 and 1. Algorithm 1 shows position update process in the Local Leader phase. In Algorithm 1, MG is the maximum number of groups in the swarm and pr is the perturbation rate which controls the amount of perturbation in the current position. The range of pr is [0.1, 0.9] (explained in Sect. 5.1).

Algorithm 1 Position update process in Local Leader Phase:

```
for each k ∈ {1, ..., MG} do
  for each member SM_i ∈ k^th group do
    for each j ∈ {1, ..., D} do
      if U(0, 1) ≥ pr then
        SMnew_ij = SM_ij + U(0,1) × (LL_kj − SM_ij) + U(−1,1) × (SM_rj − SM_ij)
      else
        SMnew_ij = SM_ij
      end if
    end for
  end for
end for
```

3.1.3 Global Leader Phase (GLP)

After completion of the Local Leader phase, the Global Leader phase (GLP) starts. In GLP phase, all the SM's update their position using experience of Global Leader and local group member's experience. The position update equation for this phase is as follows:

$$SMnew_{ij} = SM_{ij} + U(0,1) \times (GL_j - SM_{ij}) + U(-1,1) \times (SM_{rj} - SM_{ij}) \quad (3)$$

where GL_j represents the jth dimension of the global leader position and $j \in \{1, 2, ..., D\}$ is the randomly chosen index.

In this phase, the positions of spider monkeys (SM_i) are updated based on a probabilities $prob_i$ which are calculated using their fitness. In this way a better candidate will have a higher chance to make itself better. The probability $prob_i$ may be calculated using following expression (there may be some other but it must be a function of fitness):

$$prob_i = 0.9 \times \frac{fitness_i}{max\_fitness} + 0.1, \quad (4)$$

here $fitness_i$ is the fitness value of the ith SM and $max\_fitness$ is the maximum fitness in the group. Further, the fitness of the newly generated position of the SM's is calculated and compared with the old one and adopted the better position.

Algorithm 2 Position update process in Global Leader Phase (GLP) :

```
for k = 1 to MG do
  count = 1;
  GS = k^th group size;
  while count < GS do
    for i = 1 to GS do
      if U(0, 1) < prob_i then
        count = count + 1.
        Randomly select j ∈ {1...D}.
        Randomly select SM_r from k^th group s.t. r ≠ i.
        SMnew_ij = SM_ij + U(0, 1) × (GL_j − SM_ij) + U(−1, 1) × (SM_rj − SM_ij).
      end if
    end for
    if i is equal to GS then
      i = 1;
    end if
  end while
end for
```

3.1.4 Global Leader Learning (GLL) phase

In this phase, the position of the global leader is updated by applying the greedy selection in the population i.e., the position of the SM having best fitness in the population is selected as the updated position of the global leader. Further, it is checked that the position of global leader is updating or not and if not then the $GlobalLimitCount$ is incremented by 1.

3.1.5 Local Leader Learning (LLL) phase

In this phase, the position of the local leader is updated by applying the greedy selection in that group i.e., the position of the SM having best fitness in that group is selected as the updated position of the local leader. Next, the updated position of the local leader is compared with the old one and if the local leader is not updated then the $LocalLimitCount$ is incremented by 1.

3.1.6 Local Leader Decision (LLD) phase

If any Local Leader position is not updated up to a predetermined threshold called $LocalLeaderLimit$, then all the members of that group update their positions either by random initialization or by using combined information from Global Leader and Local Leader through Eq. (5), based on the pr.

$$SMnew_{ij} = SM_{ij} + U(0, 1) \times (GL_j - SM_{ij}) + U(0, 1) \times (SM_{ij} - LL_{kj}); \quad (5)$$

It is clear from the Eq. (5) that the updated dimension of this SM is attracted towards global leader and repel from the local leader. The pseudo code of LLD phase for kth group is shown in Algorithm 3. In this algorithm $LocalLimitCount_k$ is the trial counter for the local best solution of kth group.

Algorithm 3 Local Leader Decision Phase:

```
for k = {1...MG} do
  if LocalLimitCount_k > LocalLeaderLimit then
    LocalLimitCount_k = 0.
    GS = k^th group size;
    for i ∈ {1...GS} do
      for each j ∈ {1...D} do
        if U(0, 1) ≥ pr then
          SMnew_ij = SM_minj + U(0, 1) × (SM_maxj − SM_minj)
        else
          SMnew_ij = SM_ij + U(0, 1) × (GL_j − SM_ij) + U(0, 1) × (SM_ij − LL_kj)
        end if
      end for
    end for
  end if
end for
```

3.1.7 Global Leader Decision (GLD) phase

In this phase, the position of global leader is monitored and if it is not updated up to a predetermined number of iterations called $GlobalLeaderLimit$, then the global leader divides the population into smaller groups. Firstly, the population is divided into two groups and then three groups and so on till the maximum number of groups *(MG)* are formed as shown in the Figs. 2, 3, 4, 5. Each time in GLD phase, LLL process is initiated to elect the local leader in the newly formed groups. The case in which maximum number of groups are formed and even then the position of global leader is not updated

then the global leader combines all the groups to form a single group. Thus the proposed algorithm is inspired from fusion–fission structure of SMs. The working of this phase is shown in Algorithm 4. The complete pseudo-code of the proposed strategy is given in Algorithm 5:

Algorithm 4 Global Leader Decision Phase:

if $GlobalLimitCount > GlobalLeaderLimit$ **then**
  $GlobalLimitCount = 0$
  if Number of groups $< MG$ **then**
    Divide the population into groups.
  else
    Combine all the groups to make a single group.
  end if
  Update Local Leaders position.
end if

Algorithm 5 Spider Monkey Optimization (SMO) Algorithm:

1. Initialize Population, $LocalLeaderLimit$, $GlobalLeaderLimit$, pr.
2. Calculate fitness (i.e., the distance of individuals from food sources).
3. Select global leader and local leaders by applying greedy selection (see section 3.1.4, 3.1.5).

while (Termination criteria is not satisfied) **do**
  (i) For finding the objective (Food Source), generate the new positions for all the group members by using self experience, local leader experience and group members experience.[Refer Algorithm 1].
  (ii) Apply the greedy selection process between existing position and newly generated position, based on fitness and select the better one;
  (iii) Calculate the probability $prob_i$ for all the group members using equation (4).
  (iv) Produce new positions for the all the group members, selected by $prob_i$, by using self experience, global leader experience and group members experiences. [Refer Algorithm 2].
  (v) Update the position of local and global leaders, by applying the greedy selection process on all the groups (see section 3.1.4, 3.1.5).
  (vi) If any Local group leader is not updating her position after a specified number of times (*LocalLeaderLimit*) then re-direct all members of that particular group for foraging by algorithm (3)
  (vii) If Global Leader is not updating her position for a specified number of times (*GlobalLeaderLimit*) then she divides the group into smaller groups by algorithm (4).
end while

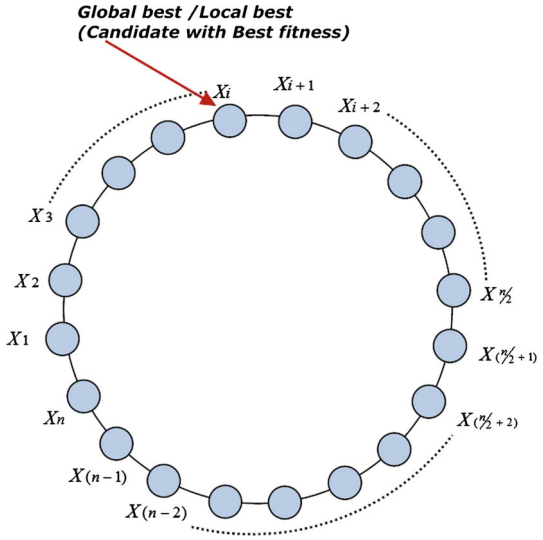

Fig. 2 SMO topology: single group

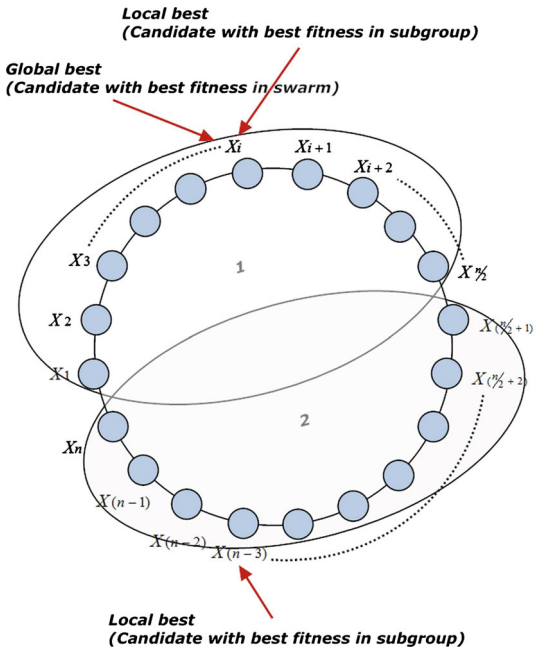

Fig. 3 SMO topology: swarm is divided into two group

3.2 Control parameters in SMO

It is clear from the above discussion that there are four control parameters in SMO algorithm: the value of $LocalLeader Limit$, $GlobalLeaderLimit$, the maximum group MG and perturbation rate pr. Some settings of control parameters are suggested as follows:

