Arab J Sci Eng
DOI 10.1007/s13369-016-2270-8

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Intelligent Multiple Search Strategy Cuckoo Algorithm for Numerical and Engineering Optimization Problems

Hojjat Rakhshani[1] · **Amin Rahati**[1]

Received: 17 March 2016 / Accepted: 27 June 2016
© King Fahd University of Petroleum & Minerals 2016

Abstract This paper presents intelligent multiple search strategy algorithm (IMSS) as a new modification of cuckoo search (CS) to improve performance of the conventional algorithm. To do so, the proposed IMSS algorithm adopts a multiple search strategy and Q-learning technique. The introduced multiple search strategy couples CS and covariance matrix adaptation evolution strategy (CMAES) to explore search space more efficiently and also to reduce computational time of finding the optimal solution. More precisely, CS enables the IMSS to achieve better accuracy of final solutions through Lévy flights, and CMAES enhances its convergence rate via a concept known as evolution path. To provide an intelligent balance between the exploration and exploitation behaviors, the IMSS employs Q-learning method and thereby acquires information about the performance of each search strategy. Then, it uses this information to dynamically select the best strategy for evolving candidate solutions as optimization process progress. In other words, the IMSS algorithm transforms the task of learning the optimal policy in Q-learning into the search for an efficient and adaptive optimization behavior. The IMSS is evaluated on CEC 2005 and CEC 2013 test suites, and its results are compared with results produced by several state-of-the-art algorithms. For further validation, the presented approach is also applied on two well-studied engineering design problems. The obtained results indicate that the IMSS provides very competitive results compared to other algorithms on the aforementioned optimization problems.

Keywords Cuckoo search · Covariance matrix adaptation evolution strategy · Reinforcement learning · Engineering design problems

1 Introduction

Optimization algorithms are tools that can be used to find optimal solutions in a more rapid and accurate way. Over the past decades, many encouraging advances have been made in the optimization algorithm domain due to its major influence on the cost and time efficiency. The application of optimization algorithms covers a wide area of real-world problems that varies from engineering [1] to biomedical studies [2]. The early works on this subject were based on mathematical programming methods. However, these approaches may provide suboptimal results and are not effective in solving multimodal and high-dimensional problems. Alternatively, metaheuristic algorithms such as genetic algorithm (GA) [3], harmony search (HS) [4], particle swarm optimization (PSO) [5], differential evolution (DE) [6], artificial bee colony (ABC) [7], and CS [8] have been emerged. CS is one of the successful metaheuristic algorithms which has been recently introduced by Yang and Deb. It is inspired by obligate brood parasitism of some cuckoo species and foraging patterns of animals and insects (Lévy flights). The long jump length distribution provided by Lévy flights is effective for highly complex and nonlinear optimization problems [8]. CS is also characterized by the ease of implementation and having few parameters. Altogether, it has been shown

Electronic supplementary material The online version of this article (doi:10.1007/s13369-016-2270-8) contains supplementary material, which is available to authorized users.

✉ Amin Rahati
a.rahati@cs.usb.ac.ir

1 Department of Computer Science, Faculty of Mathematics, University of Sistan and Baluchestan, Zahedan 98135-674, Iran

to be a potential tool for solving optimization problems [9,10].

Real-world optimization problems are usually highly nonlinear and dynamic (their environment changes along with time). Typical examples are nonlinear process control [11], structural design [12], grouping problems [13], and text mining [14]. In such condition, an optimization algorithm should detect the environment changes and then adapts itself; otherwise, it encounters two main problems of invalid memory and loss of diversity [15]. Since the introduction of CS, several variants have been proposed to address aforementioned problems. In particular, researchers adopted different approaches to make a better balance between the exploration and exploitation by using information exchange mechanism [16], quantum computing principles [17], fuzzy system [18], orthogonal learning [19], adaptive strategy [20,21], and hybrid approaches [22]. Although these extensions have been proved to be successful, there is no guarantee that they will provide the best performance for all optimization problems. This behavior is consistent with "no free lunch" theorem [23]. For this reason, new approaches with different characteristics are continuously developing in the literature.

In this study, we examine the potential of a multiple search strategy which combines the CS and CMAES [24] algorithms to address the above-mentioned problem. The CS approach prevents the IMSS of being trapped into the local optimum, and the CMAES accelerates its convergence rate. The IMSS is also assisted by a machine learning method which enables the algorithm to dynamically make a balance between exploration and exploitation. Indeed, Q-learning technique [25] helps the IMSS to learn the utility of employing the best search strategy. Toward this goal, it assigns a reward or penalty to each search algorithm depending on whether its achieved results has been improved. Thereafter, these reward/penalty values are used in a selection mechanism to adaptively select the best search algorithm for producing the next generation of candidate solutions. These considerations allow the IMSS to use necessary knowledge about the optimization problem and to optimize its behavior.

