Expert Systems With Applications 95 (2018) 384–420

Contents lists available at ScienceDirect

Expert Systems With Applications

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

New cuckoo search algorithms with enhanced exploration and exploitation properties

Rohit Salgotra[a,*], Urvinder Singh[a], Sriparna Saha[b]

[a] Electronics & Communication Engineering Department, Thapar University, Punjab, India
[b] Department of Computer Science, Indian Institute of Technology Patna, India

ARTICLE INFO

Article history:
Received 11 May 2017
Revised 17 November 2017
Accepted 21 November 2017
Available online 22 November 2017

Keywords:
Cuckoo search
Cuckoo Version 1.0
Cuckoo Version 2.0
Cuckoo Version 3.0
Numerical optimization
Benchmark problems

ABSTRACT

Cuckoo Search (CS) algorithm is nature inspired global optimization algorithm based on the brood parasitic behavior of cuckoos. It has proved to be an efficient algorithm as it has been successfully applied to solve a large number of problems of different areas. CS employs Lévy flights to generate step size and to search the solution space effectively. The local search is carried out using switch probability in which certain percentages of solutions are removed. Though CS is an effective algorithm, still its performance can be improved by incorporating the exploration and exploitation during the search process. In this work, three modified versions of CS are proposed to improve the properties of exploration and exploitation. All these versions employ Cauchy operator to generate the step size instead of Lévy flights to efficiently explore the search space. Moreover, two new concepts, division of population and division of generations, are also introduced in CS so as to balance the exploration and exploitation. The proposed versions of CS are tested on 24 standard benchmark problems with different dimension sizes and varying population sizes and the effect of probability switch has been studied. Apart from this, the best of the proposed versions is also tested on CEC 2015 benchmark suite. The modified algorithms have been statistically tested in comparison to the state-of-the-art algorithms, namely grey wolf optimization (GWO), differential evolution (DE), firefly algorithm (FA), flower pollination algorithm (FPA) and bat algorithm (BA). The numerical and statistical results prove the superiority of the proposed versions with respect to other popular algorithms available in the literature.

1. Introduction

Cuckoo search (CS) algorithm (Yang & Deb, 2009) is a recently introduced meta-heuristic algorithm and is based on the obligate brood parasitic behavior of cuckoo species found in nature. The algorithm starts by dividing the search process into two phases which are a global and a local phase. In the global phase, the formation of new nests takes place while in the local phase, removal of a fraction of worst nests is followed. Here global phase refers to the exploration where as local phase corresponds to the exploitation. The global phase is governed by Lévy flight based random walks rather than simple Brownian or Gaussian walks (Pavlyukevich, 2007). The main reason for the use of Lévy flight is because of their heavy tail, infinite mean and variance, which helps in exploring the search space in a potentially more efficient way. The local phase is governed by selecting two random solutions from the search space with a certain probability, which controls the extent of exploitation. So overall there are three parameters which control the working capability of CS algorithm (Yang & Deb, 2009). The first parameter is the Lévy flight component which controls the exploration search equation, second is the exploitation or local search equation controlled by two random solutions and third is the probability which decides the extent of exploration and exploitation.

The CS algorithm, because of its simple structure, has gained attraction from the research community in the recent years and a large number of articles have been proposed to improve its working capability such as Zhang, Wang, and Wu (2012), Kanagaraj, Ponnambalam, and Jawahar (2013), Ilunga-Mbuyamba et al. (2016), Long, Liang, Huang, and Chen (2014), Wang and Zhong (2015) and others. A detailed discussion about the recent advances on CS is given in Section 3. It has been found that CS algorithm lacks proper balance between the exploration and exploitation phase and more work is required to be done to achieve the balance (Zhang et al., 2012). Also, probability, which decides the extent of exploration and exploitation, is a very important factor and no proper study has been done to find an appropriate value

* Corresponding author.
E-mail addresses: rohit.salgotra@thapar.edu (R. Salgotra), sriparna@iitp.ac.in (S. Saha).

https://doi.org/10.1016/j.eswa.2017.11.044
0957-4174/

of this parameter. Further, a very little work has been done to test what set of parameters gives the best performance of CS algorithm. Most of the works done till date on CS has focused only on its application in different domains and very limited work has been done to improve its performance. The algorithm though is very efficient in exploring the search space; much is required to improve its exploitative tendencies. Several other problems are also associated with CS. Those are discussed in detail in Section 3. Motivated by these, in the current work at first we developed some new improved versions of CS which have good exploration and exploitation properties. A thorough sensitivity study of different parameters namely probability, population size, and dimension of CS is also conducted and parameter combinations which give the best results for a particular set of problems are reported.

To achieve the above-said objectives, three new variants have been proposed in Section 3 in this a new concept of division of population and generations. By division of generations, the authors aim to divide the population into two halves. This is done because the exploitation is weak in CS, so it requires a better search equation which can provide better results. There are two advantages of dividing the population into two halves. For the first half of population, the algorithm is able to perform extensive exploration using the original equations and for the second half of the population, intensive exploitation can be performed using the new search equation. Also division of population has been done to help the members of the population to change their positions abruptly and converge faster towards the end of iterations. And both division of population and generations is done to achieve a balance between exploration and exploitation phase. More discussion about how the division of population and generation is applied is given in the consequent subsections. For improving the exploration, instead of using Lévy flight based random walks, Cauchy based walks are employed and to improve the exploitation of the local search, different searching strategies are used to evaluate the final solution. More justification for the use of Cauchy based random walks has been presented in subsequent subsections. Further to analyze the performance of CS and the proposed variants experimentally, a set of twenty-four benchmark functions (Suganthan et al., 2005) has been used and the effects of different parameters like switching probability, population size and dimension have been discussed. The effect of switching probability has been analyzed by choosing five different probability values and the best fit probability is estimated. To analyze the effect of population size, three different population sizes are used. In order to show the effect of population size and probability, only CS and proposed variants are compared and best among the proposed variants are analyzed. The best among the proposed variants is then compared with the well-known state of art algorithms. Also, five different dimension sizes varying from 30 to 1000 are used for analyzing the applicability of proposed variant on higher dimensional problems. Statistical tests are also done to validate the end results. Further to validate the best-proposed algorithm in solving some highly challenging datasets, it is applied to CEC 2015 benchmark problems (Liang, Qu, Suganthan, & Chen, 2014) and comparative study with respect to other well-known algorithms is presented.

The article is divided into six sections and is outlined as follows: Section 1 details about the introduction part, dealing with the motivation, objectives, and contribution of this research. Section 2 gives details of the basic CS algorithm while in Section 3, the literature review or the related work is presented. Section 4 outlines the proposed approaches along with their mathematical models and theoretical analysis. This section also presents the complexity analysis of proposed versions. In Section 5, extensive results are presented. Here detailed results with respect to the parameters of CS and the proposed variants along with a comparative analysis of state-of-the-art algorithms are presented. In the subsequent subsections of Section 5, main findings of research, limitations and insightful implications from the proposed work are also presented. The final Section 6 presents the concluding remarks and future scope. The outline of the article is given in Fig. 1.

2. Cuckoo search algorithm

Cuckoo Search (CS) is a new heuristic algorithm inspired from the obligate brood parasitic behavior of some cuckoo species as they lay their eggs in the nests of host birds. Some cuckoos have a specialty of imitating colors and patterns of eggs of a few chosen host species. This reduces the probability of eggs being abandoned. If host bird discovers foreign eggs, they either abandon the eggs or throw them away. Parasitic cuckoos choose a nest where the host bird just lays its eggs. Eggs of cuckoo hatch earlier than their host eggs and when it hatches, it propels the host eggs out of the nests. Hence cuckoo chicks get a good share of food and sometimes they even imitate the voice of host chicks to get more food (Payne, 2005). Mostly cuckoos search food by a simple random walk, where the random walk is a Markov chain whose next position is based on current position and transition probability of next position. Using Lévy flights instead of simple random walks improve the search capabilities. Lévy flight is a random walk in step-lengths following a heavy-tailed probability distribution (Yang & Deb, 2009). Each cuckoo acts as a potential solution to the problem under consideration. The main aim is to generate a new and potentially better solution (cuckoo) to be replaced with a not so good solution. Each nest has one egg but as the problem complexity increases, multiple eggs can be used to represent a set of solutions. There are three basic idealized rules of CS. First rule says that each cuckoo lays one egg and dumps it in a random nest. The second rule is that the nest with the highest fitness will carry over to next generations whereas the final rule defines that the number of available host nests is kept fixed and the egg laid by cuckoo is discovered by host bird with a probability $p \in [0, 1]$. And depending on p, the host bird either throws the egg away or abandons the nest. It is assumed, that only a fraction p of nests is replaced by new nests.

Based on the three rules, the cuckoo search has been implemented. To generate a new solution $x_i^{t+1}$for ith cuckoo, Lévy flight is performed. This step is called global random walk and is given by

$$x_i^{t+1} = x_i^t + \alpha \otimes Le'vy(\lambda)\left(x_{best} - x_i^t\right) \tag{1}$$

The local random walk is given by:

$$x_i^{t+1} = x_i^t + \alpha \otimes H(p - \epsilon) \otimes \left(x_j^t - x_k^t\right) \tag{2}$$

where x_i^t is the previous solution, $\alpha > 0$ is the step size related to problem scales and $\otimes$ is entry wise multiplication. Here x_j^t *and* x_k^t are randomly selected solutions and x_{best} is the current best solution. In present work, the random step length via Lévy flight is considered due to more efficiency of Lévy flights in exploring the search space and is drawn from a Lévy distribution having infinite variance and mean.

$$Le'vy \sim \left\{ \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \; \frac{1}{s^{1+\lambda,}} (s \gg s_0 > 0) \right. \tag{3}$$

As some fractions of new solutions are generated by Lévy flights, so the local search speeds up. Here some of the solutions should be generated by far field randomization which will keep the system away from getting trapped in local optimum, $\Gamma(\lambda)$ is the gamma function, p is the switch probability, ε is a random number and $(1 < \lambda \le 3)$. The step length in cuckoo search is heavy-tailed and any large step is possible due to large scale randomization. The pseudo-code for CS is given in Algorithm 1.

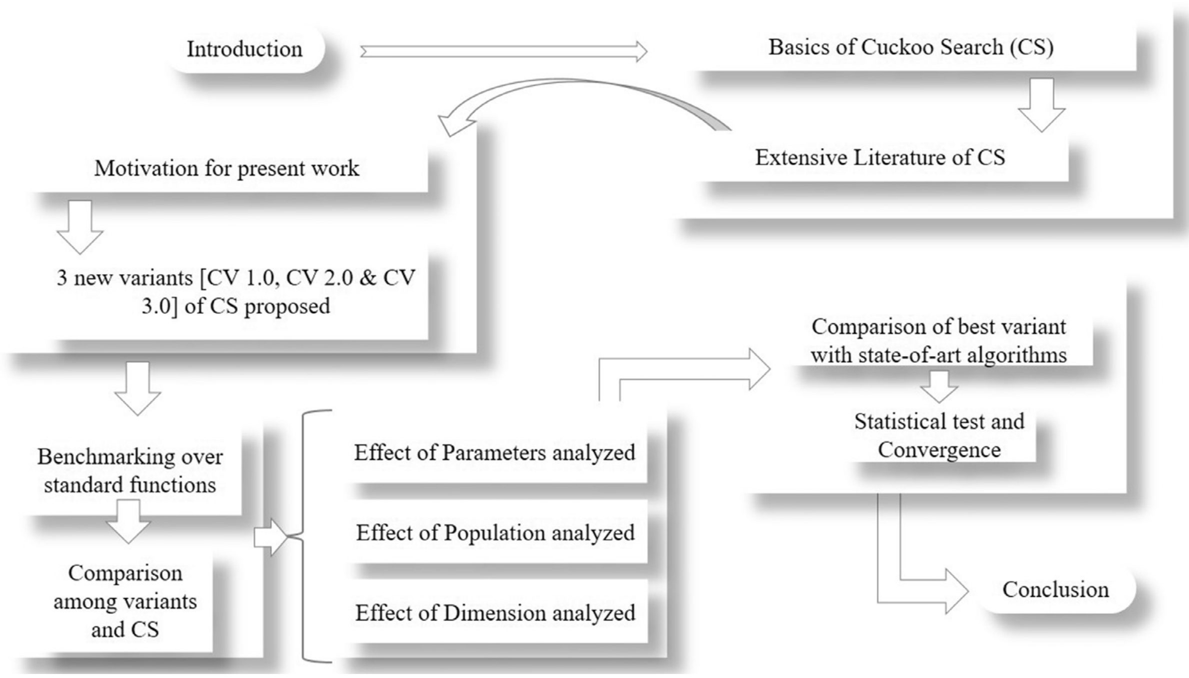

Fig. 1. Outline of the article.

3. Literature review

In this sections first of all, the background of the related work is presented. Secondly, the detailed literature of CS with respect to modifications and applications is discussed. Thirdly, inferences are drawn from the literature which acts as the basis for present work and finally, the main originalities of present work are discussed.

The fact that nature is the perfect problem solver is well known and it has been doing so for millions of years. Biological systems such as perception have emerged to recognize different patterns, immune systems have evolved to eliminate foreign bodies, decision making has been used explicitly to design robots, learning process and others working in collaboration to solve high-end problems. These systems depict high-level computing present in nature and in this context, a large number of optimization algorithms have been proposed for the past two decades. The algorithms are known as nature inspired meta-heuristic optimization algorithms and are finding popularity not only in engineering optimization problems but also in fields such as the stock market for profit or loss prediction, business management, finding optimal path while traveling and others. The main cause of their popularity is their flexibility and faster response as compared to classical optimization techniques (Gutjahr, 2009). One more reason is that they are population based algorithms and don't require an initial guess while solving any problem. Also because of their linear nature and fewer parameters to tune, they are found suitable even for highly complex optimization problems.

Broadly, nature inspired meta-heuristic algorithms are classified into two categories namely evolutionary algorithms and swarm intelligent algorithms. Evolutionary algorithms are based upon the phenomena of the evolution of species. A large number of algorithms have been proposed in this context. The first such algorithm was proposed in the late 70s and is named as an evolutionary strategy (ES) (Rechenberg, 1978). The trend followed and a lot of other algorithms were proposed namely genetic algorithm (GA) (Holland, 1992), genetic programming (GP) (Koza, 1992), differential evolution (DE) (Storn & Price, 1997), biogeography based optimization (BBO) (Simon, 2008), ant lion algorithm (ALO) (Mirjalili, 2015) and others. The other group of the algorithm is swarm intelligent algorithms which are based on the social behavior of animals. These include particle swarm optimization (PSO) (Eberhart & Kennedy, 1995), ant colony optimization (ACO) (Dorigo, Birattari, & Stutzle, 2006), artificial bee colony (ABC) (Karaboga & Basturk, 2007), grey wolf optimizer (Mirjalili, Mirjalili, & Lewis, 2014), bat algorithm (BA) (Yang, 2010a), firefly algorithm (FA) (Yang, 2010b), flower pollination algorithm (FPA) (Yang, 2012) and bat flower pollinator (BFP) (Salgotra & Singh, 2016). All these algorithms have proven their worth in terms of competitiveness. When compared to evolutionary algorithms, swarm intelligent algorithms are found to have some advantages. These algorithms preserve some of the previous information over subsequent iterations while evolutionary algorithms discard previous information on the creation of new population. Also, swarm intelligent algorithms require a minimal set of parameters to be tuned and hence can be implemented easily.

CS is one such swarm intelligent algorithm introduced in the recent past and has proved its worth in various fields of research. Since its inception, a large number of modifications have been proposed. Modified adaptive CS with division into subgroups was proposed by Zhang et al. (2012). It was seen that grouping strategy achieves a balance between exploration and exploitation tendencies of CS algorithm. As the algorithm was tested only on seven benchmark functions and that too from two different datasets, it can be said that only functions providing better results were used and others were neglected. CS with orthogonal learning was proposed in Li, Wang, and Yin (2014). The authors discussed the effect of population size and tested the performance of proposed variant on 23 benchmark problems. It was found that orthogonal based learning enhances the exploration capability of CS and also help in providing better convergence. The authors of the article didn't provide any discussion about the parameters of CS. Hybrid CS with initial crossover and mutation was proposed by Kanagaraj et al. (2013). This version aims at improving the premature convergence and was applied to reliability redundancy allocation problem. Though the proposed variant provided good results no proper analytical study about CS parameters was presented. Other modifications include hybrid CS for knapsack problem (Feng, Wang, Feng, & Zhao, 2014), hybrid CS with soils and wet local search algorithm (Long et al., 2014), parameter adjustment with chaotic map was introduced in Wang and Zhong (2015) and it was found that chaotic maps improve the performance of CS. Multi-objective CS was proposed by Yang and Deb for design optimization (Yang & Deb, 2013), discrete CS was proposed for solving travelling salesman problem (Ouaarab, Ahoid, & Yang, 2014), hybrid CS with wind driven algorithm was proposed for satellite image segmentation (Bhandari, Singh, Kumar, & Singh, 2014), improved CS for hybrid flow shop scheduling was proposed in Marichelvam, Prabaharan, and Yang (2014), hybrid CS and harmony search was proposed for numerical optimization in Wang, Gandomi, Zhao, and Chu (2016) enhanced chaos based CS was proposed for global optimization in Huang, Ding, Yu, Wang, and Lu (2016).

In terms of applications, CS has been applied to a wide range of optimization problems such as CS for distributed networks (Nguyen, Truong, & Phung, 2016), feature extraction (Wang, Han, Shen, & Li, 2014), electrocardiogram signal watermarking (Dey, Samanta, Yang, Das, & Chaydhuri, 2013), multi-reservoir system (Ming, Chang, Huang, Wang, & Huang, 2015), CS for Sudoku problem (Soto, Crawford, Galleguillos, Monfroy, & Paredes, 2014), web search clustering (Cobos et al., 2014), PCB drill path problem (Lim, Kanagaraj, & Ponnambalam, 2014), finite impulse response fractional order differentiator design (Kumar & Rawat, 2015), designing 1-dimensional and 2-dimensional recursive filters (Sarangi, Panda, & Dash, 2014), video tracking system (Walia & Kapoor, 2014) and other applications such as economic load dispatch by Bindu and Reddy (2013), design of planer EBG structures (Pani, Nagpal, Malik, & Gupta, 2013) and a brief review of CS was proposed by Fister, Yang, Fister, and Fister (2014). Other applications include embedded system design (Kumar & Chakarverty, 2011), playing Mario game using CS (Speed, 2010), flood forecasting (Chaowanawatee & Heednacram, 2012), Levenberg Marquardt based using back propagation training using CS (Nawi, Khan, & Rehman, 2013), brain image segmentation (Ilunga-Mbuyamba et al., 2016) and others. Most of these articles focus on the problem itself without proper configuring the parameters of CS algorithm.

3.1. Inference drawn from literature and motivation behind present work

As seen in the section above, CS has been applied to a large number of optimization problems. Four main inferences are drawn from the literature. Firstly, no proper discussion about the effect of parameters on the performance of CS was presented in most of the cases. Secondly, most of the articles based on applications didn't even mention about the parameters of CS. Thirdly, effect of population size and dimension was exploited in very few cases and no proper justification of their analysis is presented and finally, it can be estimated from the literature that grouping can help to improve the performance of CS algorithm.

These inferences have motivated the authors to estimate that dividing the population into sub-groups can improve the diversity in the population and hence improve the exploration on a whole and exploitation within the group. Also, proper values of CS parameters should be identified for proper performance evaluation and effect of population and dimension should be analyzed to check the aplicability of CS for high dimension problems.

Based on the above inference and motivation, the main contributions of this research are highlighted as follows. First of all, division of population and generations have been applied to improve the exploration and exploitation. Secondly, Cauchy based random walks are used instead of Lévy flight for exploring the search space. Based on these two concepts, three different variants of CS are proposed. To analyse the performance, first of all, five different probability values are used to identify the best probability value which gives the best results. Here probability values are chosen in the range of 0 and 1. After analyzing the best probability value, analysis with three different population sizes is presented to check the performance of CS and proposed variants for a maximum number of function evaluations. Finally, five different dimensions are used to test the performance of the best proposed variants on higher dimensional problems.

In the next section, three new variants of CS are proposed to address all the issues discussed above.

4. The proposed variants

This section is also detailed as: first, drawbacks of original CS and reasons for the proposal of new variants are presented. Secondly, details about the modifications and how they are implemented are presented. Thirdly, three versions of CS are proposed and finally, complexity analysis of proposed versions is presented.

CS algorithm uses a combination of global explorative random walk and local exploitative random walk which is controlled by switching probability. Based on the switching probability, local search becomes very intensive occurring for only one-fourth of the time and global search is extensive taking place for around three-fourth of the times. Also, cuckoo search follows Lévy flight behavior which is far better than a simple random walk. Lévy flights have infinite variance and mean which helps CS to explore the search space in a better way. This advantage along with local and global search makes CS more efficient in achieving global convergences. Over the past few decades, the complexity of problems to be optimized is increasing day by day and so do the evolutionary algorithms. Wolpert and Macready proved that no algorithm can be generalized for all optimization problems (Wolpert & Macready, 1997). So, it becomes necessary to design new algorithms as per particular set of problems. In this paper, local search and global search phase of CS are modified to achieve a better position of the new cuckoo. Different variants are proposed and these variants are based on these new ideas. Two new concepts based on the division of generations and population are proposed. Based on these two concepts, three new versions of CS have been proposed. The first version uses enhanced global and dual division in local search phase whereas the second version uses dual division in global search only and in the final version, a four-fold division of population is followed in global search phase. Here, note that the division of generations is applied in all the three versions whereas division of population is applied to local and global search phases. Further, the reason for the use of above discussed ideas is given below:

- *Why division of generations and population is used:* There are two factors namely exploration and exploitation which decide whether an algorithm is efficient or not. Exploration decides how good an algorithm is in escaping local minima and exploitation decides its convergence properties. So, an algorithm which can balance both this phenomenon is able to achieve better results (Črepinšek, Liu, & Mernik, 2013). For a generalized algorithm, the extensive global search should be followed at the beginning of search process and as the search progresses; more intensive exploitation should be done. So, to maintain a balance in exploration and exploitation, division of generations and population has been implemented. By division of generations, we mean to say that for half of the iterations a standard equation having best explorative tendencies should be used and for rest of the iterations more intensive exploitation should be there. This will maintain a balance between exploration and exploitation. For the division of population, we allocate the population to multiple locations or simply into smaller groups and this process is carried out by use of different search equation in each group. It will help the members of the population to change their positions abruptly in the initial stages and converge faster as the iterations proceed. Thus, both the modifications will add up to maintain a balance between the exploration and exploitation.

- *How division of generations is applied:* The generations have been divided into two halves. For the first half of the generations, Cauchy based global search is adopted and the reason for its use has been explained in the subsequent subsection. The local search phase follows the original equation of CS. For the second half of the generations, division of generations is followed. Also, to avoid repetitiveness, the first half of the generations which is same in all the three cases have not been discussed explicitly in every modification proposed. Only modifications based on the second half of generations are presented.
- *How division of population is applied:* The division of population has been employed using multiple search equations. Using different division strategies, three modified versions have been proposed. In the first version, modified global search and dual local search is followed. By dual local search, we mean to say that population is divided into two parts and two search equations are used to implement this phenomenon. In the second version, the dual search equations are employed in global search phase. For the third version, the population is divided into four parts and four search equations are employed to update the solution.
- *Cauchy based mutation operator:* Though Lévy mutated step size is good for exploring the search space, still there is more room for a better exploration operator. So, in this context, Cauchy based mutation operator is applied to generate step size instead of Lévy mutation operator. The Cauchy based mutation operator is used to generate a Cauchy distributed random number δ. This random number is then integrated in the global search equation to generate a new solution. The first concept for such mutation was formulated in Yao, Liu, and Lin (1999). The Cauchy density function is given by

$$f_{Cauchy(0,g)}(\delta) = \frac{1}{\pi}\frac{g}{g^2+\delta^2} \tag{4}$$

The Cauchy distribution function is given by:

$$y = \frac{1}{2} + \frac{1}{\pi}\arctan\left(\frac{\delta}{g}\right) \tag{5}$$

where g is the scale parameter and its value is taken as 1 and $y \in [0,\ 1]$. Solving Eq. (5) for the value of δ, we get

$$\delta = \tan\left(\pi\left(y-\frac{1}{2}\right)\right) \tag{6}$$

The above equation will generate a Cauchy distributed random number in the range of 0 to 1. Based on this the general Eq. (2) of the global search phase changes to

$$x_i^{t+1} = x_i^t + \alpha \otimes Cauchy(\delta)\left(x_i^t - x_j^t\right) \tag{7}$$

The use of Cauchy distribution because of its fatter tail, generate larger steps, which helps the algorithm in exploring the search space in a better way. This helps the algorithm in escaping local minima (Salgotra & Singh, 2017). This has been done because the best solution guided difference prevents the algorithm from exploring the search space and this may lead to premature convergence. The Cauchy based mutation operator has been employed in all the proposed versions for the first half of the iterations and has not been discussed explicitly in every subsection. Also in the pseudo-code of proposed versions, only the second half of the generations are presented. This is because, for the first half of the generations, the proposed versions are same as that of the basic CS algorithm except for the change in Eq. (1) which is replaced by Eq. (7).