- $MG = N/10$, i.e., it is chosen such that minimum number of SM's in a group should be 10
- $GlobalLeaderLimit \in [N/2, 2 \times N]$,
- $LocalLeaderLimit$ should be $D \times N$,
- $pr \in [0.1, 0.9]$,

here, N is the swarm size.

4 Discussion

Exploration and exploitation are the two important characteristics of the population (or swarm) based optimization algorithms [9,17,35,25]. In optimization algorithms, the explo-

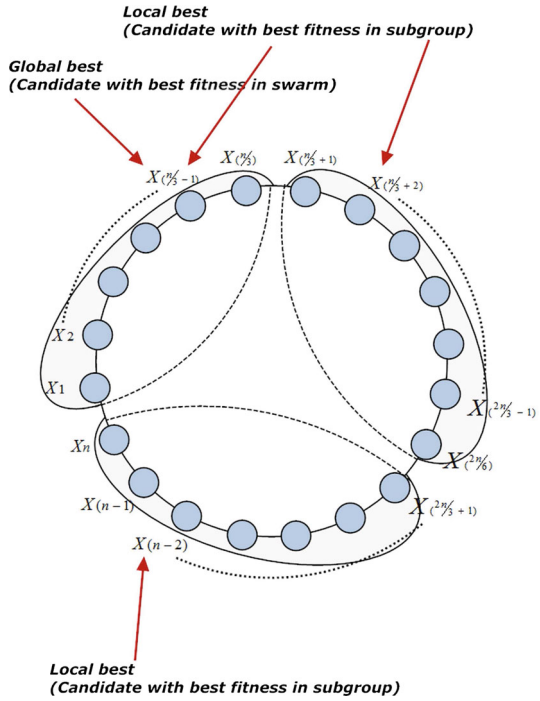

Fig. 4 SMO topology: swarm is divided into three group

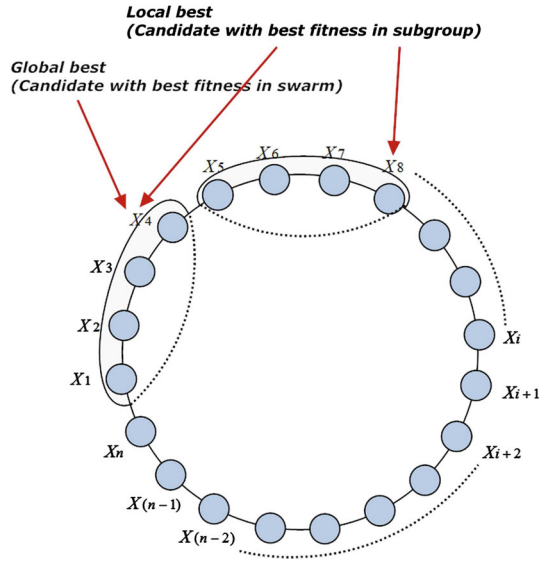

Fig. 5 SMO topology: minimum size group

ration represents the ability to discover the global optimum by investigating the various unknown regions in the solution search space. While, the exploitation represents the ability to find better solutions by implementing the knowledge of the previous good solutions. In behavior, the exploration and exploitation contradict with each other, however both abilities should be well balanced to achieve better optimization performance [40]. It is expected from a good search process that it should explore the new solutions, while maintaining satisfactory performance by exploiting existing solutions [12].

The inherent drawback with most of the population based stochastic algorithms is premature convergence. ABC, DE and PSO are not exceptions [2,15,20]. Dervis Karaboga and Bahriye Akay [15] compared the different variants of ABC and found that ABC shows poor performance and remains inefficient in exploring the search space. The solution search equation of ABC is significantly influenced by a random quantity which helps in exploration at the cost of exploitation of the search space [42]. Further, Mezura-Montes et al. [20] analyzed DE and its variants for global optimization and found that DE has deficiency of premature convergence and stagnation. Also some studies proved that DE sometimes stops proceeding toward the global optima even though the population has not converged to local optima or any other point [18]. Price et al. [28] also drawn the same conclusions regarding DE. However the standard PSO has the capability to get a good solution at a significantly faster rate but, when it is compared to other optimization techniques, it is weak to refine the optimum solution, mainly due to less diversity in later search [2]. On the different side, problem-based tuning of parameters is required in PSO, to get an optimum solution accurately and efficiently [33]. Therefore, it is clear that if a population based algorithm is capable of balancing between exploration and exploitation of the search space, then the algorithm is regarded as an efficient algorithm. From this point of view ABC, PSO and DE are not efficient algorithms. The problems of premature convergence and stagnation is a matter of serious consideration for designing a comparatively efficient nature inspired algorithms (NIAs). By keeping in mind the existing drawbacks of NIAs, SMO is designed in this paper.

In the proposed algorithm, the first phase named '*Local Leader phase*' is used to explore the search region as in this phase all the members of the groups update their positions with high perturbation in the dimensions. The perturbation is high for initial iterations and gradually reducing in later iterations. The second phase '*Global Leader phase*' promotes the exploitation as in this phase, better candidates get more chance to update and in position update process, only single randomly selected dimension is updated. The third and fourth phase namely '*Local Leader Learning phase*' and '*Global Leader Learning phase*', are used to check that the search process is not stagnated. In these two phases, it is checked that the local best and global best solutions are updating or not in a predefined number of trials. If not then the solution is considered stagnated. The fifth phase '*Local Leader Decision phase*' is used to avoid the stagnation or premature convergence of local solutions. In this phase, if the local best solution is not updated in a predefined number of trials (*LocalLeaderLimit*) then all the members of that group are re-initialized. In this phase, all the dimensions of the individ-

uals are initialized either randomly or by using global best solution and local best solution. Further, the *Global Leader Decision phase* is used to avoid stagnation of the global best solution. In this phase if the global best solution is not updated within a predefined number of trials (*GlobalLeaderLimit*) then the group is divided into smaller subgroups. The benefit of this structured group strategy is that initially there is a single group so every newly generated food source is attracted towards the best food source (in this case the global best will be the local best also), thereby converging faster to the solution. But as a results of such exploitative tendency, in many cases, the population may skip the global minima and can get stuck into local minima. Therefore, to avoid this situation, if global minima is not updating itself for a predefined number of times then the group is divided into subgroups. Now every new solution will be attracted towards the respective subgroup's local best food source, hence contributes in the exploration of the search space. When the maximum number of subgroups have been formed and even though the global optima is not updating its position then all the subgroups are combined to form a single group and process repeats itself. Therefore, this phase helps to balance the exploration and exploitation capability of the algorithm while maintaining the convergence speed. From above discussion, it is clear that SMO tries to balance the diversity in the population/swarm and hence can be considered as a new candidate in the field of population based algorithms like ABC [15], PSO [17], DE [35] etc. In ABC, DE and PSO, the position update equation is based on the difference vector and so is the case with SMO. Therefore, it may be considered in the category of ABC, PSO and DE algorithms.