The rest of this paper is organized as follows. Section 2 briefly reviews related works in the literature. Section 3 presents an overview of the standard CS algorithm. Section 4 introduces the standard CMAES algorithm. In Sect. 5, the proposed IMSS algorithm and its implementation details are elaborated. In Sect. 6, we compare performance of the IMSS with several state-of-the-art algorithms on CEC 2005 and CEC2013 test suites. The performance of IMSS to find optimal parameters of real-world engineering problems and obtained results are discussed in the same section. Finally, Sect. 7 summarizes the paper and draws conclusions.

2 Literature Review

Up to now, different variants of the standard CS have been developed in the literature. The early work was motivated by the notion of an information sharing mechanism and a self-adaptive step size parameter [16]. In another study, Yang and Deb [26] formulated a multi-objective CS to solve structural design problems. In [17], a modified CS based on quantum computing principles has been proposed to solve knapsack problems. Wang et al. [27] introduced a hybrid variant of CS using the krill herd algorithm for numerical optimization tasks. In [28], CS algorithm adopted for secondary protein structure prediction. Wang et al. [29] put forwarded an improved CS with varied scaling factor to solve numerical optimization problems. In [19], an orthogonal-based learning strategy has been incorporated into the CS with the aim of improving exploitation capability of the standard algorithm. Interestingly, Ilunga-Mbuyamba et al. proposed a multi-population CS for brain tumor images segmentation [30]. In [31], a new CS extensions for optimally tuning parameters of traffic signal controllers have been developed. Naumann et al. adopted a new computational aerodynamic shape optimization algorithm based on modified CS [32]. Suresh and Lal proposed a new extension of CS for segmenting satellite images [33]. Similarly, Kanagaraj et al. presented a hybrid metaheuristic based on the CS and GA algorithms [22]. In [20], Zhang et al. proposed a self-adaptive CS in order to make a good exploration-exploitation balance. Wang et al. introduced a solution based on the CS algorithm for solar radiation prediction [34]. In [35], Huang et al. utilized chaotic sequences and adaptive step size parameter to enhance diversity of the solutions. Moreover, Liu and Fu proposed an improved CS algorithm based on the frog leaping local search and chaos theory [36]. In [37], a novel one-rank CS proposed for solving economic load dispatch (ELD) problems. Huang et al. developed a hybrid algorithm based on CS and teaching-learning-based algorithms to address optimization problems in structure designing and machining processes [38]. Chakraverty et al. presented a fuzzy-based CS for design space exploration of distributed multiprocessor embedded systems [18]. In [39], authors developed a method based on the CS and balanced Bayesian information criterion for clustering Web search results. In [40], authors introduced a modified CS based on DE search operators and self-adaptive parameters. Akkoyunlu1 et al. used a CS-based algorithm for accurate estimation of thermal power plant heat curve parameters [41].

Different from the aforementioned researches, this study formulates an intelligent hybrid algorithm based on CS and CMAES algorithms to enhance the performance of CS algorithm. In line with this improvement, we also employ the Q-learning technique in order to provide a good balance between the exploration and exploitation abilities of the proposed algorithm

3 Cuckoo Search

CS is a nature-inspired optimization algorithm that simulates aggressive reproduction behavior of some cuckoo species such as *ani* and *guira* [8]. This behavior arises from the brood parasitism according to which they lay their eggs in nests of other host birds. This reproduction strategy increases survival probability of their eggs. However, in some cases the host bird may finds such alien eggs and either remove them or abandon its nest and build a new one [8]. The aforementioned considerations are incorporated in the basic version of CS. Yang and Deb use the following rules to describe the main algorithm steps [8]:

- Each cuckoo deposits one egg at a time and places it in a randomly picked nest (crossover operator),
- The best nests that include eggs (solutions) with high quality will be transferred to the next generation (elitism),
- The number of available host nests is fixed, and a host bird can find an alien egg with a probability equal to $p_a \in [0, 1]$ (mutation operator).

