Soft Comput
DOI 10.1007/s00500-016-2419-0

FOUNDATIONS

Spider monkey optimization algorithm for constrained optimization problems

Kavita Gupta[1] · Kusum Deep[1] · Jagdish Chand Bansal[2]

Abstract In this paper, a modified version of spider monkey optimization (SMO) algorithm for solving constrained optimization problems has been proposed. To the best of author's knowledge, this is the first attempt to develop a version of SMO which can solve constrained continuous optimization problems by using the Deb's technique for handling constraints. The proposed algorithm is named constrained spider monkey optimization (CSMO) algorithm. The performance of CSMO is investigated on the well-defined constrained optimization problems of CEC2006 and CEC2010 benchmark sets. The results of the proposed algorithm are compared with constrained versions of particle swarm optimization, artificial bee colony and differential evolution. Outcome of the experiment and the discussion of results demonstrate that CSMO handles the global optimization task very well for constrained optimization problems and shows better performance in comparison with compared algorithms. Such an outcome will be an encouragement for the research community to further explore the potential of SMO in solving benchmarks as well as real-world problems, which are often constrained in nature.

Communicated by A. Di Nola.

✉ Kavita Gupta
gupta.kavita3043@gmail.com

Kusum Deep
kusumdeep@gmail.com

Jagdish Chand Bansal
jcbansal@gmail.com

[1] Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee, Uttarakhand 247667, India

[2] Department of Applied Mathematics, South Asian University, Akbar Bhawan, Chankyapuri, New Delhi 110021, India

Keywords Spider monkey optimization · Constrained optimization · CEC2006 · CEC2010 · Deb's technique

1 Introduction

Most of the real-world optimization problems are constrained in nature. Solving such optimization problems involves finding the values of a set of decision variables such that the objective function value is optimized while satisfying all the constraints and variable bounds. The general mathematical form of a single objective nonlinear constrained optimization problem is as follows:

$$\begin{aligned}
&\text{Minimize}\ \ f(X),\ \ X=[x_1,x_2,\ldots,x_D]\ \ \text{s.t.}\\
&g_i(X)\le 0\ \ i=1,2,\ldots,\mathrm{p}\\
&h_k(X)=0\ \ k=p+1,\ \ p+2,\ldots,\mathrm{m}\\
&L_j\le x_j\le U_j\ \ j=1,2,\ldots,\mathrm{D}
\end{aligned}\tag{1}$$

where $X=[x_1,x_2,\ldots,x_D]$ is a D-dimensional solution vector. $g_i(X)$ is the ith inequality constraint, and $h_k(X)$ is the kth equality constraint. L_j and U_j are the lower and upper bound on the jth decision variable, respectively. f, g_i's and h_k's are nonlinear.

Constrained nonlinear optimization problems are comparatively hard to solve as for such type of problems, the aim is not only to find the optimal solution, but also to keep the focus on the feasible solutions. There are various contributing factors for increasing the complexity of the constrained optimization problems. Some of them are the type of constraints, type of objective function(s), percentage of the feasible region in the search space, number of equality and inequality constraints, number of active constraints. Classical derivative-based optimization techniques

Published online: 02 November 2016

fail most of the times to solve such problems. Metaheuristics play a significant role in solving such optimization problems. Last few decades witness the development of many metaheuristics which are nature-inspired. Some of the nature-inspired algorithms are genetic algorithm (Holland 1975), particle swarm optimization (Kennedy and Eberhart 1995), differential evolution (Storn and Price 1997), artificial bee colony (Karaboga 2005), ant colony optimization (Dorigo 1992), bacterial foraging optimization (Passino 2002), biogeography-based optimization (Simon 2008), artificial immune algorithm (Farmer et al. 1986), teaching–learning-based algorithm (Rao et al. 2012), gravitational search algorithm (Rashedi et al. 2009), evolutionary membrane algorithm (Han et al. 2014). Vortex search algorithm (Doğan and Olmez 2015), optics inspired optimization (Kashan 2015), water wave optimization (Zheng 2015) are some of the latest additions in the list of metaheuristics. Vortex search algorithm is inspired from the vortex pattern formed by the vertical flow of stirred fluids. Optics inspired optimization algorithm finds its source of inspiration from the converging/diverging behavior of a concave/convex mirror. Water wave optimization algorithm simulates the different behaviors like propagation, refraction and breaking of water waves to develop an efficient algorithm for global optimization problems. Metaheuristics have been applied successfully in solving real-world optimization problems (Ali et al. 2014; Esmaeili and Dashtbayazi 2014; Imran and Kowsalya 2014; Kumar et al. 2014; Kuo et al. 2014). Exploration and exploitation are the two driving forces behind the execution of a metaheuristic. Exploration is the ability of a metaheuristic to explore different areas of the search space with an objective to find promising regions, and exploitation is the ability to refine the already found good solutions. Exploration and exploitation seem to be conflicting in nature. So, a proper balance between exploration and exploitation is needed for the success of a metaheuristic in solving optimization problems. Because a lack of balance between these two (exploration and exploitation) can affect the performance of a metaheuristic significantly. Excessive exploration leads to the production of new solutions, but it slows down the convergence, while, on the other hand, excessive exploitation increases the possibility of the search to be trapped in a local optimum. Modifications have been reported by various researchers in the existing algorithms (Gao et al. 2014; Jia 2013; Khajehzadeh 2012) to balance the exploration and exploitation with an objective to improve the performance of these algorithms.

Search algorithm and constraint handling technique both play an important role in solving constrained optimization problems. Since nature-inspired algorithms are formulated in a way in their ingenious design that they are suitable only for unconstrained optimization, various constraint handling techniques have been developed over the decades to be incorporated in these algorithms to remove this deficiency. Though the method for handling constraints is different in all the constraint handling techniques, yet the objective of mostly all of these techniques is same. The objective is to prefer feasible solutions over infeasible solutions. This objective is the key force to drive the search toward feasible region of the search space. Each constrained handling technique has its own advantages and disadvantages. So, which constrained handling technique is the most suitable for a particular nature-inspired algorithm is still an open research question. The paper (Coello 2002) presents a comprehensive though not exhaustive survey of different constraint handling techniques which have been developed to incorporate constraint handling mechanism in nature-inspired algorithms. Constraint handling techniques have been divided into five categories: (1) penalty approach, (2) special representations and operators, (3) repair algorithms, (4) separation of objective function and constraints and (5) hybrid methods. Each technique has been discussed in detail along with its usage in different metaheuristics. There is a vast literature available on constraint handling techniques in which several techniques have been proposed and developed, but there are only few techniques which have been proved competitive in solving constrained optimization problems. Among them are penalty approach, stochastic ranking, Deb's feasibility rules. Since its inception, Deb's technique is one of the most widely used constraint handling technique because of its ease of implementation and parameter-free approach. Though penalty approach is still popular in handling constraints, requirement of additional penalty parameter sometimes makes it difficult to implement. GA is the oldest, yet one of the most popular choices among researchers these days too. Mostly used constraint handling mechanism with GA is the penalty approach. Most popular constraint handling technique with DE is Deb's feasibility rules (Deb 2000). Also, with swarm intelligent techniques such as PSO and ABC, Deb's technique for constrained optimization is the most popular one.

In this paper, spider monkey optimization (SMO) algorithm which is inspired from the food searching behavior of spider monkeys has been incorporated with Deb's constraint handling technique in order to deal with constrained optimization problems and the performance of proposed algorithm has been investigated over constrained benchmark problems. Potential of SMO for solving unconstrained optimization problems has already been explored in Bansal et al. (2014), Gupta and Deep (2016a, b), Gupta et al. (2016), Sharma (2016).

Before proceeding further, it is important to understand the usefulness of working on this new metaheuristic SMO when there are already well-established metaheuristics for solving unconstrained as well as constrained optimization problems. The reasons can be the increasing complexity of modern-world optimization problems which demand for the development of efficient search techniques for solving them at low computational cost. Also, no free lunch theorem (Wolpert and Macready 1997) makes room for the development of new algorithms by stating that there is no best algorithm for all the optimization problems. So, a new algorithm showing competitive performance in comparison with other state-of-the-art algorithms on most of the optimization problems in the benchmark set deserves to be explored and developed. Moreover, the main question is why to use SMO particularly for optimization purpose? The advantage of SMO over other well-established metaheuristics such as PSO, ABC and DE is that it has the provision to handle problems such as stagnation or premature convergence in its original design. Such a mechanism is not present in the original designs of other metaheuristics such as PSO, ABC and DE. However, these algorithms have been improved enough to handle these problems very well. In simple words, we can say, advanced versions of PSO, ABC and DE are available, which handle the problem of stagnation or premature convergence very efficiently. But not every user who needs to solve an optimization problem has enough knowledge to deal with these algorithms. They do not even know about the occurrence of such problems (stagnation or premature convergence) during the execution of these algorithms. Even if they do know about these problems, they may not know which version to select in order to avoid the problem of stagnation or premature convergence. So, SMO is beneficial for such users who are not having much knowledge about metaheuristics and want to use basic version of a metaheuristic to solve an optimization problem using black box approach.

The rest of the paper is organized as follows: Sect. 2 gives introduction to spider monkey optimization technique. Section 3 provides the details of Deb's feasibility rules for constraint handling. In Sect. 4, the proposed algorithm is discussed in detail. In Sect. 5, experimental setup is provided. In Sect. 6, experimental results are summarized and discussed. Finally, concluding remarks and possible future directions are given in Sect. 7.

2 Spider monkey optimization

Spider monkey optimization is a recently developed nature-inspired stochastic optimization technique, which particularly finds its place in the category of swarm intelligent techniques. Spider monkey is a species of monkeys which belongs to the class of fission–fusion social structure (FFSS)-based animals (Symington 1990). These monkeys live in groups and exhibit intelligent foraging behavior. They search for their food in different directions by sharing information with other group members. Their intelligent food searching strategy is the source of motivation behind the development of this algorithm. Search space of the optimization problem under consideration can be seen as food searching area of spider monkeys. Each solution of the optimization problem is represented by the position of spider monkey in this food searching area. Collection of all the solutions is called the swarm. Optimal point is the food source. Fitness of a solution indicates how near is the spider monkey to the food source. Inspired from the food foraging behavior of spider monkeys, this technique follows the rules of information sharing and continuous learning for position update among the whole swarm like other swarm intelligent techniques. In SMO, each spider monkey strives to update its current position with a better one by using its own experience as well as experience of other members of the swarm. Detailed mathematical formulation of this technique can be found in Bansal et al. (2014). SMO has four user-defined parameters, namely perturbation rate, maximum number of groups, global leader limit and local leader limit. Like all other metaheuristics, this algorithm starts with uniformly generated random initial positions of all the spider monkeys in the swarm. These positions get updated with iterations. Best solution of the whole swarm is called the global leader. If the global leader is not improving for specified number of iterations, the whole swarm gets divided into groups. The best solution in each group is known as the local leader of that group. Initially, there is only one group. So, initially local leader is same as the global leader. In addition to initialization, SMO has six iterative phases which contribute to improve the position of spider monkeys. These phases are Local Leader Phase, Global Leader Phase, Global Leader Learning Phase, Local Leader Learning Phase, Local Leader Decision Phase and Global Leader Decision Phase. Each phase has a different purpose to serve in the execution of the entire SMO algorithm. Local Leader Phase and Global Leader Phase are responsible for generating a new trial position for each spider monkey. If this newly generated position is better than the current position, the spider monkey updates its position with the new one; otherwise, it retains its current position. Global Leader Learning Phase and Local Leader Learning Phase are there for selecting global leader and local leaders, respectively. Local Leader Decision Phase and Global Leader Decision Phase are used to check and handle stagnation and premature convergence in local groups and the whole swarm, respectively. Notations used in the algorithms are provided in Table 1.

A brief description of each phase is given below:

2.1 Local leader phase

This phase is responsible for updating the current swarm. For this purpose, a new trial position is created for each spider monkey with the help of its current position, posi-

Table 1 Notations

$fitness_i$	Fitness of the position of ith spider monkey
f_{worst}	Objective function value of the worst feasible solution of the swarm
SM_i	Position of ith spider monkey
$f(SM_i)$	Objective function value of ith spider monkey
gl_j	jth Co-ordinate of the position of current global leader
ll_{kj}	jth Co-ordinate of the position of local leader of kth group
$probability_i$	Probability of selection of ith member in the swarm (here member means spider monkey)
SM_{new}	A trial vector for creating a new position of a spider monkey
sm_{ij}	jth Co-ordinate of the position of ith spider monkey
sm_{maxj}	Upper bound on the jth co-ordinate
sm_{minj}	Lower bound on the jth co-ordinate
sm_{rj}	jth Co-ordinate of the position of a randomly selected say rth member of a group
$viol_i$	Sum of constraint violations of ith spider monkey
eps	Tolerance limit to convert an equality constraint into an inequality constraint
D	Number of decision variables
Global_Count	Counter for recording the number of iterations since the global leader of the swarm has last updated its position
Global_Leader_Limit	Global leader limit
Local_Count [k]	Counter for recording the number of iterations since the local leader of the kth group has last updated its position
Local_Leader_Limit	Local leader limit
MaxFitness	Maximum fitness in the swarm
MaxGroups	Maximum number of groups allowed in the swarm
member [k][0]	Index of first member of the kth group
member [k][1]	Index of last member of the kth group
members [k]	Number of spider monkeys in the kth group
numgroups	Number of groups in the current swarm
pr	Perturbation rate
R (p,q)	Uniform random number between p and q
ss	Number of spider monkeys in the swarm (swarm size)
m	Total number of inequality and equality constraints

tion of the local leader and a randomly selected member of that group. Every dimension of the solution gets a chance to be updated based on the value of the perturbation rate. Equation for generating trial position is given below:

$$sm_{new\ j} = \begin{cases} sm_{ij} + R(0,1) \times (ll_k - sm_{ij}) & \\ \quad +R(-1,1) \times (sm_{ij} - sm_{ij}) & \text{if } R(0,1) \geq pr \\ sm_{ij} & \text{otherwise} \end{cases} \tag{2}$$

Here $R(0, 1)$ is generated for each dimension. If this newly generated position of ith spider monkey has better fitness value than its current position, then this new position is adopted; otherwise, current position is retained.

2.2 Global leader phase

Like Local Leader Phase, updation of the swarm takes place in this phase also, but in a different manner. Unlike Local Leader Phase, here only one randomly selected dimension of the solution gets updated. Which spider monkey will get a chance to be updated depends on its probability. Formula for calculating probability of ith spider monkey is given below:

$$probability_i = 0.9 * \left(\frac{fitness_i}{MaxFitness}\right) + 0.1 \tag{3}$$

Trial position is generated by the following equation:

$$\begin{aligned} sm_{newj} = {} & sm_{ij} + R\,(0,1) \times \left(gl_j - sm_{ij}\right) \\ & +R\,(-1,1) \times \left(sm_{rj} - sm_{ij}\right) \end{aligned} \tag{4}$$

Again comparison between the between fitness value of the newly generated position and the current position will be made and the better one will be adopted.