5 Experimental results

In order to analyze the performance of SMO algorithm, 26 different global optimization problems (f_1 to f_{26}) are selected (listed in Table 1). These are continuous, un-biased optimization problems and have different degrees of complexity and multimodality. Test problems $f_1 - f_{19}$ and $f_{24} - f_{26}$ are taken from [1] and test problems $f_{20} - f_{23}$ are taken from [36] with the associated offset values. This set is large enough to include different kinds of problems such as unimodal, multimodal, separable and non separable. A unimodal function $f(x)$ has a single extremum (minimum or maximum in the range specified for x. On the other hand if a function has more than one peaks in the search space i.e., local extremum, this function is called multimodal. Multimodal functions are used to test the ability of algorithms getting rid of local minima. If the exploration process of an algorithm is poor then it will not be able to search the whole search space efficiently and will stuck to some local optima. Functions having flat surfaces are also difficult for algorithms since, the flatness of the function does not give any information to the algorithm to direct the search process towards the minima.

5.1 Experimental setting

Swarm size, perturbation rate (pr), LLL, GLL and maximum number of groups (MG) are the parameters that affect the performance of the SMO, significantly. To fine tune (finding most suitable values) these parameters, sensitivity analyses with different values of these parameters have been carried out. Swarm size is varied from 40 to 160 with step size 20, pr is varied from 0.1 to 0.9 with step size 0.1, MG is varied from 1 to 6 with step size 1, LLL is varied from 100 to 2500 with step size 200 and GLL is varied from 10 to 220 with step size 30. At a time only one parameter is varied while all other parameters are kept fixed. This fine tuning is done with the following assumptions:

- pr is varied from 0.1 to 0.9 while MG, LLL, GLL and Swarm size are fixed to be 5, 1500, 50, and 50, respectively.
- MG is varied from 1 to 6 while LLL, GLL and Swarm size are fixed to be 1500, 50, and 50, respectively. pr is linearly increasing from 0.1 to 0.4 through iterations.
- GLL is varied from 10 to 220 while LLL, MG and Swarm size are fixed to be 1500, 5, and 50, respectively. pr is linearly increasing from 0.1 to 0.4 through iterations.
- LLL is varied from 100 to 2500 while MG, GLL and Swarm size are fixed to be 5, 50, and 50, respectively. pr is linearly increasing from 0.1 to 0.4 through iterations.
- Swarm size is varied from 40 to 160 while LLL, GLL and MG are fixed to be 1500, 50, and 5, respectively. pr is linearly increasing from 0.1 to 0.4 through iterations.

For the purpose of sensitivity analysis, 6 test problems are considered and each problem is simulated 30 times. Effects of these parameters are shown in Fig. 6a–f respectively. It is clear from Fig. 6a that SMO is very sensitive towards pr. SMO performs better on some problems with small values of pr while on some, it performs better with large values of pr. Therefore, the value of pr is adopted linearly increasing over iterations to consider the dynamic nature of parameter pr. It should be noted that this setting of pr is not general. For a particular problem a different setting of pr may provide better results. Further, by analyzing Fig. 6b, it can be stated that the value of $MG = 5$ gives comparatively better results for the given set of test problems. Sensitivity of $GlobalLeaderLimit(GLL)$ and $LocalLeaderLimit(LLL)$ can be analyzed by Fig. 6c, d. It is observed that the value of $GLL = 50$ and $LLL = 1500$ gives better results on the considered benchmark optimization problems. Further, swarm size is analyzed in Fig. 6e, f. It is clear from Fig. 6e that SMO is quite sensitive with respect

Table 1 Benchmark functions used in experiments

Test Problem	Objective function	Search Range	Optimum Value	D	C	AE
Schwefel function 1.2	$f_1(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	$[-100, 100]$	0	30	UN	$1.0E-03$
Step function	$f_2(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	$[-100, 100]$	0	30	US	$1.0E-03$
Schwefel function	$f_3(x) = -\sum_{i=1}^{D} (x_i sin\sqrt{\lvert x_i \rvert})$	$[-500, 500]$	$-418.9829 \times D$	30	MS	$1.0E-03$
Rastrigin	$f_4(x) = 10D + \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i)]$	$[-5.12, 5.12]$	0	30	MS	$1.0E-03$
Levy function 1	$f_5(x) = \frac{\pi}{D}[10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2(1 + 10\sin^2(\pi y_{i+1}))$ $+ (y_D - 1)^2] + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$, where $y_i = 1 + \frac{1}{4}(x_i + 1)$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a; \\ 0, & -a \le x_i \le a; \\ k(-x_i - a)^m, & x_i < a. \end{cases}$	$[-50, 50]$	0	30	MN	$1.0E-03$
Levy function 2	$f_6(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}[(x_i - 1)^2(1 + \sin^2(3\pi x_{i+1}))] + (x_D - 1)^2$ $(1 + \sin^2(2\pi x_D)) + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]$	0	30	MN	$1.0E-03$
Shekel Foxholes Function	$f_7(x) = [\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - A_{ij})^6}]^{-1}$	$[-65.536, 65.536]$	0.998	2	MN	$1.0E-03$
Kowalik function	$f_8(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	$[-5, 5]$	0.0003075	4	MN	$1.0E-03$
Six-hump camel back	$f_9(x) = (4 - 2.1x_1^2 + x_1^4/3)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	$[-5, 5]$	-1.0316	2	MN	$1.0E-03$
Branin RCOS function	$f_{10}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	$x_1 \in [-5, 10]$, $x_2 \in [0, 15]$	0.397887	2	MN	$1.0E-03$
Goldstien & Price function	$f_{11}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $* [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 - 48x_2 - 36x_1x_2 + 27x_2^2)]$	$[-2, 2]$	3	2	MN	$1.0E-03$
Hartmann function 3	$f_{12}(x) = -\sum_{i=1}^{4} \alpha_i exp[-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2]$	$[0, 1]$	-3.86278	3	MN	$1.0E-03$
Hartmann function 6	$f_{13}(x) = -\sum_{i=1}^{4} \alpha_i exp[-\sum_{j=1}^{6} B_{ij}(x_j - Q_{ij})^2]$	$[0, 1]$	-3.32237	6	MN	$1.0E-03$
Shekel function 5	$f_{14}(x) = -\sum_{j=1}^{5}[\sum_{i=1}^{4}(x_i - C_{ij})^2 + \beta_j]^{-1}$	$[0, 10]$	-10.1532	4	MN	$1.0E-03$
Shekel function 7	$f_{15}(x) = -\sum_{j=1}^{7}[\sum_{i=1}^{4}(x_i - C_{ij})^2 + \beta_j]^{-1}$	$[0, 10]$	-10.4029	4	MN	$1.0E-03$
Shekel function 10	$f_{16}(x) = -\sum_{j=1}^{10}[\sum_{i=1}^{4}(x_i - C_{ij})^2 + \beta_j]^{-1}$	$[0, 10]$	-10.5364	4	MN	$1.0E-03$
Cigar	$f_{17}(x) = {x_0}^2 + 100000 \sum_{i=1}^{D} {x_i}^2$	$[-10, 10]$	0	30	US	$1.0E-05$
Axis parallel hyper-ellipsoid	$f_{18}(x) = \sum_{i=1}^{D} i x_i^2$	$[-5.12, 5.12]$	0	30	US	$1.0E-05$
Beale	$f_{19}(x) = [1.5 - x_1(1 - x_2)]^2 + [2.25 - x_1(1 - x_2^2)]^2 + [2.625 - x_1(1 - x_2^3)]^2$	$[-4.5, 4.5]$	0	2	UN	$1.0E-05$
Shifted Sphere	$f_{20}(x) = \sum_{i=1}^{D} z_i^2 + f_{bias}, z = x - o, x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	$[-100, 100]$	-450	10	US	$1.0E-05$
Shifted Schwefel	$f_{21}(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} z_j)^2 + f_{bias}, z = x - o, x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	$[-100, 100]$	-450	10	UN	$1.0E-05$

Table 1 continued

Test Problem	Objective function	Search Range	Optimum Value	D	C	AE
Shifted Griewank	$f_{22}(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{z_i}{\sqrt{i}}) + 1 + f_{bias}, z = (x - o), x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[−600, 600]	−180	10	MN	$1.0E-05$
Shifted Ackley	$f_{23}(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)) + 20 + e + f_{bias}, z = (x - o), x = (x_1, x_2, ...x_D), o = [o_1, o_2, ...o_D]$	[−32, 32]	−140	10	MS	$1.0E-05$
Easom's function	$f_{24}(x) = -\cos x_1 \cos x_2 e^{((-(x_1-\pi)^2-(x_2-\pi)^2))}$	[−10, 10]	−1	2	UN	$1.0E-13$
Dekkers and Aarts	$f_{25}(x) = 10^5 x_1^2 + x_2^2 - (x_1^2 + x_2^2)^2 + 10^{-5}(x_1^2 + x_2^2)^4$	[−20, 20]	−24777	2	MN	$5.0E-01$
Shubert	$f_{26}(x) = -\sum_{i=1}^{5} i\cos((i+1)x_1 + 1)\sum_{i=1}^{5} i\cos((i+1)x_2 + 1)$	[−10, 10]	−186.7309	2	MN	$1.0E-05$

D Dimensions, *C* Characteristic, *U* Unimodal, *M* Multimodal, *S* Separable, *N* Non-Separable, *AE* Acceptable Error

to the swarm size. The optimum value of swarm size around 40 can be observed from the Fig. 6e, f.