4.1. Cuckoo Version 1.0

In this version, two modifications are proposed in the standard CS and it has been named as Cuckoo Version 1.0 (CV 1.0). Firstly, the global pollination phase has been enhanced by taking the mean of first three solutions from the pool of search agents. These three solutions are generated based upon the current best solution. The equations proposed in this regard are given by

$$x_1 = x_i - A_1\left(C_1.x_{best} - x_i^t\right);\, x_2 = x_i - A_2\left(C_2.x_{best} - x_i^t\right);\, x_3 = x_i - A_3\left(C_3.x_{best} - x_i^t\right) \tag{8}$$

$$x_{new} = \frac{x_1+x_2+x_3}{3} \tag{9}$$

Here the current best solution x_{best} is moved to a new location by multiplying it with a random variable and the random solution x_i is subtracted from it to generate the new solution x_{new}. The basic concept for this modification has been inspired from the search equations of GWO (Mirjalili et al., 2014). In GWO, the solution is generated by adjusting the parameters with respect to the position of best search agent. The same concept has been employed to obtain new solution x_{new}. This new solution is then subjected to Eq. (7) of the global search phase which leads to

$$x_i^{t+1} = x_{new}^t + \alpha \otimes Cauchy(\delta)\left(x_{best} - x_{new}^t\right) \tag{10}$$

where A_1, A_2, A_3 *and* C_1, C_2, C_3 belongs to A and C respectively. A and C are given by

$$A = 2a.r_1 - a;\, C = 2.r_2 \tag{11}$$

where a is a linearly decreasing random number in the range of [0, 2] with respect to iterations, r_1 *and* r_2 are two uniformly distributed random numbers in the range of [0,1]. These random numbers r_1 *and* r_2 allow the search agents to reach any position within the search space and hence increase the explorative tendencies of CS algorithm.

Secondly, the local search phase is enhanced to improve the performance of basic CS. The population, in this case, is divided into halves and two different searching strategies are used to evaluate the final solution. For the first half of the population, the general Eq. (2) of the standard CS is used and for the other half, a new search equation is introduced. This new search equation is given by

$$x_i^t = x_i^t + F.\left(\left(x_j^t - x_k^t\right) + \left(x_l^t - x_m^t\right)\right) \tag{12}$$

where x_j^t, x_k^t, x_l^t *and* x_m^t are four random solutions and the new solution will be updated based on them. Here because of the use of same random number F (in the range of 0 to 1), the equation becomes efficient for smaller search space. So, using the same for second half of the population will definitely provide better results. On a whole, the combination of Cauchy based global search and dual local search can provide an exact balance between exploration and exploitation of the proposed variants. The pseudo-code for CV 1.0 is given in Algorithm 2.

4.2. Cuckoo Version 2.0

For the second proposed version, division of population has been applied for both local and global phase. Here two-fold division has been followed in both the phases. For global search, half of the population is updated based on the Eq. (13) and the next half is updated based on the Eq. (9). As towards the later stages, more exploitation is the requirement, so here the Eq. (7) has been modified by using the difference of best solution with a random solution. This will create copies of the candidate solution around the current best solution and hence will add to the exploitative tendencies. The modified equation is given by

$$x_i^{t+1} = x_i^t + \alpha \otimes Cauchy(\delta)\left(x_{best} - x_i^t\right) \tag{13}$$

Both the solutions from Eqs. (9) and (13) are then combined to generate a new solution. For local search phase, Eqs. (2) and (12) have been used to evaluate the new solution. Also for the global search phase, Cauchy based equation will ensure extensive exploration for the initial set of the population whereas Eq. (9) will ensure intensive exploitation. The justification for local search equation has been elaborated in Section 3.1. The pseudo-code for CV 2.0 is given in Algorithm 3.

4.3. Cuckoo Version 3.0

The use of Cauchy based mutation operator in the first half of the iterations has added to the explorative capabilities of basic CS and it is required to maintain a balanced exploitation also. So, for this case, in order to balance the exploitative tendencies of CS, a four-fold division of population in the global search phase is followed. By four-fold division, we mean that the population is divided into four parts and each part is evaluated by a different search equation. The first part is evaluated using Eq. (9), second part by using Eq. (12), third by using Eq. (13) and the fourth equation is given by

$$x_i^{t+1} = x_i^t + b.\left(x_i^t - x_j^t\right) \tag{14}$$

where b is a uniformly distributed random number in the range of [0, 1]. The local search phase is kept same as that of the original CS. The four-fold division strategy will help the population to be selected from four different smaller groups. This will add diversity to each group and will enhance the exploitation capabilities. So here again, a balance of exploration and exploitation can be achieved. The pseudo code for CV 3.0 is given in Algorithm 4.

4.4. Complexity analysis of proposed versions

The computational complexity of the original CS algorithm can be given by $O(n.D.\boldsymbol{t_{max}})$ where n is the maximum number of iterations, D is the dimension size and $\boldsymbol{t_{max}}$ is the maximum number of iterations. The main aim of providing this complexity analysis is to analyse the basic operations of CS and the proposed versions, and to summarize their worst case complexities. The equation is derived based on the fact that the algorithm is initialized with a population of size n and the dimension factor (D) decides the complexity of the algorithm. For each population member, we need to perform $O(D)$ number of operations, resulting in O (n. D) complexity. The above mentioned complexity is for a single iteration, but in general CS runs for a number of iterations. So the total complexity depends on the maximum number of iterations. This procedure gives the total complexity of CS as $O(n.D.\boldsymbol{t_{max}})$. Compared to the original CS algorithm, all the proposed versions have no extra computational burden. It is also evident from the comparison results in Section 5 that for variable population sizes, results for all the proposed versions along with CS algorithm change and same is the effect of dimension size. Also, all the new parameters added to the proposed version choose very small random values and have no effect on the space or time complexity. Therefore, the overall time complexity of all the proposed versions is almost same as the original CS algorithm.

5. Result and discussion

In this section, a detailed analysis of proposed versions of CS is presented to check whether they improve the performance of basic CS or not. Here, an experimental study of proposed versions with respect to standard CS is presented for benchmark test problems and then the best among the proposed versions are compared with state-of-the-art algorithms. This section is divided into nine consecutive subsections. The first subsection gives details about the benchmark functions and the parameter settings of various algorithms for experimental testing. The second subsection deals with the first parameter that is a probability. This parameter is very important in deciding the extent of exploration and exploitation and on a whole adds a balance between the two. So a proper value of probability needs to be identified for the algorithm to be efficient. From the literature, it has been found that the value of probability mostly lies in the range of 0 to 1 and a value of 0.25 gives best results for some problems while the value of 0.50 for others. So here five different sets of probabilities are used and experimental analysis is done to prove which set gives the best results. After the selection of a particular probability, the next work is to identify a proper population size. So in the third subsection, five different population sizes are used and results of CS are compared with the proposed variants. The main focus here is to use a minimal number of function evaluations and to achieve maximum output. Since we know that the maximum number of function evaluations is equal to the multiplication of population size and the total number of iterations, so to keep the number of function evaluations fixed, the total iteration count and population size are kept same for all the algorithms. After finding the exact population size and probability for the proposed variants, the best variant is identified and is subjected to comparative analysis with respect to other state-of-the-art algorithms in the fourth subsection. These comparative results are also subjected to statistical testing in the fifth subsection and best variant is identified. The best variant as obtained is also tested on higher dimensional set to prove its worthiness in the sixth subsection. The seventh subsection deals with another dataset of CEC 2015 benchmark problems for further analysis of the applicability of proposed version to highly complex benchmark problems. Note that the statistical testing is also performed on CEC 2015 benchmark functions as well. Apart from these experimental results, a sensititvity study of the parameters of proposed versions has also been added in the eight subsection and finally, the summary of results is presented in final subsection. Summary of results include the main findings of the research, insightful implications, limitations and some future prospects. The detailed discussion is presented in the consecutive subsections.

For performance testing, 100 runs have been performed and results are presented in terms of best of 100 runs, worst of 100 runs, mean of 100 runs, standard deviation of 100 runs and Wilcoxon rank-sum test for statistical analysis. Apart from this, all the algorithms are experimentally tested on Dell Inspiron 1464 with Intel core i3 processor having 4.00GB RAM, using MATLAB version R2016a in windows 10, x64 bit personal computer.

5.1. Test suite and parameter settings

5.1.1. Test suite

This section deals with the test suite used to validate the performance of proposed versions of CS. Three types of test functions namely unimodal, multimodal and fixed dimensional functions are used. Twenty-four functions have been used, with eight unimodal, ten multimodal, and five fixed dimension functions. Unimodal functions define how efficient an algorithm is in exploiting the search space. They usually have only one global minimum and are simple functions. Multimodal functions, on the other hand, have many local minima and these are used to judge the exploratory power in any algorithm. The third set of functions which are fixed dimensional functions have fixed dimension size and these functions define the consistency of any algorithm in

Table 1
Description of test problems.

Test Problems	Objective Function	Search Range	Optimum Value	D
Unimodal Functions				
Sphere function	$f_1(x) = \sum_{i=1}^{D} x_i^2$	[−100, 100]	0	30
Step function	$f_2(x) = \sum_{i=1}^{D} (\lvert x_i + 0.5 \rvert)^2$	[−100, 100]	0	30
Quartic function	$f_3(x) = \sum_{i=1}^{D} x_i^4 + random(0, 1)$	[−1.28, 1.28]	0	30
Tablet function	$f_4(x) = 10^6 x_i^2 + \sum_{i=1}^{D} x_i^2$	[−5, 5]	0	30
Sum of different powers function	$f_1(x) = \sum_{i=1}^{D} x_i^{i+1}$	[−1, 1]	0	30
Elliptic function	$f_6(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	[−100, 100]	0	30
Scaffer function	$f_7(x) = [\frac{1}{n-1} \sqrt{s_i}.(\sin(50.0 s_i^{\frac{1}{5}}) + 1)]^2 s_i = \sqrt{x_i^2 + x_{i+1}^2}$	[−100, 100]	0	30
Schwefel function	$f_8(x) = \sum_{i=1}^{D} [x_i sin(\sqrt{\lvert x_i \rvert})]$	[−500, 500]	−418.9829 × D	30
Multimodal Functions				
Ackley function	$f_9(x) = -20 \exp(-0.2\sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2}) - \exp\ (\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i)) + 20 + e$	[−100, 100]	0	30
Powell function	$f_{10}(x) = \sum_{i=1}^{d} [(x_{4i-3} - 10 x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^2]$	[−4, 5]	0	30
Dixon & Price function	$f_{11}(x) = (x_1 - 1)^2 + \sum_{i=2}^{d} i(2x_i^2 - x_{i-1})^2$	[−10,10]	0	30
Zakharov function	$f_{12}(x) = \sum_{i=1}^{d} x_i^2 + (\sum_{i=1}^{d} 0.5 i x_i)^2 + (\sum_{i=1}^{d} 0.5 i x_i)^4$	[−5, 10]	0	30
Cigar function	$f_{13}(x) = x_0^2 + 10000 \sum_{i=1}^{D} x_i^2$	[−10, 10]	0	30
Griewank Function	$f_{14} = \frac{1}{4000} \sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos(\frac{x_i}{\sqrt{i}}) + 1$	[−600, 600]	0	30
Penalized 1 Function	$f_{15} = \frac{\pi}{n} \{10\sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 sin^2(\pi y_{i+1})] + (y_n - 1)2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)\ y_i = 1 + \frac{x_{i+1}}{4}$ $u(x_i, a, k, m) = \{ \begin{matrix} k(x_i - a)^m x_i > a \\ 0\ -a < x_i < a \\ k(-x_i - a)^m\ x_i < -a \end{matrix}$	[−50, 50]	0	30
Penalized 2 Function	$f_{16} = 0.1\{(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$ $u(x_i, a, k, m) = \{ \begin{matrix} k(x_i - a)^m x_i > a \\ 0\ -a < x_i < a \\ k(-x_i - a)^m\ x_i < -a \end{matrix}$	[−50, 50]	0	30
Rastrigin function	$f_{17}(x) = 10D + \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i)]$	[−5.12, 5.12]	0	30
Weierstrass function	$f_{18}(x) = \sum_{i=1}^{D} \sum_{k=0}^{kmax} [a^k \cos(2\pi b^k (x_i + 0.5))] - D \sum_{k=0}^{kmax} [a^k \cos(2\pi b^k .0.5)];$ *where* $a = 0.5,\ b = 3,\ kmax = 20$	[−0.5, 0.5]	0	30
Fixed dimension functions				
Beale function	$f_{19}(x) = [1.5 - x_1(1 - x_2)]^2 + [2.25 - x_1(1 - x_2^2)]^2 + [2.625 - x_1(1 - x_2^3)]^2$	[−4.5, 4.5]	0	2
Hartmann function 3	$f_{20}(x) = -\sum_{i=1}^{4} \alpha_i \exp[-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2]$	[0, 1]	−3.86278	3
Hartmann function 6	$f_{21}(x) = -\sum_{i=1}^{4} \alpha_i \exp[-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2]$	[0, 1]	−3.32237	6
Six Hump Camel function	$f_{22}(x) = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1 x_2 + (-4 + 4x_2^2)x_2^2$	[−5, 5]	−1.0316	2
Michalewicz	$f_{23}(x) = -\sum_{i=1}^{D} \sin(x_i) sin^{2m}(\frac{i x_i^2}{\pi})$	[0, π]	−9.6601	10
Easom function	$f_{24}(x) = -cos x_1 cos x_2 e^{(-(x_1-\pi)^2 - (x_2-\pi)^2)}$	[−10, 10]	−1	2

finding global minima or the final best solution. The description of test functions is given in Table 1.

5.1.2. Parameter settings

This subsection gives the details about the various algorithms used for comparison. The algorithms used are CS, GWO, FA, DE, BA, FPA, BFP, particle swarm optimization-gravitational search optimization (PSOGSA) (Mirjalili & Hashim, 2010) and the proposed variants CV 1.0, CV 2.0 and CV 3.0. The parameter settings for all the algorithms used for comparison purpose are obtained from their respective papers. As far as CS and proposed versions are concerned, only switch probability is to be defined. For an initial guess the switching probability is taken randomly but after computing the effect of probability switch (the detailed sensitivity study is presented in Section 5.3), it has been found that a probability value of 0.5 fits better for most of the cases. Also in order to make a fair comparison, the original CS is executed with the same parameter settings as done with the proposed versions. Moreover, the parameter values such as population size and a maximum number of iterations are chosen after careful testing and the present set of val-

Table 2
Parameter settings of various algorithms.

Algorithm	Parameters	Values
CS	Switch probability	0.5
	Max Iteration	500
	Stopping Criteria	Max Iteration.
CV 1.0, CV 2.0 & CV 3.0	$\vec{a}$	Linearly decreased from 2 to 0
	Switch probability	0.5
	Maximum Cycles	500
	Stopping Criteria	Max Iteration.
GWO	$\vec{a}$	Linearly decreased from 2 to 0.
	Max Iteration	500
	Stopping Criteria	Max Iteration.
FA	Number of fireflies	20
	Gamma (γ)	1
	Max Iteration	500
	Stopping Criteria	Max Iteration.
DE	F	0.5
	CR	0.5
	Max Iteration	500
	Stopping Criteria	Max Iteration.
BA	Loudness	0.5
	Pulse rate	0.5
	Max Iteration	500
	Stopping Criteria	Max Iteration.
FPA	Probability Switch	0.8
	Max Iteration	500
	Stopping Criteria	Max Iteration.
BFP	Population size	20
	Loudness	0.25
	Pulse rate	0.25
	Probability switch	0.8
	Max Iteration	500
	Stopping criteria	Max Iteration.
PSOGSA	Population size	20
	C1 & C2 (weighting factors)	2
	Weighting function (w)	[0, 1]
	Gravitational Constant (G)	1
	Alpha (α)	20
	Max Iteration	500
	Stopping criteria	Max Iteration.

ues fit better for most of the cases. This is because lower values of population size and iterations degrade the performance whereas higher values improve the performance marginally but a maximum number of function evaluations increases many folds. This point has been proved experimentally in the effect of population subsection (Section 5.4).The various parameters selected for different algorithms are given in Table 2. These parameters are chosen from the existing literature of the respective algorithms. For GWO (Mirjalili et al., 2014), parameter $\vec{a}$, is of major concern. This parameter gives the best results for linearly decreasing values while rest of the parameters are random numbers. For CS and proposed variants, switch probability p_a is a very important factor which decides the extent of exploration and exploitation. This parameter has been chosen after careful investigation and detailed results are presented in the Section 5.2. For FA, there are three control parameters: the randomization parameter (α), the attractiveness (β) and the absorption coefficient (γ). The parameters α and β are simple random numbers in the range of [0, 1] and pose little effect on the performance of FA. These parameters generally decide the type of random steps taken by FA in evaluating the solution. The final parameter γ is a very crucial parameter and it decides the convergence speed of FA and its value lies in the range of 0.1–10 (Fister, Fister, Yang, & Brest, 2013). Further, in Yang (2010b,c) it has been given that a value of ‘1’ for parameter γ is suitable for most of the cases. So for present case, $\gamma = 1$ is chosen. In case of DE algorithm, F and CR are two major parameters and numerous studies have been proposed to find an optimal set of these parameters. The values of these parameters generally lie in the range of [0, 1] but Wang, Cai, and Zhang (2011) found that a high value of F = 1.0 and CR = 0.9, improve the exploration capabilities of DE whereas high values of F = 0.8 and low value of CR = 0.2 help in improving the exploitation capabilities. Also Draa, Bouzoubia, and Boukhalfa (2015) estimated that a value of 0.5 for F and [0.5–0.9] for CR will provide better results for most of the cases. So in present case, the values of both CR and F are kept as 0.5. For BA, pulse rate and loudness are two important parameters and in Fister, Fister, and Yang, (2013) it has been found that a value of 0.5 for both pulse rate and loudness performs better for most of the cases. So in present case, it has been set to 0.5 for both loudness and pulse rate. For FPA, probability switch is the only parameter which decides the rate of exploration and exploitation or in simple terms, probability decides the performance of FPA. In Yang (2012) it has explicitly written that a probability value of 0.8 is best for most of the cases and hence has been used as such in the present scenario. For BFP, because of its hybrid structure, it has both the parameters of BA and FPA as important parameters, so the same criterion has been followed to set the parameters as used in BA and FPA. The original article of BFP (Salgotra & Singh, 2016) also provided proper explanation about the various parameter settings and same has been used in present work. For the final algorithm that is PSOGSA, the parameter setting is kept same as used by the original article (Mirjalili & Hashim, 2010).

Table 3
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS at $\boldsymbol{p} = 0.05$.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	27.2652	4.30E+02	1.07E+01	61.0355
	CV 1.0	3.23E−87	1.39E−70	3.51E−72	1.97E−71
	CV 2.0	2.21E−88	2.86E−72	8.05E−74	4.43E−73
	CV 3.0	5.75E−80	4.68E−47	9.81E−49	6.42E−48
f_2	CS	32	6.39E+02	2.00E+02	1.07E+02
	CV 1.0	0	0	0	0
	CV 2.0	0	0	0	0
	CV 3.0	0	0	0	0
f_6	CS	3.67E+04	4.83E+05	1.13E+05	6.91E+04
	CV 1.0	2.83E−82	2.82E−68	6.51E−70	3.57E−69
	CV 2.0	1.18E−90	7.32E−65	7.35E−67	7.32E−66
	CV 3.0	7.86E−81	5.60E−45	8.44E−47	6.01E−46
f_8	CS	−8.61E+03	−7.12E+03	−7.70E+03	2.60E+02
	CV 1.0	−8.79E+03	−6.80E+03	−7.67E+03	3.98E+02
	CV 2.0	−8.12E+03	−6.85E+03	−7.52E+03	2.75E+02
	CV 3.0	−8.43E+03	−6.90E+03	−7.50E+03	3.29E+02
f_{14}	CS	0.00E−00	1.55E−15	1.88E−17	1.58E−16
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{15}	CS	1.1798	25.0136	7.4155	4.4720
	CV 1.0	0.0093	12.1838	3.1391	3.3608
	CV 2.0	0.0039	10.0282	2.3183	3.0204
	CV 3.0	0.0049	16.9770	5.4412	5.6506
f_{16}	CS	18.214	2.25E+04	1.36E+03	3.61E+03
	CV 1.0	0.1080	0.9222	0.4408	0.1833
	CV 2.0	0.1072	0.5905	0.2729	0.0991
	CV 3.0	0.0746	52.791	1.2674	7.0507
f_{17}	CS	0.00E−00	3.41E−13	6.03E−15	3.54E−14
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{18}	CS	8.4077	18.9194	13.356	1.9902
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	1.42E−14	0.42E−16	1.42E−15
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{20}	CS	−3.8628	−3.8628	−3.8628	1.01E−09
	CV 1.0	−3.8628	−3.8628	−3.8628	4.11E−15
	CV 2.0	−3.8628	−3.8628	−3.8628	6.53E−15
	CV 3.0	−3.8628	−3.8628	−3.8628	3.27E−15
f_{21}	CS	−3.3224	−3.2032	−3.3176	0.0059
	CV 1.0	−3.3224	−3.3224	−3.3224	6.38E−09
	CV 2.0	−3.3224	−3.2032	−3.3200	7.71E−08
	CV 3.0	−3.3224	−3.3224	−3.3224	0.0119
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.11E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	1.07E−15
	CV 2.0	−1.0316	−1.0316	−1.0316	1.42E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	1.11E−15

5.2. Effect of switch probability

In the literature, it can be seen that some detailed works have been done on evaluation of proper switch probability but no proper justification about how the study was done is present. In the present work, the effect of switch probability (p) has been analyzed using different values of p. For analysis of these values, 13 benchmark functions from Table 1 are used. Here four unimodal (f_1, f_3, f_6 *and* f_8), five mulit-modal ($f_{14}, f_{15}, f_{16} f_{17}$ *and* f_{18}) and four fixed dimensional functions (f_{20}, f_{21} *and* f_{22}) are used. The population size of 20 and the total number of iterations of 500 are considered for the purpose of comparison. In Tables 3, 4, 5, 6 and 7, comparison results for $p = 0.05,\ 0.25,\ 0.50,\ 0.75$ *and* 0.95 respectively, are reported. The results are discussed as below:

- *Comparison of Best values:* From the best values of all the tables, it can be seen that for f_1, CS was not able to reach global optimum while all other proposed versions provided a very near optimum solution with CV 2.0 providing the best among all at $p = 0.75$. For f_2 and f_{18} CS was not able to reach global optimum whereas all the proposed versions provided exact global optimum. For f_6, again CV 2.0 was found to be the best for $p = 0.75$. For $f_8$all the algorithms provided competitive results. For f_{14} and f_{17}, all the algorithms provided exact global optima for $p = 0.05,\ 0.25$ *and* 0.50 and for $p = 0.75$ *and* 0.95, CS was not able to achieve exact minima while for others it was very difficult to analyse which one is the best. For f_{15} and f_{16} CS is found to be the worst among all while all others provided very competitive results with CV 3.0 providing best results for f_{15} at $p = 0.75$ and for f_{16} at $p = 0.50$. For $f_{20,}$ f_{21} and f_{22} all the algorithms provided the same optimums.
- *Comparison of Worst values:* From worst values of all the tables, it can be said that for f_1 and $f_{6,}$ CS was found to be the worst and all the proposed algorithms were competitive with CV 2.0 providing the best among all at $p = 0.75$ for both the functions. For f_2 and f_{18} all the proposed versions provided exact global optimum except CS. For f_8 all the algorithms provided competitive results. For f_{14} and f_{17}, all the algorithms were very competitive for $p = 0.05,\ 0.25$ *and* 0.50 and for $p = 0.95$, CS was not able to achieve exact minima while for others it was very difficult to analyse which one is the best. For f_{15} and f_{16} CS provided the worst results and all others provided very competi-

Table 4
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS at $\boldsymbol{p} = 0.25$.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	1.6842	47.6422	10.2935	7.9390
	CV 1.0	4.28E−85	3.12E−72	4.21E−74	3.14E−73
	CV 2.0	2.32E−93	2.57E−69	2.92E−71	2.60E−70
	CV 3.0	6.86E−80	2.58E−46	9.34E−48	2.82E−47
f_2	CS	7	29.9300	15.1172	1.05E+01
	CV 1.0	0	0	0	0
	CV 2.0	0	0	0	0
	CV 3.0	0	0	0	0
f_6	CS	2.86E+03	7.08E+04	1.77E+04	1.06E+04
	CV 1.0	4.50E−82	1.66E−68	4.62E−70	2.52E−69
	CV 2.0	2.18E−93	3.55E−67	7.03E−69	4.70E−68
	CV 3.0	9.40E−70	1.97E−42	2.46E−44	2.00E−43
f_8	CS	−9.01E+03	−7.37E+03	−7.95E+03	2.93E+02
	CV 1.0	−9.03E+03	−6.73E+03	−7.75E+03	3.84E+02
	CV 2.0	−8.17E+03	−6.66E+03	−7.51E+03	3.12E+02
	CV 3.0	−8.59E+03	−6.91E+03	−7.67E+03	3.88E+02
f_{14}	CS	0.00E−00	4.40E−13	1.22E−14	6.24E−14
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{15}	CS	1.2670	12.9524	4.8678	2.1996
	CV 1.0	0.0071	6.0306	1.0298	1.7926
	CV 2.0	0.0063	12.474	2.1431	3.1386
	CV 3.0	0.0017	8.7164	1.9770	2.2416
f_{16}	CS	4.6216	1.61E+01	23.8676	19.1747
	CV 1.0	0.0731	1.1738	0.4483	0.1986
	CV 2.0	0.0686	0.6458	0.2828	0.1279
	CV 3.0	0.0369	0.6275	0.1718	0.1003
f_{17}	CS	0.00E−00	2.38E−13	1.38E−14	4.36E−14
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{18}	CS	7.1970	16.8691	11.0874	2.0709
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{20}	CS	−3.8628	−3.8628	−3.8628	5.92E−15
	CV 1.0	−3.8628	−3.8628	−3.8628	6.60E−15
	CV 2.0	−3.8628	−3.8628	−3.8628	6.63E−15
	CV 3.0	−3.8628	−3.8628	−3.8628	6.70E−15
f_{21}	CS	−3.3224	−3.3224	−3.3224	1.83E−04
	CV 1.0	−3.3224	−3.3224	−3.3224	1.00E−08
	CV 2.0	−3.3224	−3.3224	−3.3224	5.84E−08
	CV 3.0	−3.3224	−3.3224	−3.3224	1.61E−07
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.11E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	1.09E−15
	CV 2.0	−1.0316	−1.0316	−1.0316	1.48E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	1.11E−15

tive results with CV 1.0 providing best results for f_{15} at $p = 0.95$ and CV 3.0 for f_{16} at $p = 0.50$. For f_{21}, the results of CS and CV 3.0 deviated to some extent from global optima at $p = 0.95$ and for f_{20} and f_{22} all the algorithms provided the same optimums.