2.3 Global leader learning phase

In this phase, a spider monkey whose position is having best fitness value among all the members of the swarm will be updated as global leader of the swarm. If the position of the global leader does not get updated, then global limit count is incremented by 1. Global limit count keeps track of the number of iterations of no updating in the position of global leader.

2.4 Local leader learning phase

Like global leader, local leaders are selected for each group by applying greedy selection in that group. If the position of the local leader of a particular group does not get updated, its limit count will be incremented by 1.

2.5 Local leader decision phase

This phase checks whether there is stagnation or premature convergence in any group with the help of local limit count and reinitialize the group if stagnation is there.

```
Begin
    Set value of control parameters
    Random initialization of the swarm using uniform distribution.
    Calculate the fitness value of each member of the swarm
    Select global leader and local leaders by greedy selection based on fitness value
    Set iteration=0
    While (termination criterion is not fulfilled) do
        //Local Leader Phase
        //Calculate Probability of each member of the swarm
        //Global Leader Phase
        //Global Leader Learning Phase
        //Local Leader Learning Phase
        //Local Leader Decision Phase
        //Global Leader Decision Phase
        Iteration=iteration+1
    End while
End
```

Algorithm 1. Pseudocode for SMO

2.6 Global leader decision phase

This phase checks for stagnation or premature convergence in the whole swarm. The whole swarm is then divided into groups in case of stagnation or premature convergence.

Pseudocode for SMO is provided in Algorithm 1.

3 Deb's technique for handling constraints

Deb's technique (2000) is one of the most widely used techniques for handling constraints in the area of metaheuristics. It is based on the following rules, popularly known as *three feasibility rules*:

- A feasible solution is always preferred over an infeasible solution.
- Between two feasible solutions, the one having higher fitness value is preferred.
- Between two infeasible solutions, the one having less constraint violation is preferred.

Here, the constraint violation $viol_i$ of any solution X for the constrained optimization problem (1) is calculated as follows:

$$viol_i = \sum_{i=1}^{p} G_i(X) + \sum_{j=p+1}^{m} H_j(X) \tag{5}$$

where

$$G_i(X) = \begin{cases} g_i(X) & \text{if } g_i(X) > 0 \\ 0 & \text{if } g_i(X) \le 0 \end{cases}$$

$$H_j(X) = \begin{cases} \left|h_j(X)\right| & \text{if } \left|h_j(X)\right| - \text{ eps } > 0 \\ 0 & \text{if } \left|h_j(X)\right| - \text{ eps } \le 0 \end{cases}$$

It is a well-known fact that constraints handling technique adopted to handle constraints in an optimization problem plays a significant role in the performance of a metaheuristic technique. It highly influences the performance of the metaheuristic technique in which it is used. In spite of having so many constraint handling techniques existing in the literature, there are some reasons for using Deb's technique as a mean for handling constraints in SMO. This technique is easy to understand and implement. Since its inception, it is the most widely used constraint handling technique (Coello 2002). It has been used with various metaheuristics and has shown its potential over other constraint handling techniques. It does not require any additional parameters other than the parameters used for original algorithm. Also, it retains the original characteristics of the search technique.

Feasibility rules have been used with various nature-inspired optimization techniques for solving constrained optimization problems. Mezura-Montes et al. (2004, 2005, 2006) used the feasibility rules for the selection between target vector and trial vector. Although the proposed approach was easy to implement, it faced the problem of premature convergence in some test problems. This required two modifications in the DE algorithm: (1) for each target vector, there will be more than one trial vector; (2) a new DE variant was designed. Zielinski and Laur (2006) used feasibility rules with a local best PSO. The approach caused the problem of premature convergence in some test problems having high number of equality constraints due to the lack of diversity-preserving mechanism. Sun et al. (2009a, b) used the same approach with global best PSO for mixed variable optimization problems. He and Wang (2007) used feasibility rules with PSO for updating global best and personal best of each particle. Simulated annealing was used as a local search operator and applied to global best solution. Though the usage of SA improved the performance of PSO, it increased the number of user-defined parameters of both PSO and SA. In Muñoz-Zavala et al. (2006, 2005), Toscano-Pulido and Coello (2004), mutation operators were used with PSO to avoid the problem of converging to a local optimum. In Karaboga and Akay (2011), feasibility rules have been used with ABC for solving constrained optimization problems. The probability was defined based on the fitness value of feasible solutions as well as infeasible solutions. In Mezura-Montes and Hernández-Ocaña (2009), feasibility rules have been used as a constraint handling mechanism with bacterial foraging optimization technique for solving constrained optimization problems. In Elsayed et al. (2011), feasibility rules have been used with a modified GA for solving constrained optimization problems.

4 Proposed version of SMO for constrained optimization

This paper aims to design a constrained version of SMO. The proposed CSMO differs from original SMO only in two ways: calculation of fitness value and selection of the solution while comparing two solutions. In original SMO, fitness of a solution is based on its objective function value while in CSMO; fitness of a solution is based on its feasibility. Formula for calculating fitness value in CSMO is given in Algorithm 4. In original SMO, the comparison of two solutions is made on the basis of their objective function value. In CSMO, the comparison of two solutions is made on the basis of three feasibility rules mentioned in Sect. 3.

Execution steps of the proposed algorithm are given below:

4.1 Initialization

This is the first step in an optimization process. CSMO does not make any assumption about the feasibility of the initial swarm as initialization of the swarm with feasible solutions may require high computational cost depending on the size of the feasible region. Also, it is nearly impossible to generate initial feasible population in some cases where the ratio of feasible region to its search space is very small. So, in CSMO, initial swarm is randomly generated between lower and upper bounds of the decision variables using uniform distribution accepting both feasible and infeasible solutions. Initialization steps are provided in Algorithm 2.

```
Begin
    For i = 1 to ss do
        For j = 1 to D do
            sm_ij = sm_minj + R(0,1) × (sm_maxj − sm_minj)
        End For
    End For
End
```

$$sm_{ij} = sm_{minj} + R(0,1) \times (sm_{maxj} - sm_{minj})$$

Algorithm 2. Steps for initialization of the swarm

4.2 Local leader phase

Local Leader Phase in CSMO is same as in SMO except the selection of solution for comparison. In SMO, the fitness value of the newly created position of a spider monkey is compared with the fitness value of the current position. If the newly generated position is having better fitness value than its current position, then the spider monkey updates its position; otherwise, it retains its current position. In CSMO, position updating of spider monkeys is based on Deb's three rules of feasibility. New position and current position of a spider monkey are compared according to these rules, and then updating or retaining of the current position of the spider monkey is performed. Working steps of Local Leader Phase is provided in Algorithm 3.

4.3 Global leader phase

In Global Leader Phase, a solution gets a chance to be updated based on its probability. Probability of a solution is based on its fitness value. In CSMO, fitness of every solution is calculated in a manner such that feasible solution would always get the preference over the infeasible solutions. Fitness and probability calculation procedure is provided in Algorithm 4. Execution steps of Global Leader Phase are provided in Algorithm 5.

4.4 Global leader learning phase

In this phase, a solution having best fitness value in the swarm is found. Then, it is compared with the current global leader of the swarm using three feasibility rules. The current global leader gets updated by the best member of the swarm if the best member of the swarm is better; otherwise, global leader retains its current position. Procedure of selecting global leader is provided in Algorithm 6.

4.5 Local leader learning phase

In this phase, local leader selection is performed in every group. In each group, solution with best fitness value is found, and then, it is compared with the local leader of that group based on three feasibility rules. Local leader of the group gets updated by the best member of that group if it is better; otherwise, the local leader retains its current position. Selection procedure is provided in Algorithm 7.

4.6 Local leader decision phase

This phase helps in the re-initialization of a group if its local leader is not updating its position for the specified local leader limit. Re-initialization of the group may cause the member of the group to enter into infeasible region from a feasible region. But it is also necessary sometimes if the feasible regions of the search space are disjoint and the optimum lies in the other feasible region. This phase maintains the exploration capability of the algorithm. Working steps of this phase are provided in Algorithm 8.

4.7 Global leader decision phase

In this phase, the whole swarm is divided into groups if the global leader is not updated for the specified global leader limit. Execution steps of this phase are provided in Algorithm 9.

Begin
    For k = 1 to *numgroups* **do**
        For i = *member*[k][0] to *member*[k][1] **do**
            For j = 1 to D **do**
                Generate sm_{newj} using eq. (2)
            End For
            Apply the selection process between SM_{new} and SM_i based on Deb's *Three Feasibility Rules*
        End For
    End For
End

Algorithm 3. Steps for executing *Local Leader Phase*

Begin
    For i = 1 to ss **do**
        If$(viol_i = 0)$ **then**
            $fitness_i = f(SM_i)$
        Else
            $$fitness_i = f_{worst} + \sum_{j=1}^{m} viol_j$$
        End If
    End For
    For i = 1 to ss **do**
        $$probability_i = 0.9 \times \left(\frac{fitness_i}{MaxFitness}\right) + 0.1$$
    End For
End

Algorithm 4. Steps for calculating fitness and probability of solutions

```
Begin
    For k = 1 to numgroups do
        GS = k^th group size
        t = 0, i = 1
        While (t<ss) do
            For i = 1 to GS do
                If (R(0,1)< probability_i) then
                    t = t+1
                    Randomly select j from {1,2,..., D}
                    Randomly select SM_r from k^th group
                    Generate sm_newj using eq.(4)
                End If
                Apply the selection process between SM_new and SM_i based on Deb's Three
                Feasibility Rules
            End For
            i = i+1
            If (i = ss) then
                i = 1
            End If
        End While
    End For
End
```

Algorithm 5. Steps for executing *Global Leader Phase*

```
Begin
    //update position of the global leader of the swarm by applying three feasibility rules
    If (position of global leader is updated from previous position) then
        Global_Count = 0
    Else
        Global_Count= Global_Count+1
    End If
End
```

Algorithm 6. Steps for executing *Global Leader Learning Phase*

```
Begin
    For k =1 to numgroups do
        //update position of the leader of the group using three feasibility rules
        If (position of local leader is updated from previous position) then
            Local_Count[k] = 0
        Else
            Local_Count[k] = Local_Count[k] + 1
        End If
    End For
End
```

Algorithm 7. Steps for executing *Local Leader Learning Phase*

```
Begin
    For k = 1 to numgroups do
        If (Local_Count[k]> Local_Leader_Limit) then
            Local_Count[k] = 0
            For i = member[k][0] to member[k][1] do
                For j = 1 to D do
                    If (R(0, 1) ≥ pr) then
                        sm_ij = sm_minj + R(0,1) × (sm_maxj − sm_minj)
                    Else
                        sm_ij = sm_ij + R(0,1) × (gl_j − sm_ij) + R(0,1) ×
                                (sm_ij − ll_kj)
                    End If
                End For
            End For
        End If
    End For
End
```

Algorithm 8. Steps for executing *Local Leader Decision Phase*

```
Begin
    If (Global_Count > Global_Leader_Limit) then
        Global_Count = 0
        If (numgroups< MaxGroups) then
            numgroups = numgroups +1
        Else
            numgroups = 1
        End If
        Apply Local Leader Learning Phase
    End If
End
```

Algorithm 9. Steps for executing *Global Leader Decision Phase*

5 Experimental setup

Due to non-availability of analytical methods to predict the performance, metaheuristics are subjected to some empirical investigation. Analysis of experimental results helps to understand which optimization technique is preferable to others on a given set of problems. This can be considered as a first step in the validation and application of a metaheuristic. Before applying metaheuristics to real-life problems, their performance is analyzed on a set of benchmark functions via some experimentation. These benchmark problems may contain characteristics of the real-life problems which someone is interested to solve. Though there are numerous benchmark functions available in the literature, standard benchmarks such as CEC (http://www.ntu.edu.sg/home/epnsugan), BBOB (http://coco.gforge.inria.fr/doku.php?id=program) should be used to avoid any kind of bias regarding the selection of benchmark functions. In this paper, CEC2006 (Liang 2006) and CEC2010 (Mallipeddi and Suganthan 2010) benchmark sets have been considered for this purpose. Both the sets contain single objective constrained optimization problems. These two sets have also been used for experimentation in Asafuddoula et al. (2015), Parouha and Das (2015).

5.1 State-of-the-art algorithms used for comparison of results

The results of CSMO have been compared with ABC (Karaboga and Akay 2011), CHDE (Mezura-Montes et al. 2004) and PESO (Muñoz-Zavala et al. 2005). All the algorithms have been implemented in C. In this paper, global best topology has been used in place of ring topology used in PESO (Muñoz-Zavala et al. 2005). ABC, DE and PSO are some of the most widely used metaheuristics for solving constrained optimization problems, and SMO has some features similar to these algorithms. These factors led to the selection of constrained versions of these algorithms (ABC, DE and PSO) for comparison with constrained version of SMO.

Since the performance of a metaheuristic is highly sensitive to the constraint handling technique used, it will be more appropriate to compare the performance of SMO with those versions of these algorithms which have used Deb's technique for handling constraints. The constraint handling technique used in the compared algorithms is also Deb's technique. Using the same constraints handling technique helps to maintain the consistency in the comparison of algorithms and to access the potential of a metaheuristic.

5.2 Setting of control parameters

Search efficiency of an algorithm is highly sensitive to the choice of its control parameters. The set of parameters which produces optimal solution for a particular problem may result in a complete failure for the other problem. So optimal parameter setting is also an important issue while considering the performance of different algorithms. Parameter setting also requires analysis to be performed in order to find optimal parameter setting. In this paper, such an analysis has been avoided and none of the algorithm has been meta-optimized to improve its performance on a particular benchmark function. Parameter setting for every algorithm has been adopted as it is mentioned in their original research articles, and it is provided in Table 2. Only population size or swarm size is kept same for all the algorithms for a fair comparison. In this paper, our aim is not to the find best algorithm for a particular benchmark function, but to get an idea of search potential of SMO for solving constrained optimization problems.