From the implementation point of view, each egg in a nest represents a solution, each cuckoo egg shows a new solution, and the goal is to replace not so-good eggs in the host nests with potentially better eggs. In this schema, there are also two important components for the purpose of intensification and diversification, respectively. The first one is the crossover search operator which uses Lévy flights to increase the exploration capability of CS. As Lévy flights distribution has infinite mean and variance, step lengths that are provided by this approach are more efficient than regular random walk or Brownian motions [26]. CS algorithm uses Lévy flights in order to generate new solution x^{t+1} for cuckoo i, as below [8]:

$$x_i^{t+1} = x_i^t + a \otimes \text{Lévy}(\beta) \tag{1}$$

and

$$a = a_0 \otimes \left(x_j^t - x_i^t\right), \tag{2}$$

$$\text{Lévy}(\beta) = \frac{u}{|v|^{1/\beta}} \tag{3}$$

where $\otimes$ represents entry-wise multiplications, β is Lévy flights exponent, a_0 is the step size scaling factor, x_j^t shows a randomly selected solution, and $a > 0$ is the step size parameter. For each optimization problem, the step size should be proportionate to the scales of that problem. Finally, u and v are two numbers with zero means and associated variance, as presented in Eq. (4).

```
CS Algorithm
procedure CS
  initialize a population of c host nests each with N solutions x_i, i = 1 ... N
  f  ←  compute fitness value for all the solutions
  repeat
      stochastically choose two nests (say x and y)
      for i = 1: N  do
          x_i^{t+1} ← generate ith solution acoording to Lévy flights by Equation (1)
          if solution x_i^{t+1} is better than y_i
                 replace y_i by new solution  x_i^{t+1}
          end if
      end for
      throw out a fraction (p_a) of worst nests
      for each abandoned nest K do
          for i = 1: N  do
              k_i^{t+1} ← generate ith solution by Equation (5)
              if solution k_i^{t+1} is better than k_i^t
                     replace k_i^t by new solution  k_i^{t+1}
              end if
          end for
      end for
      rank the solutions and find the current best
  until (stop condition = false)
post process results and visualization
end procedure
```

Fig. 1 Pseudocode of CS algorithm

$$\sigma_u = \left\{ \frac{\text{Gamma}\,(1+\beta)\sin(\pi\beta/2)}{\text{Gamma}\,[1+\beta/2]\,\beta 2^{\beta-1/2}} \right\}^{1/\beta}, \quad \sigma_v = 1 \tag{4}$$

The second component in this schema is the mutation search operator and performs global search. Solutions that are generated in this phase help CS to escape from local optima. CS employs Eq. (5) to generate new solution x_i^{t+1} as follows [9]:

$$x_i^{t+1} = x_i^t + r \otimes H\,(p_a - \epsilon) \otimes \left(x_j^t - x_k^t\right) \tag{5}$$

Here, x_j^t and x_k^t are different solutions selected randomly, r and ϵ are some random numbers with uniform distribution, and $H\,(u)$ is the Heaviside function. Also, the switching probability p_a is responsible for making a balance between the local and global search characteristics. When p_a increases, the probability for global optimization is reduced and vice versa. According to aforementioned steps, the pseudocode of CS is illustrated in Fig. 1.

4 Covariance Matrix Adaptation Evolution Strategy

The CMAES is a state-of-the-art evolutionary algorithm for numerical optimization of non-convex and nonlinear problems. This algorithm is able to learn second-order model of underlying fitness function by means of the covariance matrix adaption within an iterative procedure [24]. In contrast to classical methods such as Quasi-Newton, deriva-

```
CMAES Algorithm
procedure CMAES
  λ ← number of samples per iteration
  μ ← number of recombinition points
  initialize state variables m, σ, C = I, p_σ = 0, p_c = 0
  repeat
    for i = 1: λ do
      x_i^{t+1} ← sample ith solution from a multivariate normal distribution by Equation (6)
      f_i ← compute fitness value for the ith solution
    end for
    rank the new solutions and find the first μ solutions
    m^{t+1} ← update the mean value by Equation (7)
    p_c^{t+1} ← update anisotropic evolution path by Equation (9)
    C^{t+1} ← update the covariance matrix by Equation (10)
    p_σ^{t+1} ← update isotropic evolution path by Equation (11)
    σ^{t+1} ← update the step − size using isotropic path length by Equation (12)
  until (stop condition = false)
post process results and visualization
end procedure
```

Fig. 2 Pseudocode of CMAES algorithm

tives or gradients of the fitness function are not required by this method. Such characteristic makes the CMAES feasible on non-smooth and non-continuous optimization problems. It turns out to be a particularly efficient and has been used in many computer science researches and real-world applications [42,43].