- *Comparison of Mean values:* It can be seen from the tables that mean values give almost the same results as given by the best and worst cases. Also for mean values of CS only, it gives worst mean value at $p = 0.05$ while at $p = 0.25,\ 0.50$ *and* 0.75 the results are best and again for $p = 0.95$, the results become worse except for f_{14} and f_{17}, where best values at $p = 0.05$. For fixed dimension function $f_{21,}$ the mean value is not exactly the global optima for $p = 0.95$ and for other cases all fixed dimensional functions provide exact global optima. The results for all the proposed variants don't deviate much for any value of p.
- *Comparison of standard deviation values:* From the standard deviation section in the tables, it can be seen that for f_1 and f_6 CV 2.0 provides the best results at $p = 0.95$. For f_2, f_{14}, f_{17} and f_{18} CS provides the worst results while from the proposed variants, all the proposed variants provide the exact zero standard deviation. For f_8 the results are competitive. For $f_{15}$and f_{16}, CV 1.0 provides best result at $p = 0.95$ and CV 3.0 at $p = 0.50$ respectively. For all the fixed dimension problems that is for $f_{20,}$ f_{21} and f_{22} CS is found to be highly competitive for all the cases but still CV, 1.0 was found to be the best among all. Also for these functions, it should be noted here that CV 1.0 provided best results for all the values of p.
- *Inference drawn:* Overall from the results of all the five tables, it can be seen that for most of the cases, the performance of CS is worst for $p = 0.05$ while for $p = 0.25,\ 0.50$ *and* 0.75 its performance is better and for $p = 0.95$, the results degrade. This proves that performance of CS depends upon the value of p but when we compare it with the proposed versions, there is either no change in performance or is modest. Hence, we can say that the proposed techniques have no effect of p on their performance.

5.3. Effect of population

In meta-heuristic algorithms, it has been seen that the number of function evaluations depends on the population size and iterations. For a fixed number of iterations, smaller is the population size, less is the number of function evaluations required to achieve

Table 5
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS at $\boldsymbol{p} = 0.50$.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	2.2151	14.0305	6.2216	2.4490
	CV 1.0	4.66E−86	4.96E−73	6.74E−74	4.96E−73
	CV 2.0	9.61E−93	8.23E−72	1.30E−72	8.23E−72
	CV 3.0	1.28E−73	3.78E−48	4.84E−49	3.78E−48
f_2	CS	9	36	16.9700	5.1902
	CV 1.0	0	0	0	0
	CV 2.0	0	0	0	0
	CV 3.0	0	0	0	0
f_6	CS	4.72E+03	3.17E+04	1.42E+04	5.44E+03
	CV 1.0	8.70E−83	3.71E−68	7.55E−70	4.25E−69
	CV 2.0	7.88E−87	7.66E−68	8.38E−70	7.68E−69
	CV 3.0	1.68E−69	1.39E−41	1.84E−43	1.44E−42
f_8	CS	−8.82E+03	−7.33E+03	−8.18E+03	2.77E+02
	CV 1.0	−9.07E+03	−7.21E+03	−7.97E+03	3.95E+02
	CV 2.0	−8.37E+03	−6.78E+03	−7.52E+03	3.42E+02
	CV 3.0	−9.17E+03	−7.05E+03	−7.94E+03	3.80E+02
f_{14}	CS	4.44E−16	5.70E−08	1.26E−09	6.94E−09
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{15}	CS	1.4688	11.167	3.6256	1.2798
	CV 1.0	0.0063	8.9164	0.2786	1.1849
	CV 2.0	0.0076	13.901	2.6156	3.5781
	CV 3.0	0.0024	4.2793	0.1813	0.5352
f_{16}	CS	2.0356	8.6997	7.7810	4.6404
	CV 1.0	0.1574	0.9189	0.4142	0.1548
	CV 2.0	0.0872	1.4004	0.2930	0.1470
	CV 3.0	0.0298	0.3737	0.1420	0.0793
f_{17}	CS	2.50E−10	5.67E−11	6.13E−12	2.90E−11
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{18}	CS	7.1719	15.7463	11.6801	1.5381
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	2.13E−14	2.13E−16	2.13E−15
f_{20}	CS	−3.8628	−3.8628	−3.8628	3.61E−13
	CV 1.0	−3.8628	−3.8628	−3.8628	6.40E−15
	CV 2.0	−3.8628	−3.8628	−3.8628	6.65E−15
	CV 3.0	−3.8628	−3.8628	−3.8628	4.09E−13
f_{21}	CS	−3.3224	−3.3223	−3.3224	6.31E−06
	CV 1.0	−3.3224	−3.3224	−3.3224	1.11E−08
	CV 2.0	−3.3224	−3.3224	−3.3224	3.72E−08
	CV 3.0	−3.3224	−3.3224	−3.3224	3.01E−07
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.25E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	8.42E−14
	CV 2.0	−1.0316	−1.0316	−1.0316	7.93E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	1.76E−15

the global optima and greater is the population size more are the function evaluations. Here five set of population sizes namely 20, 40, 60, 80 and 100 are used to analyze the effect of population on the performance of CS and the proposed approaches. The total number of iterations is kept fixed at 500. As seen in the previous section, the value of $p = 0.25,\ 0.50\ and\ 0.75$ provides comparable results for CS algorithm. For the proposed version, the value of p has no effect on their performance. So, in this section we have used $p = 0.50$ as the standard value for analysing the effect of population. Thirteen test functions from Table 1 are selected to check the effect of population size on the performance of proposed variants and the standard CS. These functions are unimodal ($f_1, f_3, f_4\ and\ f_6$), multimodal ($f_9, f_{13}, f_{15} f_{16}\ and\ f_{18}$) and fixed dimension functions ($f_{19}, f_{20}, f_{21}\ and\ f_{22}$).

- *Population size 40:* In Table 8, the simulation results for population size 40 are presented and it can be seen that for f_1, CS is not able to attain global minima while all the proposed versions provided very good results and among them, CV 2.0 is the best. In f_3, CV 1.0 provides better optimum results. For rest of the variants, the results are competitive. For f_4, CV 2.0 gives the best results while all other variants give quite competitive results. For f_6, CS gives the worst results while other are still competitive. The best algorithm, in this case, is the CV 2.0. For f_9, CV 1.0, CV 2.0 and CV 3.0 provide the same best and worst and the comparison is in terms of mean and standard deviation where CV 1.0 is found to be the best. For f_{13}, CV 2.0 provides the best results. For f_{15} and f_{16} it is found that CV 1.0 and CV 3.0 gave best results respectively. For f_{18}, the values of best, worst, mean and standard deviation are same for all the proposed variants. Here it is difficult to comment which one is better. For function f_{19}, CV 2.0 is highly competitive with CV 1.0 for best values where as in terms of worst, mean and standard deviation, CV 1.0 is the best. So here CV 1.0 can be considered as the best. For f_{20} and f_{21}, CV 1.0 gives the best standard deviation while for f_{22}, CV 3.0 is the best. In this case, CS is found to be better for no function, CV 1.0 for seven functions, CV 2.0 for five functions and CV 3.0 for three functions. So, overall CV 1.0 is found to be the best among all the proposed variants for the population size of 40.
- *Population size 60:* The simulation results for population size 60 are found in Table 9 and it can be seen that or f_1, f_4, f_6 and f_{13}

Table 6
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS at $\boldsymbol{p} = 0.75$.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	5.3774	19.6146	12.3785	3.4401
	CV 1.0	7.80E−85	2.75E−70	1.85E−72	2.79E−71
	CV 2.0	1.97E−93	1.80E−68	2.07E−70	1.81E−69
	CV 3.0	3.61E−78	6.70E−46	6.78E−48	6.70E−47
f_2	CS	13	48	27.6000	6.4980
	CV 1.0	0	0	0	0
	CV 2.0	0	0	0	0
	CV 3.0	0	0	0	0
f_6	CS	1.69E+04	9.62E+04	4.71E+04	1.62E+04
	CV 1.0	1.32E−81	5.90E−65	5.92E−67	5.90E−66
	CV 2.0	7.79E−88	7.76E−69	8.14E−71	7.76E−70
	CV 3.0	2.69E−71	1.64E−39	1.83E−41	1.65E−40
f_8	CS	−9.52E+03	−8.36E+03	−8.81E+03	2.14E+02
	CV 1.0	−9.55E+03	−7.39E+03	−8.31E+03	3.75E+02
	CV 2.0	−8.12E+03	−6.85E+03	−7.52E+03	2.75E+02
	CV 3.0	−9.79E+03	−7.92E+03	−8.71E+03	3.32E+02
f_{14}	CS	2.19E−10	1.62E−04	5.21E−06	1.91E−05
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{15}	CS	1.1940	4.3962	2.8984	0.5593
	CV 1.0	0.0038	0.1288	0.0208	0.0141
	CV 2.0	0.0071	9.6460	2.4805	3.1419
	CV 3.0	0.0019	0.6402	0.0335	0.0889
f_{16}	CS	1.8313	7.6083	4.5154	1.1883
	CV 1.0	0.1158	1.0545	0.4355	0.2003
	CV 2.0	0.1454	0.5364	0.2730	0.0828
	CV 3.0	0.0410	0.4156	0.1220	0.0606
f_{17}	CS	4.61E−14	1.56E−06	5.77E−08	2.25E−07
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{18}	CS	6.7923	15.7841	11.3448	1.3619
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	7.10E−15	7.10E−17	7.10E−16
f_{20}	CS	−3.8628	−3.8628	−3.8628	1.89E−08
	CV 1.0	−3.8628	−3.8628	−3.8628	1.96E−14
	CV 2.0	−3.8628	−3.8628	−3.8628	6.66E−15
	CV 3.0	−3.8628	−3.8628	−3.8628	6.28E−10
f_{21}	CS	−3.3224	−3.3224	−3.3224	2.61E−06
	CV 1.0	−3.3224	−3.3224	−3.3224	6.53E−09
	CV 2.0	−3.3224	−3.3224	−3.3224	1.48E−07
	CV 3.0	−3.3224	−3.3224	−3.3224	6.04E−07
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.38E−12
	CV 1.0	−1.0316	−1.0316	−1.0316	1.58E−13
	CV 2.0	−1.0316	−1.0316	−1.0316	1.18E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	8.00E−12

CV 2.0 provides the best results in terms of best, worst, mean and even standard deviation. In f_3 CV 1.0 provides better optimum results. For rest of the variants, the results are competitive. For f_9, CV 1.0, CV 2.0 and CV 3.0 provide the same best and worst and the comparison is in terms of mean and standard deviation. In this case, CV 1.0 is found to be the best. For f_{15} and f_{16} it is found that CV 1.0 gave best results. For f_{18}, the values of best, worst, mean and standard deviation are exactly zero for all the proposed variants and no one can be termed as the best. For f_{19}, CV 1.0 is better. For f_{20} and f_{21}, CV 1.0 gives the best standard deviation and for f_{22}, CV 3.0 gives the best results. In this case, CV 1.0 for eight functions, CV 2.0 for five functions, CV 3.0 for two functions and CS is not better for any function. So, overall CV 1.0 is found to be the best among all the proposed variants for the population size of 60.

- *Population size 80:* The results for this case are presented in Table 10. For most of the functions, the results are similar to that of the previous case of population size 60. For this case, CV 1.0 is better for eight functions, CV 2.0 for five functions, CV 3.0 for two functions and CS for none. Hence again CV 1.0 is better for this population size also.
- *Population size 100:* Simulation results for population size 100 are presented in Table 11. Here also the results are similar to population size 60. For this case, we find that CS is found to be better for none, CV 1.0 for eight functions, CV 2.0 for five functions and CV 3.0 for two functions. This case also shows that CV 1.0 is better for 100 population size.
- *Inference drawn:* It can be seen from the Table 5, 8, 9, 10 and 11 that for variable population size, the change in the results is large for 20 population size and as we move towards higher population sizes, the change is only moderate. This change is seen in only some of the unimodal and multi-modal functions namely f_1, f_4, f_6, and f_{13}. But for rest of the test functions, there are modest changes in the results. Overall, we can say that the results don't vary too much and thus any population size can be taken as standard population size. Here for a fixed number of iterations, if we double the population size, the number of function evaluations also doubles. So, we can say that if the number of iterations is 500 and population size is 20, the number of function evaluations will be 10,000, for population size of 40, maximum number of function evaluations becomes 20,000 and for the other three cases of 60, 80 and 100

Table 7
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS at $\boldsymbol{p} = 0.95$.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	84.0617	2.95E+02	1.47E+02	42.1674
	CV 1.0	1.15E−86	3.56E−70	4.33E−72	3.57E−71
	CV 2.0	5.65E−94	1.78E−68	1.79E−70	1.78E−69
	CV 3.0	2.50E−78	1.36E−43	1.37E−45	1.36E−44
f_2	CS	69	394	2.08E+02	58.9524
	CV 1.0	0	0	0	0
	CV 2.0	0	0	0	0
	CV 3.0	0	0	0	0
f_6	CS	2.01E+05	1.85E+06	8.85E+05	3.53E+05
	CV 1.0	1.73E−87	2.68E−68	8.57E−70	3.56E−69
	CV 2.0	2.80E−91	7.10E−69	1.49E−70	8.67E−70
	CV 3.0	1.39E−76	2.24E−46	2.85E−41	2.30E−45
f_8	CS	−1.03E+04	−9.31E+03	−9.79E+03	2.20E+02
	CV 1.0	−1.02E+04	−8.01E+03	−9.00E+03	3.73E+02
	CV 2.0	−8.96E+03	−7.02E+03	−7.59E+03	3.13E+02
	CV 3.0	−1.06E+04	−9.19E+03	−9.78E+03	2.85E+02
f_{14}	CS	5.30E−05	0.1493	0.0529	0.0550
	CV 1.0	0.00E−00	0.1456	0.0058	0.0287
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{15}	CS	1.2179	3.6053	2.8151	0.5646
	CV 1.0	0.0053	0.1497	0.0215	0.0186
	CV 2.0	0.0066	3.6824	2.4880	3.6500
	CV 3.0	0.0022	1.0463	0.0214	0.0744
f_{16}	CS	4.1138	20.332	11.801	2.8387
	CV 1.0	0.0681	1.1885	0.4924	0.2127
	CV 2.0	0.0947	0.7503	0.2892	0.1100
	CV 3.0	0.0555	0.4843	0.1540	0.0683
f_{17}	CS	4.35E−05	0.5578	0.0435	0.0821
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
f_{18}	CS	8.8965	14.2200	11.8011	1.0094
	CV 1.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 2.0	0.00E−00	0.00E−00	0.00E−00	0.00E−00
	CV 3.0	0.00E−00	7.10E−15	1.42E−16	1.00E−15
f_{20}	CS	−3.8627	−3.8101	−3.8592	7.40E−03
	CV 1.0	−3.8628	−3.8628	−3.8628	1.68E−14
	CV 2.0	−3.8628	−3.8628	−3.8628	6.63E−15
	CV 3.0	−3.8628	−3.8627	−3.8628	1.71E−05
f_{21}	CS	−3.3224	−3.3179	−3.3219	4.72E−04
	CV 1.0	−3.3224	−3.3224	−3.3224	1.84E−07
	CV 2.0	−3.3224	−3.3223	−3.3224	3.66E−08
	CV 3.0	−3.3224	−3.3217	−3.3222	9.63E−05
f_{22}	CS	−1.0316	−1.0316	−1.0316	6.06E−08
	CV 1.0	−1.0316	−1.0316	−1.0316	5.04E−12
	CV 2.0	−1.0316	−1.0316	−1.0316	2.75E−10
	CV 3.0	−1.0316	−1.0316	−1.0316	4.21E−07

population sizes, the maximum number of function evaluations becomes 30,000, 40,000 and 50,000, respectively. But an algorithm providing best results by using a minimum number of function evaluations is always considered as the best one. For the present case, since all the test functions provide same comparative results, so we can choose a population size of 20 as the best population size.

5.4. Comparative study

From the above discussion, it is found that CV 1.0 is the best among all the proposed variants. It can be seen from the inferences in Section 5.3 that the optimum probability switch is taken to be 0.5 and from the inferences in Section 5.4, it is evident that population size of 20 is the best. The reason for no change in performance at higher populations is because of the division of iterations and population which helps in proper exploration and exploitation of the search space, helping the algorithm in achieving better results by employing a minimum number of function evaluations. Now to prove CV 1.0 to be state-of-the-art, it is compared with well-known algorithms. The major algorithms used in this paper are GWO, CS, DE, BA, FA, FPA and parameter setting for these algorithms is given in Table 2.

The simulation results for comparison are presented in Table 12. It can be seen from the table that for functions f_1 *and* f_{17} FA, GWO, CV 1.0, CV 2.0 and CV 3.0 are able to reach near optimum results while other algorithms got stuck in some local minima, for these function CV 2.0 is able to find the best values but in terms of worst, mean and standard deviation, CV 1.0 was found to be the best. So here we can say that CV 1.0 provides the best results for this case. For f_2 GWO, CV 2.0 and CV 1.0 have equal performance and it is not evident which algorithm is best among both. For f_3, f_4, f_5, f_{10}, f_{12} *and* f_{14} CV 1.0 is the best and GWO, CV 2.0 and CV 3.0 are only slightly competitive with respect to it. For function f_6, f_{13}, CV 2.0 is found to be the best. For function f_7, f_9 *and* f_{18} only the proposed algorithms are able to achieve global minima and no other algorithm is able to match their performance. For f_8, FA, CV 1.0, CV 2.0 and CV 3.0 are very competitive where as others provide highly deviated solutions. For f_{11}, only CV 1.0 achieves global optimal, rest of functions stuck in local minima. For f_{15} and f_{16}, CV 1.0, CV 2.0 and CV 3.0 are competitive but for these functions, FA provides the best results. For CV 1.0 provides the best results.

Table 8
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS for population size 40.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	9.3303	30.5712	18.0249	3.9600
	CV 1.0	5.07E−99	7.11E−83	7.12E−85	7.11E−84
	CV 2.0	**7.91E−105**	**2.54E−86**	**5.14E−88**	**3.30E−87**
	CV 3.0	1.64E−87	6.93E−64	6.98E−66	6.93E−65
f_3	CS	0.0675	0.2562	0.1398	0.0389
	CV 1.0	4.27E−05	**0.0050**	**0.0010**	**9.27E−04**
	CV 2.0	4.07E−05	0.0213	0.0024	0.0027
	CV 3.0	4.33E−04	0.0310	0.0059	0.0055
f_4	CS	0.04530	0.1642	0.0929	0.0232
	CV 1.0	2.00E−105	7.97E−88	1.35E−89	8.39E−89
	CV 2.0	**1.37E−107**	**1.20E−90**	**1.39E−92**	**1.20E−91**
	CV 3.0	3.47E−90	9.88E−67	9.89E−69	9.88E−68
f_6	CS	1.43E+04	6.94E+04	3.84E+04	1.05E+04
	CV 1.0	1.19E−95	1.75E−80	1.87E−82	1.75E−81
	CV 2.0	**5.39E−102**	**8.65E−83**	**8.78E−85**	**8.65E−84**
	CV 3.0	6.58E−85	3.61E−64	6.40E−66	4.21E−65
f_9	CS	7.1409	15.4349	10.6564	1.7526
	CV 1.0	8.88E−16	4.44E−15	**1.38E−15**	**1.23E−15**
	CV 2.0	8.88E−16	4.44E−15	1.63E−15	1.45E−15
	CV 3.0	8.88E−16	4.44E−15	2.20E−15	1.72E−15
f_{13}	CS	1.79E+02	8.11E+02	3.81E+02	95.1921
	CV 1.0	5.26E−100	9.97E−85	2.02E−86	1.36E−85
	CV 2.0	**4.68E−106**	**2.45E−85**	**2.75E−87**	**2.45E−86**
	CV 3.0	4.35E−87	7.89E−65	8.68E−67	7.92E−66
f_{15}	CS	2.1853	5.5465	3.8651	0.6468
	CV 1.0	**0.0013**	**0.1233**	**0.0099**	**0.0220**
	CV 2.0	0.0028	2.2010	0.0715	0.3003
	CV 3.0	0.0027	0.5215	0.0227	0.0677
f_{16}	CS	3.8000	16.3724	8.3004	2.2980
	CV 1.0	0.0371	0.4395	0.1718	0.0974
	CV 2.0	0.0640	0.5447	0.1982	0.0828
	CV 3.0	0.0499	**0.3148**	**0.1518**	**0.0578**
f_{18}	CS	12.4288	18.4737	15.6275	1.2561
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**0**	**0**	**0**
f_{19}	CS	1.88E−16	4.73E−09	5.85E−11	4.73E−10
	CV 1.0	2.53E−24	**6.84E−19**	**4.04E−20**	**1.07E−19**
	CV 2.0	7.05E−25	1.04E−16	4.49E−18	1.62E−17
	CV 3.0	1.09E−18	1.01E−11	2.96E−13	1.12E−12
f_{20}	CS	−3.8628	−3.8628	−3.8628	3.00E−13
	CV 1.0	−3.8628	−3.8628	−3.8628	**1.83E−14**
	CV 2.0	−3.8628	−3.8628	−3.8628	4.15E−14
	CV 3.0	−3.8628	−3.8628	−3.8628	1.48E−13
f_{21}	CS	−3.3224	−3.3224	−3.3224	2.14E−07
	CV 1.0	−3.3224	−3.3224	−3.3224	**1.46E−08**
	CV 2.0	−3.3224	−3.3224	−3.3224	2.01E−07
	CV 3.0	−3.3224	−3.3224	−3.3224	8.91E−08
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.19E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	4.73E−15
	CV 2.0	−1.0316	−1.0316	−1.0316	2.16E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	**1.07E−15**

For the fixed dimension function, CV 1.0, CV2 2.0 and CV 3.0 have marginal variation in their performance and here CV 1.0 has an upper edge for f_{21} and for function f_{24} where results of CV 1.0 and PSOGSA are same and it is not clear that which algorithm performs better whereas for function f_{20}, CV 1.0 and CV 2.0 have the equivalent performance and for f_{22} and f_{23} CV 1.0 is the best. So, from the twenty-four-function used in this comparison, CV 1.0 is found to be better for seventeen functions, CV 2.0 for seven functions, CV 3.0 for four functions, FA for two functions, PSOGSA and GWO for one function whereas DE, BA, BFP, and FPA were found to be best for none. Note that all the proposed version are better than other algorithms and among these three proposed variants, CV 1.0 is the best.

For a generalized algorithm, because of extensive exploration in the initial stages, the convergence is slow due to smaller steps at the start but as the algorithm approaches towards the final stages, intensive exploitations start, making the algorithm to move faster. This property makes the algorithm to converge faster towards the later stages. Both extensive exploration and exploitation in any generalized algorithm ensure that the algorithm is efficient in achieving a balanced exploration and exploitation. From the convergence profiles in the Fig. 2, it is evident that CV 1.0 is a very good algorithm with the ability of extensive exploration, intensive exploitation and maintaining a balance between the two. Thus, both comparative results from Table 12 and convergence plots from Fig. 2 validate the hypothesis of CV 1.0 to be considered in the category of highly competitive algorithms.