5.3 Test functions and evaluation criterion

CEC2006 and CEC2010 problems have been considered for testing the performance of proposed constrained version of SMO. Both the benchmark sets contain constrained optimization problems of different types. In CEC2006 benchmark set, there are 24 problems, g01–g24. The number of design variables is different for each optimization problem. Objective function is categorized as linear, nonlinear, quadratic, cubic or polynomial in this benchmark set. The mathemati-

Table 2 Parameter setting for CSMO, ABC, CHDE, PESO

Algorithm	Parameter setting
CSMO	Perturbation rate (pr) = linearly increasing ([0.1, 0.4])
	Maximum number of groups (MG) = 5
	Local leader limit = 1500
	Global leader limit = 50
ABC	Modification rate (MR) = 0.8
	Maximum cycle number (MCN) = (maximum number of function evaluations)/100
	Limit = 0.5 × ss × D, where ss is the swarm size and D is the dimension of the problem
	SPP = 0.5 × ss × D
CHDE	F = generated randomly between [0.3, 0.9] per run using a uniform distribution
	CR = generated randomly between [0.8, 1.0] per run using a uniform distribution
PESO	c1 = 0.1
	c2 = 1
	Inertia weight (w) = generated randomly between [0.5,1] using uniform distribution

Table 3 Classification of problems in CEC2006 benchmark set on the basis of the type of constraints

Type of constraints	Problems
Only equalities	g03, g11, g13, g14, g15, g17
Only inequalities	g01, g02, g04, g06, g07, g08, g09, g10, g12, g16, g18, g19, g24
Both equalities and inequalities	g05, g20, g21, g22, g23

cal definition of each problem can be found in Liang (2006). Classification of problems on the basis of type of constraints is provided in Table 3. For each problem in the benchmark set, each algorithm has 25 independent runs performed for each problem. The stopping criterion is when 500,000 number of function evaluations have been performed. For comparison of results, the feasibility rate and success rate are recorded. The formulas for calculating them are given below:

$$\text{feasibility rate} = \frac{\text{number of feasible runs}}{\text{total number of runs}} \times 100$$

A run is said to be a 'feasible run' if at least one feasible solution is found.

$$\text{success rate} = \frac{\text{number of successful runs}}{\text{total number of runs}} \times 100$$

Table 4 Classification of problems in CEC2010 benchmark set on the basis of the type of constraints

Type of constraints	Problems
Only equalities	C03, C04, C05, C06, C09, C10, C11,
Only inequalities	C01, C07, C08, C13, C14, C15,
Both equalities and inequalities	C02, C12, C16, C17, C18

A run is declared 'successful run' if $|f(X) - f(X^*)| \leq 0.0001$, where $f(X)$ is the best value obtained by the algorithm and $f(X^*)$ is the known global optimal value.

Some statistical measures have been used for the comparison of results as suggested by Liang (2006). For this purpose, best, median, worst, average and standard deviation of the function error values $|f(X) - f(X^*)|$ for the achieved best solution after 5,00,000 function evaluations are recorded. The method for sorting error values of achieved best solutions of all the runs is given below:

- Feasible solutions are sorted in front of infeasible solutions.
- Feasible solutions are sorted in an increasing order according to their function error value $|f(X) - f(X^*)|$
- Infeasible solutions are sorted in an increasing order according to their mean value of the violations of all constraints.

Mean violation at $X = \frac{\left(\sum_{i=1}^{p} G_i(X) + \sum_{j=p+1}^{m} H_j(X)\right)}{m}$
where

$$G_i(X) = \begin{cases} g_i(X) & \text{if } g_i(X) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$H_j(X) = \begin{cases} \left|h_j(X)\right| & \text{if } \left|h_j(X)\right| - \text{eps} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Also, the best, median, worst, average and standard deviation of the number of function evaluations for the successful runs have been recorded.

There are 18 problems, C01–C18, in CEC2010 benchmark set. The mathematical definition of each problem in this benchmark set can be found in Mallipeddi and Suganthan (2010). Table 4 provides a classification of problems on the basis of type of constraints. Unlike CEC2006, in CEC2010, all the problems are of same dimension. In CEC2010, results have been considered for 10 dimensions and 30 dimensions. Objective function is categorized as separable or non-separable. Optimal value of the test problems has not been provided. For comparison of results, feasibility rate and best, median, worst, average and standard deviation of obtained global optima obtained after fixed number of function evaluations are recorded. Fixed numbers of function evaluations for 10 dimensions and 30 dimensions are 2, 00,000 and 6, 00,000, respectively (Mallipeddi and Suganthan 2010). A total of 25 independent runs have been conducted for each problem. Following criterion has been adopted to sort the objective function values of achieved best solutions obtained in 25 runs:

- Feasible solutions are sorted in front of infeasible solutions.
- Feasible solutions are sorted in an increasing order according to their objective function value.
- Infeasible solutions are sorted in an increasing order according to their mean value of the violations of all constraints.

Definition of feasibility rate and mean value of constraints violations is same as that for CEC2006 problems.

6 Discussion of experimental results

The results of algorithms on both the benchmark sets (CEC2006 and CEC2010) have been discussed separately. First, the performance of algorithms has been analyzed on CEC2006 problems and then on CEC2010 optimization problems for both 10 and 30 dimensions. Results have been presented in the form of tables and convergence graphs. In the convergence graphs, value on the horizontal axis represents the number of iterations and the vertical axis shows the function error value and objective function value in case of CEC2006 and CEC2010, respectively. In order to observe whether the results are significantly different or not, Wilcoxon rank-sum test at 5% ($\alpha = 0.05$) significance level is performed between CSMO-ABC, CSMO-CHDE and CSMO-PESO for both CEC2006 and CEC2010 problems. If there is no significant difference between the results, '=' sign appears in the table, and when there is significant difference between the results, '+' or '-' sign appears based on CSMO is performing better or worse than the other algorithm. Tables 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 and 32, 33, 34, 35, 36, 37, 38, 39, 40, 41 present the results for CEC2006 benchmark problems, CEC2010 benchmark problems for 10 dimensions and CEC2010 benchmark problems for 30 dimensions, respectively. The entries in the cell of the tables contain both feasible and infeasible solutions. Infeasible solutions are the entries with a number in parentheses, which indicates the number of

Table 5 Feasibility rate (F.R.) and best, median, worst, mean and standard deviation (SD) of the function error values obtained by CSMO with 25 independent runs on CEC2006 benchmark problems

Problems	F.R.	Best	Median	Worst	Mean	SD
g01	100	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
g02	100	2.55E−05	1.12E−02	4.96E−02	1.54E−02	1.52E−02
g03	100	2.39E−01	4.88E−01	8.86E−01	4.91E−01	1.45E−01
g04	100	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
g05	80	4.79E−02	5.49E+02	5.30E−03(1)	3.68E+02	3.92E+02
g06	100	2.79E−08	6.93E−07	2.40E−04	1.43E−05	4.80E−05
g07	100	3.05E−02	2.81E−01	7.18E−01	3.01E−01	2.09E−01
g08	100	4.16E−17	4.16E−17	4.16E−17	4.16E−17	6.29E−33
g09	100	6.75E−04	5.10E−03	1.07E−02	5.27E−03	2.99E−03
g10	100	1.70E+01	2.06E+02	9.91E+02	2.69E+02	2.20E+02
g11	100	5.99E−06	1.55E−02	2.50E−01	8.49E−02	1.03E−01
g12	100	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
g13	88	4.91E−01	9.09E−01	2.14E−04(1)	8.58E−01	1.46E−01
g14	100	5.10E−01	4.09E+00	6.09E+00	3.90E+00	1.43E+00
g15	100	1.66E−03	6.46E−01	1.06E+01	2.98E+00	3.65E+00
g16	100	6.49E−12	6.68E−10	1.49E−07	9.79E−09	3.13E−08
g17	96	5.49E+01	1.59E+02	8.49E−05(1)	2.21E+02	1.39E+02
g18	100	4.34E−05	1.94E−04	2.42E−03	3.47E−04	4.74E−04
g19	100	1.30E+00	5.95E+00	1.49E+01	6.33E+00	3.77E+00
g20	0	3.72E−03(6)	1.84E−02(11)	3.04E−02(15)	N.A.	N.A.
g21	0	1.51E−03(2)	5.44E−03(3)	1.71E−02(3)	N.A.	N.A.
g22	0	2.31E−01(16)	5.06E+00(10)	4.26E+0419)	N.A.	N.A.
g23	16	4.00E+02	7.36E−04(4)	8.32E−03(4)	4.00E+02	0.00E+00
g24	100	1.24E−14	1.24E−14	1.24E−14	1.24E−14	3.22E−30

Table 6 Success rate (S.R.) and best, median, worst, mean and standard deviation (SD) of the number of function evaluations obtained by CSMO with successful runs out of 25 independent runs on CEC2006 benchmark problems

Problems	S.R.	Best	Median	Worst	Mean	SD
g01	100	8950	10,750	28,150	13,714	6026
g02	8	217,736	255,548	293,361	255,548	53,474
g03	0	N.A.	N.A.	N.A.	N.A.	N.A.
g04	100	17,550	21,850	30,150	22,690	2973
g05	0	N.A.	N.A.	N.A.	N.A.	N.A.
g06	96	165,267	201,324	326,597	216,177	42,756
g07	0	N.A.	N.A.	N.A.	N.A.	N.A.
g08	100	650	950	1250	934	146
g09	0	N.A.	N.A.	N.A.	N.A.	N.A.
g10	0	N.A.	N.A.	N.A.	N.A.	N.A.
g11	24	274,204	359,310	476,918	374,996	80,233
g12	100	350	1050	1650	1026	274
g13	0	N.A.	N.A.	N.A.	N.A.	N.A.
g14	0	N.A.	N.A.	N.A.	N.A.	N.A.
g15	0	N.A.	N.A.	N.A.	N.A.	N.A.
g16	100	12,850	26,050	80,724	34,930	20,422
g17	0	N.A.	N.A.	N.A.	N.A.	N.A.
g18	20	219,036	372,280	476,428	348,277	109,348
g19	0	N.A.	N.A.	N.A.	N.A.	N.A.
g20	0	N.A.	N.A.	N.A.	N.A.	N.A.
g21	0	N.A.	N.A.	N.A.	N.A.	N.A.
g22	0	N.A.	N.A.	N.A.	N.A.	N.A.
g23	0	N.A.	N.A.	N.A.	N.A.	N.A.
g24	100	2950	4250	5850	4258	796

Table 7 Comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate (F.R.) and success rate (S.R.) on CEC2006 benchmark problems

Problems	Feasibility rate				Success rate			
	CSMO	ABC	CHDE	PESO	CSMO	ABC	CHDE	PESO
g01	**100**	**100**	**100**	**100**	**100**	**100**	56	88
g02	**100**	**100**	**100**	**100**	8	**52**	12	0
g03	**100**	**100**	**100**	**100**	0	0	0	0
g04	**100**	**100**	**100**	**100**	**100**	**100**	92	**100**
g05	**80**	0	0	72	0	0	0	0
g06	**100**	**100**	92	**100**	96	**100**	80	**100**
g07	**100**	4	12	**100**	0	0	4	0
g08	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**
g09	**100**	40	40	**100**	0	0	**36**	0
g10	**100**	0	0	**100**	0	0	0	0
g11	**100**	0	0	**100**	24	0	0	**88**
g12	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**
g13	88	0	0	**100**	0	0	0	0
g14	**100**	0	96	80	0	0	**40**	0
g15	**100**	0	0	**100**	0	0	0	0
g16	**100**	**100**	**100**	**100**	**100**	**100**	76	28
g17	96	0	0	**100**	0	0	0	0
g18	**100**	**100**	96	**100**	20	12	**88**	12
g19	**100**	**100**	**100**	**100**	0	0	**48**	0

Table 7 continued

Problems	Feasibility rate				Success rate			
	CSMO	ABC	CHDE	PESO	CSMO	ABC	CHDE	PESO
g20	0	0	0	**72**	0	0	0	**36**
g21	0	0	0	0	0	0	0	0
g22	0	0	0	0	0	0	0	0
g23	16	28	**56**	24	0	0	**16**	0
g24	**100**	**100**	**100**	**100**	**100**	**100**	96	**100**

Best values are in bold

Table 8 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate (F.R.) and success rate (S.R.) on CEC2006 benchmark problems

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Feasibility rate (F.R.)	9	14	1
	Success rate (S.R.)	2	19	3
	Total	11	33	4
CHDE	Feasibility rate (F.R.)	11	12	1
	Success rate (S.R.)	6	11	7
	Total	17	23	8
PESO	Feasibility rate (F.R.)	2	18	4
	Success rate (S.R.)	4	17	3
	Total	6	35	7

constraints violated by that infeasible solution. N.A. entry in a cell indicates non-availability of results.

6.1 Discussion of results for CEC2006 benchmark problems

Tables 5 and 6 present the results obtained by CSMO on CEC2006 benchmark problems. From Table 5, it can be seen that CSMO has 100 percent feasibility rate in 17 problems and zero percent feasibility rate in three (g20, g21 and g22) problems. In the problems having inequality constraints only, CSMO has obtained 100 percent feasibility rate. Also, CSMO has good feasibility rate on the problems with equality constraints only. The problems where CSMO has failed completely to enter the feasible region in any run (g20, g21, g22) are the problems having both equality and inequality constraints. From Table 6, it can be seen that CSMO has 100 percent success rate in six problems. Though CSMO successfully enters the feasible region in the problems having equality constraints, it fails to obtain global optima in any of these problems (zero success rate) except the problem g11. Among the problems in which CSMO has positive success rate are the ones having inequality constraints only except problem g11. Table 7 presents the feasibility rate and success rate of all the algorithms. It can be seen that all the algorithms have 100 percent feasibility rate on g1, g2, g3, g4, g8, g12, g16, g19 and g24 and 100 percent success rate on g8 and g12. g21 and g22 can be considered difficult problems for these algorithms as all the algorithms fail to reach the feasibility region in all the runs. All the algorithms have zero percent success rate on g3, g5, g10, g13, g15, g17, g21 and g22. Summary of pairwise comparison of CSMO against each of ABC, CHDE and PESO in terms of feasibility rate and success rate is provided in Table 8. From this table, it can be seen that CSMO has better feasibility rate than ABC and CHDE, but it performs inferior to PESO. All the algorithms perform almost equal in terms of success rate. So, from the pairwise comparison of CSMO with all the algorithms in terms of feasibility rate and success rate, it can be concluded that though CSMO performs better than ABC and CHDE in locating the feasible region, it performs almost equal to these two in locating a near-optimal solution. But this is just an observation which needs further experimentation. Tables 9, 10 and 11 present the best, median and worst of function error values, respectively, obtained by all the algorithms in 25 runs. Tables 12, 13 provide the mean and standard deviation of function error values of feasible runs only. Summary of pairwise comparison of CSMO with ABC, CHDE and PESO on the basis of best, median, worst, mean and standard deviation values is provided in Table 14. From this table, it is clear that CSMO performs better than ABC, CHDE and PESO in terms of function error values.