The present study considers the CMAES algorithm with weighted intermediate recombination and step size adaptation. In this approach, solutions are generated from a multivariate normal distribution N with mean m and covariance C. Following Hansen and Ostermeier [24], we produce new solution x^{t+1} as follows:

$$x^{t+1} = m^t + \sigma^t N\left(0, C^t\right) \tag{6}$$

$$m^t = \sum_{i=1}^{\mu} w_i x^t_{i:\lambda} \tag{7}$$

$$w_i = \log\left(\mu + \frac{1}{2}\right) - \log(i), \sum\nolimits_{i=1}^{\mu} w_i = 1 \tag{8}$$

where m^t is the weighted mean of the μ best solutions and $x^t_{i:\lambda}$ determines the ith ranked individual. Also, overall standard deviation σ^t is the step size parameter. At each iteration, algorithm adopts a covariance matrix C^t by an evolution path p_c^{t+1}, as given in Eqs. (9) and (10).

$$p_c^{t+1} = (1 - c_c)\, p_c^t + \sqrt{c_c(2 - c_c)} \frac{\sqrt{\mu}}{\sigma^t}\left(m^{t+1} - m^t\right) \tag{9}$$

$$C^{t+1} = (1 - c_{\text{cov}})\, C^t + c_{\text{cov}}\, p_c^{t+1} \left(p_c^{t+1}\right)^{\mathrm{T}} \tag{10}$$

Here, c_c and $c_{\text{cov}} \in [0, 1]$ are learning rates for the given evolution path. and the covariance matrix C^t, respectively.

Also, it adapts the step size parameter through the evolution path p_σ^{t+1} as below:

$$p_\sigma^{t+1} = (1 - c_\sigma)\, p_\sigma^t + \sqrt{c_\sigma(2 - c_\sigma)}\sqrt{\mu} B^t m^{t+1} \tag{11}$$

$$\sigma^{t+1} = \sigma^{t+1} \exp\left(\frac{\| p_\sigma^{t+1} \| - \hat{\chi}_n}{d_\sigma \hat{\chi}_n}\right) \tag{12}$$

$$\hat{\chi}_n \approx \sqrt{n}\left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right) \tag{13}$$

In these equations, n represents the problem dimension, B^t is the normalized eigenvectors of C^t, c_σ controls the learning rate, and $d_\sigma > 1$ is damping parameter. The pseudocode of the CMAES algorithm is outlined in Fig. 2. In this figure, the main steps of the algorithm are as follows: (1) sampling the new solutions, (2) ranking the sampled solutions based on their fitness value, and (3) updating the internal state variables. More details about the algorithm and its parameter settings are discussed in [24].

5 The Proposed Algorithm

The proposed IMSS algorithm adopts two main components. The first one is a multiple search strategy which explores the fitness landscape through information exchange between CS and CMAES. The main goal of this component is to avoid premature convergence and to decrease the computation time efforts. The second component is a reinforcement learning technique which guides the search process by making a proper balance between the CS and CMAES algorithms. This technique allows the IMSS to take into account the underlying mathematical properties and domain-specific search knowledge of the problem. The following subsections first describe two components that are adopted in the IMSS and then present the overall framework.

5.1 Multiple Search Strategy

The incorporated multiple search strategy divides the execution process of the IMSS into four steps. The main focus of the first step is on the intensification strategy and searching around the promising areas found during the optimization process. This exploitation behavior can enhance solutions quality and convergence speed of the conventional algorithm. For this purpose, we employ CMAES algorithm in order to increase the probability of raising successful solutions. The CMAES algorithm is considered as one of the best algorithms for solving optimization problems and thus can be a good candidate [24,44–46]. Thereafter, second step ranks population in descending order based on their fitness and finds the μ best candidate solutions. This consideration is known as migration strategy. The third step uses the acquired knowledge of the search process and enhances the exploration capability of the introduced algorithm. Toward this goal, the selected μ superior solutions in previous migration step are evolved by the search operators of CS (i.e., crossover and mutation). This elitism perturbation approach can improve solutions' diversity and prevents the algorithm of being trapped into a local optima. Finally, the fourth step introduces an information sharing mechanism for transferring information from CS into the CMAES. In this step, the new search direction of the CMAES is determined using the standard covariance matrix from Eq. (10) and CS covariance matrix. Inspired by [47], we provide a new way for both the algorithms to cooperate with each other and to take advantages of the shared information. This method places equal emphasis on the obtained information throughout the entire evolutionary process, as shown in Eq. (14).