5.5. Statistical testing

To test the performance of the proposed CV 1.0 version, Wilcoxon rank-sum test is performed (Derrac, García, Molina, & Herrera, 2011). This test uses two different samples of the population and returns a p-value corresponding to the two. This p value determines the significance of the two population sets. In terms

Table 9
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS for population size 60.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	13.9323	54.0312	33.4056	7.5642
	CV 1.0	2.76E−112	8.57E−94	2.05E−95	1.22E−94
	CV 2.0	**3.05E−121**	**4.66E−98**	**1.52E−99**	**7.07E−99**
	CV 3.0	3.86E−95	2.44E−73	2.44E−75	2.44E−74
f_3	CS	0.0719	0.2598	0.1675	0.0390
	CV 1.0	**3.79E−05**	**0.0032**	**6.70E−04**	**4.88E−04**
	CV 2.0	4.91E−05	0.0074	0.0018	0.0014
	CV 3.0	2.67E−04	0.0169	0.0031	0.0027
f_4	CS	0.0893	0.3221	0.1607	0.0362
	CV 1.0	8.57E−113	5.79E−96	5.98E−98	5.79E−97
	CV 2.0	**6.07E−118**	**1.78E−99**	**1.99E−101**	**1.78E−100**
	CV 3.0	2.06E−98	2.26E−79	2.41E−81	2.26E−80
f_6	CS	2.14E+04	9.56E+04	6.07E+04	1.35E+04
	CV 1.0	1.82E−107	3.15E−91	7.21E−93	4.32E−92
	CV 2.0	**8.43E−111**	**7.16E−93**	**8.69E−95**	**7.25E−94**
	CV 3.0	4.35E−96	9.20E−72	9.20E−74	9.20E−73
f_9	CS	8.7049	15.4846	11.8192	1.2896
	CV 1.0	8.88E−16	4.44E−15	**1.24E−15**	**1.07E−15**
	CV 2.0	8.88E−16	4.44E−15	1.52E−15	1.37E−15
	CV 3.0	8.88E−16	4.44E−15	2.30E−15	1.74E−15
f_{13}	CS	3.38E+02	1.16E+03	6.27E+02	1.68E+02
	CV 1.0	5.19E−111	5.65E−92	7.53E−94	5.73E−93
	CV 2.0	**2.03E−113**	**1.33E−95**	**1.61E−97**	**1.34E−96**
	CV 3.0	5.21E−101	1.08E−71	1.08E−73	1.08E−72
f_{15}	CS	2.6612	7.0328	4.6809	0.7840
	CV 1.0	**3.08E−04**	**0.1081**	**0.0029**	**0.0107**
	CV 2.0	0.0028	2.1425	0.0497	0.2948
	CV 3.0	0.0033	2.4923	0.0581	0.2949
f_{16}	CS	6.8706	21.7676	12.7678	2.7665
	CV 1.0	**0.0055**	**0.2659**	**0.0879**	**0.0550**
	CV 2.0	0.0368	0.4745	0.2041	0.0899
	CV 3.0	0.0526	0.4800	0.1827	0.0779
f_{18}	CS	13.0020	21.2100	17.6203	1.3688
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**0**	**0**	**0**
f_{19}	CS	2.63E−15	2.93E−11	1.94E−12	3.99E−12
	CV 1.0	**3.36E−23**	**2.36E−18**	**6.55E−20**	**2.58E−19**
	CV 2.0	9.10E−22	6.57E−15	1.57E−16	7.57E−16
	CV 3.0	8.98E−19	1.23E−12	1.16E−13	2.13E−13
f_{20}	CS	−3.8628	−3.8628	−3.8628	2.56E−13
	CV 1.0	−3.8628	−3.8628	−3.8628	**6.31E−15**
	CV 2.0	−3.8628	−3.8628	−3.8628	6.97E−15
	CV 3.0	−3.8628	−3.8628	−3.8628	4.53E−13
f_{21}	CS	−3.3224	−3.3224	−3.3224	1.81E−07
	CV 1.0	−3.3224	−3.3224	−3.3224	**2.72E−08**
	CV 2.0	−3.3224	−3.3224	−3.3224	1.43E−07
	CV 3.0	−3.3224	−3.3224	−3.3224	6.51E−08
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.20E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	1.11E−14
	CV 2.0	−1.0316	−1.0316	−1.0316	4.60E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	**1.02E−15**

of algorithmic studies, two algorithms are tested using the same method and corresponding p-value is checked at 5% level of significance. If two algorithms satisfy these conditions, they are said to be statistically significant. For the present case, the significance of CV 1.0 is to be calculated with respect to others to prove its efficiency statistically. This is done by comparing CV 1.0 with BA, CV1.0/DE, CV 1.0/FA, CV 1.0/FPA, CV 1.0/BFP, CV 1.0/PSOGSA and CV 1.0/GWO. Since we cannot compare the best algorithm with itself, so NA (Not applicable) has been incorporated in its place. The '∼' sign has been added where ever two algorithms provide same statistical results. These algorithms are either identical or don't have any statistical relevance so can't be compared (Salgotra & Singh, 2017). The statistical results are presented in Table 13. It can be seen from the table that CV 1.0 provides better statistical significance of more than 80% of the cases and hence making it a potential candidate for becoming a state-of-the-art algorithm.

5.6. Effect of dimension

From the comparative analysis, it is imperative that CV 1.0 is the best-fit candidate for becoming a state-of-the-art algorithm. But for any algorithm to be called as a generic problem solver, it should be able to provide good results on higher dimension problems. So, to test this hypothesis, CV 1.0 is tested on higher dimension functions. Here from Table 1, seventeen functions consisting of unimodal and multi-modal sets are used and four different dimension sizes are used to validate the results. The dimension set includes 30-dimensional functions whose comparisons are shown in Section 5.5. For analyzing the effect of dimension, dimension sizes of 50, 100, 500 and 1000 are used. Tables 14–17 present the results of the dimensional analysis. It is evident from the tables that though the performance of CV 1.0 reduces to some extent with increase in dimension with respect to the basic dimension size of 30, the decrease in results is marginal. Also, the solution quality with

Table 10
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS for population size 80.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	25.8047	73.2406	44.4418	9.4418
	CV 1.0	2.86E−115	1.20E−98	2.03E−100	1.99E−99
	CV 2.0	**1.25E−120**	**5.96E−103**	**8.63E−105**	**6.07E−104**
	CV 3.0	2.89E−108	8.30E−84	8.39E−86	8.30E−85
f_3	CS	0.0772	0.2926	0.1873	0.0418
	CV 1.0	**2.85E−05**	**0.0013**	**4.46E−04**	**2.98E−04**
	CV 2.0	1.54E−05	0.0047	0.0010	9.30E−04
	CV 3.0	7.27E−05	0.0127	0.0029	7.82E−14
f_4	CS	0.1063	0.3253	0.2243	0.0025
	CV 1.0	1.38E−119	1.56E−100	1.60E−102	1.56E−101
	CV 2.0	**3.14E−123**	**7.12E−107**	**1.68E−108**	**8.90E−108**
	CV 3.0	2.61E−104	6.51E−89	1.47E−90	7.33E−90
f_6	CS	3.74E+04	1.22E+05	7.36E+04	1.57E+04
	CV 1.0	4.18E−113	1.69E−95	1.93E−97	1.70E−96
	CV 2.0	**4.13E−117**	**1.23E−97**	**1.25E−99**	**1.23E−98**
	CV 3.0	3.30E−97	**2.60**E−80	2.60E−82	2.60E−81
f_9	CS	9.4334	14.7056	12.3275	1.2282
	CV 1.0	**8.88E−16**	**4.44E−15**	**1.10E−15**	**8.47E−16**
	CV 2.0	**8.88E−16**	**4.44E−15**	1.66E−15	1.47E−15
	CV 3.0	**8.88E−16**	**4.44E−15**	1.98E−15	1.65E−15
f_{13}	CS	4.71E+02	1.30E+03	8.73E+02	1.75E+02
	CV 1.0	1.75E−113	5.52E−97	5.65E−99	5.52E−98
	CV 2.0	**3.89E−123**	**3.56E−101**	**4.48E−103**	**3.66E−102**
	CV 3.0	2.86E−99	4.14E−82	4.26E−84	4.14E−83
f_{15}	CS	2.9452	7.7995	5.2843	0.7429
	CV 1.0	**1.46E−04**	**0.0046**	**8.12E−04**	**6.98E−04**
	CV 2.0	0.0020	0.2429	0.0105	0.0285
	CV 3.0	0.0014	1.3381	0.0307	0.1497
f_{16}	CS	8.3627	25.294	16.432	3.5843
	CV 1.0	**0.0042**	**0.2344**	**0.0531**	**0.0491**
	CV 2.0	0.0485	0.5401	0.1849	0.0990
	CV 3.0	0.0454	0.4043	0.1767	0.0742
f_{18}	CS	16.1464	20.9889	18.7617	1.0789
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**0**	**0**	**0**
f_{19}	CS	2.36E−17	7.38E−11	2.81E−12	9.69E−12
	CV 1.0	**5.91E−23**	**1.93E−18**	**1.12E−19**	**2.36E−19**
	CV 2.0	3.39E−21	4.14E−15	6.96E−17	4.26E−16
	CV 3.0	1.85E−16	7.90E−13	7.11E−14	1.41E−13
f_{20}	CS	−3.8628	−3.8628	−3.8628	5.26E−13
	CV 1.0	−3.8628	−3.8628	−3.8628	**2.23E−14**
	CV 2.0	−3.8628	−3.8628	−3.8628	5.82E−14
	CV 3.0	−3.8628	−3.8628	−3.8628	1.04E−13
f_{21}	CS	−3.3224	−3.3224	−3.3224	1.35E−07
	CV 1.0	−3.3224	−3.3224	−3.3224	**2.38E−08**
	CV 2.0	−3.3224	−3.3224	−3.3224	3.28E−07
	CV 3.0	−3.3224	−3.3224	−3.3224	6.86E−08
f_{22}	CS	−1.0316	−1.0316	−1.0316	7.18E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	4.34E−15
	CV 2.0	−1.0316	−1.0316	−1.0316	4.94E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	**1.03E−15**

respect to other algorithms does not deviate from standard results shown in Section 5.5 in Table 12. This shows that even with the change in dimension size, the algorithm is able to provide global optimal results. This is because of the presence of enhanced global and local search phases. On a whole, we can say that CV 1.0 is a very competitive algorithm and performs very well for higher dimension functions.

5.7. Comparison results for CEC 2015 benchmark problems

As a further extension, CV 1.0 is applied to CEC 2015 benchmark functions to prove its superiority over the other algorithms. The CEC 2015 benchmark problems consist of fifteen functions with a distribution of two unimodal functions, three multimodal functions, three hybrid functions and seven composite functions. All the problems are minimization problems and have bound constraints, shifted scalable and rotated functions. A general summary of all the functions is given in Table 18 and for more detailed discussion one can refer to Liang, Qu, Wang, Suganthan, and Chen (2014).These functions are highly complex functions and a maximum number of function evaluations recommended by Liang, Qu, Wang, Suganthan, and Chen (2014) is 10, 000 $\times D$, where D is the problem dimension. In present case the dimension size is taken to be 10, so the total number of function evaluations becomes 100,000. Also we know that, the total number of function evaluations is equal to the population size multiplied by the total number of iterations or generations. So to achieve the total function evaluations of 100,000, the population size is chosen to be 50 and the maximum number of iterations is kept 2000, making the total number of function evaluations same in both the cases.

From the results in Table 19, it can be seen that for function F_1, F_2 and F_6 except from CV 1.0, no other algorithm is able to achieve the global minima. Here for CV 1.0, the best, worst and mean values are exactly same as global minima. The standard deviation, in this case, is also best for CV 1.0. For functions F_3 and F_7, all the

Table 11
Comparison of CV 1.0, CV 2.0, CV 3.0 and CS for population size 100.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	CS	34.9296	74.4182	54.2697	8.9788
	CV 1.0	1.73E−118	3.18E−103	5.26E−105	3.53E−104
	CV 2.0	**5.84E−128**	**2.40E−104**	**2.40E−106**	**2.40E−105**
	CV 3.0	1.55E−111	3.47E−88	3.51E−90	3.47E−89
f_3	CS	0.0719	0.3276	0.2054	0.0571
	CV 1.0	**2.25E−05**	**0.0016**	**4.92E−04**	**3.57E−04**
	CV 2.0	7.60E−05	0.0056	7.95E−04	7.68E−04
	CV 3.0	6.22E−05	0.0067	0.0015	8.37E−04
f_4	CS	0.1400	0.3889	0.2561	0.0013
	CV 1.0	2.60E−124	3.42E−107	8.17E−109	4.61E−108
	CV 2.0	**4.82E−128**	**6.09E−109**	**9.71E−111**	**7.02E−110**
	CV 3.0	2.13E−113	3.24E−94	4.22E−96	3.33E−95
f_6	CS	4.46E+04	1.29E+05	8.27E+04	1.61E+04
	CV 1.0	5.61E−115	4.11E−101	7.84E−103	5.03E−102
	CV 2.0	**2.63E−125**	**2.15E−104**	**2.19E−106**	**2.15E−105**
	CV 3.0	3.60E−102	5.24E−88	7.87E−90	5.44E−89
f_9	CS	9.7686	15.4366	13.0067	1.0487
	CV 1.0	8.88E−16	4.44E−15	**1.17E−15**	**9.68E−16**
	CV 2.0	8.88E−16	4.44E−15	1.70E−15	1.50E−15
	CV 3.0	8.88E−16	4.44E−15	2.06E−15	1.67E−15
f_{13}	CS	6.11E+02	1.82E+03	1.07E+03	2.15E+02
	CV 1.0	3.43E−117	4.35E−102	4.45E−104	4.34E−103
	CV 2.0	**3.92E−124**	**1.15E−107**	**2.31E−109**	**1.54E−108**
	CV 3.0	1.04E−108	1.62E−88	2.18E−90	1.69E−89
f_{15}	CS	3.3389	7.2958	5.6025	0.7273
	CV 1.0	**6.90E−05**	**0.0026**	**4.53E−04**	**3.96E−04**
	CV 2.0	0.0021	0.1301	0.0093	0.0207
	CV 3.0	0.0025	1.4931	0.0546	0.2396
f_{16}	CS	9.6351	26.632	17.969	3.3586
	CV 1.0	**0.0014**	**0.1204**	**0.0249**	**0.0268**
	CV 2.0	0.0557	0.6394	0.2005	0.1053
	CV 3.0	0.0491	0.3893	0.1596	0.0663
f_{18}	CS	17.1639	21.570	19.2243	0.9863
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**0**	**0**	**0**
f_{19}	CS	8.62E−16	2.54E−11	1.60E−12	3.47E−12
	CV 1.0	**1.61E−22**	**2.35E−18**	**1.05E−19**	**2.77E−19**
	CV 2.0	4.07E−20	1.32E−15	5.80E−17	1.66E−16
	CV 3.0	6.86E−17	5.04E−13	5.45E−14	8.81E−14
f_{20}	CS	−3.8628	−3.8628	−3.8628	1.76E−13
	CV 1.0	−3.8628	−3.8628	−3.8628	**1.97E−14**
	CV 2.0	−3.8628	−3.8628	−3.8628	2.22E−13
	CV 3.0	−3.8628	−3.8628	−3.8628	9.00E−14
f_{21}	CS	−3.3224	−3.3224	−3.3224	7.97E−08
	CV 1.0	−3.3224	−3.3224	−3.3224	**2.27E−08**
	CV 2.0	−3.3224	−3.3224	−3.3224	2.26E−07
	CV 3.0	−3.3224	−3.3224	−3.3224	4.73E−08
f_{22}	CS	−1.0316	−1.0316	−1.0316	1.25E−15
	CV 1.0	−1.0316	−1.0316	−1.0316	4.37E−15
	CV 2.0	−1.0316	−1.0316	−1.0316	1.31E−15
	CV 3.0	−1.0316	−1.0316	−1.0316	**1.02E−15**

algorithms provide marginal variation from the global minima and the comparative results among these algorithms are highly competitive. Here standard deviation decides the best algorithm under study and it can be seen that CV 1.0 provides the best results. For functions F_4, F_5, F_9, F_{10} and F_{12}, FA, GWO and CV 1.0 provide very competitive results and it is very difficult to judge which one is the best. Here based on the best value we can say that CV 1.0 provides the best global optima for F_4 and for F_5, FA is the best and based on standard deviation, FA provides best results for function F_9 and CV 1.0 for functions F_{10} and F_{12}. For function F_8, only CV 1.0 provides the best results. For function F_{13}, all the algorithms are able to achieve global minima and based on the standard deviation, BA provides the best results. For F_{14} and F_{15}, the variation in optimality is minimal, here PSOGSA is found to be the best for function F_{14} and CV 1.0 for function F_{15}. Overall, we can say that FA is better for two functions, BA for one function, PSOGSA for one function and CV 1.0 for eleven functions. From this very observation, we can say that CV 1.0 is a very competitive algorithm and performs better than its other counterparts. The results of Wilcoxon rank-sum test reported in Table 20 also validate the supremacy of CV 1.0 statistically.

5.8. Sensitivity study of parameter settings for the proposed CV 1.0/2.0/3.0 versions

It can be seen from the above results that all the three proposed versions are good in terms of finding global minima and are better than the other algorithms. These proposed algorithms though are very efficient but their performance depends on certain set of parameters. The most important parameter is the switch probability. A detailed discussion about this parameter is presented in Section 5.2 and inferences about its use are discussed in that particular section. It was found that, though there is marginal

Table 12
Comparison of CV 1.0 with the state-of-the-art algorithms.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	BA	8.28E+03	5.18E+04	2.56E+04	7.73E+03
	BFP	3.98E+04	9.01E+04	6.63E+04	1.01E+04
	CV 1.0	**1.16E−84**	**7.46E−71**	**7.61E−73**	**7.46E−72**
	CV 2.0	**2.34E−93**	**1.04E−65**	**1.04E−67**	**1.04E−66**
	CV 3.0	**1.08E−71**	**3.28E−45**	**3.48E−47**	**3.28E−46**
	CS	**2.09E+00**	**1.33E+01**	**5.79E+00**	**2.25E+00**
	DE	2.19E+04	8.92E+04	6.83E+04	1.02E+04
	FA	0.0055	0.0385	0.0170	0.0068
	FPA	1.07E+03	6.21E+03	3.15E+03	9.69E+02
	GWO	8.69E−28	2.46E−24	7.70E−26	2.66E−25
	PSOGSA	1.2602	2.00E+04	4.03E+03	5.57E+03
f_2	BA	9328	52,816	2.69E+04	7.86E+03
	BFP	35,883	86,733	6.77E+04	1.07E+04
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**1**	**1.00E−02**	**1.00E−02**
	CS	**8**	**35**	**1.69E+01**	**5.26E+00**
	DE	32,224	8.64E+04	6.81E+04	1.02E+04
	FA	0	2	0.1600	0.3949
	FPA	1060	5922	3.29E+03	1.01E+03
	GWO	**0**	**0**	**0**	**0**
	PSOGSA	231	20,388	5.37E+03	4.68E+03
f_3	BA	4.5650	6.9961	14.4307	6.9961
	BFP	38.8144	187.80	106.932	29.079
	CV 1.0	**1.57E−04**	0.0012	0.0017	0.0012
	CV 2.0	**2.01E−04**	0.0476	0.0068	0.0083
	CV 3.0	**1.00E−03**	0.0419	0.0137	0.0108
	CS	**4.67E−02**	0.2670	0.1368	0.0376
	DE	36.6303	26.1571	1.29E+02	26.1571
	FA	0.0190	0.0487	0.0804	0.0487
	FPA	0.0761	1.3440	0.5802	0.3440
	GWO	2.33E−04	**7.92E−04**	**0.0014**	**7.92E−04**
	PSOGSA	0.0543	8.3080	0.3791	0.8568
f_4	BA	4.6385	9.92E+02	1.82E+02	1.70E+02
	BFP	1.12E+02	4.39E+03	5.15E+02	7.17E+02
	CV 1.0	**2.07E−87**	**1.60E−76**	**1.97E−75**	**1.62E−74**
	CV 2.0	**2.92E−95**	**4.09E−71**	**4.09E−73**	**4.09E−72**
	CV 3.0	**2.38E−82**	**1.16E−49**	**1.99E−51**	**1.34E−50**
	CS	**1.05E−02**	**6.46E−02**	**3.21E−02**	**1.20E−02**
	DE	31.4995	1.03E+03	2.72E+02	1.83E+03
	FA	8.4605	79.5762	34.4397	12.8443
	FPA	5.2794	42.5364	17.5376	6.6479
	GWO	4.24E−30	8.41E−28	1.73E−28	2.12E−28
	PSOGSA	0.0646	75.0787	10.3139	16.2801
f_5	BA	1.35E−07	4.72E−06	1.05E−06	8.36E−07
	BFP	0.1357	1.2790	0.5920	0.2262
	CV 1.0	**2.10E−268**	**2.63E−246**	**3.30E−248**	**0**
	CV 2.0	**1.10E−165**	**1.84E−109**	**2.52E−111**	**1.95E−110**
	CV 3.0	**5.02E−109**	**8.33E−59**	**8.33E−61**	**8.33E−60**
	CS	**9.13E−15**	**4.61E−10**	**1.14E−11**	**5.22E−11**
	DE	0.1203	1.8686	0.9617	0.3539
	FA	6.23E−08	1.48E−05	2.42E−06	2.50E−06
	FPA	4.72E−08	3.28E−04	1.84E−05	3.95E−05
	GWO	1.96E−102	4.36E−85	4.59E−87	4.36E−86
	PSOGSA	3.61E−10	1.61E−07	3.04E−08	3.00E−08
f_6	BA	2.87E+08	3.93E+09	1.17E+09	6.57E+08
	BFP	7.26E+08	5.62E+09	2.83E+09	9.23E+08
	CV 1.0	**6.24E−82**	**2.41E−69**	**5.82E−71**	**2.91E−70**
	CV 2.0	**9.42E−92**	**1.42E−68**	**1.45E−70**	**1.42E−69**
	CV 3.0	**2.12E−72**	**7.20E−44**	**1.35E−45**	**8.98E−45**
	CS	**5.37E+03**	**3.15E+04**	**1.39E+04**	**6.44E+03**
	DE	1.10E+08	2.88E+09	8.06E+08	5.60E+08
	FA	1.69E+06	3.73E+07	1.05E+07	6.34E+06
	FPA	3.09E+06	5.51E+07	2.02E+07	1.04E+07
	GWO	1.37E−24	4.88E−21	1.89E−22	5.55E−22
	PSOGSA	4.16E+05	1.52E+09	1.55E+08	2.50E+08
f_7	BA	2.75E−13	0.4890	0.1680	0.1290
	BFP	0.0184	0.4940	0.3553	0.1150
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**0**	**0**	**0**
	CS	**6.91E−13**	**6.24E−06**	**2.17E−07**	**9.35E−07**
	DE	0	0.4018	0.0793	0.0810

(*continued on next page*)