Tables 15, 16, 17, 18 and 19 show the best, median, worst, mean and standard deviation, respectively, of function evaluations for successful runs for all the algorithms. Summary of pairwise comparison of CSMO with ABC, CHDE and PESO on the basis of number of function evaluations is provided in Table 20. From this table, it can be concluded that CSMO performs better than ABC. CSMO performs inferior and almost equal to CHDE and PESO, respectively.

Results of Wilcoxon rank-sum test for CEC 2006 problems based on mean function error value are provided in Table 21. It can be observed from the outcome of Wilcoxon rank-sum test that CSMO performs significantly better than other algorithms.

Convergence graphs of problems g02, g03 and g06 are shown in Figs. 1, 2 and 3, respectively.

Table 9 Comparison of CSMO against ABC, CHDE and PESO in terms of best of function error values obtained with 25 independent runs on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g02	2.55E−05	3.00E−05	**9.23E−08**	1.27E−02
g03	2.39E−01	5.14E−01	3.86E−01	**4.85E−02**
g04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g05	**4.79E−02**	2.02E−05(1)	2.01E−05(1)	1.06E−01
g06	2.79E−08	**1.18E−11**	**1.18E−11**	**1.18E−11**
g07	3.05E−02	8.81E−02	**1.81E−13**	7.66E−01
g08	**4.16E−17**	**4.16E−17**	**4.16E−17**	**4.16E−17**
g09	6.75E−04	2.72E−03	**0.00E+00**	1.94E−03
g10	**1.70E+01**	2.11E−05(1)	2.14E−04(1)	5.35E+01
g11	5.99E−06	3.38E−04(1)	1.37E−04(1)	**4.99E−09**
g12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g13	4.91E−01	3.34E−05(1)	3.35E−05(1)	**9.06E−02**
g14	5.10E−01	4.51E−05(1)	**2.13E−14**	4.16E+00
g15	1.66E−03	5.00E−05(1)	5.15E−05(1)	**1.29E−03**
g16	6.49E−12	**4.88E−15**	**4.88E−15**	**4.88E−15**
g17	**5.49E+01**	2.61E−05(1)	2.55E−05(1)	9.97E+01
g18	4.34E−05	3.61E−05	**3.33E−16**	7.22E−06
g19	1.30E+00	1.40E+00	**2.13E−14**	4.36E+00
g20	3.72E−03(6)	8.46E−03(7)	7.19E−03(5)	**3.20E−03**
g21	1.51E−03(2)	2.56E−04(2)	**1.71E−05(1)**	2.59E−04(2)
g22	**2.31E−01(16)**	3.54E+03(19)	3.98E+00(6)	1.99E+01(11)
g23	4.00E+02	4.00E+02	**0.00E+00**	3.26E+02
g24	**1.24E−14**	**1.24E−14**	**1.24E−14**	**1.24E−14**

Best values are in bold

Table 10 Comparison of CSMO against ABC, CHDE and PESO in terms of median of function error values obtained with 25 independent runs on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**0.00E+00**	**0.00E+00**	1.78E−15	1.44E−13
g02	1.12E−02	**1.02E−04**	2.58E−02	5.15E−02
g03	**4.88E−01**	8.15E−01	8.17E−01	5.56E−01
g04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g05	**5.49E+02**	2.41E−05(1)	2.89E−05(1)	9.86E+02
g06	6.93E−07	2.36E−11	**1.18E−11**	**1.18E−11**
g07	**2.81E−01**	2.96E−01(1)	3.84E−01(1)	3.57E+00
g08	**4.16E−17**	**4.16E−17**	**4.16E−17**	**4.16E−17**
g09	**5.10E−03**	1.64E+00(1)	2.51E+00(1)	8.34E−02
g10	**2.06E+02**	9.34E−03(1)	8.90E−03(1)	1.52E+03
g11	1.55E−02	1.35E−02(1)	6.09E−03(1)	**4.82E−06**
g12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g13	9.09E−01	4.26E−05(1)	4.76E−05(1)	**7.65E−01**
g14	4.09E+00	1.41E−03(3)	**1.46E−03**	1.03E+01
g15	**6.46E−01**	5.14E−05(1)	7.62E−05(1)	4.14E+00
g16	6.68E−10	**4.88E−15**	**4.88E−15**	1.64E−03
g17	**1.59E+02**	3.08E−03(3)	3.21E−05(1)	1.72E+02
g18	1.94E−04	7.50E−04	**1.11E−11**	2.47E−03
g19	5.95E+00	1.93E+00	**2.73E−05**	8.61E+00

Table 10 continued

Problems	CSMO	ABC	CHDE	PESO
g20	1.84E−02(11)	1.71E−02(15)	9.14E−03(6)	**3.60E−02**
g21	5.44E−03(3)	1.63E−03(2)	**3.25E−05(1)**	4.07E−02(1)
g22	**5.06E+00(10)**	6.71E+04(19)	6.50E+05(19)	2.04E+05(11)
g23	7.36E−04(4)	3.18E−03(1)	**4.00E+02**	1.11E−01(1)
g24	**1.24E−14**	**1.24E−14**	**1.24E−14**	**1.24E−14**

Best values are in bold

Table 11 Comparison of CSMO against ABC, CHDE and PESO in terms of worst of function error values obtained with 25 independent runs on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**0.00E+00**	**0.00E+00**	6.00E+00	3.00E+00
g02	4.96E−02	**1.10E−02**	3.44E−01	1.02E−01
g03	**8.86E−01**	9.24E−01	9.97E−01	1.00E+00
g04	**0.00E+00**	**0.00E+00**	1.43E+02	3.64E−12
g05	5.30E−03(1)	3.81E−03(1)	**1.40E−04(3)**	5.40E−03(1)
g06	2.40E−04	4.76E−10	1.10E+00(2)	**1.55E−11**
g07	**7.18E−01**	1.59E+00(2)	2.73E+00(4)	1.91E+01
g08	**4.16E−17**	**4.16E−17**	5.55E−17	5.55E−17
g09	**1.07E−02**	4.80E+01(2)	3.95E+01(2)	2.64E−01
g10	**9.91E+02**	1.04E−01(3)	1.09E−01(2)	2.52E+03
g11	**2.50E−01**	1.18E−01(1)	1.18E−01(1)	**2.50E−01**
g12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g13	2.14E−04(1)	7.36E−04(1)	1.30E−04(1)	**5.32E+00**
g14	**6.09E+00**	2.19E−03(3)	6.23E−02(1)	6.67E−05(1)
g15	**1.06E+01**	8.15E−05(1)	3.50E−04(2)	**1.06E+01**
g16	1.49E−07	**6.88E−15**	4.42E−01	3.34E−03
g17	8.49E−05(1)	2.73E−02(3)	1.58E−04(2)	**4.59E+02**
g18	**2.42E−03**	3.25E−03	2.05E+00(7)	2.14E−01
g19	1.49E+01	**2.70E+00**	4.15E+02	3.78E+01
g20	3.04E−02(15)	**1.98E−02(20)**	3.45E−02(7)	8.30E−02(1)
g21	1.71E−02(3)	2.20E−02(2)	**8.18E−03(3)**	1.22E−01(1)
g22	**4.26E+04(19)**	1.69E+05(19)	2.38E+06(19)	1.07E+06(13)
g23	**8.32E−03(4)**	7.40E−02(3)	1.48E+00(1)	5.83E−01(2)
g24	**1.24E−14**	**1.24E−14**	2.35E−02	**1.24E−14**

Best values are in bold

Table 12 Comparison of CSMO against ABC, CHDE and PESO in terms of mean of function error values obtained with feasible runs out of 25 independent runs on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**0.00E+00**	**0.00E+00**	1.34E+00	2.80E−01
g02	1.54E−02	**1.62E−03**	6.32E−02	5.13E−02
g03	**4.91E−01**	8.01E−01	7.69E−01	5.49E−01
g04	**0.00E+00**	**0.00E+00**	7.26E+00	4.37E−13
g05	**3.68E+02**	N.A.	N.A.	5.96E+02
g06	1.43E−05	6.45E−11	9.10E+00	**1.22E−11**
g07	3.01E−01	8.81E−02	**1.04E−03**	5.27E+00
g08	**4.16E−17**	**4.16E−17**	4.22E−17	4.72E−17
g09	**5.27E−03**	6.55E−03	1.52E+00	8.99E−02
g10	**2.69E+02**	N.A.	N.A.	1.42E+03
g11	8.49E−02	N.A.	N.A.	**1.00E−02**

Table 12 continued

Problems	CSMO	ABC	CHDE	PESO
g12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g13	8.58E−01	N.A.	N.A.	**8.51E−01**
g14	3.90E+00	N.A.	**8.09E−01**	1.01E+01
g15	**2.98E+00**	N.A.	N.A.	4.69E+00
g16	9.79E−09	**5.68E−15**	2.32E−02	1.52E−03
g17	**2.21E+02**	N.A.	N.A.	2.53E+02
g18	**3.47E−04**	8.56E−04	8.02E−03	1.50E−02
g19	6.33E+00	**1.92E+00**	2.34E+01	1.07E+01
g20	N.A.	N.A.	N.A.	**4.84E−02**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	4.00E+02	4.00E+02	**2.30E+02**	5.76E+02
g24	**1.24E−14**	**1.24E−14**	9.40E−04	**1.24E−14**

Best values are in bold

Table 13 Comparison of CSMO against ABC, CHDE and PESO in terms of standard deviation of function error values obtained with feasible runs out of 25 independent runs on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**0.00E+00**	**0.00E+00**	1.97E+00	7.92E−01
g02	1.52E−02	**3.04E−03**	8.82E−02	2.69E−02
g03	1.45E−01	**1.09E−01**	2.03E−01	3.40E−01
g04	**0.00E+00**	**0.00E+00**	2.93E+01	1.21E−12
g05	**3.92E+02**	N.A.	N.A.	4.39E+02
g06	4.80E−05	9.73E−11	3.92E+01	**1.23E−12**
g07	2.09E−01	N.A.	**1.70E−03**	4.84E+00
g08	**6.29E−33**	**6.29E−33**	2.78E−18	6.95E−18
g09	2.99E−03	**2.91E−03**	4.81E+00	7.08E−02
g10	**2.20E+02**	N.A.	N.A.	7.28E+02
g11	1.03E−01	N.A.	N.A.	**5.00E−02**
g12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
g13	**1.46E−01**	N.A.	N.A.	9.70E−01
g14	1.43E+00	N.A.	**1.32E+00**	3.52E+00
g15	**3.65E+00**	N.A.	N.A.	3.94E+00
g16	3.13E−08	**1.00E−15**	8.95E−02	1.36E−03
g17	1.39E+02	N.A.	N.A.	**1.20E+02**
g18	**4.74E−04**	7.57E−04	3.90E−02	4.25E−02
g19	3.77E+00	**3.03E−01**	8.63E+01	7.21E+00
g20	N.A.	N.A.	N.A.	**6.34E−02**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	**0.00E+00**	**0.00E+00**	1.82E+02	2.39E+02
g24	**3.22E−30**	**3.22E−30**	4.70E−03	**3.22E−30**

Best values are in bold

Table 14 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of best, median, worst, mean and standard deviation (SD) of function error values on CEC2006 benchmark problems

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Best	14	6	4
	Median	13	5	6
	Worst	13	5	6
	Mean	3	6	5
	SD	2	6	5
	Total	45	28	26
CHDE	Best	9	5	10
	Median	12	4	8
	Worst	20	1	3
	Mean	11	1	3
	SD	12	1	2
	Total	64	12	26
PESO	Best	9	5	10
	Median	16	4	4
	Worst	17	4	3
	Mean	16	2	3
	SD	16	2	3
	Total	74	17	23

6.2 Discussion of results for CEC2010 benchmark problems for 10 dimensions

Results obtained by CSMO on CEC2010 problems for 10 dimensions are presented in Table 22. From Table 22, it can

Table 15 Comparison of CSMO against ABC, CHDE and PESO in terms of best of number of function evaluations obtained with successful runs out of 25 independent on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**8950**	23,850	9150	17,300
g02	217,736	316,453	**154,950**	N.A.
g03	N.A.	N.A.	N.A.	N.A.
g04	17,550	65,050	**7900**	15,950
g05	N.A.	N.A.	N.A.	N.A.
g06	165,267	159,380	**5250**	29,600
g07	N.A.	N.A.	**70,100**	N.A.
g08	650	950	**250**	1700
g09	N.A.	N.A.	**17,200**	N.A.
g10	N.A.	N.A.	N.A.	N.A.
g11	274,204	N.A.	N.A.	**31,100**
g12	**350**	850	1600	800
g13	N.A.	N.A.	N.A.	N.A.
g14	N.A.	N.A.	**40,300**	N.A.
g15	N.A.	N.A.	N.A.	N.A.
g16	12,850	41,050	**8200**	13,850
g17	N.A.	N.A.	N.A.	N.A.
g18	219,036	422,568	**16,300**	21,800
g19	N.A.	N.A.	**64,500**	N.A.
g20	N.A.	N.A.	N.A.	**8900**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	N.A.	N.A.	**95,850**	N.A.
g24	2950	7151	**2150**	6800

Best values are in bold

Table 16 Comparison of CSMO against ABC, CHDE and PESO in terms of median of number of function evaluations obtained with successful runs out of 25 independent on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**10,750**	24,950	14,100	83,825
g02	**255,548.5**	376,953	290,100	N.A.
g03	N.A.	N.A.	N.A.	N.A.
g04	21,850	77,150	**9950**	17,450
g05	N.A.	N.A.	N.A.	N.A.
g06	201,324	175,484	**6500**	33,200
g07	N.A.	N.A.	**70,100**	N.A.
g08	**950**	1950	1100	3050
g09	N.A.	N.A.	**29,250**	N.A.
g10	N.A.	N.A.	N.A.	N.A.
g11	359,310	N.A.	N.A.	**86,750**
g12	**1050**	2450	4300	2300
g13	N.A.	N.A.	N.A.	N.A.
g14	N.A.	N.A.	**77,775**	N.A.
g15	N.A.	N.A.	N.A.	N.A.
g16	26,050	45,250	**12,650**	14,600
g17	N.A.	N.A.	N.A.	N.A.
g18	372,280	427,268	81,375	**23,000**
g19	N.A.	N.A.	**165,750**	N.A.
g20	N.A.	N.A.	N.A.	**12,200**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	N.A.	N.A.	**196,775**	N.A.
g24	4250	12,151	**2925**	9350