$$C^{t+1} = \frac{1}{2} C^{t+1}_{\mathrm{CMAES}} + \frac{1}{2} C^{t+1}_{\mathrm{CS}} \tag{14}$$

$$C^{t+1}_{\mathrm{CS}} = B^{t}_{\mathrm{rot}} \left(D^{t}\right)^{2} \left(B^{t}_{\mathrm{rot}}\right)^{\mathrm{T}} \tag{15}$$

$$B^{t}_{\mathrm{rot}} = \mathrm{RB} \tag{16}$$

Here, D^t is a diagonal matrix and its elements are the square roots of the eigenvalues of C^t. Also, R is a rotation matrix. In this schema, the incorporated matrix R rotates C^{t}_{CMAES}, such that the columns of B are aligned with the vector $p_{\mathrm{g}} = \mathrm{CS}_{\mathrm{best}} - \mathrm{m}^t$. More generally, it rotates the covariance matrix C^{t}_{CMAES} toward the best solution that is obtained by CS algorithm (i.e., $\mathrm{CS}_{\mathrm{best}}$). This yields a new covariance matrix C^{t+1}_{CS} that is used in Eq. (14). The complete algorithm for computing the rotation matrix R can be found in [47].

5.2 Reinforcement Learning Technique

As pointed in optimization studies, there is a strong need to maintain a good balance between the exploration and exploitation capabilities. Indeed, main differences between the existing optimization algorithms lie in the particular way of achieving this balance [48]. The present study adopts Q-learning method in order to address this main task. Basically, Q-learning is a reinforcement method that can learn the utility of performing actions based on the acquired knowledge of the environment. This method is model-free and can be used to handle different problems with stochastic transitions and rewards states. This property of Q-learning makes it suitable for identifying the evolutionary states of the IMSS. In particular, our approach attempts to learn the utility of employing the best search algorithm using this value-iteration method as below [25]:

$$Q^{t+1}(s,a) = Q^{t}(s,a) + \rho.\left(R^{t+1} + \gamma \cdot F - Q^{t}(s,a)\right) \tag{17}$$

Here, R^{t+1} is the received reward/penalty after taking action a in state s at iteration t, F shows estimate of feature reward/penalty value, γ is the discount factor which determines the importance of future rewards/penalty, and finally, ρ is the learning rate parameter. In this equation, we also consider the following two assumptions:

- $\mathrm{S} = \{s_1, s_2, \ldots, s_\lambda\}$ is defined to be the set of individual states,
- $\mathrm{A} = \{\mathrm{CMAES}, \mathrm{CS}\}$ is a set of actions that the proposed IMSS can select.

For each state s, $R^{t+1} = \left|\text{fitness}\left(x^{t+1}_{s}\right) - \text{fitness}\left(x^{t}_{s}\right)\right|$ is determined to be the immediate reward/penalty after performing an action a in iteration $t+1$. In the case of increasing solution quality, $Q^{t+1}(s,a)$ will be updated with a positive reward $(+R^{t+1})$; otherwise, it will be computed with a negative penalty $(-R^{t+1})$. The F parameter is computed in the same way. In this approach, a selected action $a \in A$ is first executed. Then, candidate solutions are evolved and Q-learning assigns a reward or penalty to each solution depending on whether the obtained result has improved. The obtained rewards/penalties are stored in a lookup table, the so-called Q-table which will be used to measure the overall search efficiency, as given in Eq. (18).