To prove the efficiency of SMO algorithm, it is compared with four state-of-art algorithms, namely PSO [4] (based on Standard PSO 2006 but with linearly decreasing inertia weight, modified velocity update equation and a different parameters setting), ABC [14], DE ($DE/rand/bin/1$) [35] and Covariance Matrix Adaptation Evolution Strategies (CMA-ES) [11]. For the comparison, same stopping criteria, number of simulations and maximum number of function evaluations are used for all the considered algorithms. The values of parameters for the considered algorithms are as follows:

SMO parameters setting:

- The Swarm size $N = 50$,
- $MG = 5$,
- $GlobalLeaderLimit = 50$,
- $LocalLeaderLimit = 1500$,
- $pr \in [0.1, 0.4]$, linearly increasing over iterations,

$$pr_{G+1} = pr_G + (0.4 - 0.1)/MIR; \quad pr_1 = 0.1. \tag{6}$$

  where, G is the iteration counter, MIR is the maximum number of iterations.
- The stopping criteria is either maximum number of function evaluations (which is set to be 2.0×10^5) is reached or the corresponding acceptable error (mentioned in Table 1) have been achieved,
- The number of simulations (or run) =100.

ABC parameters setting:

- Colony size $SN = 100$,
- Number of food sources $SN/2$,
- $limit = 1500$ [14],

DE parameters setting:

- The crossover probability $CR = 0.9$ [8],
- The scale factor which controls the implication of the differential variation $F = 0.5$ [27],
- Population size $NP = 50$.

PSO parameters setting:

- Inertia Weight (w), decreases linearly from 1 to 0.1,
- Acceleration coefficients ($c_1 = 2, c_2 = 2$),
- Swarm size $S = 50$.

CMA-ES parameters setting [10]:
All parameter settings for CMA-ES is kept same as in [10]. The results of CMA-ES are obtained by the source code provided at its developer's webpage, http://www.bionik.tu-berlin.de/user/niko/index.html.

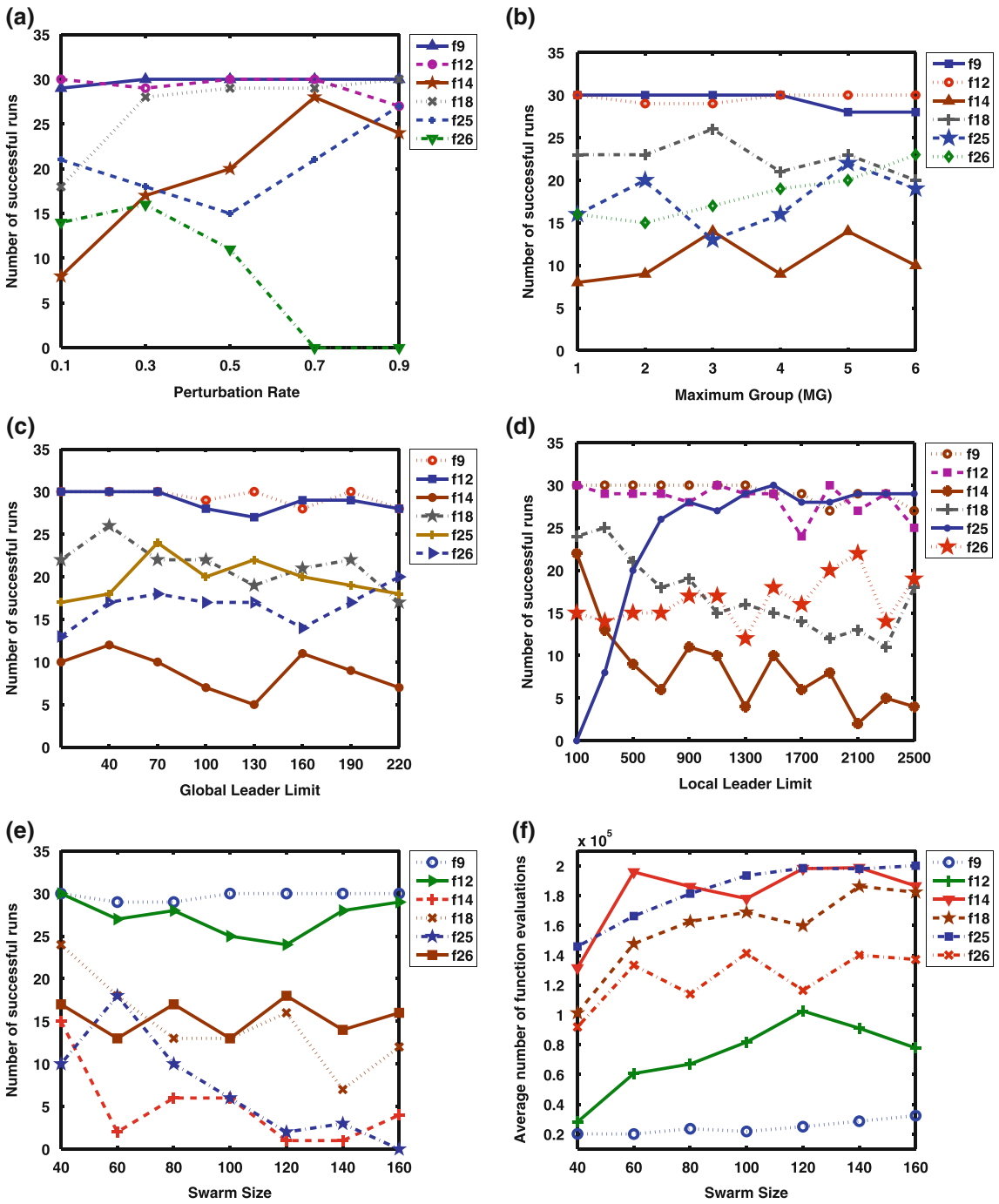

Fig. 6 Effect of parameters on success rate for functions f_9, f_{12}, f_{14}, f_{18}, f_{25} and f_{26} **a** when pr is varied, **b** when MG is varied, **c** when Global Leader Limit is varied, **d** when Local Leader Limit is varied, **e** Swarm Size v/s Successful Runs, **f** Swarm Size v/s Average number of function evaluations

5.2 Results analysis of experiments

Numerical results for benchmark problems of Table 1 with the experimental settings given in Sect. 5.1 are given in Table 2. In Table 2, standard deviation (SD), mean error (ME), average number of function evaluations (AFE), and success rate (SR) are reported for test problems f_1 to f_{26}. The *AFE* is the average of the function evaluations that are required to reach at the termination criteria in 100 runs. In other words,

$$AFE = \frac{\sum_{i=1}^{100} \#\text{ of function evaluations to meet the termination criteria for run } i}{100}$$