Best values are in bold

Table 17 Comparison of CSMO against ABC, CHDE and PESO in terms of worst of number of function evaluations obtained with successful runs out of 25 independent on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	28,150	27,250	**20,300**	366,800
g02	**293,361**	441,355	372,900	N.A.
g03	N.A.	N.A.	N.A.	N.A.
g04	**30,150**	85,451	35,750	97,100
g05	N.A.	N.A.	N.A.	N.A.
g06	326,597	218,293	**9150**	53,600
g07	N.A.	N.A.	**70,100**	N.A.
g08	**1250**	2750	1650	4550
g09	N.A.	N.A.	**54,800**	N.A.
g10	N.A.	N.A.	N.A.	N.A.
g11	476,918	N.A.	N.A.	**329,300**
g12	**1650**	4950	6450	3500
g13	N.A.	N.A.	N.A.	N.A.
g14	N.A.	N.A.	**323,750**	N.A.
g15	N.A.	N.A.	N.A.	N.A.
g16	80,724	53,050	43,600	**20,150**
g17	N.A.	N.A.	N.A.	N.A.
g18	476,428	465,770	236,350	**23,000**
g19	N.A.	N.A.	**437,950**	N.A.
g20	N.A.	N.A.	N.A.	**192,500**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	N.A.	N.A.	**267,900**	N.A.
g24	5850	14,650	**4600**	10,850

Best values are in bold

be seen that CSMO has 100 percent feasibility rate in 12 problems. In problems C05, C06 and C11, CSMO fails to enter the feasible region in any run. CSMO has 100 percent feasibility rate in all the problems having inequality constraints only or both inequality and equality constraints only. CSMO has either low or zero feasibility rate on the problems having

Table 18 Comparison of CSMO against ABC, CHDE and PESO in terms of mean of number of function evaluations obtained with successful runs out of 25 independent on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	**13,714**	25,198	14,642	120,840
g02	**255,548**	379,636	272,650	N.A.
g03	N.A.	N.A.	N.A.	N.A.
g04	22,690	76,624	**11,165**	21,038
g05	N.A.	N.A.	N.A.	N.A.
g06	216,177	179,669	**6830**	33,992
g07	N.A.	N.A.	**70,100**	N.A.
g08	**934**	1886	1086	3068
g09	N.A.	N.A.	**30,311**	N.A.
g10	N.A.	N.A.	N.A.	N.A.
g11	374,996	N.A.	N.A.	**115,400**
g12	**1026**	2526	4142	2264
g13	N.A.	N.A.	N.A.	N.A.
g14	N.A.	N.A.	**113,065**	N.A.
g15	N.A.	N.A.	N.A.	N.A.
g16	34,930	45,022	**14,402**	15,457
g17	N.A.	N.A.	N.A.	N.A.
g18	348,277	438,535	94,531	**22,600**
g19	N.A.	N.A.	**173,146**	N.A.
g20	N.A.	N.A.	N.A.	**37,450**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	N.A.	N.A.	**189,325**	N.A.
g24	4258	11,878	**3064**	9116

Best values are in bold

Table 19 Comparison of CSMO against ABC, CHDE and PESO in terms of standard deviation of number of function evaluations obtained with successful runs out of 25 independent on CEC2006 benchmark problems

Problems	CSMO	ABC	CHDE	PESO
g01	6026	**869**	3308	99,405
g02	53,474	**37,772**	110,017	N.A.
g03	N.A.	N.A.	N.A.	N.A.
g04	**2973**	4726	5623	16,017
g05	N.A.	N.A.	N.A.	N.A.
g06	42,756	15,270	**1131**	4401
g07	N.A.	N.A.	N.A.	N.A.
g08	**146**	396	338	813
g09	N.A.	N.A.	**13,775**	N.A.
g10	N.A.	N.A.	N.A.	N.A.
g11	80,233	N.A.	N.A.	**75,240**
g12	**274**	1092	1181	673
g13	N.A.	N.A.	N.A.	N.A.
g14	N.A.	N.A.	**84,753**	N.A.
g15	N.A.	N.A.	N.A.	N.A.
g16	20,422	2879	7981	**2159**
g17	N.A.	N.A.	N.A.	N.A.
g18	109,348	23,702	75,516	**692**
g19	N.A.	N.A.	**98,839**	N.A.
g20	N.A.	N.A.	N.A.	**59,787**
g21	N.A.	N.A.	N.A.	N.A.
g22	N.A.	N.A.	N.A.	N.A.
g23	N.A.	N.A.	**70,820**	N.A.
g24	796	2062	**542**	987

Best values are in bold

Table 20 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of best, median, worst, mean and standard deviation (SD) of number of function evaluations with successful runs out of 25 independent runs on CEC2006 benchmark problems

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Best	8	0	1
	Median	8	0	1
	Worst	5	0	4
	Mean	8	0	1
	SD	4	0	5
	Total	33	0	12
CHDE	Best	2	0	7
	Median	4	0	5
	Worst	4	0	5
	Mean	4	0	5
	SD	4	0	5
	Total	18	0	27
PESO	Best	5	0	4
	Median	4	0	5
	Worst	5	0	4
	Mean	4	0	5
	SD	5	0	4
	Total	23	0	22

equality constraints only. Feasibility rate of all the algorithms is provided in Table 23. Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate is provided in Table 24. Table 24 shows that CSMO has higher feasibility rate in more number of problems than ABC and CHDE, while PESO outperforms CSMO in this case.

Tables 25, 26 and 27 present the best, median and worst of objective function values obtained by all the algorithms in 25 runs. Tables 28, 29 present the mean and standard deviation of objective function values for feasible runs only. Summary of pairwise comparison is provided in Table 30, and it can be seen that CSMO outperforms the other three algorithms

Table 21 Wilcoxon rank-sum test based on mean function error values with a significance level of $\alpha = 0.05$ for CEC2006 problems ('+' indicates CSMO is significantly better, '-' indicates CSMO is significantly worse, and '=' indicates there is no significant difference)

Problems	Pairwise comparison of CSMO versus		
	ABC	CHDE	PESO
g01	=	+	+
g02	−	+	+
g03	+	+	=
g04	=	+	=
g05	+	+	=
g06	−	+	−
g07	=	−	+
g08	=	=	+
g09	+	+	+
g10	+	+	+
g11	=	=	−
g12	=	=	=
g13	+	+	−
g14	+	−	+
g15	+	+	+
g16	−	+	+
g17	+	+	=
g18	+	+	+
g19	−	+	+
g20	*	*	=
g21	*	*	*
g22	*	*	*
g23	=	=	+
g24	=	=	=

* Represents that there is no comparison available

Table 22 Feasibility rate (F.R.) and best, median, worst, mean and standard deviation (SD) of the objective function values obtained by CSMO with 25 independent runs on CEC2010 benchmark problems for 10 dimensions

Problems	F.R.	Best	Median	Worst	Mean	SD
C01	100	−7.47E−01	−7.47E−01	−7.47E−01	−7.47E−01	0.00E+00
C02	100	4.96E−01	2.10E+00	3.55E+00	2.22E+00	9.03E−01
C03	100	1.29E+08	2.73E+12	5.22E+14	4.82E+13	1.16E+14
C04	16	1.55E−03	9.02E−04(2)	2.28E+00(2)	7.00E+00	8.08E+00
C05	0	3.08E−04(2)	5.92E−03(2)	1.73E−02(2)	N.A.	N.A.
C06	0	6.02E−04(2)	9.14E−03(2)	2.31E−02(2)	N.A.	N.A.
C07	100	6.37E−02	1.57E+00	7.20E+01	9.63E+00	1.88E+01
C08	100	4.12E−05	1.06E+01	1.02E+03	6.27E+01	2.03E+02
C09	8	1.35E+12	3.90E−04(1)	5.79E−03(1)	1.50E+13	1.93E+13
C10	8	3.15E+11	5.29E−04(1)	2.88E−03(1)	4.68E+11	2.16E+11
C11	0	7.24E−04(1)	8.53E−01(1)	1.89E+01(1)	N.A.	N.A.
C12	100	−5.70E+02	−2.62E+02	2.14E+01	−2.94E+02	2.75E+02
C13	100	−6.84E+01	−6.74E+01	−6.23E+01	−6.66E+01	2.21E+00
C14	100	4.95E−03	5.43E−01	7.42E+00	1.34E+00	1.95E+00
C15	100	8.23E+10	1.15E+12	8.53E+12	1.90E+12	2.27E+12
C16	100	8.33E−01	1.02E+00	1.05E+00	9.96E−01	5.27E−02
C17	100	8.19E+01	3.04E+02	1.01E+03	3.83E+02	2.33E+02
C18	100	2.32E+03	6.78E+03	1.70E+04	7.48E+03	3.54E+03

Table 23 Comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate obtained on CEC2010 benchmark problems for 10 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	100	100	100	100
C02	**100**	0	0	**100**
C03	**100**	0	44	88
C04	16	0	**44**	4
C05	0	0	0	**100**
C06	0	0	0	**100**
C07	100	100	100	100
C08	100	100	100	100
C09	8	0	0	**100**
C10	8	0	0	**100**
C11	0	0	**60**	12
C12	**100**	56	76	36
C13	100	100	100	100
C14	**100**	12	16	**100**
C15	**100**	0	0	**100**
C16	**100**	0	0	**100**
C17	**100**	0	0	**100**
C18	**100**	0	0	**100**

Best values are in bold

Table 24 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate (F.R.) on CEC2010 benchmark problems for 10 dimensions

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Feasibility rate (F.R.)	11	7	0
CHDE	Feasibility rate (F.R.)	10	6	2
PESO	Feasibility rate (F.R.)	3	10	5

in terms of obtained objective function values. Results of Wilcoxon rank-sum test based on mean objective function value for 10 dimensions problems are presented in Table 31. It shows CSMO performs significantly better than other algorithms on most of the problems.

6.3 Discussion of results for CEC2010 benchmark problems for 30 dimensions

Tables 35, 36 and 37 present the results for best, median and worst of objective function values in 25 runs, respectively. Tables 38, 39 present the mean and standard deviation of objective function values obtained in feasible runs only. From Table 40, it is clear that CSMO performs better than the other three algorithms in terms of objective function values for 30 dimensions also. Table 41 provides outcome of Wilcoxon rank-sum test for 30 dimension problems showing CSMO performing significantly better than other algorithms.

Table 25 Comparison of CSMO against ABC, CHDE and PESO in terms of best of objective function values obtained with 25 independent runs on CEC2010 benchmark problems for 10 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	**−7.47E−01**	**−7.47E−01**	**−7.47E−01**	**−7.37E−01**
C02	4.96E−01	3.40E−05(1)	3.53E−05(1)	**−1.69E−02**
C03	1.29E+08	1.02E−02(1)	**0.00E+00**	2.58E+09
C04	1.55E−03	1.48E−04(2)	**−1.00E−05**	9.22E−04
C05	3.08E−04(2)	9.80E−03(2)	5.25E−05(1)	**1.80E+02**
C06	6.02E−04(2)	3.55E−02(2)	5.55E−05(1)	**3.88E−01**
C07	6.37E−02	9.24E−02	**0.00E+00**	9.16E−06
C08	4.12E−05	7.24E−03	**0.00E+00**	2.34E−05
C09	**1.35E+12**	1.00E−04(1)	1.82E−04(1)	2.82E+12
C10	**3.15E+11**	1.64E−04(1)	4.57E−04(1)	4.45E+12
C11	7.24E−04(1)	1.15E−03(1)	**−1.52E−03**	−1.17E−03
C12	**−5.70E+02**	−1.99E−01	−3.05E+02	−1.99E−01
C13	**−6.84E+01**	**−6.84E+01**	**−6.84E+01**	−6.56E+01
C14	4.95E−03	2.70E+11	3.50E+00	**4.57E−07**
C15	**8.23E+10**	5.40E−03(1)	3.40E−01(1)	4.02E+12
C16	8.33E−01	5.28E−05(2)	5.15E−05(2)	**7.14E−01**
C17	**8.19E+01**	3.40E−05(1)	3.57E−05(1)	1.77E+02
C18	**2.32E+03**	2.07E−08(1)	3.32E−06(1)	4.15E+03

Best values are in bold

Table 26 Comparison of CSMO against ABC, CHDE and PESO in terms of median of objective function values obtained with 25 independent runs on CEC2010 benchmark problems for 10 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	**−7.47E−01**	**−7.47E−01**	**−7.47E−01**	−6.69E−01
C02	**2.10E+00**	5.10E−05(1)	1.27E−04(1)	3.78E+00
C03	**2.73E+12**	3.36E−01(1)	1.01E−04(1)	8.99E+13
C04	9.02E−04(2)	2.98E−03(4)	**2.60E−05(1)**	6.77E−02(3)
C05	5.92E−03(2)	1.16E−01(2)	1.62E−01(2)	**5.06E+02**
C06	9.14E−03(2)	2.83E−01(2)	2.28E−01(2)	**3.88E−01**
C07	1.57E+00	3.44E+00	**2.31E−19**	1.21E−01
C08	1.06E+01	**2.12E−01**	1.06E+01	4.51E+01
C09	3.90E−04(1)	2.66E−03(1)	8.22E−03(1)	**1.71E+13**
C10	5.29E−04(1)	4.45E−03(1)	5.48E−03(1)	**1.86E+13**
C11	8.53E−01(1)	3.16E+00(1)	**−1.52E−03**	4.34E−01(1)
C12	**−2.62E+02**	−1.68E−01	−1.99E−01	1.41E+01(1)
C13	−6.74E+01	−6.82E+01	**−6.84E+01**	−6.23E+01
C14	**5.43E−01**	2.34E+00(1)	1.59E+00(1)	3.99E+00
C15	**1.15E+12**	1.03E+00(1)	2.08E+00(1)	8.29E+13
C16	**1.02E+00**	1.38E−04(2)	2.93E−04(2)	1.05E+00
C17	**3.04E+02**	6.97E−05(1)	1.19E−04(1)	5.93E+02
C18	**6.78E+03**	1.49E−05(1)	1.18E−04(2)	1.38E+04