$$\mathrm{SE_a} = \sum_{i=1}^{\lambda} Q^{t+1}(s_i, a) \tag{18}$$

At each iteration, the proposed $\mathrm{SE_a}$ value is computed and will be used to find the best search algorithm. More precisely, the IMSS continues searching with CS algorithm if $\mathrm{SE_{CS}} > \mathrm{SE_{CMAES}}$, or toggles to the CMAES search algorithm in the case of performance deterioration. This procedure allows the IMSS to optimize its exploration and exploitation behaviors more efficiently.

```
IMSS Algorithm
1: procedure IMSS
2:   x ← initialize a population with λ solutions, i = 1 … λ
3:   f ← compute fitness value for all the solutions
4:   Q ← initialize the Q-Table enteries with zero
5:   initialize state variables m, σ, C ← I, p_σ ← 0, p_c ← 0, SE_CS ← 0, SE_CMAES ← 0
6:   repeat
7:     if SE_CMAES > SE_CS do
8:       SE_CMAES ← 0
9:       for i = 1: λ do
10:        x_i^{t+1} ← sample ith solution from a multivariate normal distribution by Equation (6)
11:        f_i ← compute fitness value for the ith solution
12:        Q(i, CMAES) ← Update-Q-Table (CMAES, x, f, Q, i)
13:        SE_CMAES ← SE_CMAES + Q(i, CMAES)
14:      end for
15:      rank the new solutions and select the first μ solutions
16:      update the mean value, covariance matrix and step-size parameter by Equations (7), (9) − (12)
17:    else
18:      SE_CS ← 0
19:      k ← rank the solutions and find index of the first μ solutions
20:      for each i ∈ k do
21:        x_i^{t+1} ← generate ith solution acoording to Lévy flights by Equation (1)
22:        f_i ← compute fitness value for the ith solution
23:        if solution x_i^{t+1} is better than x_i
24:          replace x_i by new solution x_i^{t+1}
25:        end if
26:        Q(i, CS) ← Update-Q-Table (CS, x, f, Q, i)
27:        if rand(0,1) > p_a
28:          x_i^{t+1} ← generate ith solution by Equation (5)
29:          f_i ← compute fitness value for the ith solution
30:          if solution x_i^{t+1} is better than x_i
31:            replace x_i by new solution x_i^{t+1}
32:          end if
33:          Q(i, CS) ← Update-Q-Table (CS, x, f, Q, i)
34:        end if
35:        SE_CS ← SE_CS + Q(i, CS)
36:      end for
37:      update covariance matrix by Equation (14)
38:    end if
39:  until (stop condition = false)
40: post process results and visualization
41: end procedure
```

Fig. 3 Pseudocode of the proposed IMSS algorithm

5.3 IMSS Algorithm

In this section, a detailed pseudocode of the IMSS approach is presented. Figure 3 shows the overall procedure, and Fig. 4 demonstrates the employed Q-learning algorithm. The algorithm is started by initializing a population of candidate solutions using a random method. Next, fitness of the generated candidate solutions is evaluated. Then, the Q-table entries are initialized with zero values. In this way, both the IMSS and CMAES algorithms have equal selection chance during the optimization process. Before proceeding to the next step, algorithm checks whether the stopping criteria are satisfied. Then, in the main loop, the CMAES (lines 8–16) and CS (lines 21–35) algorithms are executed asynchronously. In this loop, the multiple search component allows the algorithms to share their best search information (line 37). Also, the incorporated reinforcement learning component updates the Q-table information (lines 12, 26 and 33) and decides whether its behavior needs to be changed (line

```
Update-Q-Table
procedure Update-Q-Table
  input parameters: solver, x, f, Q, i
  R^{t+1} ← |f_i^{t+1} − f_i^t|
  F^{t+1} ← R^{t+1}
  if solution x_i^{t+1} is better than x_i
    Q(i, solver) ← Q(i, solver) + ρ.( R^{t+1} + γ.F^{t+1} − Q(i, solver))
  else
    Q(i, solver) ← Q(i, solver) + ρ.( −R^{t+1} − γ.F^{t+1} − Q(i, solver))
  end if
end procedure
```

Fig. 4 Pseudocode of the adopted Q-learning

Table 1 Benchmark functions

No.	Name	Properties	Search range	Optimum solution
F1	Shifted Sphere Function	U, SH, SE, SC	[−100,100]	−450
F2	Shifted Schwefel's Problem 1.2	U, SH, NS, SC	[−100,100]	−450
F3	Shifted Rotated High Conditioned Elliptic Function	U, SH, R, NS, SC	[−100,100]	−450
F4	Shifted Schwefel's Problem 1.2 with Noise in Fitness	U, SH, NS, SC, N	[−100,100]	−450
F5	Schwefel's Problem 2.6 with Global Optimum on Bounds	U, NS, SC	[−100,100]	−310
F6	Shifted Rosenbrock's Function	M, SH, NS, SC	[−100,100]	390
F7	Shifted Rotated Griewank's Function without Bounds	M, R, SH, NS, SC	[0, 600]	−180
F8	Shifted Rotated Ackley's Function with Global Optimum on Bounds	M, R, SH, NS, SC	[−32,32]	−140
F9	Shifted Rastrigin's Function	M, SH, SE, SC	[−5, 5]	−330
F10	Shifted Rotated Rastrigin's Function	M, R, SH, NS, SC	[−5, 5]	−330
F11	Shifted Rotated Weierstrass Function	M, R, SH, NS, SC	[−0.5,0.5]	90
F12	Schwefel's Problem 2.13	M, SH, NS, SC	$[-\pi, \pi]$	−460
F13	Expanded Extended Griewank's plus Rosenbrock's Function	M, SH, NS, SC	[−3,1]	−130
F14	Shifted Rotated Expanded Scaffer's	M, SH, NS, SC	[−100,100]	−300

U unimodal, *M* multimodal, *SH* shifted, *R* rotated, *SE* separable, *NS* non-separable, *SC* scalable, *N* noisy

7). If so, it will adapt a new search behavior in order to make a balance between the search capabilities. Execution of the main loop continues until a stopping criterion is met.