Table 2 Experimental results

Test Function	Algorithm	SD	ME	AFE	SR
f_1	DE	1.42E−04	8.68E−04	27378	100
	PSO	6.72E−05	9.34E−04	45914.5	100
	ABC	2.02E−04	7.57E−04	35901	100
	CMA-ES	2.90E−04	7.10E−04	21248	100
	SMO	8.38E−05	8.88E−04	15128.19	100
f_2	DE	3.54E−01	1.00E−01	25858	95
	PSO	2.36E−04	2.53E−04	38273.5	100
	ABC	5.92E−04	6.35E−04	20244	100
	CMA-ES	1.77E+00	1.44E+00	72184	36
	SMO	1.20E−05	2.34E−04	12018.41	100
f_3	DE	6.12E+02	2.24E+03	200000	0
	PSO	6.70E+02	2.80E+03	200000	0
	ABC	1.18E+01	1.19E+00	170335	76
	CMA-ES	8.96E+02	7.64E+03	200000	0
	SMO	1.11E+02	7.67E+01	180525.04	65
f_4	DE	4.93E+00	1.54E+01	200000	0
	PSO	1.35E+01	3.80E+01	200000	0
	ABC	3.14E−04	4.72E−04	87039	100
	CMA-ES	1.36E+01	5.18E+01	200000	0
	SMO	2.33E−04	3.39E−04	83158.66	100
f_5	DE	1.45E−02	2.90E−03	24490.5	98
	PSO	1.44E−02	2.98E−03	52777.5	98
	ABC	2.19E−04	7.48E−04	29301	100
	CMA-ES	3.40E−02	7.20E−04	23254	88
	SMO	1.61E−04	6.52E−04	16176	100
f_6	DE	1.78E−03	1.15E−03	26753	97
	PSO	2.58E−03	1.62E−03	51446	93
	ABC	1.96E−04	7.89E−04	32604	100
	CMA-ES	4.50E−03	1.70E−03	13756	86
	SMO	1.43E−04	1.07E−04	23728.83	100
f_7	DE	$7.15E-05$	2.39E−04	2632.5	100
	PSO	3.09E−04	3.06E−04	3778	100
	ABC	2.79E−04	1.93E−04	1306	100
	CMA-ES	6.87E+00	1.04E+01	200000	0
	SMO	2.68E−04	2.02E−04	919.71	100
f_8	DE	2.14E−04	8.32E−04	8170	97
	PSO	1.33E−04	8.39E−04	1957	100
	ABC	1.34E−04	8.43E−04	7525.37	100
	CMA-ES	4.20E−03	1.50E−03	13434	88
	SMO	1.38E−04	8.51E−04	2214.37	100
f_9	DE	1.01E−04	4.44E−04	1686.5	100
	PSO	3.05E−04	4.80E−04	1318	100
	ABC	3.11E−04	5.08E−04	899	100
	CMA-ES	5.20E−11	6.03E−04	619	100
	SMO	2.99E−04	4.02E−04	529.65	100
f_{10}	DE	1.32E−04	4.89E−04	2081	100
	PSO	2.82E−04	4.81E−04	1445.5	100
	ABC	2.79E−04	4.77E−04	1480	100
	CMA-ES	1.40E−07	3.98E−04	594	100
	SMO	2.91E−04	4.25E−04	673.2	100
f_{11}	DE	1.20E−04	4.78E−04	1608	100
	PSO	2.70E−04	4.83E−04	1900.5	100
	ABC	3.08E−04	4.88E−04	2925.11	100
	CMA-ES	2.51E−01	1.43E−02	2052	78
	SMO	2.96E−04	4.85E−04	866.25	100
f_{12}	DE	1.04E−04	5.01E−04	1334	100
	PSO	2.46E−04	5.77E−04	1080.5	100
	ABC	2.64E−04	5.48E−04	1415	100
	CMA-ES	4.80E−08	9.25E−04	996	100
	SMO	2.66E−04	5.15E−04	598.95	100
f_{13}	DE	5.22E−02	8.84E−02	149112	26
	PSO	5.63E−02	4.29E−02	76074	64
	ABC	2.33E−04	6.81E−04	4652	100
	CMA-ES	5.80E−02	3.80E−03	22330	48
	SMO	1.18E−02	1.86E−03	27278.86	99
f_{14}	DE	1.05E+00	1.50E−01	7962	98
	PSO	1.36E+00	2.75E−01	16708.5	96
	ABC	2.56E−04	5.74E−04	6656	100
	CMA-ES	2.58E−02	3.18E−02	42561	40
	SMO	2.58E−04	6.40E−04	17592.18	100
f_{15}	DE	1.65E−04	5.63E−04	3659.5	100
	PSO	2.35E−04	6.89E−04	5435	100
	ABC	2.93E−04	5.90E−04	8222.32	100
	CMA-ES	1.74E−01	1.25E−02	35632	48
	SMO	2.57E−04	6.62E−04	9519.46	100
f_{16}	DE	6.67E−01	6.76E−02	5620	99
	PSO	2.56E−04	6.57E−04	5463.5	100
	ABC	2.82E−04	5.75E−04	9584.35	100
	CMA-ES	2.54E−02	2.15E−03	11234	52
	SMO	2.48E−04	6.53E−04	7605.82	100
f_{17}	DE	1.34E−06	8.63E−06	40678	100
	PSO	5.46E−07	9.38E−06	69416.5	100
	ABC	1.82E−06	8.25E−06	63993	100
	CMA-ES	6.61E−06	2.51E−05	10318	100
	SMO	9.45E−07	8.94E−06	22477.95	100
f_{18}	DE	1.31E−06	8.60E−06	26463.5	100
	PSO	6.13E−07	9.37E−06	44129.5	100
	ABC	2.00E−06	8.18E−06	41861	100
	CMA-ES	1.06E−06	7.03E−06	16463	100
	SMO	7.62E−07	8.96E−06	14679.72	100
f_{19}	DE	1.13E−06	4.72E−06	1849	100
	PSO	2.67E−06	4.22E−06	2762	100
	ABC	2.42E−06	7.81E−06	31948.76	100
	CMA-ES	1.01E−06	1.17E−06	1247.16	100

Table 2 continued

Test Function	Algorithm	SD	ME	*AFE*	SR
	SMO	2.58E−06	4.81E−06	1569.15	100
f_{20}	DE	2.03E−06	7.46E−06	10805.5	100
	PSO	1.47E−06	8.07E−06	15854.5	100
	ABC	2.16E−06	7.35E−06	17112	100
	CMA-ES	8.12E−06	6.38E−06	13805.5	100
	SMO	1.86E−06	7.65E−06	5898.42	100
f_{21}	DE	3.39E+03	1.21E+05	200000	0
	PSO	1.01E+03	7.84E+02	200000	0
	ABC	3.96E+03	1.31E+04	200000	0
	CMA-ES	1.00E+03	1.84E+02	200000	0
	SMO	1.18E+03	1.84E+04	200000	0
f_{22}	DE	1.20E−02	1.34E−02	165684	22
	PSO	2.77E−02	4.28E−02	198768.5	2
	ABC	2.86E−03	1.09E−03	101707.2	86
	CMA-ES	5.63E−04	1.03E−04	95707.2	96
	SMO	6.03E−03	2.68E−03	130922.94	77
f_{23}	DE	1.21E−06	8.85E−06	15959	100
	PSO	8.69E−07	9.10E−06	24687.5	100
	ABC	1.71E−06	8.09E−06	32415	100
	CMA-ES	2.35E−06	5.37E−06	17365	100
	SMO	1.13E−06	8.66E−06	9069.39	100
f_{24}	DE	7.36E−15	4.50E−14	5210	100
	PSO	2.87E−14	5.13E−14	9778	100
	ABC	5.64E−07	5.71E−08	128925.18	52
	CMA-ES	8.17E−14	7.83E−14	9612	100
	SMO	2.69E−14	4.71E−14	11789.91	100
f_{25}	DE	1.12E−03	4.90E−01	2725.5	100
	PSO	5.64E−03	4.91E−01	4979	100
	ABC	5.25E−03	4.90E−01	2567	100
	CMA-ES	6.07E−03	7.91E−01	1725.5	100
	SMO	4.98E−03	4.89E−01	1258.29	100
f_{26}	DE	2.16E−06	4.78E−06	9663	100
	PSO	4.01E−04	1.01E−04	72252.5	83
	ABC	5.95E−06	5.32E−06	8248.56	100
	CMA-ES	6.27E−06	7.32E−06	14262	100
	SMO	5.58E−06	4.97E−06	4379.76	100

Table 2 shows that most of the time SMO outperforms in terms of reliability, efficiency and accuracy. Some more intensive statistical analyses based on boxplots, the Mann–Whitney *U* rank sum test, performance indices [6], and acceleration rate (AR) [29] have been carried out for the results of DE, PSO, ABC, CMA-ES and SMO.