Table 12 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	FA	3.51E−13	0.0031	3.12E−05	3.12E−04
	FPA	2.90E−10	8.29E−06	4.03E−07	1.22E−06
	GWO	**0**	**0**	**0**	**0**
	PSOGSA	0	0.0677	0.0014	0.0071
f_8	BA	−4.08E+227	−5.66E+24	−4.97E+225	5.69E+134
	BFP	−4.81E+15	−1.53E+03	−7.31E+13	5.06E+14
	CV 1.0	**−9.07E+03**	−7.04E+03	−7.84E+03	**3.77E+02**
	CV 2.0	**−8.44E+03**	−6.87E+03	−7.52E+03	**2.83E+02**
	CV 3.0	**−8.85E+03**	−7.11E+03	−7.95E+03	**3.73E+02**
	CS	**−9.02E+03**	−7.76E+03	−8.24E+03	**2.53E+02**
	DE	−1.39E+127	−5.04E+116	−2.32E+125	1.61E+125
	FA	−8.91E+03	**−4.75E+03**	−6.66E+03	8.38E+02
	FPA	−1.36E+40	−3.17E+31	−3.46E+38	1.84E+38
	GWO	−8.65E+31	−2.24E+25	−2.10E+30	9.38E+31
	PSOGSA	−8.03E+03	−4.89E+03	−6.66E+03	7.41E+02
f_9	BA	11.8266	18.6199	16.9291	1.0371
	BFP	19.7632	20.8843	20.4850	0.2579
	CV 1.0	**8.88E−16**	**4.44E−15**	**1.35E−15**	**1.20E−15**
	CV 2.0	**8.88E−16**	**4.44E−15**	**2.02E−15**	**1.66E−15**
	CV 3.0	**8.88E−16**	**4.44E−15**	**2.14E−15**	**1.76E−15**
	CS	**3.46E+00**	**12.6364**	**7.2292**	**2.0449**
	DE	16.9460	20.9508	20.1775	0.8037
	FA	0.0344	0.9482	0.09397	0.1225
	FPA	8.7570	14.9551	11.7919	1.2539
	GWO	9.32E−14	4.20E−13	1.68E−13	5.16E−14
	PSOGSA	9.8404	19.9631	17.5044	1.8909
f_{10}	BA	1.49E+02	5.21E+03	1.41E+03	1.07E+03
	BFP	6.30E+03	3.07E+04	1.45E+04	5.08E+03
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**1.10E−91**	**7.71E−15**	**7.80E−17**	**7.17E−15**
	CV 3.0	**4.86E−75**	**2.34E−10**	**2.80E−12**	**2.38E−11**
	CS	**0.2644**	**5.2889**	**1.9142**	**1.1353**
	DE	1.08E+03	1.49E+04	5.21E+03	3.63E+03
	FA	0.8101	64.7743	11.1998	9.7588
	FPA	44.5663	4.19E+02	1.59E+02	83.4629
	GWO	8.70E−07	1.14E−04	2.20E−05	2.27E−05
	PSOGSA	0.0518	2.90E+03	66.2695	3.45E+02
f_{11}	BA	2.16E+04	1.03E+06	2.08E+05	1.68E+05
	BFP	3.96E+05	2.83E+06	1.76E+06	5.11E+05
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0.6667**	**0.6695**	**0.6670**	**4.00E−04**
	CV 3.0	**0.6670**	**0.6783**	**0.6696**	**1.70E−03**
	CS	**2.7783**	**21.0306**	**8.0838**	**3.2242**
	DE	3.98E+05	2.75E+06	1.93E+06	4.21E+05
	FA	0.7051	1.09E+02	13.9782	22.7202
	FPA	6.07E+02	1.83E+04	5.10E+03	3.37E+03
	GWO	0.6667	0.6725	0.6668	5.83E−04
	PSOGSA	0.667	3.88E+05	1.12E+04	5.16E+04
f_{12}	BA	2.82E+02	3.25E+05	9.83E+03	4.36E+04
	BFP	4.43E+02	1.89E+03	1.40E+08	3.68E+08
	CV 1.0	**3.6E−266**	**3.6E−247**	**4.4E−249**	**0**
	CV 2.0	**4.08E+00**	**6.23E+01**	**3.11E+01**	**1.35E+01**
	CV 3.0	**2.88E+01**	**1.16E+02**	**5.89E+01**	**1.67E+01**
	CS	**1.36E+02**	**3.47E+02**	**2.30E+02**	**4.57E+01**
	DE	1.54E+03	5.74E+05	2.44E+04	6.02E+04
	FA	43.4116	3.38E+02	1.54E+02	54.7164
	FPA	1.39E+02	1.81E+03	6.13E+02	3.11E+02
	GWO	1.71E−07	0.4637	0.0117	0.0513
	PSOGSA	35.4683	3.76E+02	1.40E+02	65.1312
f_{13}	BA	1.13E+05	1.17E+06	3.65E+05	1.62E+05
	BFP	6.66E+05	2.02E+06	1.55E+06	2.51E+05
	CV 1.0	**5.32E−89**	**3.07E−70**	**6.18E−72**	**3.61E−71**
	CV 2.0	**2.10E−96**	**1.84E−66**	**1.90E−68**	**1.84E−67**
	CV 3.0	**3.34E−76**	**2.60E−40**	**2.60E−42**	**2.60E−41**
	CS	**3.66E+01**	**2.99E+02**	**1.22E+02**	**5.17E+01**
	DE	2.40E+05	2.07E+06	1.65E+06	3.07E+05
	FA	1.7965	139.4007	31.5047	21.8411
	FPA	3.40E+04	1.45E+05	7.49E+04	2.40E+04
	GWO	1.83E−26	1.05E−23	8.64E−25	1.53E−25
	PSOGSA	1.00E−29	2.50E+05	1.50+04	5.96E+04
f_{14}	BA	0.1456	76.0727	12.0498	14.0377
	BFP	1.5980	3.73E+02	82.0362	75.9781
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**

(*continued on next page*)

Table 12 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	CV 3.0	**0**	**0**	**0**	**0**
	CS	**1.11E−16**	**1.01E−07**	**1.56E−09**	**1.05E−08**
	DE	0	14.1035	0.4844	1.6131
	FA	3.36E−08	1.11E−05	2.06E−06	2.04E−06
	FPA	1.05E−08	2.10E−04	1.45E−05	2.93E−05
	GWO	0	0.1456	0.00873	0.0348
	PSOGSA	0	0.5365	0.0904	0.0943
f_{15}	BA	2.72E+05	2.39E+08	4.64E+07	4.63E+07
	BFP	9.38E+07	9.79E+08	5.60E+08	1.86E+02
	CV 1.0	0.0071	13.4494	1.5052	2.5976
	CV 2.0	0.0055	14.3060	2.2586	3.2817
	CV 3.0	0.0018	3.4577	0.2927	0.6965
	CS	1.3836	5.6063	3.5324	0.8679
	DE	2.05E+08	9.80E+08	6.23E+08	1.57E+08
	FA	**1.71E−04**	**0.3937**	**0.0142**	0.0479
	FPA	11.0790	2.45E+05	4.50E+03	2.56E+04
	GWO	0.0187	0.5673	0.0693	**0.0617**
	PSOGSA	8.1835	2.56E+08	1.28E+07	5.60E+07
f_{16}	BA	1.03E+07	5.16E+08	1.36E+08	1.13E+08
	BFP	2.39E+08	1.80E+09	1.05E+09	3.50E+08
	CV 1.0	0.1067	1.2982	0.4382	0.1981
	CV 2.0	0.1202	0.7024	0.3016	0.1191
	CV 3.0	0.0292	0.3018	0.1284	0.0518
	CS	1.4167	17.7862	6.8440	3.4614
	DE	7.53E+08	1.84E+09	1.22E+09	2.23E+08
	FA	**0.0011**	**0.0274**	**0.0066**	**0.0051**
	FPA	1.16E+03	2.76E+06	4.39E+05	5.34E+05
	GWO	0.2134	1.4344	0.8340	0.2359
	PSOGSA	28.8520	8.20E+08	3.69E+07	1.31E+08
f_{17}	BA	2.47E−08	12.9344	3.0346	3.0111
	BFP	0.0106	19.8991	6.3343	4.1938
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	0	**0**	**0**	**0**
	CV 3.0	0	**0**	**0**	**0**
	CS	0	**7.30E−11**	**2.58E−12**	**7.30E−11**
	DE	0	5.3503	1.4886	1.1273
	FA	9.29E−10	0.9950	0.0099	0.0995
	FPA	4.04E−07	0.0535	0.0029	0.0073
	GWO	**0**	**0**	**0**	**0**
	PSOGSA	0	3.9798	0.6368	0.7940
f_{18}	BA	25.4258	44.2524	34.9182	3.9018
	BFP	35.9973	51.1303	46.1738	2.8954
	CV 1.0	**0**	**0**	**0**	**0**
	CV 2.0	**0**	**0**	**0**	**0**
	CV 3.0	**0**	**0**	**0**	**0**
	CS	**8.2042**	**14.6325**	**11.7295**	**1.4052**
	DE	15.3664	40.7489	25.3598	4.3389
	FA	13.0235	25.1890	18.7263	2.9922
	FPA	27.7361	39.1185	34.7068	1.8148
	GWO	1.42E−14	9.5890	2.0476	1.7815
	PSOGSA	9.0048	26.4511	17.2063	3.4552
f_{19}	BA	2.01E−11	1.0926	0.1675	0.3073
	BFP	1.20E−04	13.8882	1.7511	2.5430
	CV 1.0	1.11E−29	**3.68E−20**	**5.73E−22**	**3.77E−21**
	CV 2.0	2.46E−31	**1.10E−16**	**1.49E−18**	**1.12E−17**
	CV 3.0	2.80E−19	**1.51E−11**	**3.78E−13**	**1.71E−12**
	CS	3.36E−17	**1.01E−09**	**4.14E−11**	**1.63E−10**
	DE	**0**	0.4526	0.0087	0.0498
	FA	7.27E−12	4.41E−09	1.01E−09	9.70E−10
	FPA	4.22E−14	3.24E−05	1.45E−06	4.45E−06
	GWO	9.46E−10	0.4966	0.0245	0.1072
	PSOGSA	8.30E−25	0.7621	0.1677	0.3173
f_{20}	BA	−3.8628	−1.30E−10	−1.5088	1.5275
	BFP	−2.7902	−1.36E−77	−0.0869	0.3720
	CV 1.0	−3.8628	−3.8627	−3.8627	**2.64E−15**
	CV 2.0	−3.8628	−3.8628	−3.8628	**6.65E−15**
	CV 3.0	−3.8628	−3.8628	−3.8628	**6.86E−14**
	CS	−3.8628	−3.8628	−3.8628	**3.49E−13**
	DE	−3.8628	−3.3718	−3.8525	0.0616
	FA	−3.8628	−3.8627	−3.8628	1.45E−07
	FPA	−3.8628	−2.5732	−3.8207	0.1602
	GWO	−3.8628	−3.8485	−3.8593	0.0050
	PSOGSA	−3.8628	−2.32E−05	−2.8477	1.4904
f_{21}	BA	−3.3224	−3.1842	−3.2741	0.0595
	BFP	−3.1589	−0.6032	−1.9290	0.5449

(*continued on next page*)

Table 12 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	CV 1.0	−3.3224	−3.3224	−3.3224	**1.28E−08**
	CV 2.0	−3.3224	−3.3224	−3.3224	**4.16E−07**
	CV 3.0	−3.3224	−3.3224	−3.3224	**3.71E−07**
	CS	−3.3224	−3.3223	−3.3224	**4.61E−06**
	DE	−3.3224	−2.5067	−3.1668	0.1351
	FA	−3.3224	−3.0811	−3.2725	0.0690
	FPA	−3.3224	−3.1886	−3.2828	0.0360
	GWO	−3.3224	−2.8366	−3.2623	0.0901
	PSOGSA	−3.3224	−2.8104	−3.2510	0.1115
f_{22}	BA	−1.0316	2.1043	−0.9187	0.3921
	BFP	−1.0316	7.7792	0.0685	1.2546
	CV 1.0	−1.0316	−1.0316	−1.0316	**1.32E−15**
	CV 2.0	−1.0316	−1.0316	−1.0316	**3.42E−15**
	CV 3.0	−1.0316	−1.0316	−1.0316	**1.75E−15**
	CS	−1.0316	−1.0316	−1.0316	**1.41E−15**
	DE	−1.0316	−0.8620	−1.0295	0.0172
	FA	−1.0316	−1.0316	−1.0316	3.95E−09
	FPA	−1.0316	−1.0316	−1.0316	2.17E−07
	GWO	−1.0316	−1.0316	−1.0316	1.71E−08
	PSOGSA	−1.0316	−1.0316	−1.0316	1.11E−15
f_{23}	BA	−7.7026	−2.6872	−5.1747	1.1381
	BFP	−5.7377	−1.8558	−3.8721	0.7463
	CV 1.0	−8.8318	−6.6938	−7.7693	**0.4876**
	CV 2.0	−8.4235	−6.6294	**−8.3147**	**0.4255**
	CV 3.0	−9.3203	−7.9024	**−8.6880**	**0.3150**
	CS	−9.4115	−8.4752	**−8.9059**	**0.2235**
	DE	**−9.5711**	−6.8238	−7.8564	0.5938
	FA	−9.4846	−5.9727	−5.7255	0.8182
	FPA	−6.6186	−4.3771	−7.4287	0.3782
	GWO	−9.1372	−4.8622	−6.8307	0.8742
	PSOGSA	−8.8248	−3.0740		0.9596
f_{24}	BA	−1.0000	−0.9779	−8.11E−05	0.2190
	BFP	−0.9604	−1.56E−20	−0.1420	0.2638
	CV 1.0	−1.0000	−1.0000	−1.0000	**0**
	CV 2.0	−1.0000	−1.0000	−1.0000	**0**
	CV 3.0	−1.0000	−1.0000	−1.0000	**2.23E−17**
	CS	−1.0000	−1.0000	−1.0000	**1.57E−17**
	DE	−1.0000	−0.9985	−0.7911	0.0269
	FA	−1.0000	−1.0000	−1.0000	3.22E−09
	FPA	−1.0000	−1.0000	−1.0000	1.05E−08
	GWO	−1.0000	−1.0000	−1.0000	4.12E−07
	PSOGSA	−1.0000	−1.0000	−1.0000	**0**

Table 13
p-test values of various algorithms.

Function	DE	BA	GWO	FA	FPA	BFP	PSOGSA	CS	CV 2.0	CV 3.0	CV 1.0
f_1	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	5.93E−08	2.64E−34	NA
f_2	5.64E−39	5.63E−39	∼	5.98E−05	5.64E−39	5.64E−39	5.64E−39	5.27E−39	∼	3.22E−01	NA
f_3	2.56E−34	2.56E−34	NA	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	4.07E−13	2.73E−26	0.3627
f_4	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.45E−06	2.39E−33	NA
f_5	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA
f_6	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA	9.84E−34	8.69E−10
f_7	2.15E−38	5.64E−39	∼	5.64E−39	5.64E−39	5.64E−39	3.72E−04	5.64E−39	∼	∼	NA
f_8	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	9.60E−15	6.35E−25	4.34E−15	3.11E−02	1.90E−10	NA
f_9	2.97e−37	2.97e−37	2.86e−37	2.97e−37	2.97e−37	2.97E−37	2.97E−37	2.97E−37	1.30E−03	2.46E−06	NA
f_{10}	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	2.56E−34	1.40E−08	2.32E−31	NA
f_{11}	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−34	5.64E−34	NA
f_{12}	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA
f_{13}	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA	8.23E−34	1.91E−05
f_{14}	4.06F−36	5.64E−39	0.0133	5.64E−39	5.64E−39	5.64E−39	1.13E−17	5.64E−39	∼	∼	NA
f_{15}	2.56E−34	2.56E−34	2.40E−34	NA	2.56E−34	2.56E−34	2.56E−34	1.84E−12	9.17E−01	4.72E−12	1.79E−24
f_{16}	2.56E−34	2.56E−34	2.56E−34	NA	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.32E−31	1.40E−08	2.56E−34
f_{17}	2.15E−38	5.64E−39	∼	5.64E−39	5.64E−39	5.64E−39	6.02E−16	7.95E−32	∼	∼	NA
f_{18}	5.64E−39	5.64E−39	5.62E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	5.64E−39	∼	∼	NA
f_{19}	3.88E−11	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	8.25E−14	2.56E−34	9.98E−09	2.56E−34	NA
f_{20}	2.18E−05	7.63E−37	7.63E−37	7.63E−37	7.63E−37	7.63E−37	6.58E−11	1.07E−33	7.00E−01	4.25E−31	NA
f_{21}	2.56E−34	2.56E−34	2.56E−34	1.26E−25	2.56E−34	2.56E−34	0.0487	1.98E−14	1.64E−06	7.75E−02	NA
f_{22}	0.6950	1.54E−35	1.54E−35	1.54E−35	1.54E−35	1.54E−35	9.91E−11	1.92E−02	3.03E−01	6.08E−01	NA
f_{23}	7.99E−11	2.51E−32	0.0078	0.2974	2.56E−34	2.56E−34	4.64E−14	4.20E−33	1.50E−03	2.75E−27	NA
f_{24}	2.71E−06	5.64E−39	5.64E−39	5.64E−39	2.15E−38	5.64E−39	∼	1.58E−01	∼	3.22E−01	NA

Table 14
Comparison of CV 1.0 with state-of-the-art for D = 50.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	BA	1.97E+04	8.02E+04	4.49E+04	1.31E+04
	CV 1.0	**3.12E−80**	**3.36E−66**	**4.12E−68**	**3.43E−67**
	DE	9.76E+04	1.50E+05	1.30E+05	9.80E+03
	FA	0.0528	0.1772	0.1009	0.0247
	FPA	4.43E+03	1.25E+04	7.63E+03	1.80E+03
	GWO	6.16E−20	6.31E−18	9.75E−19	1.03E−18
	BFP	7.43E+04	1.46E+05	1.20E+05	1.71E+04
	PSOGSA	5.96E+02	3.18E+04	9.36E+03	7.28E+03
f_2	BA	20,373	138,834	4.64E+04	1.65E+04
	CV 1.0	**0**	**0**	**0**	**0**
	DE	100,571	142,255	1.25E+05	8.64E+03
	FA	0	4	0.9400	1.0900
	FPA	4520	11,565	7.67E+03	1.58E+03
	GWO	0	0	0	0
	BFP	66,634	150,919	1.19E+05	1.77E+04
	PSOGSA	4303	42,099	1.96E+04	7.71E+03
f_3	BA	2.4749	44.6535	14.9609	7.0858
	CV 1.0	1.99E−05	0.0109	0.0019	0.0017
	DE	46.7308	1.90E+02	1.31E+02	29.0180
	FA	0.0129	0.2820	0.0688	0.0466
	FPA	0.1205	1.3552	0.5137	0.2693
	GWO	2.29E−04	0.0041	0.0015	8.09E−04
	BFP	24.1849	1.80E+02	1.02E+02	30.1528
	PSOGSA	0.0584	0.6442	0.2710	0.0994
f_4	BA	27.6526	1.96E+03	3.13E+02	3.28E+02
	CV 1.0	7.42E−85	2.42E−71	3.88E−73	2.54E−72
	DE	1.12E+02	1.94E+03	4.08E+02	2.79E+02
	FA	34.4326	1.43E+02	86.5176	19.7481
	FPA	20.5215	80.3545	37.2576	10.7715
	GWO	3.45E−23	1.17E−19	6.58E−21	1.34E−20
	BFP	2.24E+02	7.54E+03	6.88E+02	8.77E+02
	PSOGSA	2.6128	1.81E+02	26.2423	21.8542
f_5	BA	1.29E−07	8.55E−06	1.14E−06	1.08E−06
	CV 1.0	1.73E−180	1.82E−152	3.45E−154	2.39E−153
	DE	0.4149	2.2686	1.2029	0.3870
	FA	8.97E−08	2.25E−05	2.08E−06	2.54E−06
	FPA	3.92E−08	2.01E−04	1.86E−05	3.33E−05
	GWO	1.62E−94	8.84E−74	8.89E−76	8.84E−75
	BFP	0.0440	1.3743	0.6769	0.2979
	PSOGSA	6.25E−09	1.55E−06	1.31E−07	1.72E−07
f_6	BA	2.36E+08	3.41E+09	1.23E+09	6.18E+08
	CV 1.0	**1.17E−81**	**3.06E−67**	**4.72E−69**	**3.44E−68**
	DE	2.25E+07	2.52E+09	7.31E+08	5.16E+08
	FA	2.49E+06	4.11E+07	1.02E+07	5.80E+06
	FPA	4.07E+06	8.02E+07	2.19E+07	1.32E+07
	GWO	4.87E−25	3.80E−21	1.80E−22	4.20E−22
	BFP	1.18E+09	5.23E+09	2.66E+09	7.79E+08
	PSOGSA	1.79E+05	6.28E+08	1.22E+08	1.54E+08
f_7	BA	0.0031	0.4672	0.1803	0.1268
	CV 1.0	**0**2.05E−04	**0**	**0**	**0**
	DE	1.35E−11	0.4519	0.1084	0.1050
	FA	1.16E−10	0.0031	3.12E−05	3.12E−04
	FPA	0	6.52E−06	3.64E−07	9.60E−07
	GWO	2.52E−04	0	0	0
	BFP	0	0.4921	0.3320	0.1348
	PSOGSA		0.0328	0.0012	0.0037
f_8	BA	−1.39E+229	−9.26E+63	−1.39E+227	4.59E+185
	CV 1.0	**−1.30E+04**	−1.01E+04	−1.13E+04	5.57E+02
	DE	−9.25E+124	−1.64E+113	−4.74E+123	1.62E+124
	FA	−1.64E+04	−7.22E+03	−1.07E+04	1.38E+03
	FPA	−7.19E+40	−2.44E+32	−2.05E+39	9.07E+39
	GWO	−4.58E+31	−3.49E+21	−1.71E+30	5.42E+30
	BFP	−7.99E+17	−2.00E+03	−8.36E+15	7.99E+16
	PSOGSA	−1.31E+04	−7.88E+03	−1.01E+04	1.06E+03
f_9	BA	19.9917	20.1739	20.0204	0.0342
	CV 1.0	**8.88E−16**	19.9999	5.9101	9.0797
	DE	20.4590	21.0044	20.6889	0.1249
	FA	20.0576	20.2549	20.1346	0.0448
	FPA	21.0532	21.3447	21.2558	0.0430
	GWO	20.7586	21.3370	21.2450	0.0674
	BFP	18.8740	20.9547	20.5472	0.3911
	PSOGSA	16.6166	19.9517	18.6840	0.5916
f_{10}	BA	4.41E+02	1.45E+04	3.95E+03	2.35E+03
	CV 1.0	1.78E−80	7.64E−10	7.90E−12	7.64E−11

(*continued on next page*)

Table 14 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	DE	5.26E+03	5.85E+04	2.71E+04	1.38E+04
	FA	5.6033	1.11E+02	36.53	21.1483
	FPA	2.53E+02	1.42E+03	6.28E+02	2.27E+02
	GWO	2.65E−06	1.89E−04	4.37E−05	3.89E−05
	BFP	9.30E+03	5.73E+04	3.26E+04	9.37E+03
	PSOGSA	2.4691	2.98E+03	1.89E+02	5.71E+02
f_{11}	BA	1.29E+05	2.12E+06	6.72E+05	4.57E+05
	CV 1.0	0.6666	0.6667	0.6666	2.13E−05
	DE	4.00E+06	8.83E+06	6.40E+06	1.00E+06
	FA	3.7667	2.25E+02	37.9455	44.6759
	FPA	7.84E+03	1.17E+05	2.99E+04	1.59E+04
	GWO	0.6666	0.6673	0.6667	9.09E−05
	BFP	1.84E+06	8.21E+06	5.51E+06	1.46E+06
	PSOGSA	22.9188	1.44E+06	1.30E+05	2.80E+05
f_{12}	BA	4.89E+02	2.75E+07	4.42E+05	2.94E+06
	CV 1.0	1.02E+02	3.77E+02	2.07E+02	58.8994
	DE	2.23E+03	2.64E+06	1.24E+05	2.89E+05
	FA	2.91E+02	7.91E+02	5.01E+02	1.12E+02
	FPA	7.39E+02	7.81E+03	2.77E+03	1.47E+03
	GWO	0.1630	4.66E+02	52.39	73.3810
	BFP	1.63E+03	2.13E+11	8.06E+09	2.42E+10
	PSOGSA	3.73E+02	1.07E+03	6.67E+02	1.37E+02
f_{13}	BA	1.69E+06	8.40E+06	3.72E+06	1.35E+06
	CV 1.0	4.12E−80	7.10E−66	1.26E−	8.04E−67
	DE	1.02E+07	1.50E+07	671.25E+07	9.74E+05
	FA	1.44E+02	1.15E+03	5.49E+02	2.01E+02
	FPA	3.51E+05	1.14E+06	7.02E+05	1.53E+05
	GWO	2.28E−18	6.96E−16	1.04E−16	1.19E−16
	BFP	1.29E+06	3.55E+06	2.761E+06	5.22E+05
	PSOGSA	0.0487	2.50E+05	1.75E+04	6.41E+04
f_{14}	BA	1.83E+02	8.54E+02	4.20E+02	1.31E+02
	CV 1.0	**0**	0.0762	0.0010	0.0079
	DE	9.24E+02	1.32E+03	1.16E+03	79.1890
	FA	0.0154	0.0885	0.0388	0.0125
	FPA	40.9002	1.07E+02	65.4108	12.7698
	GWO	0	0.0414	0.0065	0.0110
	BFP	0.9086	3.92E+02	79.5364	77.0673
	PSOGSA	0	0.1954	0.0816	0.0765
f_{15}	BA	8.85E+06	7.47E+08	1.31E+08	1.16E+08
	CV 1.0	0.0105	10.3264	0.6086	1.8601
	DE	6.95E+08	1.80E+09	1.25E+09	2.17E+08
	FA	2.42E−04	5.9320	1.4757	1.4027
	FPA	16.9266	4.39E+05	4.50E+04	8.69E+04
	GWO	0.0135	0.7330	0.1306	0.1302
	BFP	1.14E+08	1.66E+09	1.06E+09	3.41E+08
	PSOGSA	4.49E+03	5.12E+08	9.51E+07	1.56E+08
f_{16}	BA	2.77E+07	9.81E+08	2.83E+08	1.78E+08
	CV 1.0	0.2024	1.0672	0.5238	0.1801
	DE	1.35E+09	3.05E+09	2.40E+09	3.14E+08
	FA	0.0012	19.1050	0.3026	2.0883
	FPA	1.32E+05	9.69E+06	1.83E+06	1.67E+06
	GWO	0.2998	1.8559	1.2334	0.3075
	BFP	3.87E+08	3.08E+09	1.96E+09	6.61E+08
	PSOGSA	7.42E+04	1.23E+09	1.28E+08	2.22E+08
f_{17}	BA	1.00E+02	3.90E+02	1.93E+02	61.6807
	CV 1.0	**0**	3.29E+02	1.80E+02	1.06E+02
	DE	6.49E+02	8.64E+02	7.87E+02	39.9670
	FA	69.0339	1.96E+02	1.25E+02	27.3938
	FPA	2.91E+02	4.04E+02	3.60E+02	21.0419
	GWO	2.84E−13	52.9028	6.5015	7.1207
	BFP	1.57E−06	17.7019	6.7410	3.9040
	PSOGSA	0	4.9747	0.8158	0.9420

effect of switch probability on the performance of all the proposed variants, a probability value of 0.5 gives better results for most of the cases. Apart from this parameter, population size is also an important parameter and it can be seen from Section 5.3 that with increase in population size, there is marginal variation in results but the computational complexity increases many folds. A detailed discussion about the same is given in Section 5.3 and it was found that for all the proposed versions, the population size of 20 is optimal. One more parameter which has been added to the newly proposed algorithms is the $\vec{a}$ parameter. This parameter acts as deciding factor for controlling the extent of exploration and exploitation in case of GWO (Mirjalili et al., 2014). In present case, this parameter is used for second half of the iterations. So that the search equations employing Cauchy mutation operator are able to provide proper exploration operation in the first half of iterations and after that $\vec{a}$ parameter guides the algorithm to converge to global solution towards the second half of iterations. The value of this parameter is generally linearly decreasing in the range of 2 to 0. Apart from this there is no other parameter which is necessary for the operation of the proposed versions. Note that present