6 Experimental Studies

In this section, a set of experiments are conducted on numerical and real-world optimization problems to validate convergence and robustness behaviors of the proposed algorithm. In Sects. 6.1 and 6.2, a detailed description of the numerical problems used in this study, experimental setup, performance evaluation of the IMSS, and statistical analysis are presented. Thereafter, in Sect. 6.3 application and efficiency of the IMSS on a set of real-world engineering problems are investigated. The IMSS was implemented in MATLAB 7.8 environment under Windows 7 operating system. All simulations were conducted on an Intel i5, 2.5 GHz CPU and 6GB of RAM.

6.1 Comparison Between IMSS and State-of-the-Art Algorithms

We use a set of 14 benchmark functions given in CEC 2005 [49] to evaluate performance of the IMSS. This set contains unimodal (F1–F5), multimodal (F6–F12), and expanded multimodal (F13–F14) minimization functions. They have different characteristics including separable, non-separable, shifted, rotated, scalable, ill-condition, and noisy. The considered problems were widely used in previous studies and are particularly challenging for any metaheuristic algorithm. A brief summary of the benchmark functions is provided in Table 1. A complete definition of the benchmark functions can be found in [49].

The benchmark functions used for comparison are usually multimodal and their complexity enhances as problem dimension increases. So, we examine performance of the IMSS for both 10- and 30-dimensional cases. For the purpose of comparison, we used GA, DE, CMAES, ABC, PSO, and CS optimization algorithms. For each algorithm, initial population was sampled uniformly at random within the search bounds except for F7 and F25. Initial bounds for these problems are reported in CEC 2005 special session [49]. The maximum number of function evaluations is set to 1e+05 for 10-dimensional and 3e+05 for 30-dimensional cases [49]. The stopping criterion reaches either before exceeding the maximum number of function evaluations, or if the error value (difference between the global optimum and the obtained solution) becomes less than or equal to 10-8 [49]. To avoid negative effects of the random initial population, each algorithm are run 25 times. The results for the GA, DE, CMAES, and ABC are directly taken from the evolutionary

Table 2 Error values for the 10-dimensional functions F1–F5 at 1e+05 (*FEs* function evaluations)

Function		GA	DE	CMAES	PSO	ABC	CS	IMSS
F1	1st (Min)	6.4899e−9T	0.00e+000	1.81e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	7th	8.3963e−9T	0.00e+000	3.83e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	13th (Median)	8.9819e−9T	0.00e+000	5.39e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	19th	9.8060e−9T	0.00e+000	6.58e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	25th (Max)	9.9931e−9T	0.00e+000	8.59e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	Mean	8.8967e−9	0.00e+000	5.14e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	Std	9.3915e−10	0.00e+000	1.82e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
F2	1st (Min)	8.7414e−9T	0.00e+000	2.41e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	7th	9.5342e−9T	0.00e+000	3.80e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	13th (Median)	9.7326e−9T	0.00e+000	4.99e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	19th	9.8336e−9T	0.00e+000	6.48e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	25th (Max)	9.9951e−9T	0.00e+000	8.76e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	Mean	9.6317e−9	0.00e+000	5.31e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	Std	3.2989e−10	0.00e+000	1.77e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
F3	1st (Min)	7.7457e+2	9.67e−011	1.35e−9	1.583e+004	7.83e+02	5.030e−003	0.000e+000
	7th	5.6899e+4	2.80e−008	4.30e−9	4.394e+004	3.82e+03	5.458e−002	0.000e+000
	13th (Median)	8.6585e+4	1.57e−007	5.58e−9	1.022e+005	6.33e+03	1.391e−001	0.000e+000
	19th	1.3310e+5	8.31e−007	5.97e−9	1.446e+005	8.05e+03	3.618e−001	0.000e+000
	25th (Max)	3.5216e+5	2.01e−005	7.01e−9	3.518e+005	1.42e+04	1.276e+001	0.000e+000
	Mean	1.0806e+5	1.94e−006	4.94e−9	1.160e+005	6.27e+03	8.981e−001	0.000e+000
	Std	8.7160e+4	4.63e−006	1.45e−9	9.093e+004	2.83e+03	2.560e+000	0.000e+000
F4	1st (Min)	7.6909e−9T	0.00e+000	3.99e−9	5.684e−014	0.00e+00	0.000e+000	0.000e+000
	7th	9.1910e−9T	0.00e+000	2.12e−7	3.979e−013	0.00e+00	0.000e+000	0.000e+000
	13th (Median)	9.5550e−9T	0.00e+000	2.45e+4	1.705e−012	0.00e+00	0.000e+000	0.000e+000
	19th	9.8676e−9T	0.00e+000	2.25e+5	5.912e−012	0.00e+00	0.000e+000	0.000e+000
	25th (Max)	9.9807e−9T	1.14e−013	1.65e+7	4.621e−011	0.00e+00	5.684e−014	0.000e+000
	Mean	9.3788e−9	9.09e−015	1.79e+6	6.894e−012	0.00e+00	2.274e−015	0.000e+000
	Std	6.3274e−10	3.15e−014	4.66e+6	1.146e−011	0.00e+00	1.114e−014	0.000e+000
F5	1st (Min)	7.9417e−9T	0.00e+000	3.35e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	7th	8.8876e−9T	0.00e+000	5.19e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	13th (Median)	9.2950e−9T	0.00e+000	6.83e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	19th	9.7643e−9T	0.00e+000	7.61e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	25th (Max)	9.8735e−9T	0.00e+000	9.83e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	Mean	9.1535e−9	0.00e+000	6.57e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	Std	6.3186e−10	0.00e+000	1.88e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000