5.3 Statistical analysis

DE, PSO, ABC, CMA-ES and SMO are compared based on SR, AFE, and ME. First SR of all these algorithms is compared and if it is not possible to distinguish the performance of algorithms based on SR then comparison is made on the basis of AFE. ME is used for comparison if the comparison is not possible on the basis of SR and AFE both. From the results shown in Table 2, it is clear that SMO costs less on 14 function (f_1, f_2, f_4, f_5, f_6, f_7, f_9, f_{11}, f_{12}, f_{18}, f_{20}, f_{23}, f_{25}, f_{26}) among all the considered algorithms. As these functions include unimodel, multimodel, separable, non separable, lower and higher dimension functions, it can be stated that SMO balances the exploration and exploitation capabilities efficiently. ABC outperforms SMO over five test functions (f_3, f_{13}, f_{14}, f_{15}, f_{21}, f_{22}) and four are multimodel functions. It shows that ABC perform better on multimodel functions as the solution search equation of ABC is significantly influenced by a random quantity which helps in exploration at the cost of exploitation of the search space [32]. CMA-ES outperforms over SMO on six test functions (f_{10}, f_{17}, f_{19}, f_{21}, f_{22}, f_{24}) among these four test functions are unimodel functions. Generally speaking, the cost of CMA-ES is lower than those of SMO, ABC, DE and PSO for the unimodal functions. This is because CMA-ES is a local method devised for optimal exploitation of local information [21]. DE outperform over only two test functions (f_{15}, f_{24}) in which f_{15} is unimodel and f_{24} is multimodel. Further, PSO performs better than SMO over five test function (f_8, f_{15}, f_{16}, f_{21}, f_{24}) which all are non separable functions as well as four are multimodel functions. Overall, SMO is better than DE over 24 test functions, PSO over 21 test functions, ABC over 20 test functions, and CMA-ES over 20 test functions of mixed characteristics, when compared separately. It means that when the results of all functions are evaluated together, the SMO algorithm is the cost effective algorithm for most of the functions.

For the purpose of comparison in terms of consolidated performance, boxplot analyses have been carried out for all the considered algorithms. The empirical distribution of data is efficiently represented graphically by the boxplot analysis tool [41]. The boxplots of average number of function evaluations for DE, PSO, ABC, CMA-ES and SMO are shown in Fig. 7. It is clear from Fig. 7a that SMO is cost effective in terms of function evaluations as interquartile range and median of average number of function evaluations are low for SMO. When the considered algorithms are compared on the basis of standard deviation and success rate, it can be observed from Fig. 7b, c that ABC and SMO have equal performance, while they perform better than DE, PSO and CMA-ES.

Though, it is clear from box plots that SMO is cost effective than DE, PSO, ABC, and CMA-ES i.e., SMO's result differs from the other, now to check, whether there exists any significant difference between algorithm's output or this difference is due to some randomness, we require another statistical test. It can be observed from boxplots Fig. 7a that

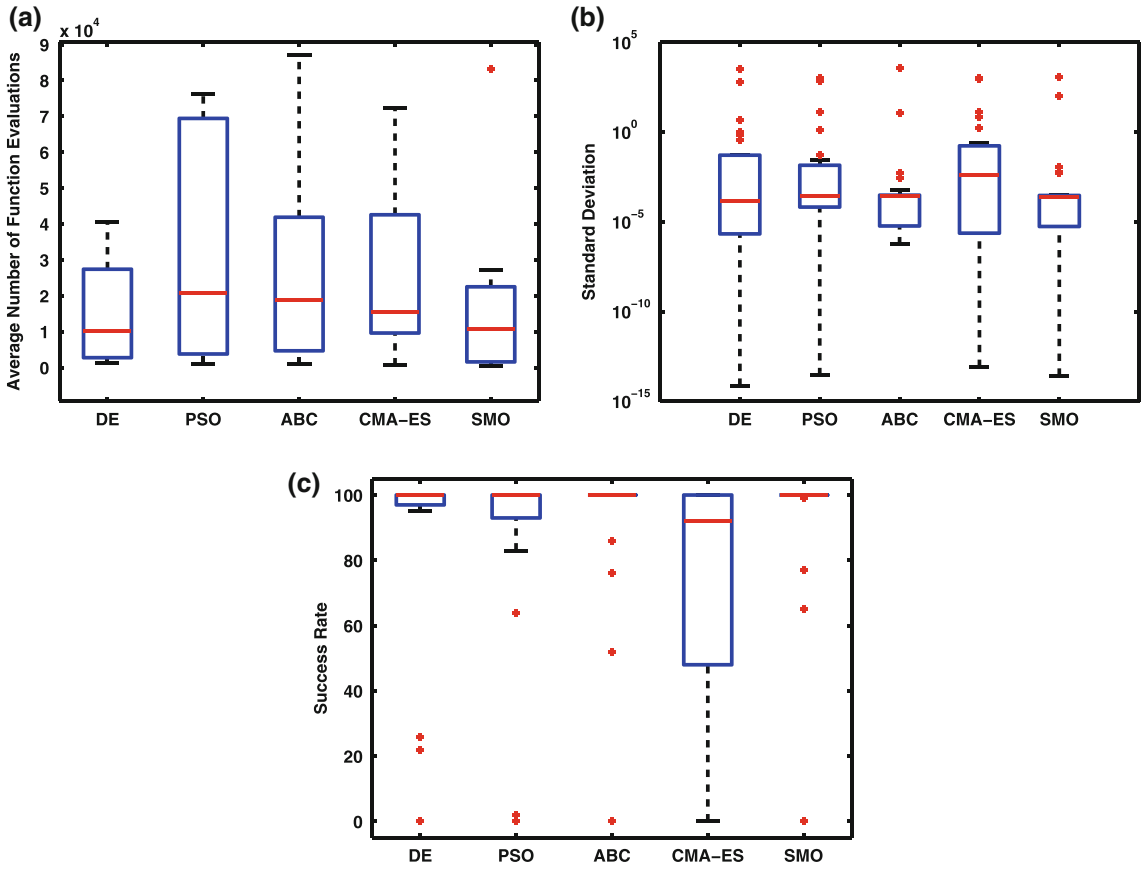

Fig. 7 Boxplots graph for **a** average number of function evaluation, **b** standard deviation, and **c** success rate

average number of function evaluations used by the considered algorithms to solve the different problems are not normally distributed, so a non-parametric statistical test is required to compare the performance of the algorithms. The Mann–Whitney U rank sum [19], a non-parametric test, is well established test for comparison among non-Gaussian data. In this paper, this test is performed at 5% level of significance ($\alpha = 0.05$) between SMO–DE, SMO–PSO, SMO–ABC, and SMO–CMA-ES.

Table 3 shows the results of the Mann–Whitney U rank sum test for the average function evaluations of 100 simulations. First we observe the significant difference by Mann–Whitney U rank sum test i.e., whether the two data sets are significantly different or not. If significant difference is not seen (i.e., the null hypothesis is accepted) then sign '=' appears and when significant difference is observed i.e., the null hypothesis is rejected then compare the average number of function evaluations. And we use signs '+' and '−' for the case where SMO takes less or more average number of function evaluations than the other algorithms, respectively. Therefore in Table 3, '+' shows that SMO is significantly better and '−' shows that SMO is worse. As Table 3 includes 79 '+' signs out of 104 comparisons. Therefore, it can be concluded that the results of SMO is significantly cost effective than DE, PSO, ABC and CMA-ES over considered test problems.