Table 15
Comparison of various algorithms at D = 100.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	BA	4.47E+04	1.60E+05	9.48E+04	2.37E+04
	CV 1.0	**7.12E−80**	**3.39E−64**	**5.56E−66**	**3.72E−65**
	DE	2.38E+05	3.17E+05	2.78E+05	1.51E+04
	FA	1.3331	1.05E+02	15.3040	18.2001
	FPA	1.20E+04	2.80E+04	1.87E+04	3.09E+03
	GWO	1.55E−12	9.85E−11	1.25E−11	1.32E−11
	BFP	1.32E+05	3.10E+05	2.56E+05	3.92E+04
	PSOGSA	1.89E+04	8.97E+04	5.53E+04	1.57E+04
f_2	BA	4940	1920	9.80E+04	2.71E+04
	CV 1.0	**0**	**0**	**0**	**0**
	DE	2397	3041	2.79E+05	1.32E+04
	FA	23	347	89.2500	56.5695
	FPA	1303	2630	1.84E+04	2.81E+03
	GWO	0	0	0	0
	BFP	134,762	307,310	2.54E+05	4.08E+04
	PSOGSA	41,230	108,669	7.40E+04	1.61E+04
f_3	BA	1.8989	53.9159	17.0121	10.0332
	CV 1.0	6.48E−05	0.0055	0.0017	0.0011
	DE	54.1812	1.78E+02	1.22E+02	26.3385
	FA	0.0111	0.1976	0.0674	0.0413
	FPA	0.1335	1.5332	0.5436	0.2461
	GWO	1.10E−04	0.0037	0.0014	7.16E−04
	BFP	18.9062	1.66E+02	1.06E+02	28.7360
	PSOGSA	0.0968	0.5750	0.2671	0.0980
f_4	BA	91.5310	2.03E+03	5.14E+02	2.75E+02
	CV 1.0	2.73E−80	1.81E−65	1.87E−67	1.81E−66
	DE	3.59E+02	3.69E+03	8.94E+02	5.94E+02
	FA	1.44E+02	3.44E+02	2.44E+02	38.4330
	FPA	47.9007	1.81E+02	84.5641	22.5370
	GWO	4.77E−15	2.41E−13	4.24E−14	3.94E−14
	BFP	4.41E+02	9.83E+03	1.28E+03	1.14E+03
	PSOGSA	42.8679	2.53E+02	1.04E+02	36.3011
f_5	BA	1.14E−07	8.51E−06	1.25E−06	1.32E−06
	CV 1.0	**6.39E−185**	**6.66E−143**	**7.00E−145**	**6.671E−144**
	DE	0.4643	2.7975	1.6391	0.4847
	FA	2.22E−07	1.79E−05	3.11E−06	3.20E−06
	FPA	3.40E−07	0.0010	4.64E−05	1.24E−04
	GWO	1.98E−70	1.77E−31	1.77E−33	1.77E−32
	BFP	0.1457	2.0068	0.9084	0.4702
	PSOGSA	3.57E−08	1.14E−05	8.72E−07	1.58E−06
f_6	BA	3.13E+08	4.19E+09	1.24E+09	6.45E+08
	CV 1.0	**1.19E−82**	**1.59E−69**	**3.20E−71**	**1.70E−70**
	DE	3.13E+07	3.21E+09	9.38E+08	7.03E+08
	FA	2.54E+06	2.91E+07	1.05E+07	5.22E+06
	FPA	5.13E+06	7.40E+07	2.17E+07	1.44E+07
	GWO	3.62E−25	8.45E−22	9.28E−23	1.49E−22
	BFP	9.38E+08	5.28E+09	2.87E+09	9.06E+08
	PSOGSA	3.44E+05	1.38E+09	1.70E+08	2.50E+08
f_7	BA	5.72E−14	0.4592	0.1647	0.1330
	CV 1.0	**0**	**0**	**0**	**0**
	DE	0.0018	0.4577	0.1202	0.1160
	FA	4.18E−12	0.0031	3.12E−05	3.12E−04
	FPA	1.38E−10	1.04E−05	3.61E−07	1.20E−06
	GWO	0	0	0	0
	BFP	3.50E−04	0.4946	0.3215	0.1224
	PSOGSA	0	0.0123	0.0013	0.0029
f_8	BA	−1.31E+226	−6.28E+80	−1.89E+224	7.89E+123
	CV 1.0	−2.95E+04	−1.54E+04	−1.80E+04	1.60E+03
	DE	−1.96E+126	−5.21E+114	−2.02E+124	1.96E+125
	FA	−2.73E+04	−1.53E+04	−2.04E+04	2.29E+03
	FPA	−3.25E+40	−7.57E+31	−7.21E+38	3.88E+39
	GWO	−8.84E+31	−1.90E+22	−2.62E+30	1.23E+31
	BFP	−9.49E+16	−2.69E+03	−9.57E+14	9.49E+15
	PSOGSA	−1.89E+04	−1.25E+04	−1.54E+04	1.42E+03
f_9	BA	20.0474	20.5516	20.2956	0.1078
	CV 1.0	**8.88E−16**	19.9999	4.7994	8.5828
	DE	20.7544	21.2920	21.0000	0.1004
	FA	20.2002	20.4781	20.3231	0.0532
	FPA	21.3343	21.4573	21.4061	0.0243
	GWO	21.2039	21.4624	21.4067	0.0375
	BFP	19.6578	21.0411	20.7464	0.3105
	PSOGSA	18.9430	19.9667	19.4067	0.2950
f_{10}	BA	1.66E+03	3.00E+04	1.11E+04	5.63E+03
	CV 1.0	1.37E−78	9.53E−15	1.21E−16	9.86E−16

(*continued on next page*)

Table 15 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	DE	5.78E+04	1.42E+05	1.07E+05	1.78E+04
	FA	79.8549	3.60E+02	1.66E+02	56.2953
	FPA	9.99E+02	4.21E+03	2.09E+03	6.07E+02
	GWO	3.76E−06	7.79E−04	1.19E−04	1.13E−04
	BFP	2.17E+04	1.36E+05	9.37E+04	2.40E+04
	PSOGSA	1.21E+02	4.92E+03	7.69E+02	1.02E+03
f_{11}	BA	5.54E+05	9.75E+06	3.49E+06	2.08E+06
	CV 1.0	0.6666	0.6669	0.6667	4.03E−05
	DE	2.15E+07	3.43E+07	2.91E+07	2.74E+06
	FA	2.56E+02	4.13E+03	1.19E+03	8.97E+02
	FPA	7.16E+04	3.68E+05	1.82E+05	6.64E+04
	GWO	0.6666	1.0000	0.6968	0.0958
	BFP	7.64E+06	3.29E+07	2.44E+07	6.81E+06
	PSOGSA	2.60E+04	5.81E+06	7.13E+05	1.17E+06
f_{12}	BA	3.73E+03	2.61E+11	5.64E+09	3.60E+10
	CV 1.0	2.39E+02	1.20E+03	6.24E+02	2.03E+02
	DE	1.34E+04	2.53E+07	1.67E+06	3.32E+06
	FA	1.10E+03	2.68E+03	1.67E+03	2.98E+02
	FPA	4.46E+03	7.88E+04	1.83E+04	1.29E+04
	GWO	1.03E+03	9.18E+03	3.17E+03	1.52E+03
	BFP	1.42E+05	7.74E+13	4.14E+12	1.06E+13
	PSOGSA	1.61E+03	3.28E+03	2.46E+03	2.78E+02
f_{13}	BA	4.09E+06	1.60E+07	7.67E+06	2.19E+06
	CV 1.0	3.41E−78	1.90E−60	1.90E−62	1.90E−61
	DE	2.39E+07	2.97E+07	2.75E+07	1.30E+06
	FA	4.33E+03	3.28E+04	1.13E+04	5.08E+03
	FPA	1.22E+06	2.70E+06	1.83E+06	3.03E+05
	GWO	6.14E−11	4.99E−09	1.24E−09	9.58E−10
	BFP	3.08E+06	7.53E+06	6.27E+06	1.01E+06
	PSOGSA	1.40E+04	5.10E+05	8.94E+04	9.86E+04
f_{14}	BA	4.03E+02	2.08E+03	8.85E+02	2.73E+02
	CV 1.0	**0**	0.0459	4.59E−04	0.0045
	DE	2.02E+03	2.80E+03	2.51E+03	1.62E+02
	FA	0.2492	1.9763	1.0064	0.3483
	FPA	1.07+02	2.38E+02	1.62E+02	23.3319
	GWO	7.41E−13	0.0525	0.0043	0.0117
	BFP	0.2088	3.85E+02	87.3903	76.5991
	PSOGSA	0	0.1954	0.0631	0.0751
f_{15}	BA	2.39E+07	8.35E+08	2.16E+08	1.73E+08
	CV 1.0	**0.0115**	8.1374	0.2423	1.1115
	DE	2.06E+09	3.56E+09	2.86E+09	2.83E+08
	FA	4.6136	24.5477	9.6186	4.1352
	FPA	1.39E+04	4.02E+06	5.70E+05	6.88E+05
	GWO	0.0461	0.8856	0.2668	0.1775
	BFP	3.06E+08	3.48E+09	2.44E+09	8.24E+08
	PSOGSA	4.09E+05	1.02E+09	3.39E+08	2.73E+08
f_{16}	BA	7.76E+07	2.80E+09	5.92E+08	4.36E+08
	CV 1.0	**0.1751**	1.3981	0.6610	0.2394
	DE	4.15E+09	6.64E+09	5.51E+09	5.24E+08
	FA	21.7214	77.9825	44.5877	10.2230
	FPA	1.11E+06	3.08E+07	8.08E+06	5.00E+06
	GWO	0.6879	2.5262	1.7598	0.2889
	BFP	7.03E+08	6.37E+09	4.48E+09	1.40E+09
	PSOGSA	3.80E+07	1.67E+09	5.71E+08	4.17E+08
f_{17}	BA	3.36E+02	7.50E+02	4.91E+02	86.2923
	CV 1.0	**0**	7.11E+02	2.77E+02	3.10E+02,
	DE	1.53E+03	1.76E+03	1.66E+03	53.8458
	FA	2.60E+02	5.05+02	3.78E+02	51.5388
	FPA	7.67E+02	9.21E+02	8.34E+02	32.9445
	GWO	4.55E−08	55.2733	13.7500	8.8135
	BFP	0.0650	17.2668	6.4704	4.1493
	PSOGSA	0	4.9747	0.6765	0.8109

set of parameter values fit better for the benchmark problems only and may not be good for all the problems. So as per the problem requirement, a standard parametric study is required to be done before the application of these set of parameters.

5.9. Summary of results

5.9.1. Main findings from the obtained results

The basic CS algorithm though is competitive but is limited in scope. Also, the already available works on either applications or modifications don't provide proper justification. So in order to improve its performance, three versions of CS algorithms namely CV 1.0, CV 2.0 and CV 3.0 are proposed in this work and it has been found that CV 1.0 is best among them. The performance of CV 1.0 is better because of the presence of Cauchy based exploration which because of its heavy tail, searches the search space in a much better way and helps to improve the exploration of basic CS algorithm. Further, use of dual local search makes the algorithm efficient in exploiting the search to the best of its potential. Both these phases add up to improve the performance of CS algorithm.

Table 16
Comparison of various algorithms for D = 500.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	BA	2.75E+05	9.43E+05	5.27E+05	1.45E+05
	CV 1.0	**8.89E−73**	**7.68E−60**	**2.15E−61**	**9.86E−61**
	DE	1.44E+06	1.62E+06	1.54E+06	3.40E+04
	FA	1.67E+05	3.15E+05	2.23E+05	2.62E+04
	FPA	8.78E+04	1.30E+05	1.09E+05	9.34E+03
	GWO	0.0058	0.0685	0.0203	0.0108
	BFP	6.56E+05	1.59E+06	1.30E+06	3.02E+05
	PSOGSA	7.87E+05	9.97E+05	8.96E+05	4.47E+04
f_2	BA	2886	9200	5.09E+05	1.42E+05
	CV 1.0	**0**	**0**	**0**	**0**
	DE	1478	1597	1.54E+06	2.80E+04
	FA	1811	3013	2.33E+05	2.67E+04
	FPA	8700	1456	1.10E+05	1.04E+04
	GWO	0	28	6.3500	4.9305
	BFP	534,485	1,604,304	1.32E+06	2.96E+05
	PSOGSA	817,585	1,069,668	9.46E+05	5.64E+04
f_3	BA	3.2426	41.0137	15.1707	8.6836
	CV 1.0	**1.88E−04**	0.0087	0.0018	0.0014
	DE	51.6107	1.87E+02	1.18E+02	28.0228
	FA	0.0235	0.2053	0.0902	0.0393
	FPA	0.1516	2.0431	0.5759	0.3115
	GWO	3.13E−04	0.0037	0.0014	7.35E−04
	BFP	24.0323	1.58E+02	1.02E+02	27.2077
	PSOGSA	0.1059	8.3756	0.3607	0.8159
f_4	BA	9.33E+02	6.15E+03	2.35E+03	9.86E+02
	CV 1.0	4.92E−75	4.09E−62	5.27E−64	4.14E−63
	DE	2.21E+03	1.07E+04	3.87E+03	1.78E+03
	FA	1.58E+03	1.98E+03	1.79E+03	83.4352
	FPA	2.91E+02	7.12E+02	4.34E+02	88.5107
	GWO	2.02E−05	3.22E−04	7.58E−05	5.30E−05
	BFP	2.19E+03	9.95E+03	4.45E+03	9.26E+02
	PSOGSA	1.15E+03	2.24E+03	1.63E+03	2.55E+02
f_5	BA	1.42E−07	1.16E−05	1.38E−06	1.54E−06
	CV 1.0	2.25E−181	3.89E−137	3.89E−139	3.89E−138
	DE	1.5427	4.1305	2.7195	0.5583
	FA	8.88E−06	1.8202	0.5153	0.3753
	FPA	5.12E−07	0.0013	9.96E−05	1.73E−04
	GWO	6.15E−12	0.0042	1.02E−04	4.47E−04
	BFP	0.2005	3.2416	1.4363	0.7457
	PSOGSA	4.36E−07	0.0147	0.0022	0.0025
f_6	BA	1.79E+08	3.56E+09	1.31E+09	7.27E+08
	CV 1.0	**1.31E−82**	**4.89E−66**	**4.91E−68**	**4.89E−67**
	DE	8.43E+07	4.50E+09	1.59E+09	1.13E+09
	FA	3.04E+06	3.16E+07	9.77E+06	4.94E+06
	FPA	4.55E+06	5.23E+07	2.13E+07	1.14E+07
	GWO	8.84E−25	2.21E−21	1.07E−22	2.73E−22
	BFP	1.08E+09	5.49E+09	2.65E+09	9.29E+08
	PSOGSA	5.68E+05	1.67E+09	1.09E+08	2.17E+08
f_7	BA	8.35E−05	0.4779	0.1604	0.1255
	CV 1.0	**0**	**0**	**0**	**0**
	DE	3.52E−04	0.4864	0.1508	0.1034
	FA	3.37E−12	0.0024	2.40E−05	2.40E−04
	FPA	3.66E−10	2.57E−06	2.03E−07	3.60E−07
	GWO	0	0	0	0
	BFP	0.0063	0.4918	0.3504	0.1225
	PSOGSA	0	0.0242	0.0010	0.0032
f_8	BA	−2.10E+227	−2.11E+129	−2.47E+225	4.5E+153
	CV 1.0	−1.49E+05	−1.06E+05	−1.41E+05	6.58E+03
	DE	−6.29E+122	−7.10E+106	−1.30E+121	7.09E+121
	FA	−9.15E+04	−5.80E+04	−7.16E+04	6.89E+03
	FPA	−6.60E+41	−2.10E+32	−9.91E+39	6.76E+40
	GWO	−2.62E+30	−4.87E+04	−1.39E+29	4.92E+29
	BFP	−3.77E+17	−8.52E+03	−5.26E+15	3.97E+16
	PSOGSA	−4.51E+04	−2.64E+04	−3.60E+04	3.64E+03
f_9	BA	20.9511	21.2094	21.1278	0.0490
	CV 1.0	**8.88E−16**	19.9999	5.1975	8.8126
	DE	21.3177	21.5686	21.4081	0.0468
	FA	20.8617	21.0433	20.9500	0.0406
	FPA	21.5111	21.6075	21.5873	0.0129
	GWO	21.5544	21.6111	21.5866	0.0120
	BFP	18.9093	21.1416	20.9132	0.3458
	PSOGSA	19.9598	20.5152	20.2136	0.1810
f_{10}	BA	2.61E+04	2.60E+05	9.08E+04	4.50E+04
	CV 1.0	5.62E−75	9.30E−20	9.30E−22	9.30E−21

(*continued on next page*)

Table 16 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	DE	5.97E+05	8.46E+05	7.61E+05	4.67E+04
	FA	1.99E+04	3.94E+04	2.99E+04	4.48E+03
	FPA	1.11E+04	2.06E+04	1.43E+04	1.83E+03
	GWO	0.0142	35.1084	1.6992	5.2841
	BFP	1.77E+05	8.57E+05	6.12E+05	1.97E+05
	PSOGSA	2.55E+04	7.01E+04	4.62E+04	8.79E+03
f_{11}	BA	2.53E+07	3.41E+08	1.13E+08	5.55E+07
	CV 1.0	0.6666	0.6673	0.6667	1.06E−04
	DE	7.87E+08	9.55E+08	8.73E+08	3.49E+07
	FA	1.59E+07	5.32E+07	3.05E+07	6.86E+06
	FPA	4.30E+06	1.25E+07	6.95E+06	1.80E+06
	GWO	10.5367	2.30E+03	1.75E+02	3.11E+02
	BFP	1.88E+08	9.47E+08	7.56E+08	2.08E+08
	PSOGSA	6.90E+07	2.08E+08	1.15E+08	2.59E+07
f_{12}	BA	1.48E+05	1.47E+17	2.94E+15	1.79E+16
	CV 1.0	5.18E+03	1.00E+10	1.06E+08	1.00E+09
	DE	9.80E+05	7.34E+09	5.92E+08	1.06E+09
	FA	1.07E+04	1.95E+04	1.42E+04	1.61E+03
	FPA	1.37E+05	6.82E+07	3.39E+06	8.09E+06
	GWO	4.85E+04	2.53E+05	1.07E+05	4.12E+04
	BFP	5.04E+07	6.05E+19	3.73E+18	9.75E+18
	PSOGSA	1.57E+04	1.82E+04	1.70E+04	4.91E+02
f_{13}	BA	2.41E+07	8.78E+07	4.75E+07	1.39E+07
	CV 1.0	3.54E−71	1.63E−56	1.92E−58	1.64E−57
	DE	1.42E+08	1.61E+08	1.53E+08	3.28E+06
	FA	1.64E+07	2.85E+07	2.26E+07	2.71E+06
	FPA	8.83E+06	1.51E+07	1.11E+07	1.08E+06
	GWO	0.5426	5.3134	1.9521	0.9628
	BFP	1.22E+07	3.99E+07	3.17E+07	7.82E+06
	PSOGSA	7.20E+06	1.27E+07	9.94E+06	1.30E+06
f_{14}	BA	1.88E+03	1.09E+04	4.62E+03	1.47E+03
	CV 1.0	**0**	**0**	**0**	**0**
	DE	1.29E+04	1.45E+04	1.38E+04	3.12E+02
	FA	1.48E+03	2.76E+03	2.04E+03	2.36E+02
	FPA	7.84E+02	1.49E+03	1.00E+03	1.11E+02
	GWO	7.78E−04	0.2018	0.0299	0.0617
	BFP	3.3305	3.16E+02	69.9947	66.4696
	PSOGSA	0	0.3870	0.0743	0.0817
f_{15}	BA	1.36E+08	5.55E+09	1.43E+09	9.70E+08
	CV 1.0	**0.0156**	0.4033	0.0684	0.0578
	DE	1.58E+10	1.93E+10	1.78E+10	7.05E+08
	FA	5.66E+07	4.40E+08	1.83E+08	7.44E+07
	FPA	2.30E+06	3.88E+07	9.81E+06	5.34E+06
	GWO	0.3697	16.4043	4.0357	3.0288
	BFP	1.51E+09	1.92E+10	1.36E+10	5.60E+09
	PSOGSA	4.11E+09	7.99E+09	6.22E+09	8.89E+08
f_{16}	BA	6.02E+08	1.29E+10	3.52E+09	2.23E+09
	CV 1.0	**0.4359**	1.00E+10	1.00E+08	9.99E+08
	DE	2.78E+10	3.46E+10	3.25E+10	1.28E+09
	FA	2.89E+08	1.28E+09	6.85E+08	2.09E+08
	FPA	3.19E+07	1.58E+08	7.83E+07	2.36E+07
	GWO	2.4954	29.4601	8.2120	4.7102
	BFP	3.57E+09	3.45E+10	2.62E+10	9.34E+09
	PSOGSA	8.59E+09	1.60E+10	1.22E+10	1.72E+09
f_{17}	BA	3.64E+03	6.60E+03	4.47E+03	4.59E+02
	CV 1.0	**0**	2.1831	0.0218	0.2183
	DE	8.62E+03	9.10E+03	8.86E+03	1.08E+02
	FA	3.61E+03	4.49E+03	4.00E+03	1.71E+02
	FPA	4.76E+03	5.26E+03	4.95E+03	92.9846
	GWO	82.0936	4.38E+02	1.77E+02	67.0061
	BFP	0.1141	17.9241	6.7843	4.2555
	PSOGSA	0	1.9899	0.6367	0.6560

Another reason for the improved performance is the division of generations into two halves, which helps to infuse a proper balance between exploration and exploitation. Thus, a combination of these two modifications makes CV 1.0 an efficient tool for optimization problems. Apart from finding a new state-of-the-art algorithm, this paper also finds that the switching probability is an important factor in deciding the solution quality of basic CS algorithm. Too low and too high values of switch probability degrade the performance of CS whereas, for mid-range of values, the change in results is marginal. Also from the results showing the effect of varying population and dimension sizes, it is evident that the proposed version is a very effective tool in finding global optima even with lower population and higher dimensional problems.