Best values are in bold

Table 27 Comparison of CSMO against ABC, CHDE and PESO in terms of worst of objective function values obtained with 25 independent runs on CEC2010 benchmark problems for 10 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	**−7.47E−01**	**−7.47E−01**	**−3.27E−01**	**−5.05E−01**
C02	**3.55E+00**	1.10E−04(1)	1.28E−03(1)	5.35E+00
C03	**5.22E+14**	2.86E+01(1)	1.27E+04(1)	3.56E+00(1)
C04	2.28E+00(2)	**2.63E−02(4)**	9.83E+00(4)	3.65E+00(2)
C05	1.73E−02(2)	2.76E−01(2)	1.09E+00(2)	**5.93E+02**
C06	2.31E−02(2)	7.00E−01(2)	6.75E−01(2)	**3.88E−01**
C07	7.20E+01	**6.07E+00**	5.39E+04	1.54E+02
C08	1.02E+03	**4.16E+01**	1.79E+06	1.60E+04
C09	5.79E−03(1)	1.33E−02(1)	6.54E−02(1)	**3.03E+13**
C10	2.88E−03(1)	1.61E−02(1)	5.80E−02(1)	**2.85E+13**
C11	1.89E+01(1)	**6.13E+00(1)**	1.13E+06(1)	1.08E+03(1)
C12	**2.14E+01**	2.71E−02(1)	3.15E+07(2)	5.75E+02(2)
C13	−6.23E+01	**−6.56E+01**	−6.21E+01	−5.96E+01
C14	**7.42E+00**	4.70E+01(2)	2.54E+01(1)	5.62E+04
C15	**8.53E+12**	1.46E+02(1)	3.47E+01(1)	3.61E+14
C16	**1.05E+00**	9.75E−04(2)	7.50E−03(2)	1.13E+00
C17	1.01E+03	2.26E−04(1)	2.99E−03(1)	**4.92E+02**
C18	**1.70E+04**	7.25E−05(1)	2.43E−02(1)	3.81E+04

Best values are in bold

Table 28 Comparison of CSMO against ABC, CHDE and PESO in terms of mean of objective function values obtained with feasible runs out of 25 independent runs on CEC2010 benchmark problems for 10 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	**−7.47E−01**	**−7.47E−01**	−7.15E−01	−6.41E−01
C02	**2.22E+00**	N.A.	N.A.	3.34E+00
C03	4.82E+13	N.A.	**9.64E−07**	1.90E+14
C04	7.00E+00	N.A.	**−9.51E−06**	9.22E−04
C05	N.A.	N.A.	N.A.	**4.82E+02**
C06	N.A.	N.A.	N.A.	**3.88E−01**
C07	9.63E+00	**3.34E+00**	2.23E+03	1.84E+01
C08	6.27E+01	**2.53E+00**	7.16E+04	8.36E+02
C09	**1.50E+13**	N.A.	N.A.	1.58E+13
C10	**4.68E+11**	N.A.	N.A.	1.88E+13
C11	N.A.	N.A.	**−1.52E−03**	−4.24E−04
C12	**−2.94E+02**	−1.84E−01	−4.87E+01	1.10E+00
C13	−6.66E+01	**−6.79E+01**	−6.62E+01	−6.09E+01
C14	**1.34E+00**	3.82E+11	4.95E+00	8.13E+03
C15	**1.90E+12**	N.A.	N.A.	1.18E+14
C16	**9.96E−01**	N.A.	N.A.	1.03E+00
C17	**3.83E+02**	N.A.	N.A.	6.43E+02
C18	**7.48E+03**	N.A.	N.A.	1.43E+04

Best values are in bold

Table 29 Comparison of CSMO against ABC, CHDE and PESO in terms of standard deviation of objective function value obtained with feasible runs out of 25 independent runs on CEC2010 benchmark problems for 10 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	**0.00E+00**	**0.00E+00**	8.44E−02	7.46E−02
C02	**9.03E−01**	N.A.	N.A.	1.33E+00
C03	1.16E+14	N.A.	**3.20E−06**	3.74E+14
C04	8.08E+00	N.A.	**1.62E−06**	N.A.
C05	N.A.	N.A.	N.A.	**1.05E+02**
C06	N.A.	N.A.	N.A.	**5.67E−17**
C07	1.88E+01	**1.62E+00**	1.08E+04	4.38E+01
C08	2.03E+02	**8.32E+00**	3.58E+05	3.18E+03
C09	1.93E+13	N.A.	N.A.	**8.50E+12**
C10	**2.16E+11**	N.A.	N.A.	9.42E+12
C11	N.A.	N.A.	**2.24E−19**	8.29E−04
C12	2.75E+02	**1.80E−02**	7.84E+01	3.27E+00
C13	2.21E+00	**8.11E−01**	2.63E+00	2.77E+00
C14	1.95E+00	1.06E+11	**1.30E+00**	1.84E+04
C15	**2.27E+12**	N.A.	N.A.	1.02E+14
C16	**5.27E−02**	N.A.	N.A.	7.22E−02
C17	**2.33E+02**	N.A.	N.A.	2.62E+02
C18	**3.54E+03**	N.A.	N.A.	7.79E+03

Best values are in bold

Table 30 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of best, median, worst, mean and standard deviation (SD) of objective function values on CEC2010 benchmark problems for 10 dimensions

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Best	16	2	0
	Median	15	1	2
	Worst	12	1	5
	Mean	2	1	3
	SD	1	1	4
	Total	46	6	14
CHDE	Best	9	2	7
	Median	12	2	4
	Worst	18	0	0
	Mean	6	0	2
	SD	4	0	4
	Total	49	4	17
PESO	Best	9	0	9
	Median	12	0	6
	Worst	13	0	5
	Mean	14	0	1
	SD	12	0	2
	Total	60	0	23

Table 31 Wilcoxon rank-sum test based on mean objective function value with a significance level of $\alpha = 0.05$ for CEC2010 problems for 10 dimensions ('+' indicates CSMO is significantly better, '−' indicates CSMO is significantly worse, and '=' indicates there is no significant difference)

Problems	Pairwise comparison of CSMO versus		
	ABC	CHDE	PESO
C01	=	+	+
C02	+	+	+
C03	+	−	=
C04	=	−	=
C05	*	*	−
C06	*	*	−
C07	−	+	+
C08	−	=	+
C09	+	+	+
C10	+	+	+
C11	=	−	=
C12	+	+	+
C13	=	=	+
C14	+	+	+
C15	+	+	+
C16	+	+	+
C17	+	+	+
C18	+	+	+

* Represents that there is no comparison available

Table 32 Feasibility rate (F.R.) and best, median, worst, mean and standard deviation (SD) of the objective function values obtained by CSMO with 25 independent runs on CEC2010 benchmark problems for 30 dimensions

Problems	F.R.	Best	Median	Worst	Mean	SD
C01	100	−8.18E−01	−8.18E−01	−8.04E−01	−8.17E−01	2.80E−03
C02	100	1.38E+00	3.03E+00	4.05E+00	2.97E+00	6.29E−01
C03	0	1.12E−03(1)	2.61E+01(1)	4.53E+02(1)	N.A.	N.A.
C04	0	4.14E−03(3)	4.27E−02(3)	1.22E+00(4)	N.A.	N.A.
C05	0	2.32E−04(2)	9.78E−04(2)	2.22E−03(2)	N.A.	N.A.
C06	0	3.52E−04(2)	2.04E−03(2)	4.22E−03(2)	N.A.	N.A.
C07	100	4.72E−04	1.37E+01	1.03E+02	2.60E+01	3.50E+01
C08	100	2.11E−03	7.99E+01	1.51E+04	1.09E+03	3.23E+03
C09	16	1.78E+13	3.36E−04(1)	1.26E−03(1)	3.36E+13	2.15E+13
C10	12	1.05E+12	3.06E−04(1)	1.53E−03(1)	2.59E+13	2.33E+13
C11	4	2.61E−04	6.91E+00(1)	1.38E+02(1)	2.61E−04	N.A.
C12	92	−8.85E+02	−1.99E−01	3.23E−01(1)	−4.19E+02	4.20E+02
C13	100	−6.75E+01	−6.40E+01	−6.19E+01	−6.43E+01	1.27E+00
C14	100	1.30E−02	1.28E+01	2.87E+03	1.85E+02	5.85E+02
C15	100	7.57E+12	2.60E+13	2.22E+13	2.32E+13	1.10E+13
C16	100	1.06E+00	1.11E+00	1.17E+00	1.11E+00	2.48E−02
C17	100	8.09E+02	1.36E+03	2.44E+03	1.43E+03	3.33E+02
C18	100	1.80E+04	2.63E+04	3.52E+04	2.65E+04	4.57E+03

Table 33 Comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate obtained on CEC2010 benchmark problems for 30 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	100	100	100	100
C02	**100**	0	0	**100**
C03	0	0	0	0
C04	0	0	0	0
C05	0	0	0	**100**
C06	0	0	0	**100**
C07	100	100	100	100
C08	100	100	100	100
C09	16	0	0	**100**
C10	12	0	0	**100**
C11	4	0	**24**	0
C12	**92**	48	36	32
C13	100	100	100	100
C14	**100**	0	24	**100**
C15	**100**	0	0	**100**
C16	**100**	0	0	**100**
C17	**100**	0	0	**100**
C18	**100**	0	0	**100**

Best values are in bold

Table 34 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of feasibility rate (F.R.) on CEC2010 benchmark problems for 30 dimensions

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Feasibility rate (F.R.)	10	8	0
CHDE	Feasibility rate (F.R.)	9	8	1
PESO	Feasibility rate (F.R.)	2	12	4

Table 32 presents the result obtained by CSMO on CEC2010 problems for 30 dimensions. CSMO is successful in entering the feasible region in all the runs in 10 problems. These are the problems with inequality constraints only or both inequality and equality constraints. Table 33 presents the feasibility rate of all the algorithms obtained for 30 dimensions. From Table 34, it can be seen that CSMO outperforms ABC and CHDE, while performs inferior to PESO.

Convergence graphs for 10 dimensions and 30 dimensions for problems C09, C10, C14 and C15 are shown in Figs. 4, 5, 6, 7, 8, 9, 10 and 11.

Table 35 Comparison of CSMO against ABC, CHDE and PESO in terms of best of objective function values obtained with 25 independent runs on CEC2010 benchmark problems for 30 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	−8.18E−01	**−8.22E−01**	−8.18E−01	−7.23E−01
C02	**1.38E+00**	3.34E−05(1)	3.63E−05(1)	2.85E+00
C03	1.12E−03(1)	1.74E+01(1)	**1.00E−04(1)**	3.38E+01(1)
C04	4.14E−03(3)	5.04E−02(4)	**2.50E−05(1)**	5.41E−02(3)
C05	2.32E−04(2)	1.19E−03(2)	5.45E−05(1)	**3.12E+02**
C06	3.52E−04(2)	2.52E−02(2)	7.95E−05(1)	**6.16E−01**
C07	4.72E−04	1.61E+01	**0.00E+00**	1.02E+01
C08	2.11E−03	2.23E+01	**0.00E+00**	1.15E+01
C09	1.78E+13	1.43E−04(1)	2.20E−04(1)	**1.67E+13**
C10	**1.05E+12**	1.06E−04(1)	4.97E−04(1)	1.66E+13
C11	2.61E−04	2.08E+01(1)	**−3.92E−04**	1.37E+01(1)
C12	**−8.85E+02**	−1.98E−01	−1.99E−01	−1.96E−01
C13	−6.75E+01	−5.55E+01	**−6.76E+01**	−6.24E+01
C14	1.30E−02	9.65E−03(1)	**1.49E−12**	1.72E+01
C15	**7.57E+12**	1.05E−02(1)	2.59E−01(1)	4.53E+13
C16	**1.06E+00**	5.30E−05(2)	5.45E−05(2)	1.08E+00
C17	8.09E+02	3.45E−05(1)	3.50E−05(1)	**8.05E+02**
C18	1.80E+04	1.30E−07(1)	5.70E−07(1)	**1.59E+04**

Best values are in bold

Table 36 Comparison of CSMO against ABC, CHDE and PESO in terms of median of objective function values obtained with 25 independent runs on CEC2010 benchmark problems for 30 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	−8.18E−01	**−8.20E−01**	−7.30E−01	−6.51E−01
C02	3.03E+00	4.23E−05(1)	7.37E−05(1)	4.15E+00
C03	2.61E+01(1)	4.36E+02(1)	**4.13E+00(1)**	1.20E+03(1)
C04	**4.27E−02(3)**	4.39E−01(4)	4.55E−02(2)	1.70E+00(3)
C05	9.78E−04(2)	4.97E−02(2)	4.49E−02(2)	**5.67E+02**
C06	2.04E−03(2)	8.23E−02(2)	9.00E−02(2)	**6.16E−01**
C07	1.37E+01	2.35E+01	**1.33E+01**	2.57E+01
C08	7.99E+01	**2.49E+01**	7.29E+01	3.04E+02
C09	3.36E−04(1)	1.12E−03(1)	2.41E−03(1)	**5.65E+13**
C10	3.06E−04(1)	1.55E−03(1)	4.32E−03(1)	**4.78E+13**
C11	6.91E+00(1)	2.24E+01(1)	**6.79E+00(1)**	2.83E+01(1)
C12	**−1.99E−01**	5.01E−05(1)	3.19E−01(1)	4.91E−02(1)
C13	**−6.40E+01**	−5.03E+01	−6.33E+01	−5.96E+01
C14	**1.28E+01**	1.13E−01(1)	7.40E+00(1)	1.17E+04
C15	**2.60E+13**	1.07E−01(1)	1.64E+01(1)	1.90E+14
C16	**1.11E+00**	4.46E−04(2)	1.45E−04(2)	1.17E+00
C17	**1.36E+03**	4.86E−05(1)	5.90E−05(1)	2.25E+03
C18	**2.63E+04**	6.56E−06(1)	5.10E−05(1)	3.94E+04

Best values are in bold

Table 37 Comparison of CSMO against ABC, CHDE and PESO in terms of worst of objective function value obtained with 25 independent runs on CEC2010 benchmark problems for 30 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	−8.04E−01	**−8.16E−01**	−4.25E−01	−5.42E−01
C02	**4.05E+00**	8.51E−05(1)	4.83E−03(1)	5.06E+00
C03	**4.53E+02(1)**	1.89E+03(1)	5.94E+04(1)	1.64E+04(1)
C04	**1.22E+00(4)**	3.10E+00(4)	3.13E+02(4)	1.36E+01(3)
C05	2.22E−03(2)	9.35E−02(2)	2.46E−01(2)	**5.93E+02**
C06	4.22E−03(2)	1.94E−01(2)	3.32E−01(2)	**6.16E−01**
C07	1.03E+02	**4.83E+01**	5.58E+08	3.00E+02
C08	1.51E+04	**1.83E+02**	3.63E+08	1.10E+04
C09	1.26E−03(1)	7.30E−03(1)	1.17E−01(1)	**1.10E+14**
C10	1.53E−03(1)	9.74E−03(1)	3.16E−01(1)	**1.30E+14**
C11	1.38E+02(1)	**2.67E+01(1)**	8.02E+07(1)	1.14E+03(1)
C12	**3.23E−01(1)**	5.70E−01(1)	6.95E+09(1)	1.52E+01(1)
C13	−6.19E+01	−4.53E+01	−5.65E+01	**−6.34E+01**
C14	**2.87E+03**	9.89E−01(1)	7.37E+01(1)	2.75E+05
C15	**2.22E+13**	6.67E−01(1)	1.73E+02(1)	5.62E+14
C16	**1.17E+00**	4.80E−03(2)	7.95E−03(2)	1.28E+00
C17	**2.44E+03**	9.15E−05(1)	4.00E−04(1)	4.58E+03
C18	**3.52E+04**	2.46E−05(1)	1.48E−03(2)	1.07E+05