Further, to compare the considered algorithms, by giving weighted importance to SR, AFE and SD, performance indices (PIs) are calculated [6]. The values of PI for the DE, PSO, ABC and CMA-ES are calculated by using following equations:

$$PI = \frac{1}{N_p}\sum_{i=1}^{N_p}(k_1\alpha_1^i + k_2\alpha_2^i + k_3\alpha_3^i)$$

Where $\alpha_1^i = \frac{Sr^i}{Tr^i}$; $\alpha_2^i = \begin{cases} \frac{Mf^i}{Af^i}, & \text{if } Sr^i > 0. \\ 0, & \text{if } Sr^i = 0. \end{cases}$; and $\alpha_3^i = \frac{Mo^i}{Ao^i}$

$i = 1, 2, ..., N_p$

- Sr^i = Successful simulations/runs of ith problem.
- Tr^i = Total simulations of ith problem.
- Mf^i = Minimum of average number of function evaluations used for obtaining the required solution of ith problem.
- Af^i = Average number of function evaluations used for obtaining the required solution of ith problem.
- Mo^i = Minimum of standard deviation obtained for the ith problem.

Table 3 Comparison based on mean function evaluations and the Mann–Whitney U rank sum test at a $\alpha = 0.05$ significance level ('+' indicates SMO is significantly better, '−' indicates SMO is significantly worst and '=' indicates that there is no significant difference), TP: Test Problem

TP	Mann–Whitney U rank sum test with *SMO*				TP	Mann–Whitney U rank sum test with *SMO*			
	DE	PSO	ABC	CMA-ES		DE	PSO	ABC	CMA-ES
f_1	+	+	+	+	f_{14}	−	−	−	+
f_2	+	+	+	+	f_{15}	−	−	−	+
f_3	+	+	−	+	f_{16}	−	−	+	+
f_4	+	+	+	+	f_{17}	+	+	+	−
f_5	+	+	+	+	f_{18}	+	+	+	+
f_6	+	+	+	−	f_{19}	+	+	+	−
f_7	+	+	+	+	f_{20}	+	+	+	+
f_8	+	−	+	+	f_{21}	=	=	=	=
f_9	+	+	+	+	f_{22}	+	+	−	−
f_{10}	+	+	+	−	f_{23}	+	+	+	+
f_{11}	+	+	+	+	f_{24}	−	−	+	−
f_{12}	+	+	+	+	f_{25}	+	+	+	+
f_{13}	+	+	−	−	f_{26}	+	+	+	+

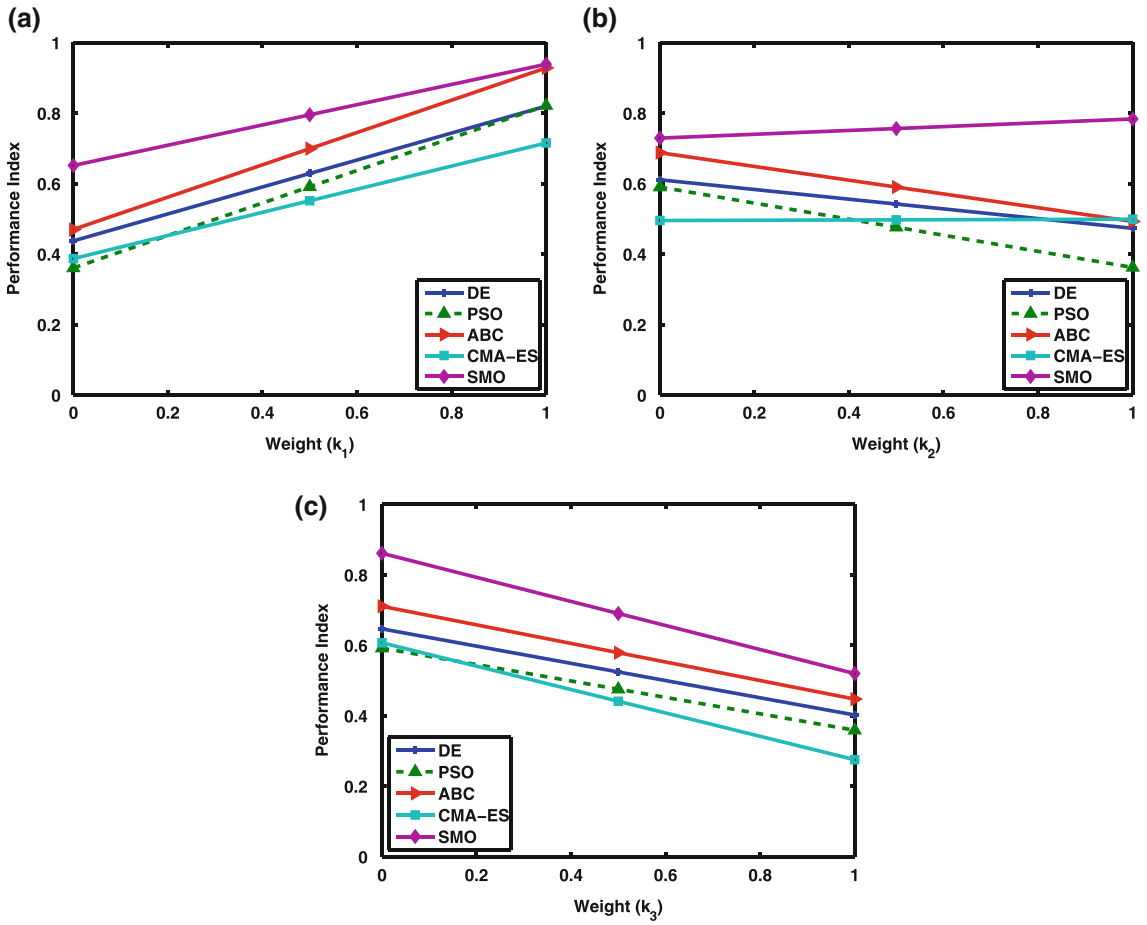

Fig. 8 Performance index for test problems; **a** for case (1), **b** for case (2) and **c** for case (3)

Table 4 Acceleration rate (AR) of SMO as compared to the DE, PSO, ABC and CMA-ES

TP	DE	PSO	ABC	CMA-ES
f_1	1.809734013	3.035029306	2.373119322	1.404530218
f_2	2.151532524	3.184572668	1.684415825	6.006118946
f_3	1.10787955	1.10787955	0.943553315	1.10787955
f_4	2.405041159	2.405041159	1.046661887	2.405041159
f_5	1.514002226	3.262704006	1.81138724	1.43756182
f_6	1.127447076	2.168079926	1.374024762	0.579716741
f_7	2.862315295	4.107816594	1.42001283	217.4598515
f_8	3.689536979	0.883772811	3.398424834	6.066736815
f_9	3.184178231	2.488435759	1.697347305	1.168696309
f_{10}	3.091206179	2.147207368	2.19845514	0.882352941
f_{11}	1.856277056	2.193939394	3.376750361	2.368831169
f_{12}	2.227230988	1.803990316	2.362467652	1.662910093
f_{13}	5.466210831	2.788752902	0.170534986	0.818582595
f_{14}	0.452587456	0.94976859	0.378349926	2.419313581
f_{15}	0.384423066	0.570935746	0.863738069	3.74306946
f_{16}	0.738907836	0.718331488	1.260133687	1.477026803
f_{17}	1.809684602	3.088204218	2.846923318	0.459027625
f_{18}	1.80272512	3.006154068	2.851621148	1.121479156
f_{19}	1.178344964	1.760188637	20.36055189	0.794799732
f_{20}	1.831931263	2.687923207	2.901115892	2.340542043
f_{21}	1	1	1	1
f_{22}	1.265507786	1.518209872	0.776847816	0.731019331
f_{23}	1.75965528	2.722068408	3.574110276	1.914682244
f_{24}	0.441903288	0.829353235	10.93521325	0.815273399
f_{25}	2.166034857	3.956957458	2.040070254	1.371305502
f_{26}	2.206285276	16.49690851	1.883336073	3.256342813

TP test problems

- Ao^i = Standard deviation obtained by an algorithm for the ith problem.
- N_p = Total number of optimization problems evaluated.

The weights assigned to SR, AFE and SD are represented by k_1, k_2 and k_3 respectively, where $k_1 + k_2 + k_3 = 1$ and $0 \leq k_1, k_2, k_3 \leq 1$. To calculate the PIs, equal weights are assigned to two variables while weight of the remaining variable vary from 0 to 1 as given in [6]. Following are the resultant cases:

1. $k_1 = W, k_2 = k_3 = \frac{1-W}{2}, 0 \leq W \leq 1;$
2. $k_2 = W, k_1 = k_3 = \frac{1-W}{2}, 0 \leq W \leq 1;$
3. $k_3 = W, k_1 = k_2 = \frac{1-W}{2}, 0 \leq W \leq 1$

The graphs corresponding to each of the cases (1), (2) and (3) for the considered algorithms are shown in Fig. 8a–c respectively. In these figures the weights k_1, k_2 and k_3 are represented by horizontal axis while the PI is represented by the vertical axis.