5.9.2. Limitations

Apart from the above discussed advantages, the proposed version has some limitations also. It can be interpreted from the above discussion that CV 1.0 is very efficient in performance but

Table 17
Comparison of various algorithms for D = 1000.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
f_1	BA	6.16E+05	1.73E+06	1.00E+06	2.41E+05
	CV 1.0	**6.98E−71**	**6.21E−55**	**6.25E−57**	**6.21E−56**
	DE	3.03E+06	3.24E+06	3.15E+06	4.65E+04
	FA	7.28E+05	9.76E+05	8.41E+05	5.76E+04
	FPA	1.91E+05	2.79E+05	2.31E+05	2.03E+04
	GWO	4.8161	1.22E+02	17.2584	13.7434
	BFP	1.15E+06	3.27E+06	2.87E+06	5.33E+05
	PSOGSA	1.95E+06	2.44E+06	2.23E+06	8.29E+04
f_2	BA	5017	1975	1.03E+06	3.08E+05
	CV 1.0	**0**	**0**	**0**	**0**
	DE	2952	3244	3.15E+06	4.99E+04
	FA	7303	1013	8.59E+05	6.52E+04
	FPA	1849	2988	2.31E+05	2.15E+04
	GWO	75	268	1.51E+02	37.9868
	BFP	1.23E+06	3.22E+06	2.82E+06	5.08E+05
	PSOGSA	2.07E+06	2.51E+06	2.29E+06	9.74E+04
f_3	BA	2.5713	49.4945	16.6947	8.9776
	CV 1.0	6.14E−05	0.0105	0.0020	0.0016
	DE	66.6980	1.88E+02	1.27E+02	24.0771
	FA	0.0259	0.3333	0.1005	0.0583
	FPA	0.1789	1.6665	0.5790	0.2961
	GWO	3.12E−04	0.0048	0.0014	8.21E−04
	BFP	37.6594	1.77E+02	1.08E+02	34.1199
	PSOGSA	0.1181	0.5537	0.2713	0.1000
f_4	BA	1.81E+03	1.20E+04	4.40E+03	1.94E+03
	CV 1.0	2.30E−76	1.12E−	1.61E−58	1.22E−57
	DE	4.36E+03	561.73E+04	6.84E+03	2.15E+03
	FA	3.47E+03	4.15E+03	3.81E+03	1.48E+02
	FPA	5.68E+02	1.23E+03	8.21E+02	1.44E+02
	GWO	0.0110	2.1201	0.1990	0.3308
	BFP	5.46E+03	2.31E+04	8.73E+03	1.91E+03
	PSOGSA	2.62E+03	5.85E+03	3.99E+03	6.36E+02
f_5	BA	9.17E−08	1.88E−05	1.58E−06	2.26E−06
	CV 1.0	7.29E−184	2.74E−129	2.74E−131	2.74E−130
	DE	1.6562	4.6239	3.1907	0.6460
	FA	0.1417	2.5948	1.1232	0.5222
	FPA	2.64E−06	0.0051	3.89E−04	7.58E−04
	GWO	6.54E−06	0.0503	0.0029	0.0066
	BFP	0.0922	3.1202	1.5265	0.7957
	PSOGSA	9.04E−04	0.1475	0.0154	0.0168
f_6	BA	1.99E+08	2.89E+09	1.20E+09	6.10E+08
	CV 1.0	**8.84E−82**	**9.52E−69**	**1.05E−70**	**9.51E−70**
	DE	7.94E+07	5.19E+09	1.87E+09	1.18E+09
	FA	2.32E+06	2.56E+07	9.64E+06	5.06E+06
	FPA	5.37E+06	9.35E+07	2.16E+07	1.60E+07
	GWO	3.45E−25	1.59E−21	1.62E−22	2.81E−22
	BFP	7.58E+08	5.81E+09	2.76E+09	9.41E+08
	PSOGSA	7.13E+05	8.51E+08	1.37E+08	1.87E+08
f_7	BA	5.08E−14	0.4731	0.1443	0.1255
	CV 1.0	**0**	**0**	**0**	**0**
	DE	9.80E−05	0.4600	0.1304	0.1113
	FA	1.74E−13	0.0031	3.12E−05	3.12E−04
	FPA	1.56E−10	5.78E−06	3.51E−07	8.62E−07
	GWO	0	0	0	0
	BFP	0.0369	0.4934	0.3347	0.1175
	PSOGSA	0	0.0183	0.0013	0.0040
f_8	BA	−9.69E+226	−1.98E+37	−1.87E+225	1.89E+156
	CV 1.0	−3.85E+05	−2.32E+05	−2.836E+05	1.69E+04
	DE	−1.89E+121	−1.62E+105	−2.02E+119	1.89E+120
	FA	−1.46E+05	−9.22E+04	−1.10E+05	1.00E+04
	FPA	−4.96E+41	−1.03E+33	−8.97E+39	5.53E+40
	GWO	−3.75E+30	−7.17E+04	−9.05E+28	4.87E+29
	BFP	−4.89E+18	−7.74E+03	−4.90E+16	4.89E+17
	PSOGSA	−6.22E+04	−3.84E+04	−5.12E+04	5.10E+03
f_9	BA	21.2019	21.3704	21.3139	0.0378
	CV 1.0	**8.88E−16**	20.0010	4.7108	8.4537
	DE	21.4391	21.5833	21.5137	0.0295
	FA	21.1118	21.2335	21.1731	0.0252
	FPA	21.6099	21.6409	21.6283	0.0070
	GWO	21.5870	21.6437	21.6268	0.0094
	BFP	19.7796	21.1701	20.8978	0.3517
	PSOGSA	19.9643	20.6917	20.3533	0.2685
f_{10}	BA	6.56E+04	5.59E+05	2.06E+05	1.02E+05
	CV 1.0	5.65E−71	8.10E−34	8.10E−36	8.10E−35

(*continued on next page*)

Table 17 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	DE	1.48E+06	1.77E+06	1.64E+06	6.58E+04
	FA	9.79E+04	1.79E+05	1.34E+05	1.54E+04
	FPA	2.26E+04	4.55E+04	3.07E+04	3.57E+03
	GWO	48.6781	7.16E+03	6.94E+02	1.08E+03
	BFP	3.40E+05	1.78E+06	1.32E+06	4.33E+05
	PSOGSA	1.32E+05	2.49E+05	1.79E+05	2.11E+04
f_{11}	BA	1.26E+08	1.32E+09	5.25E+08	2.42E+08
	CV 1.0	0.6666	0.6676	0.6667	1.16E−04
	DE	3.43E+09	3.85E+09	3.65E+09	9.50E+07
	FA	2.36E+08	5.18E+08	3.60E+08	4.83E+07
	FPA	1.90E+07	4.86E+07	3.06E+07	5.96E+06
	GWO	2.98E+04	1.07E+07	6.56+05	1.27E+06
	BFP	7.11E+08	3.85E+09	3.02E+09	9.54E+08
	PSOGSA	7.03E+08	1.24E+09	9.24E+08	9.85E+07
f_{12}	BA	5.93E+04	1.05E+20	2.41E+18	1.27E+19
	CV 1.0	1.22E+04	1.00E+10	3.21E+09	4.41E+09
	DE	3.58E+07	1.41E+13	3.69E+11	1.82E+12
	FA	2.70E+04	4.20E+04	3.31E+04	3.44E+03
	FPA	7.39E+05	1.80E+09	5.30E+07	1.92E+08
	GWO	1.26E+05	1.06E+06	3.34E+05	1.43E+05
	BFP	5.13E+15	2.76E+22	1.44E+21	3.51E+21
	PSOGSA	3.25E+04	6.65E+18	3.61E+17	1.17E+18
f_{13}	BA	5.29E+07	1.60E+08	9.47E+07	2.13E+07
	CV 1.0	1.72E−68	1.88E−53	1.89E−55	1.88E−54
	DE	2.99E+08	3.25E+08	3.15E+08	4.41E+06
	FA	7.20E+07	9.87E+07	8.40E+07	5.95E+06
	FPA	1.87E+07	3.11E+07	2.32E+07	2.27E+06
	GWO	3.64E+02	3.04E+04	1.97E+03	3.04E+03
	BFP	3.61E+07	8.07E+07	6.82E+07	1.41E+07
	PSOGSA	2.53E+07	4.22E+07	3.20E+07	3.18E+06
f_{14}	BA	4.64E+03	1.53E+04	8.51E+03	2.39E+03
	CV 1.0	**0**	0.0054	5.46E−05	5.46E−04
	DE	2.75E+04	2.95E+04	2.84E+04	4.14E+02
	FA	6.51E+03	8.59E+03	7.52E+03	5.32E+02
	FPA	1.71E+03	2.50E+03	2.05E+03	1.82E+02
	GWO	0.2867	3.4215	0.8316	0.3815
	BFP	3.6341	3.32E+02	77.2679	64.8337
	PSOGSA	0	0.5364	0.0865	0.1138
f_{15}	BA	5.39E+08	1.28E+10	2.87E+09	2.34E+09
	CV 1.0	**0.0188**	**0.4857**	**0.0957**	**0.0724**
	DE	3.46E+10	3.89E+10	3.69E+10	9.60E+08
	FA	1.28E+09	3.54E+09	2.10E+09	4.68E+08
	FPA	9.73E+06	6.65E+07	2.57E+07	1.19E+07
	GWO	22.0601	5.69E+07	1.22E+06	5.80E+06
	BFP	5.37E+09	3.92E+10	2.93E+10	1.07E+10
	PSOGSA	1.45E+10	2.45E+10	1.97E+10	1.67E+09
f_{16}	BA	1.45E+09	2.08E+10	6.55E+09	3.78E+09
	CV 1.0	**0.6417**	1.00E+10	5.33E+09	4.97E+09
	DE	6.21E+10	7.18E+10	6.75E+10	1.85E+09
	FA	3.44E+09	7.45E+09	5.47E+09	8.95E+08
	FPA	1.05E+08	3.46E+08	1.94E+08	5.57E+07
	GWO	3.28E+02	1.29E+08	5.00E+06	1.43E+07
	BFP	2.45E+10	7.15E+10	5.61E+10	1.66E+10
	PSOGSA	2.89E+10	4.49E+10	3.71E+10	2.95E+09
f_{17}	BA	8.38E+03	1.49E+04	9.88E+03	8.48E+02
	CV 1.0	**0**	**0**	**0**	**0**
	DE	1.75E+04	1.82E+04	1.79E+04	1.45E+02
	FA	9.13E+03	1.04E+04	9.73E+03	2.36E+02
	FPA	9.76E+03	1.04E+04	1.01E+04	1.20E+02
	GWO	4.16E+02	4.60E+03	1.06E+03	5.83E+02
	BFP	1.2067	18.0110	6.8489	4.2427
	PSOGSA	0	3.9798	0.7462	0.8760

due to its stochastic nature, it is not guaranteed that it will find the global optima for all the cases. It can be seen from the comparison table that it didn't perform well on all the benchmark functions. So, more work is to be done to design a standard problem solver for all optimization problems. A balanced exploration and exploitation are the basis for this proposal, but the algorithm can get stuck in local minima at later stages due to less extensive global search. Also for a balanced exploration and exploitation, the algorithm should be able to explore and exploit in every iteration. This can be considered as a drawback and more work should be done to improve the balance between exploration and exploitation phase. Though CV 1.0 is able to provide far better final results as compared to other algorithms for most of the cases, it takes more execution time because of the use of more expensive exploration operation during the initial phases. It utilizes less number of exploitation operations which take less time compared to expensive exploration operations at the initial stages. Because of this CV, 1.0 takes more time for its execution. So more work is required to be

Table 18
Summary of CEC 2015 Learning-Based Benchmark Suite.

	No.	Functions	Global Minima
Unimodal Function	F_1	Rotated High Conditional Elliptic Function	100
	F_2	Rotated Cigar Function	200
Simple Multimodal Functions	F_3	Shifted and Rotated Ackley's Function	300
	F_4	Shifted and Rotated Rastrigin's Function	400
	F_5	Shifted and Rotated Schwefel's Function	500
Hybrid Functions	F_6	Hybrid Function 1 (N = 3)	600
	F_7	Hybrid Function 2 (N = 4)	700
	F_8	Hybrid Function 3 (N = 5)	800
Composition Functions	F_9	Composition Function 1 (N = 3)	900
	F_{10}	Composition Function 2 (N = 3)	1000
	F_{11}	Composition Function 3 (N = 5)	1100
	F_{12}	Composition Function 4 (N = 5)	1200
	F_{13}	Composition Function 5 (N = 5)	1300
	F_{14}	Composition Function 6 (N = 7)	1400
	F_{15}	Composition Function 7 (N = 10)	1500
	Search Range: $[-100, 100]^D$		

Table 19
Comparison results of various algorithms for CEC 2105 test functions.

Function	Algorithm	Best	Worst	Mean	Standard Deviation
F_1	DE	1.44E+05	2.31E+08	2.91E+07	3.85E+07
	BA	1.92E+06	4.70E+08	6.64E+07	7.12E+07
	BFP	5.72E+07	8.30E+08	2.78E+08	1.59E+08
	FPA	4.22E+07	9.10E+08	3.32E+08	1.84E+08
	FA	7.13E+04	4.19E+06	1.10E+06	8.67E+05
	GWO	1.62E+05	1.77E+07	3.03E+06	2.86E+06
	PSOGSA	358.7040	3.45E+07	1.65E+06	5.41E+06
	CV 1.0	100.0000	100.0000	100.0000	5.40E−08
F_2	DE	5.75E+04	2.95E+09	5.17E+08	6.46E+08
	BA	1.63E+08	1.88E+10	6.43E+09	3.59E+09
	BFP	4.15E+09	3.70E+10	1.77E+10	6.31E+09
	FPA	4.69E+09	4.02E+10	1.97E+10	6.70E+09
	FA	252.6822	5.94E+04	7.82E+03	9.26E+03
	GWO	4.32E+03	2.52E+07	5.81E+06	5.75E+06
	PSOGSA	240.2789	320.4601	2.63E+08	6.78E+08
	CV 1.0	200.0000	200.0008	200.0002	1.30E−04
F_3	DE	320.0035	320.5480	320.2813	1.08E−01
	BA	319.9940	320.0000	319.9999	6.54E−04
	BFP	320.0003	320.1955	320.0443	4.17E−02
	FPA	320.5183	321.1876	320.9243	1.67E−01
	FA	319.9998	320.0009	320.0005	1.89E−04
	GWO	320.1403	320.5536	320.4326	7.06E−02
	PSOGSA	319.9990	487.5766	320.1041	1.14E−01
	CV 1.0	320.0421	320.3096	320.1748	5.82E−02
F_4	DE	410.0479	463.3382	431.5531	11.0855
	BA	418.9042	517.4032	450.1331	19.4060
	BFP	454.0100	565.1796	519.1702	21.2106
	FPA	479.3889	579.3497	536.2333	17.4309
	FA	401.9899	422.8841	408.7955	4.01910
	GWO	404.3039	429.3845	412.0920	4.9225
	PSOGSA	413.9294	2521.0	441.3709	16.1076
	CV 1.0	403.9798	420.8941	412.8543	3.9531
F_5	DE	1.01E+03	2845.2	1950.6	358.2341
	BA	884.1138	2678.5	1790.4	381.4069
	BFP	2134.3	3370.8	2751.5	253.6386
	FPA	2390.7	3510.2	3006.2	228.1936
	FA	500.3136	1589.4	921.1287	280.8406
	GWO	506.9979	2358.2	972.4691	297.2292
	PSOGSA	747.1840	2997.7	1446.7	350.2649
	CV 1.0	511.8313	1174.2	910.4903	132.0275
F_6	DE	1.30E+03	5.49E+06	7.97E+05	1.00E+06
	BA	1233.6	5.62E+07	2.25E+06	7.25E+06
	BFP	1.35E+07	1.45E+08	2.06E+07	2.70E+07
	FPA	8.64E+04	9.41E+07	2.25E+07	2,26E+07
	FA	1006.2	1.52E+04	3414.4	2.59E+03
	GWO	1.19E+03	2.22E+06	2.07E+04	3.37E+04

(continued on next page)

Table 19 (*continued*)

Function	Algorithm	Best	Worst	Mean	Standard Deviation
	PSOGSA	1080.9	2.99E+06	9.79E+04	3.87E+05
	CV 1.0	618.2082	683.9049	645.6423	14.1172
F_7	DE	701.6371	709.2557	702.4613	8.45E−01
	BA	705.1089	785.8340	723.1036	17.0529
	BFP	711.3150	886.4356	764.9969	37.8525
	FPA	715.5698	874.1637	768.9116	36.7324
	FA	701.0531	703.7105	702.2835	6.27E−01
	GWO	700.2994	704.6584	702.0322	7.58E−01
	PSOGSA	701.5662	708.4312	704.1815	1.3584
	CV 1.0	700.5005	701.8459	701.4104	2.54E−01
F_8	DE	893.0346	1.60E+06	1.94E+05	2.64E+06
	BA	1070.5	1.17E+07	3.43E+05	1.27E+06
	BFP	2865.8	5.46E+07	3.93E+06	6.87E+06
	FPA	2.87E+04	1.99E+07	3.58E+06	3.55E+06
	FA	933.3901	8.88E+03	2151.3	1.20E+03
	GWO	1.06E+03	4.94E+04	3.07E+03	4.85E+03
	PSOGSA	856.1880	3.09E+05	8878.5	3.09E+04
	CV 1.0	802.3415	813.5618	806.2792	2.6958
F_9	DE	1000.2	1011.1	1001.4	2.5519
	BA	1002.2	1151.6	1036.7	23.0363
	BFP	1015.4	1192.0	1090.4	38.0011
	FPA	1026.3	1172.4	1089.3	29.7804
	FA	1000.1	1000.4	1000.3	7.61E−02
	GWO	1000.1	1001.7	1000.4	1.91E−01
	PSOGSA	1000.2	1020.1	11,002.1	3.9099
	CV 1.0	1000.1	1000.3	1000.2	4.65E−02
F_{10}	DE	1286.0	8.43e+05	1.34E+05	1.97E+05
	BA	2831.8	6.51e+06	4.41E+05	1.09E+06
	BFP	7612.5	4.13E+07	3.28E+06	5.74E+06
	FPA	1.13E+04	1.18E+07	3.06E+06	2.71E+06
	FA	1328.5	6464.4	2277.1	1.05E+03
	GWO	1387.0	1651.6	4076.8	3.37E+03
	PSOGSA	1284.0	5.01E+04	5.665.4	7.92E+03
	CV 1.0	1117.5	1218.9	1205.2	20.3361
F_{11}	DE	1281.9	1772.7	1600.0	132.4487
	BA	1163.7	1874.7	1554.9	167.9876
	BFP	1415.9	1951.3	1701.3	145.9144
	FPA	1437.5	1981.4	1722.7	127.9895
	FA	1105.1	1403.6	1314.5	134.3969
	GWO	1101.4	1571.5	1302.2	80.0037
	PSOGSA	1110.5	1738.3	1438.7	97.0184
	CV 1.0	1101.2	1400.1	1240.0	148.4542
F_{12}	DE	1302.5	1400.0	1323.5	18.1248
	BA	1307.3	1365.3	1331.9	12.4189
	BFP	1325.3	1400.0	1359.5	13.9248
	FPA	1322.8	1399.3	1361.3	13.3479
	FA	1300.6	1303.2	1301.5	4.48E−01
	GWO	1301.0	1305.1	1302.2	6.34E−01
	PSOGSA	1301.8	1330.7	1306.4	5.8158
	CV 1.0	1300.9	1303.4	1302.2	4.26E−01
F_{13}	DE	1300.0	1300.0	1300.0	2.19E−04
	BA	1300.0	1300.0	1300.0	6.04E−05
	BFP	1300.0	1300.1	1300.0	6.10E−03
	FPA	1300.0	1427.5	1303.5	14.5165
	FA	1300.0	1300.0	1300.0	1.24E−04
	GWO	1300.0	1300.0	1300.0	2.41E−04
	PSOGSA	1300.0	1300.6	1300.0	5.99E−02
	CV 1.0	1300.0	1300.0	1300.0	8.55E−04
F_{14}	DE	1715.8	2851.5	1955.2	257.6795
	BA	2517.6	2.32E+04	1.30E+04	4.73E+03
	BFP	7312.1	2.88E+04	1.95E+04	4.87E+03
	FPA	7672.2	2.86E+04	1.80E+04	4.61E+03
	FA	1500.8	7919.8	2477.8	1.32E+03
	GWO	1715.6	1045.7	2763.6	1.86E+03
	PSOGSA	1500.0	1674.8	7926.4	3.316.3
	CV 1.0	1500.0	4364.5	3201.0	1.39E+03
F_{15}	DE	1600.2	1649.8	1612.5	11.4117
	BA	1650.2	3604.1	1870.9	330.5614
	BFP	1668.5	3.10E+04	5749.2	4.55E+03
	FPA	1802.1	2.89E+04	7538.6	5.97E+03
	FA	1600.0	1600.0	1600.0	1.60E−03
	GWO	1600.0	1627.1	1607.5	4.8781
	PSOGSA	1600.0	1819.2	1612.2	28.0078
	CV 1.0	1600.0	1600.0	1600.0	7.04E−12

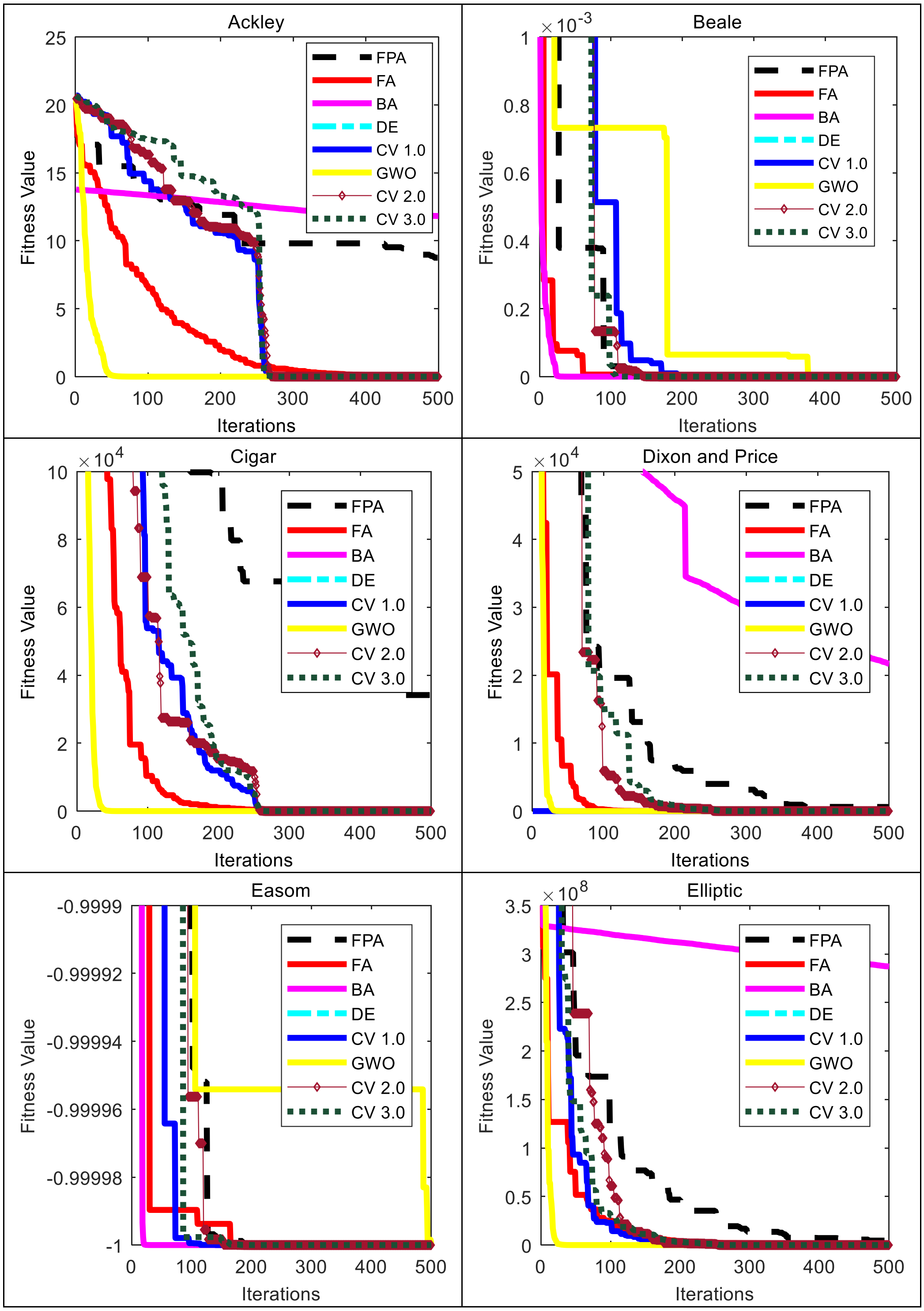

Fig. 2. Convergence profiles of various algorithms.

done, in order to design an algorithm which is faster as well as efficient in achieving global minima.

5.9.3. Insightful implications

Though the algorithm has some limitations it has proven its worth with respect to other state-of-the-art algorithms. The algorithm because of its simple structure is the best fit candidate for inclusion in hybrid intelligent and expert systems. Also, exploration and exploitation can be improved to make the algorithm fit for all engineering design problems. Further insightful implications include, designing binary version of the proposed work for the field of medical science, wireless sensor networks and others. In the field of medical science, binary CV 1.0 can be used in electroencephalogram (EEG) for reducing the total number of sensors required for its operation while maintaining the accuracy. In wireless sensor networks, it can be used to find the energy efficiency of different nodes in the system. Also, the proposed algorithms have been designed for continuous optimization problems, more

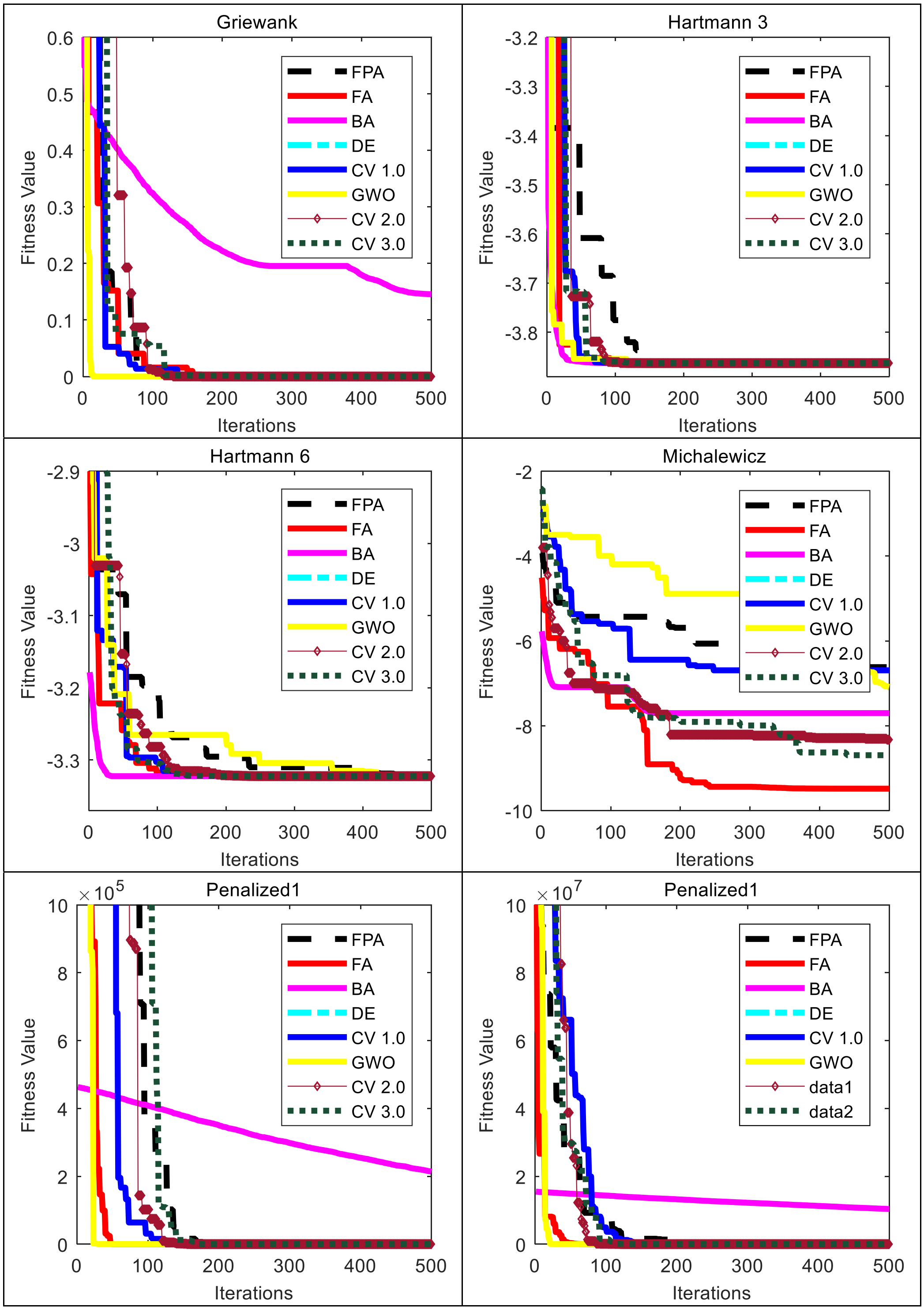

Fig. 2. Continued

work can be done to extend the algorithm to niching, dynamic and constraint optimization problems. The proposed versions can also be extended to solve some multi-objective optimization problems. The multi-objective CS algorithms are very limited in number. In future, some efficient multi-objective CS algorithms will be developed which can be further applied for solving various engineering design problems. In future, the convergence properties of CS algorithms will also be studied. Some theoretical analysis will be done in this direction.