Best values are in bold

Table 38 Comparison of CSMO against ABC, CHDE and PESO in terms of mean of objective function values obtained with feasible runs out of 25 independent runs on CEC2010 benchmark problems for 30 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	−8.17E−01	**−8.19E−01**	−7.01E−01	−6.51E−01
C02	**2.97E+00**	N.A.	N.A.	4.09E+00
C03	N.A.	N.A.	N.A.	N.A.
C04	N.A.	N.A.	N.A.	N.A.
C05	N.A.	N.A.	N.A.	**5.36E+02**
C06	N.A.	N.A.	N.A.	**6.16E−01**
C07	2.60E+01	**2.43E+01**	3.26E+07	5.74E+01
C08	1.09E+03	**4.13E+01**	2.82E+07	1.57E+03
C09	**3.36E+13**	N.A.	N.A.	5.72E+13
C10	**2.59E+13**	N.A.	N.A.	5.11E+13
C11	2.61E−04	N.A.	**−3.85E−04**	N.A.
C12	**−4.19E+02**	−1.64E−01	−1.88E−01	2.46E+00
C13	**−6.43E+01**	−5.05E+01	−6.31E+01	−5.95E+01
C14	**1.85E+02**	N.A.	1.07E+13	4.03E+04
C15	**2.32E+13**	N.A.	N.A.	2.41E+14
C16	**1.11E+00**	N.A.	N.A.	1.17E+00
C17	**1.43E+03**	N.A.	N.A.	2.35E+03
C18	**2.65E+04**	N.A.	N.A.	4.27E+04

Best values are in bold

Table 39 Comparison of CSMO against ABC, CHDE and PESO in terms of standard deviation of objective function values obtained with feasible runs out of 25 independent runs on CEC2010 benchmark problems for 30 dimensions

Problems	CSMO	ABC	CHDE	PESO
C01	2.80E−03	**1.80E−03**	1.12E−01	4.84E−02
C02	**6.29E−01**	N.A.	N.A.	6.56E−01
C03	N.A.	N.A.	N.A.	N.A.
C04	N.A.	N.A.	N.A.	N.A.
C05	N.A.	N.A.	N.A.	**8.15E+01**
C06	N.A.	N.A.	N.A.	**2.27E−16**
C07	3.50E+01	**5.46E+00**	1.14E+08	6.96E+01
C08	3.23E+03	**4.07E+01**	7.95E+07	3.06E+03
C09	**2.15E+13**	N.A.	N.A.	2.64E+13
C10	**2.33E+13**	N.A.	N.A.	2.52E+13
C11	N.A.	N.A.	**1.13E−05**	N.A.
C12	4.20E+02	4.27E−02	**3.34E−02**	5.05E+00
C13	**1.27E+00**	2.76E+00	2.66E+00	1.89E+00
C14	**5.85E+02**	N.A.	2.55E+13	6.67E+04
C15	**1.10E+13**	N.A.	N.A.	1.45E+14
C16	**2.48E−02**	N.A.	N.A.	4.08E−02
C17	**3.33E+02**	N.A.	N.A.	1.03E+03
C18	**4.57E+03**	N.A.	N.A.	1.98E+04

Best values are in bold

Table 40 Summary of pairwise comparison of CSMO against ABC, CHDE and PESO in terms of best, median, worst, mean and standard deviation (SD) of objective function value on CEC2010 benchmark problems for 30 dimensions

CSMO vs.	Criteria	Better	Equal	Worse
ABC	Best	17	0	1
	Median	16	0	2
	Worst	14	0	4
	Mean	2	0	3
	SD	1	0	4
	Total	50	0	14
CHDE	Best	8	1	9
	Median	14	0	4
	Worst	18	0	0
	Mean	6	0	1
	SD	5	0	1
	Total	51	1	15
PESO	Best	13	0	5
	Median	14	0	4
	Worst	12	0	6
	Mean	13	0	0
	SD	11	0	2
	Total	63	0	17

Table 41 Wilcoxon rank-sum test based on mean objective function value with a significance level of $\alpha = 0.05$ for CEC2010 problems for 30 dimensions ('+' indicates CSMO is significantly better, '−' indicates CSMO is significantly worse, and '=' indicates there is no significant difference)

Problems	Pairwise comparison of CSMO versus		
	ABC	CHDE	PESO
C01	−	+	+
C02	+	+	+
C03	*	*	*
C04	*	*	*
C05	*	*	−
C06	*	*	−
C07	−	=	+
C08	−	=	+
C09	+	+	+
C10	+	+	+
C11	+	=	+
C12	=	+	+
C13	+	+	+
C14	+	=	+
C15	+	+	+
C16	+	+	+
C17	+	+	+
C18	+	+	+

* Represents that there is no comparison available

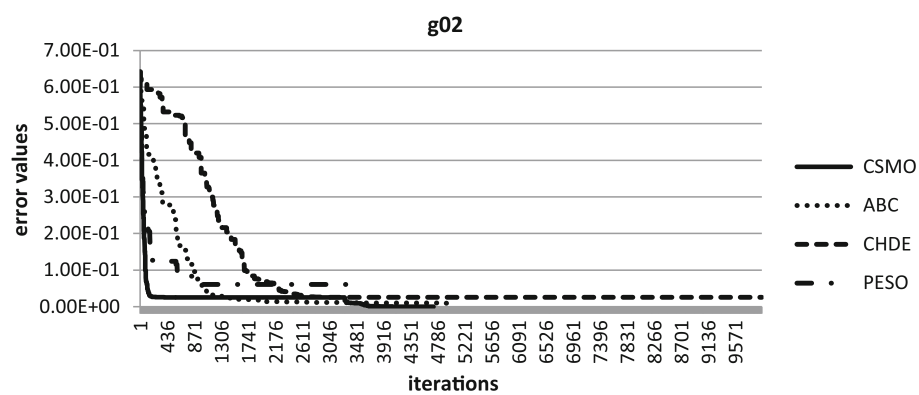


Fig. 1 Convergence graph of problem g02 (CEC2006)

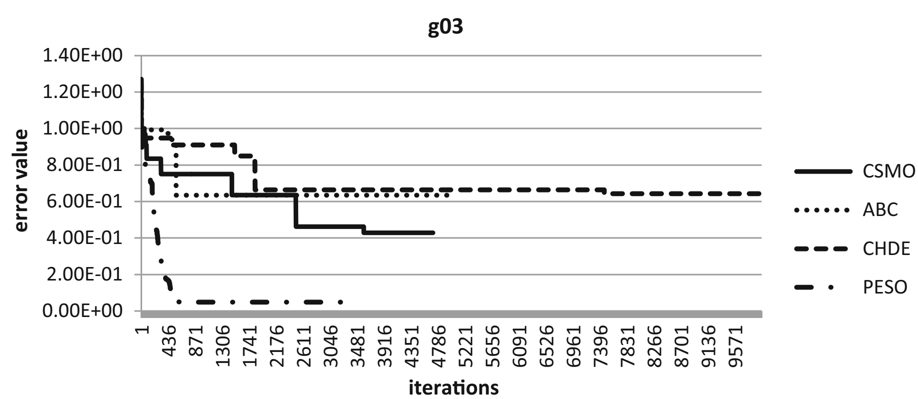


Fig. 2 Convergence graph of problem g03 (CEC2006)

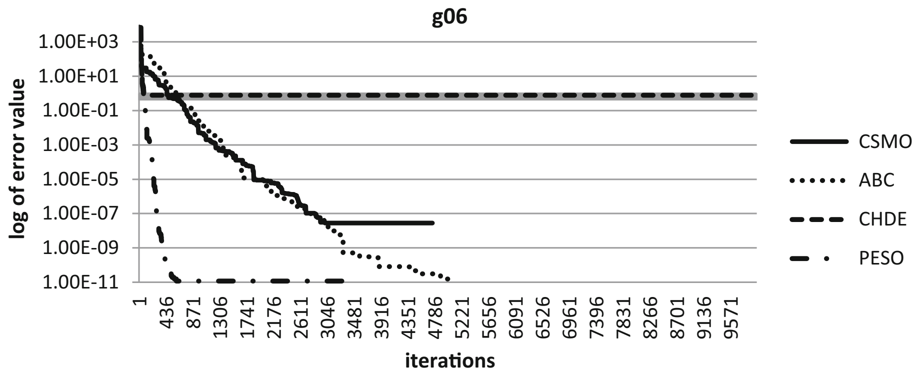


Fig. 3 Convergence graph of problem g06 (CEC2006)

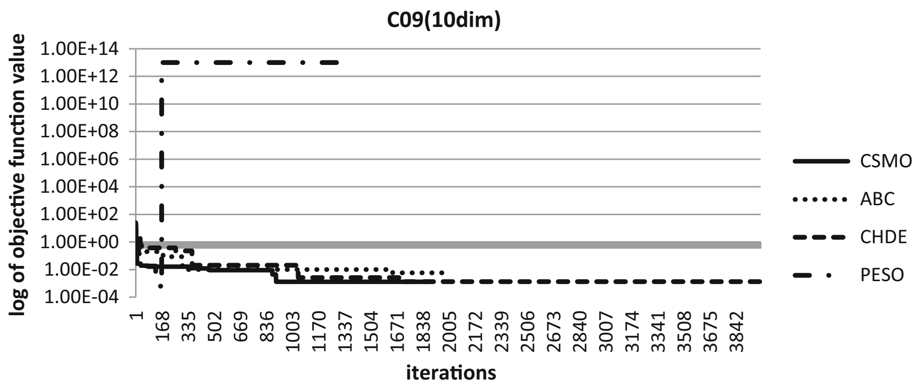


Fig. 4 Convergence graph of problem C09 for 10 dimensions (CEC2010)

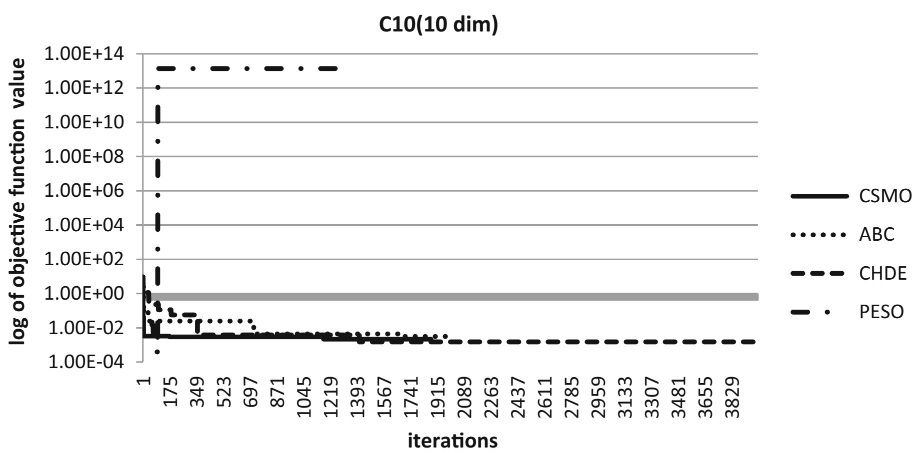


Fig. 5 Convergence graph of problem C10 for 10 dimensions (CEC2010)

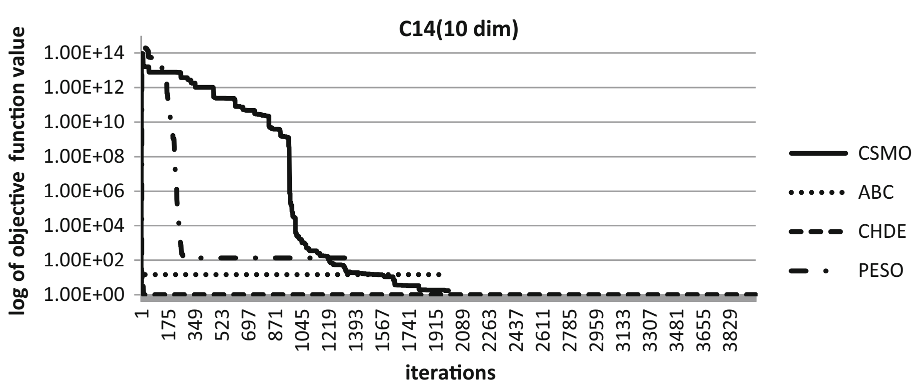


Fig. 6 Convergence graph of problem C14 for 10 dimensions (CEC2010)

Fig. 7 Convergence graph of problem C15 for 10 dimensions (CEC2010)

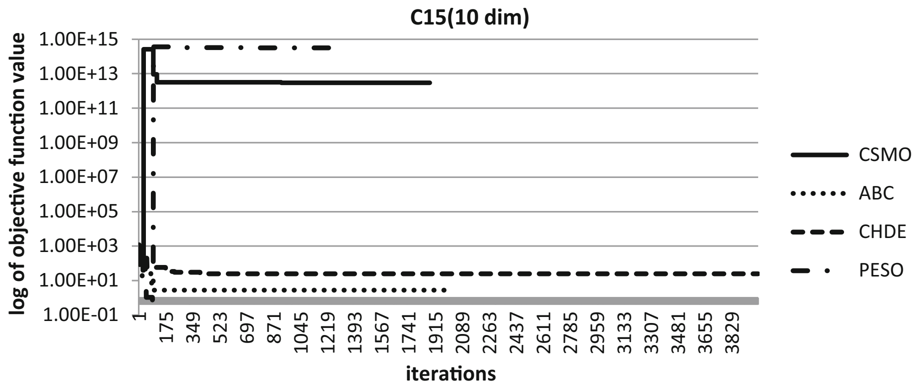


Fig. 8 Convergence graph of problem C09 for 30 dimensions (CEC2010)

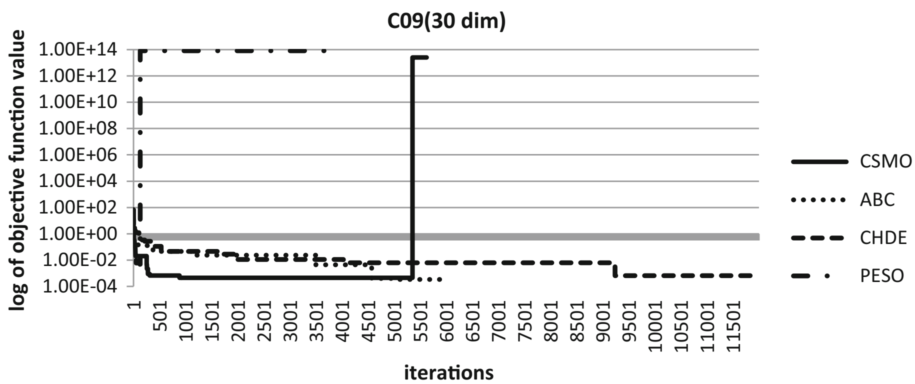


Fig. 9 Convergence graph of problem C10 for 30 dimensions (CEC2010)