In case (1), AFE and SD are given equal weights. PIs of the considered algorithms are superimposed in Fig. 8a for comparison of the performance. It is observed that PI of SMO is higher than the considered algorithms. In case (2), equal weights are assigned to SR and AFE and in case (3), equal weights are assigned to SR and AFE. It is clear from Fig. 8b, c that the algorithms perform same as in case (1).

Further, we compare the convergence speed of the considered algorithms by measuring the AFEs. A smaller AFEs means higher convergence speed. In order to minimize the effect of the stochastic nature of the algorithms, the reported function evaluations for each test problem is averaged over 100 runs. In order to compare convergence speeds, we use the acceleration rate (AR) which is defined as follows, based on the AFEs for the two algorithms ALGO and SMO:

$$AR = \frac{AFE_{ALGO}}{AFE_{SMO}}, \tag{7}$$

where, ALGO$\in$ {DE, PSO, ABC, CMA-ES} and $AR > 1$ means SMO is faster. In order to investigate the AR of the proposed algorithm as compared to the considered algorithms, results of Table 2 are analyzed and the value of AR is calculated using equation (7). Table 4 shows a comparison between SMO and DE, SMO and PSO, SMO and ABC, and SMO and CMA-ES in terms of AR. It is clear from the Table 4 that convergence speed of SMO is better than considered algorithms for most of the functions.

6 Conclusion

In this paper, a new swarm intelligence algorithm for optimization is proposed. The inspiration is from the social behavior of spider monkeys. The proposed algorithm is proved to be very flexible in the category of swarm intelligence based algorithms. With the help of numerical experiments over test problems, it has been shown that, for most of the problems the reliability (due to success rate), efficiency (due to average number of function evaluations) and accuracy (due to mean objective function value) of SMO is competitive or similar to those of DE, PSO, ABC and CMA-ES. Hence, it may be concluded that SMO is going to be a competing candidate in the field of swarm intelligence based optimization algorithms.

Acknowledgments The authors acknowledge the anonymous reviewers for their valuable comments and suggestions.

References

1. Ali MM, Khompatraporn C, Zabinsky ZB (2005) A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. J. Global Optim. 31(4):635–672

2. Angeline P (1998) Evolutionary optimization versus particle swarm optimization: philosophy and performance differences. In: Evolutionary programming VII. Springer, Berlin, pp 601–610
3. Bonabeau E, Dorigo M, Theraulaz G (1999) Swarm intelligence: from natural to artificial systems. Oxford University Press, New York
4. Clerc M (2012) A method to improve standard PSO. http://clerc.maurice.free.fr/pso/Design_efficient_PSO.pdf. Retrieved on Jan 2012
5. De Castro LN, Von Zuben FJ (1999) Artificial immune systems: Part I-basic theory and applications. Universidade Estadual de Campinas, Dezembro de, Tech. Rep
6. Thakur M. Deep K (2007) A new crossover operator for real coded genetic algorithms. Appl Math Comput 188(1):895911
7. Dorigo M, Stützle T (2004) Ant colony optimization. The MIT Press, Cambridge
8. Gamperle R, Muller SD, Koumoutsakos A (2002) A parameter study for differential evolution. Adv Intell Syst Fuzzy Syst Evol Comput 10:293–298
9. Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Professional, Upper Saddle River
10. Hansen N (2006) The cma evolution strategy: a comparing review. In: Towards a new evolutionary computation. Springer, Heidelberg, pp 75–102
11. Hansen N, Ostermeier A (1996) Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In: Proceedings of IEEE international conference on evolutionary computation, pp 312–317. IEEE
12. Hofmann K, Whiteson S, de Rijke M (2011) Balancing exploration and exploitation in learning to rank online. Adv Inform Retr 5:251–263
13. Jeanne RL (1986) The evolution of the organization of work in social insects. Monitore Zoologico Italiano 20(2):119–133
14. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Techn. Rep. TR06. Erciyes University Press, Erciyes
15. Karaboga D, Akay B (2009) A comparative study of artificial bee colony algorithm. Appl Math Comput 214(1):108–132
16. Karaboga D, Akay B (2011) A modified artificial bee colony (ABC) algorithm for constrained optimization problems. Appl Soft Comput 11(3):3021–3031
17. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, 1995, vol 4, pp 1942–1948. IEEE
18. Lampinen J, Zelinka I (2000) On stagnation of the differential evolution algorithm. In: Proceedings of MENDEL, Citeseer, pp 76–83
19. Mann HB, Whitney DR (1947) On a test of whether one of two random variables is stochastically larger than the other. Annals Math Stat 18(1):50–60
20. Mezura-Montes E, Velázquez-Reyes J, Coello CA (2006) A comparative study of differential evolution variants for global optimization. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation. ACM Press, New York, pp 485–492
21. Milano M, Koumoutsakos P, Schmidhuber J (2004) Self-organizing nets for optimization. IEEE Trans Neural Netw 15(3):758–765
22. Milton K (1993) Diet and social organization of a free-ranging spider monkey population: the development of species-typical behavior in the absence of adults. In: Juvenile primates: life history, development, and behavior. Oxford University Press, Oxford, pp 173–181
23. Norconk MA, Kinzey WG (1994) Challenge of neotropical frugivory: travel patterns of spider monkeys and bearded sakis. Am J Primatol 34(2):171–183
24. Oster GF, Wilson EO (1979) Caste and ecology in the social insects. Princeton Univ ersity Press, Princeton
25. Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst Mag 22(3):52–67
26. Passino KM (2010) Bacterial foraging optimization. Int J Swarm Intell Res (IJSIR) 1(1):1–16
27. Price KV (1996) Differential evolution: a fast and simple numerical optimizer. In: Fuzzy information processing society, 1996. NAFIPS. 1996 Biennial conference of the North American, pp 524–527. IEEE
28. Price KV, Storn RM, Lampinen JA (2005) Differential evolution: a practical approach to global optimization. Springer, Berlin
29. Rahnamayan S, Tizhoosh HR, Salama MMA (2008) Opposition-based differential evolution. IEEE Trans Evol Comput 12(1):64–79
30. Ramos-Fernandez G (2001) Patterns of association, feeding competition and vocal communication in spider monkeys, Ateles geoffroyi. Dissertations, University of Pennsylvania. http://repository.upenn.edu/dissertations/AAI3003685. 1 Jan 2001
31. Sartore J (2011) Spider monkey images. http://animals.nationalgeographic.com/animals/mammals/spider-monkey. Retrived on 21 Decmber 2011
32. Sharma H, Bansal JC, Arya KV (2012) Opposition based lévy flight artificial bee colony. Memet Comput 5(3):213–227
33. Shi Y, Eberhart R (1998) Parameter selection in particle swarm optimization. In: Evolutionary programming VII. Springer, Heidelberg, pp 591–600
34. Simmen B, Sabatier D (1996) Diets of some french guianan primates: food composition and food choices. Int J Primatol 17(5):661–693
35. Storn R, Price K (1997) Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. J Global Optim 11:341–359
36. Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL Report
37. Symington MMF (1990) Fission–fusion social organization inateles andpan. Int J Primatol 11(1):47–61
38. van Roosmalen MGM (1985) Instituto Nacional de Pesquisas da Amazônia. Habitat preferences, diet, feeding strategy and social organization of the black spider monkey (ateles paniscus paniscus linnaeus 1758) in surinam. Wageningen : Roosmalen
39. Vesterstrom J, Thomsen R (2004) A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: Congress on evolutionary computation, 2004. CEC2004., vol 2, pp 1980–1987. IEEE
40. Weise T, Chiong R, Tang K (2012) Evolutionary optimization: pitfalls and booby traps. J Comput Sci Technol 27(5):907–936
41. Williamson DF, Parker RA, Kendrick JS (1989) The box plot: a simple visual method to interpret data. Annals Intern Med 110(11):916
42. Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Computat 217(7):3166–3173