6. Conclusions and future works

This paper deals with enhancing the performance of CS by proposing three new variants of the original algorithm. One of the problems with CS is that it may converge prematurely due to low exploration capability in some conditions. To avoid this problem, the Cauchy operator has been used in the proposed modified versions of the CS. A new notion of dividing the population and generations has been done so as to balance diversification and intensi-

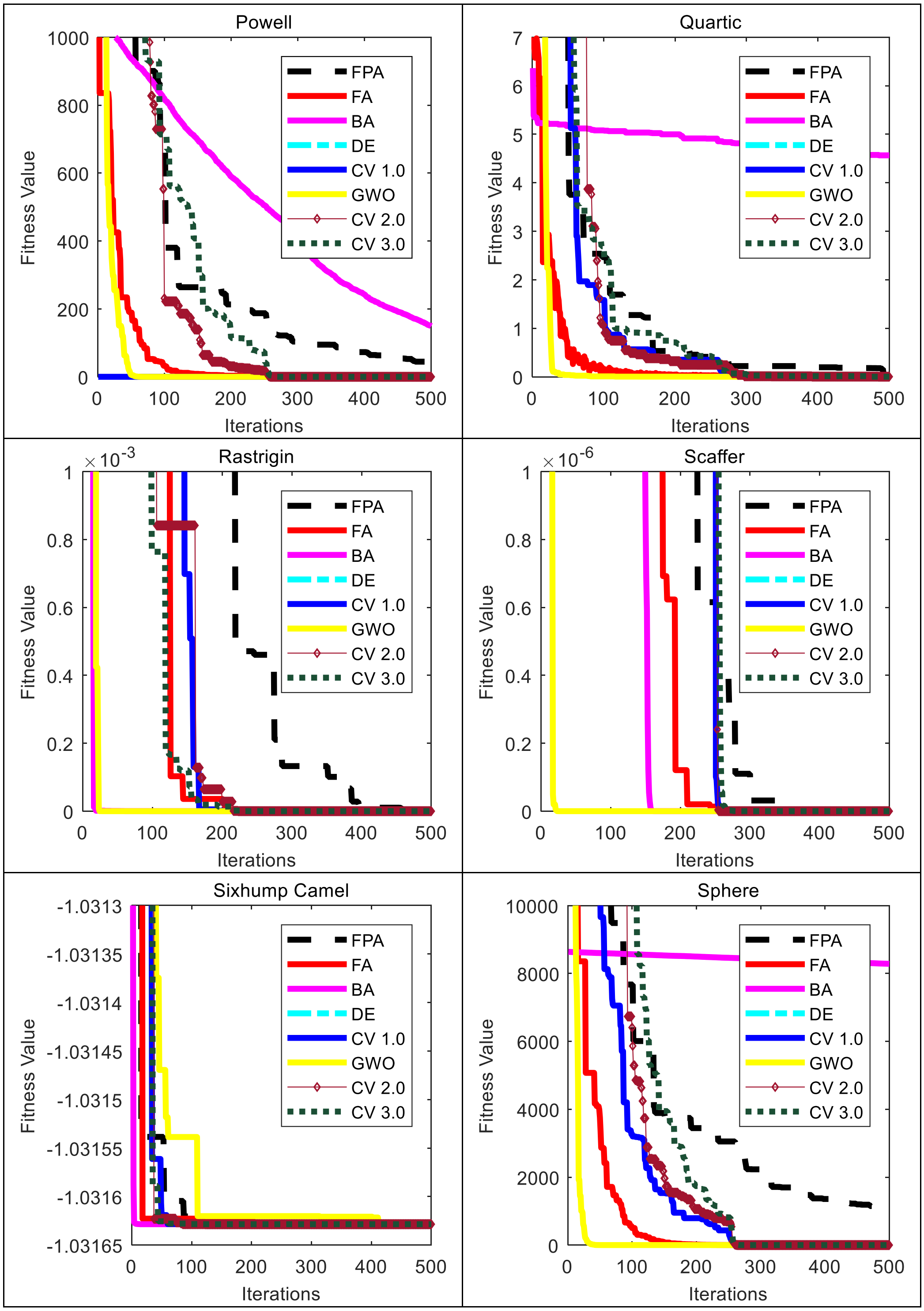

Fig. 2. Continued

fication. These modifications have resulted in significant enhancement of performance of CS. The enhanced versions of CS namely CV 1.0, CV 2.0 and CV 3.0 have been tested on various benchmark functions. A study of the increase in the effect of the population has been carried out and it is observed that the improvement is modest with respect to the increase in the computational cost. The new algorithms have been also subjected to same benchmark functions having high dimensions and again there was little effect in their performance showing their robustness to optimize functions with large variables. Apart from this, different experiments were carried out to see the effect of probability switch on the performance of basic CS and the improved algorithms. The basic CS is found to be very sensitive to probability switch with its best performance at mid values of the switch. But in the case of all the three new algorithms, there was hardly any change in the performance of algorithms in finding the global optima. This is due to the fact that due to division in population and generations the exploitation and exploration are balanced and the switch does make

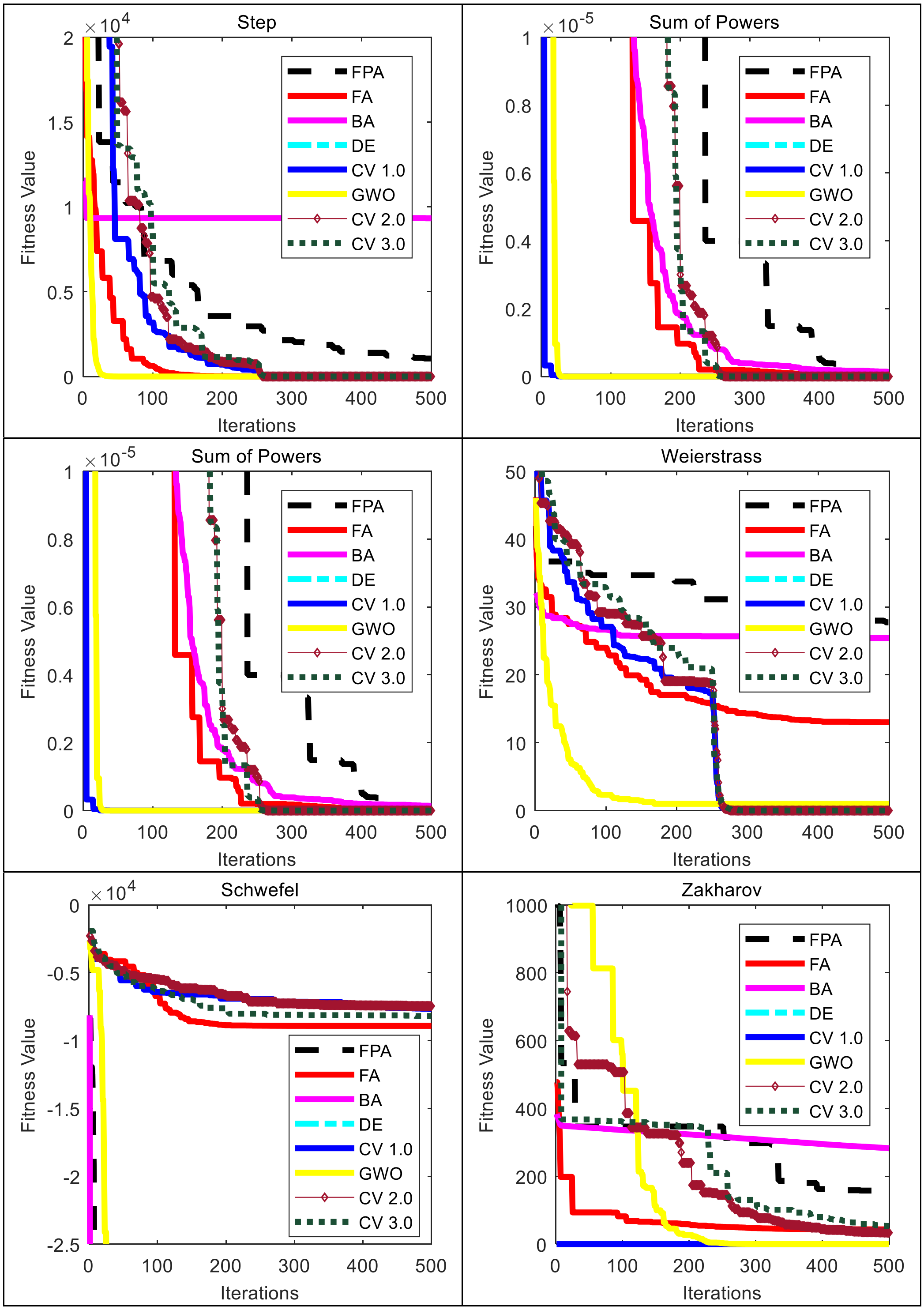

Fig. 2. Continued

any difference, hence the requirement of tuning switch for different problems is eliminated. Non-parametric statistical tests have been carried out to prove the superiority of the modified version and it is found that CV 1.0 algorithm outperforms state-of-the-art algorithms.

For future work, a balanced exploration and exploitation are one of the causes of poor performance of population based algorithms and this can be removed by employing the concept of division of population and division of generations to other algorithms. Also, instead of using the operators of GWO, more search operators available in literature can be analyzed. Also to improve the global search combined mutation operators can be applied. The proposed algorithms can be extended for solving multi-objective optimization problems and also can be applied for developing some clustering approaches for application in some real-life problems like satellite image segmentation, gene expression data clustering, cancer classification etc.

Table 20
p-test values of various algorithms for CEC 2015 test functions.

Function	DE	BA	GWO	FA	FPA	BFP	PSOGSA	CV 1.0
F_1	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA
F_2	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA
F_3	4.21E−13	2.56E−34	4.87E−33	2.56E−34	2.56E−34	2.57E−29	3.38E−09	NA
F_4	2.13E−30	4.66E−34	1.25E−01	3.06E−10	2.56E−34	2.56E−34	7.79E−33	NA
F_5	7.75E−34	1.53E−32	234E−01	NA	2.56E−34	2.56E−34	9.17E−28	1.81E−01
F_6	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA
F_7	7.52E−34	2.56E−34	2.25E−17	5.88E−29	2.56E−34	2.56E−34	2.00E−33	NA
F_8	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	NA
F_9	4.24E−32	2.56E−34	1.52E−18	NA	2.56e−34	2.56E−34	8.95E−02	7.51E−05
F_{10}	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	2.56E−34	8.95E−02	NA
F_{11}	1.01E−33	2.19E−31	4.19E−30	3.77E−12	2.56E−34	2.56E−34	1.01E−33	NA
F_{12}	7.52E−34	2.56E−34	2.02E−02	2.05E−20	2.56E−34	2.56E−34	3.99E−29	NA
F_{13}	2.56E−34	NA	2.56E−34	2.56E−34	1.43E−01	5.01E−33	2.56E−29	2.56E−34
F_{14}	1.46F−02	9.03E−33	2.61E−01	8.74E−02	5.64E−39	2.56E−34	NA	3.42E−17
F_{15}	2.55E−34	2.55E−34	2.55E−34	2.55E−34	2.56E−34	2.55E−34	2.55E−34	NA

Algorithm 1 Pseudo−code of Cuckoo Search (CS) algorithm.

Begin:
Initialize cuckoo population:*n*
Define *d*−dimensional objective function, *f(x)*
***do Until* iteration counter < maximum number of iterations**
global Search:
generate new nest x_i^{t+1}using Eq. (1)
evaluate fitness of x_i^{t+1}
choose a nest j randomly from n initial nests.
if the fitness of x_i^{t+1}better than that of x_i^t
replace j by x_i^{t+1}
end if
local search:
abandon some of the worst nests using probability switch.
create new nest using Eq. (2)
Evaluate and find the best.
end until
update final best
End

Algorithm 2 Pseudo−code of Cuckoo version 1.0 (CV 1.0).

Begin:
Initialize cuckoo population: *n*
Define *d*−dimensional objective function, *f(x)*
***do Until* iteration counter < maximum number of iterations** (t_{max})
global Search: generate new nest x_i^{t+1} using Eq. (10)
evaluate fitness of x_i^{t+1}
chose a nest j randomly from n initial nests.
if the fitness of x_i^{t+1}better than that of x_i^t
replace j by x_i^{t+1}
end if
local search: generate new solution by using Eqs. (2) **&** (12)**.**
evaluate and find the best.
end until
update final best
End

Algorithm 3 Pseudo−code of Cuckoo version 2.0 (CV 2.0).

Begin:
Initialize cuckoo population: *n*
Define *d*−dimensional objective function, *f(x)*
***do Until* iteration counter < maximum number of iterations**(t_{max})
global Search: population is updated using Eqs. (9) **&** (13)**.**
evaluate fitness of new solutionx_i^{t+1}
chose a nest j randomly from n initial nests.
if the fitness of x_i^{t+1}better than that of x_i^t
replace j by x_i^{t+1}
end if
local search: generate new solution by using Eqs. (2) **&** (12)**.**
evaluate and find the best.
end until
update final best
End

Algorithm 4 Pseudo−code of Cuckoo version 3.0 (CV 3.0).

Begin:
Initialize cuckoo population: *n*
Define *d*−dimensional objective function, *f(x)*
***do Until* iteration counter < maximum number of iterations**(t_{max})
global search: population is updated using Eqs. (9)**,** (12)**–**(14)**.**
evaluate fitness of new solution x_i^{t+1}
chose a nest j randomly from n initial nests.
if the fitness of x_i^{t+1}better than that of x_i^t
replace j by x_i^{t+1}
end if
local search: generate new solution by using Eq. (2)**.**
evaluate and find the best.
end until
update final best
End

Acknowledgements

This work has been supported under INSPIRE Fellowship (IF160215) by Directorate of Science and Technology, Government of India.

References

Bhandari, A. K., Singh, V. K., Kumar, A., & Singh, G. K. (2014). Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. *Expert Systems with Applications, 41*(7), 3538–3560.

Bindu, A. H., & Reddy, M. D. (2013). Economic load dispatch using cuckoo search algorithm. *International Journal of Engineering Research and Applications, 3*(4), 498–502.

Chaowanawatee, K., & Heednacram, A. (2012). Implementation of cuckoo search in RBF neural network for flood forecasting. In *Computational intelligence, communication systems and networks (CICSyN), 2012 fourth international conference on* (pp. 22–26). IEEE.

Cobos, C., Muñoz-Collazos, H., Urbano-Muñoz, R., Mendoza, M., León, E., & Herrera-Viedma, E. (2014). Clustering of web search results based on the cuckoo search algorithm and balanced Bayesian information criterion. *Information Sciences, 281*, 248–264.

Črepinšek, M., Liu, S. H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR), 45*(3), 35.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*(1), 3–18.

Dey, N., Samanta, S., Yang, X. S., Das, A., & Chaudhuri, S. S. (2013). Optimisation of scaling factors in electrocardiogram signal watermarking using cuckoo search. *International Journal of Bio-Inspired Computation, 5*(5), 315–326.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Artificial ants as a computational intelligence technique. *IEEE Computational Intelligence Magazine, 1*, 28–39.

Draa, A., Bouzoubia, S., & Boukhalfa, I. (2015). A sinusoidal differential evolution algorithm for numerical optimisation. *Applied Soft Computing, 27*, 99–126.

Eberhart, R. C., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science: 1* (pp. 39–43).

Feng, Y., Wang, G. G., Feng, Q., & Zhao, X. J. (2014). An effective hybrid cuckoo search algorithm with improved shuffled frog leaping algorithm for 0-1 knapsack problems. *Computational Intelligence and Neuroscience, 2014*, 36.

Fister, I., Fister Jr, I., Yang, X. S., & Brest, J. (2013). A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation, 13*, 34–46.

Fister Jr, I., Fister, D., & Yang, X. S. (2013). A hybrid bat algorithm. arXiv preprint arXiv: 1303.6310.

Fister, I., Jr, Yang, X. S., Fister, D., & Fister, I. (2014). Cuckoo search: A brief literature review. In *Cuckoo search and firefly algorithm* (pp. 49–62). Springer International Publishing.

Gutjahr, W. J. (2009). Convergence analysis of metaheuristics. In *Matheuristics* (pp. 159–187). Springer US.

Holland, J. H. (1992). Genetic algorithms. *Scientific American, 267*(1), 66–72.

Huang, L., Ding, S., Yu, S., Wang, J., & Lu, K. (2016). Chaos-enhanced cuckoo search optimization algorithms for global optimization. *Applied Mathematical Modelling, 40*(5), 3860–3875.

Ilunga-Mbuyamba, E., Cruz-Duarte, J. M., Avina-Cervantes, J. G., Correa-Cely, C. R., Lindner, D., & Chalopin, C. (2016). Active contours driven by Cuckoo search strategy for brain tumor images segmentation. *Expert Systems with Applications, 56*, 59–68.

Kanagaraj, G., Ponnambalam, S. G., &, & Jawahar, N. (2013). A hybrid cuckoo search and genetic algorithm for reliability–redundancy allocation problems. *Computers & Industrial Engineering, 66*(4), 1115–1124.

Karaboga, D., & Basturk, B. (2007). Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In *International fuzzy systems association world congress* (pp. 789–798). Berlin Heidelberg: Springer.

Kumar, A., & Chakarverty, S. (2011). Design optimization for reliable embedded system using cuckoo search. In *Electronics computer technology (ICECT), 2011 3rd international conference on: 1* (pp. 264–268). IEEE.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*: 1. MIT press.

Kumar, M., & Rawat, T. K. (2015). Optimal design of FIR fractional order differentiator using cuckoo search algorithm. *Expert Systems with Applications, 42*(7), 3433–3449.

Li, X., Wang, J., & Yin, M. (2014). Enhancing the performance of cuckoo search algorithm using orthogonal learning method. *Neural Computing and Applications, 24*(6), 1233–1247.

Liang, J. J., Qu, B. Y., Suganthan, P. N., & Chen, Q. (2014). Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization. *Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report*. Nanyang Technological University Nov.

Lim, W. C. E., Kanagaraj, G., & Ponnambalam, S. G. (2014). PCB drill path optimization by combinatorial cuckoo search algorithm. *The Scientific World Journal, 2014*, Article ID 264518, 11 pages.

Long, W., Liang, X., Huang, Y., & Chen, Y. (2014). An effective hybrid cuckoo search algorithm for constrained global optimization. *Neural Computing and Applications, 25*(3-4), 911–926.

Marichelvam, M. K., Prabaharan, T., & Yang, X. S. (2014). Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan. *Applied Soft Computing, 19*, 93–101.

Ming, B., Chang, J. X., Huang, Q., Wang, Y. M., & Huang, S. Z. (2015). Optimal operation of multi-reservoir system based-on cuckoo search algorithm. *Water Resources Management, 29*(15), 5671–5687.

Mirjalili, S., & Hashim, S. Z. M. (2010). A new hybrid PSOGSA algorithm for function optimization. In *Computer and information application (ICCIA), 2010 international conference on* (pp. 374–377). IEEE.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software, 83*, 80–98.

Nawi, N. M., Khan, A., & Rehman, M. Z. (2013). A new Levenberg Marquardt based back propagation algorithm trained with cuckoo search. *Procedia Technology, 11*, 18–23.

Nguyen, T. T., Truong, A. V., & Phung, T. A. (2016). A novel method based on adaptive cuckoo search for optimal network reconfiguration and distributed generation allocation in distribution network. *International Journal of Electrical Power & Energy Systems, 78*, 801–815.

Ouaarab, A., Ahiod, B., & Yang, X. S. (2014). Discrete cuckoo search algorithm for the traveling salesman problem. *Neural Computing and Applications, 24*(7-8), 1659–1669.

Pani, P. R., Nagpal, R. K., Malik, R., & Gupta, N. (2013). Design of planar ebg structures using cuckoo search algorithm for power/ground noise suppression. *Progress In Electromagnetics Research M, 28*, 145–155.

Payne, R. B. (2005). *The cuckoos*. Oxford.

Pavlyukevich, I. (2007). Lévy flights, non-local search, and simulated annealing. *Journal of Computational Physics, 226*(2), 1830–1844.

Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y.-P., Auger, A., et al. (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *Technical Report*. Nanyang Technological University May AND KanGAL Report #2005005, IIT Kanpur, India.

Rechenberg, I. (1978). Evolutionsstrategien. In *Simulationsmethoden in der Medizin und Biologie* (pp. 83–114). Berlin Heidelberg: Springer.

Salgotra, R., & Singh, U. (2016). A novel bat flower pollination algorithm for synthesis of linear antenna arrays. *Neural Computing and Applications*, 1–14. doi:10.1007/s00521-016-2833-3.

Salgotra, R., & Singh, U. (2017). Application of mutation operators to flower pollination algorithm. *Expert Systems with Applications, 79*, 112–129.

Sarangi, S. K., Panda, R., & Dash, M. (2014). Design of 1-D and 2-D recursive filters using crossover bacterial foraging and cuckoo search techniques. *Engineering Applications of Artificial Intelligence, 34*, 109–121.

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12*(6), 702–713.

Soto, R., Crawford, B., Galleguillos, C., Monfroy, E., & Paredes, F. (2014). A prefiltered cuckoo search algorithm with geometric operators for solving Sudoku problems. *The Scientific World Journal, 2014*, Article ID 465359, 12 pages.

Speed, E. R. (2010). Evolving a Mario agent using cuckoo search and softmax heuristics. In *2010 2nd international IEEE consumer electronics society's games innovations conference* (pp. 1–7). IEEE.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*(4), 341–359.

Walia, G. S., & Kapoor, R. (2014). Intelligent video target tracking using an evolutionary particle filter based upon improved cuckoo search. *Expert Systems with Applications, 41*(14), 6315–6326.

Wang, Y., Cai, Z., & Zhang, Q. (2011). Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Transactions on Evolutionary Computation, 15*(1), 55–66.

Wang, J.-S., Han, S., Shen, N.-N., & Li, S.-X. (2014). Features extraction of flotation froth images and BP neural network soft-sensor model of concentrate grade optimized by shuffled cuckoo searching algorithm. *The Scientific World Journal, 2014*, Article ID 208094, 17 pages.

Wang, L., & Zhong, Y. (2015). Cuckoo search algorithm with chaotic maps. *Mathematical Problems in Engineering, 2015*, Article ID 715635, 14 pages.

Wang, G. G., Gandomi, A. H., Zhao, X., & Chu, H. C. E. (2016). Hybridizing harmony search algorithm with cuckoo search for global numerical optimization. *Soft Computing, 20*(1), 273–285.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. In *Nature & biologically inspired computing, 2009. NaBIC 2009. World congress on* (pp. 210–214). IEEE.

Yang, X. S. (2010a). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)* (pp. 65–74). Springer.

Yang, X. S. (2010b). Firefly algorithm, stochastic test functions, and design optimisation. *International Journal of Bio-Inspired Computation, 2*(2), 78–84.

Yang, X. S. (2010c). *Nature-inspired metaheuristic algorithms*. Luniver press.

Yang, X. S. (2012). Flower pollination algorithm for global optimization. In *International conference on unconventional computing and natural computation* (pp. 240–249). Springer.

Yang, X. S., & Deb, S. (2013). Multiobjective cuckoo search for design optimization. *Computers & Operations Research, 40*(6), 1616–1624.

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3*(2), 82–102.

Zhang, Y., Wang, L., & Wu, Q. (2012). Modified Adaptive Cuckoo Search (MACS) algorithm and formal description for global optimisation. *International Journal of Computer Applications in Technology, 44*(2), 73–79.