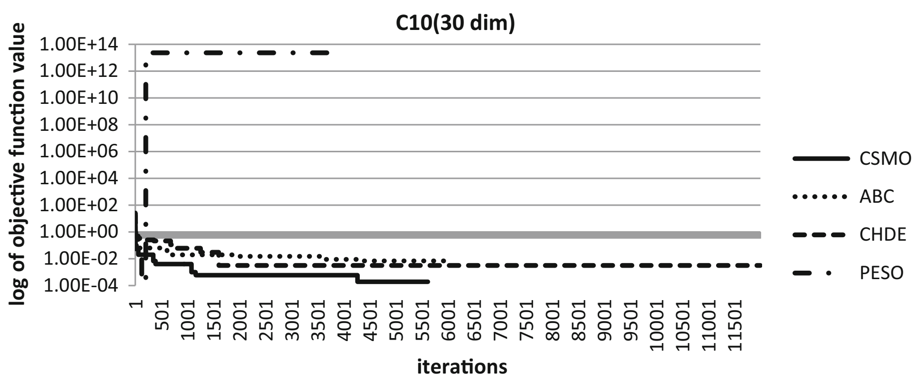
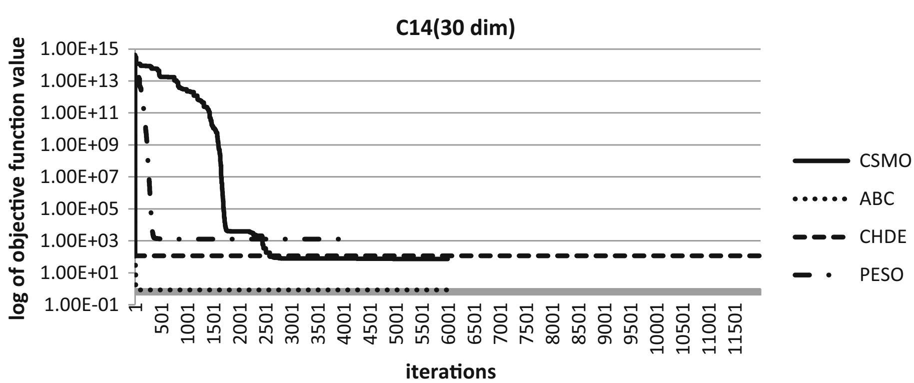


Fig. 10 Convergence graph of problem C14 for 30 dimensions (CEC2010)



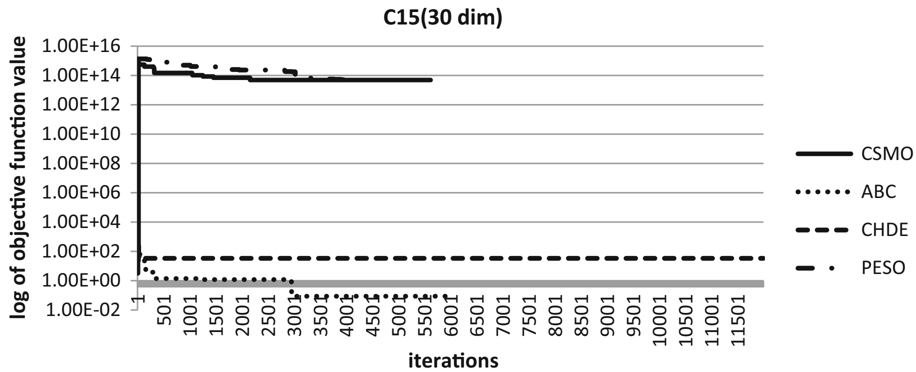


Fig. 11 Convergence graph of problem C15 for 30 dimensions (CEC2010)

An increase in the dimensions (10 to 30) does not have much effect on the performance of CSMO. CSMO has shown good performance on 30 dimensions even with the same population size. It is a general notion that performance of an algorithm generally deteriorates (though not always) with an increase in the dimensions. This is commonly known as curse of dimensionality. But if we observe carefully, for problems C09, C10 and C11, feasibility rate, though it is still less than 20 percent, has been improved. So, no comments can be made about the performance of CSMO if it is not performing well on lower dimensions, and then it will not be performing well on higher dimensions too.

For both CEC2006 and CEC2010 benchmark sets, it can be concluded that CSMO enters the feasibility region in every run in the problems having inequality constraints only. But it finds it difficult to reach the feasibility region in problems having equality constraints only. Even if it enters, it fails to locate the near-optimal solution as in case of CEC2006 problems. But again, this is just an observation which cannot be generalized. Such an insight into the performance of CSMO is important for its further development. Research and development of an algorithm should go hand in hand by knowing the limitations of an algorithm, we can choose suitable methods of its development. Since the objective of this paper is to explore the strengths and weaknesses of CSMO, an insight into the results has been made for CSMO only. Interested readers can have insight into the performance of other algorithms (ABC, CHDE and PESO) on similar lines.

7 Conclusion and future scope

In this paper, a constrained version of SMO named CSMO has been proposed. Results have been compared against PESO, ABC and CHDE on CEC2006 and CEC2010 benchmark problems. Most of the results demonstrate supremacy of CSMO over compared algorithms. Research on SMO is in its infancy stage, and therefore, a lot of research in this direction is possible. Though CSMO has shown good performance according to the criteria selected for comparing the results, we cannot generalize the conclusion in view of no free lunch theorem. But, of course, good results encourage the research community to move in the direction of developing this algorithm. In future, further study will be conducted to identify the type of problems on which CSMO perform better or worse than other state-of-the-art algorithms and to improve its performance where it is not working well. Though empirical study is important to see how an algorithm is performing on a particular set of problems, yet the theoretical study answers *why an algorithm is performing good or bad on a particular problem*. Theoretical study is important to justify the performance of an algorithm. To date, no theoretical study has been made regarding the performance of SMO. In future, an endeavor will be made in this direction. It has been pointed out in Sect. 6 that CSMO has bad performance according to the criteria for comparing the results on the problems having equality constraints only for both the benchmark sets. Future work will include the development of SMO to improve its performance on such type of problems. CSMO has been developed with least possible modification in the basic structure of SMO without involving any additional parameters other than those of SMO. In future efforts, effect of different constraint handling mechanism on the performance of SMO can be investigated to find out most compatible constraint handling mechanism for SMO while solving a particular class of optimization problems. However, performance evaluation of a particular algorithm on a set of benchmark functions is a first step in its development to find out the potential of that algorithm in solving different types of problems. But the ultimate aim of development of an algorithm is to make it capable of solving real-world optimization problems. So, in future applicability of SMO will be demonstrated on a real-world optimization problem.

Acknowledgements The first author would like to thank Ministry of Human Resource Development, Govt. of India, for funding this research under the Grant No. MHR-02-23-200-44.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Ali M, Ahn CW, Pant M (2014) Multi-level image thresholding by synergetic differential evolution. Appl Soft Comput 17:1–11

Asafuddoula M, Ray T, Sarker R (2015) A differential evolution algorithm with constraint sequencing: an efficient approach for problems with inequality constraints. Appl Soft Comput 36:101–113

Bansal JC, Sharma H, Jadon SS, Clerc M (2014) Spider monkey optimization algorithm for numerical optimization. Memet Comput 6(1):31–47

Coello CAC (2002) Theoretical and numerical constraint handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11–12):1245–1287

Deb K (2000) An efficient constraint handling method for genetic algorithm. Comput Methods Appl Mech Eng 186(2–4):311–338

Doğan B, Olmez T (2015) A new metaheuristic for numerical function optimization: vortex search algorithm. Inf Sci 293:125–145

Dorigo M (1992) Optimization, learning and natural algorithms, Ph.D. thesis, Politecnico di Milano, Italy

Elsayed SM, Sarker RA, Essam DL (2011) GA with a new multi-parent crossover for constrained optimization. In: IEEE Congress on Evolutionary Computation, CEC'2011. IEEE Press, New Orleans, USA 2011, pp 857–864

Esmaeili R, Dashtbayazi MR (2014) Modelling and optimization for microstructural properties of A1/SiC nanocomposite by artificial neural network and genetic algorithm. Expert Syst Appl 41:5817–5831

Farmer JD, Packard N, Perelson A (1986) The immune system, adaptation and machine learning. Phys. D 22(1–3):187–204

Gao WF, Liu SY, Huang LL (2014) Enhancing artificial bee colony algorithm using more information-based search equations. Inf Sci 270:112–133

Gupta K, Deep K (2016) Investigation of suitable perturbation rate scheme for spider monkey optimization algorithm. In: Proceedings of fifth international conference on soft computing for problem solving, vol. 437, Springer, Singapore, pp 839–850

Gupta K, Deep K (2016) Tournament Selection Based Probability Scheme in Spider Monkey Optimization Algorithm: In Harmony Search Algorithm, vol. 382, Springer, Berlin, pp 239–250

Gupta K, Deep K, Bansal JC (2016) Improving the local search ability of spider monkey optimization algorithm using quadratic approximation for unconstrained optimization. Comput Intell. doi:10.1111/coin.12081

Han M, Liu C, Xing J (2014) An evolutionary membrane algorithm for global numerical optimization problems. Inf Sci 276:219–241

He Q, Wang L (2007) A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. Appl Math Comput 186(2):1407–1422

Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, AnnArbor

http://coco.gforge.inria.fr/doku.php?id=program

http://www.ntu.edu.sg/home/epnsugan

Imran M, Kowsalya M (2014) Optimal size and siting of multiple distributed generators in distributions system using bacterial foraging optimization. Swarm Evolut Comput 15:58–65

Jia G et al (2013) An improved $(\mu+\lambda)$ constrained differential evolution for constrained optimization. Inf Sci 222:302–322

Karaboga D (2005) An idea based on honeybee swarm for numerical optimization. Technical Report, TR06, Erciyes University, Engineering Faculty, Computer Engineering Department

Karaboga D, Akay B (2011) A modified artificial bee colony (ABC) algorithm for constrained optimization problems. Appl Soft Comput 11(3):3021–3031

Kashan AH (2015) A new metaheuristic for optimization: optics inspired optimization (OIO). Comput Oper Res 55:99–125

Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings, IEEE International Conference, vol. 4, pp. 1942–1948

Khajehzadeh M et al (2012) A modified gravitational search algorithm for slope stability analysis. Eng Appl Artif Intell 25(8):1589–1597

Kumar V, Chhabra JK, Kumar D (2014) Automatic cluster evolution using gravitational search algorithm and its application on image segmentation. Eng Appl Artif Intell 29:93–103

Kuo RJ, Hung SY, Cheng WC (2014) Application of an optimization artificial immune network and particle swarm optimization-based fuzzy neural network to an RFID-based positioning system. Inf Sci 262:78–98

Liang JJ et al (2006) Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. In: Technical Report. Nanyang Technological University, Singapore

Mallipeddi R, Suganthan PN (2010) Problem definitions and evaluation criteria for the CEC 2010 competition on constrained real-parameter optimization. Nanyang Technological University, Singapore

Mezura-Montes E, Coello CAC, Tun-Morales EI (2004) Simple feasibility rules and differential evolution for constrained optimization. In: IMICAI'2004, LNAI 2972, pp. 707–716

Mezura-Montes E, Hernández-Ocaña B (2009) Modified bacterial foraging optimization for engineering design. In: Dagli CH et al (eds) Proceedings of the artificial neural networks in engineering conference, ANNIE'2009, in: Intelligent Engineering Systems Through Artificial Neural Networks, vol 19., ASME PressSt. Louis, MO, USA, pp 357–364

Mezura-Montes E, Velázquez-Reyes J, Coello CAC (2005) Promising infeasibility and multiple offspring incorporated to differential evolution for constrained optimization. In: Proceedings of the genetic and evolutionary computation conference, GECCO'2005, vol 1, Washington, DC, USA, ACM Press, New York, ISBN: 1-59593-010-8, pp 225–232

Mezura-Montes E, Velázquez-Reyes J, Coello CAC (2006) Modified differential evolution for constrained optimization. In: IEEE Congress on Evolutionary Computation, CEC'2006. IEEE, Vancouver, BC, Canada 2006, pp 332–339

Muñoz-Zavala AE, Aguirre AH, Diharce ERV (2005) Constrained optimization via particle evolutionary swarm optimization algorithm (PESO). In: Proceedings of the genetic and evolutionary computation conference, GECCO'2005, vol 1, Washington DC, USA, ACM Press, New York, ISBN:1-59593-010-8, pp 209–216

Muñoz-Zavala AE, et al (2006) PESO+ for constrained optimization. In: IEEE international congress on evolutionary computation, pp 231-238

Parouha RP, Das KN (2015) An efficient hybrid technique for numerical optimization. Comput Ind Eng 83:193–216

Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst Mag 22(3):52–67

Rao RV, Savsani VJ, Vakharia DP (2012) Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems. Inf Sci 183(1):1–15

Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248

Sharma A et al (2016) Ageist spider monkey optimization algorithm. Swarm Evolut Comput 28:58–77
Simon D (2008) Biogeography-based optimization. IEEE Trans Evolut Comput 12:702–713
Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
Sun C-L, Zeng J-C, Pan JS (2009) A particle swarm optimization with feasibility based rules for mixed-variable optimization problems. In: Ninth international conference on hybrid intelligent systems, 2009, HIS'09, IEEE Computer Society Press, Shenyang, China, pp 543–547
Sun C-L, Zeng J-C, Pan J-S (2009) An improved particle swarm optimization with Feasibility-based rules for mixed-variable optimization problems. In: Fourth international conference on innovative computing, information and control, IEEE Computer Society Press, 2009, pp 897–903
Symington MMF (1990) Fission-fusion social organization in ateles and pan. Int J Primatol 11(1):47–61
Toscano-Pulido G, Coello CAC (2004) A constraint-handling mechanism for particle swarm optimization. In: Proceedings of the congress on evolutionary computation 2004, CEC' 2004, vol 2, Portland, Oregon, USA, IEEE Service Center, Piscataway, New Jersey, pp 1396–1403
Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evolut Comput 1:67–82
Zheng Y-J (2015) Water wave optimization: a new nature-inspired metaheuristic. Comput Oper Res 55:1–11
Zielinski K, Laur R (2006) Constrained single-objective optimization using particle swarm optimization. In: IEEE congress on evolutionary computation, CEC'2006. IEEE, Vancouver, BC, Canada 2006, pp 443–450