computation papers [50–53]. The parameter configurations of these algorithms can be found in their corresponding references. For the other algorithms, the parameter settings are adjusted as follows. In PSO, population size is considered to be 100, cognitive and social components (C_1 and C_2) are 1.8, and inertia weight is set to 0.6 [54]. The parameter settings of CS algorithm are set according to the original paper (population size n= 15, $p_a = 0.25$, $a_0 = 0.01$ and $\beta = 1.5$) [8]. For IMSS, values of the common parameters ($\mu = n$, p_a, a_0, β) are inherited from the original CS algorithm. For λ, ρ, and γ parameters, and we conduct a grid search based on different control settings; $n = \{20, 30, \ldots, 200\}$ and $\rho = \gamma = \{0.1, 0.2, \ldots, 0.9\}$. According to our simulations, we found that $\lambda = 100$, $\rho = 0.5$, and $\gamma = 0.7$ can balance the search behavior of the IMSS.

The 1st (best), 7th, 13th (median), 19th, 25th (worst), mean, and standard deviation (Std) of the error values for the 10- and 30- dimensional problems are reported in Tables 2, 3, 4, and 5, respectively. In order to provide a more accurate comparison, the results are provided at different check points (1e+03, 1e+04, 1e+05 and 3e+05) [49]. The results at 1e+03 and 1e+04 checkpoints for both the 30 and 50

Table 3 Error values for the 10-dimensional functions F6–F14 at 1e+05 (*FEs* function evaluations)

Function		GA	DE	CMAES	PSO	ABC	CS	IMSS
F6	1st (Min)	1.8594e−2	0.00e+000	2.31e−9	4.938e−001	1.07e+00	0.000e+000	0.000e+000
	7th	8.7512e−1	0.00e+000	4.22e−9	5.599e−001	3.23e+00	0.000e+000	0.000e+000
	13th (Median)	3.7770e+0	0.00e+000	4.77e−9	6.284e−001	4.44e+00	2.552e−011	0.000e+000
	19th	4.8391e+0	1.14e−013	6.84e−9	7.573e−001	6.50e+00	6.124e−005	0.000e+000
	25th (Max)	1.4830e+2	3.99e+000	9.65e−9	4.618e+001	8.84e+00	3.987e+000	0.000e+000
	Mean	1.8909e+1	1.59e−001	5.41e−9	2.475e+000	4.69e+00	4.366e−001	0.000e+000
	Std	3.9977e+1	7.97e−001	1.81e−9	8.922e+000	2.24e+00	1.186e+000	0.000e+000
F7	1st (Min)	9.8573e−3	7.40e−003	2.32e−9	1.071e−002	2.38e−01	1.086e−002	0.000e+000
	7th	3.6926e−2	8.12e−002	3.71e−9	3.447e−002	3.05e−01	2.617e−002	0.000e+000
	13th (Median)	6.3961e−2	1.08e−001	4.79e−9	4.884e−002	3.37e−01	4.423e−002	0.000e+000
	19th	1.1073e−1	1.60e−001	6.22e−9	7.691e−002	3.90e−01	5.064e−002	0.000e+000
	25th (Max)	2.4620e−1	6.14e−001	7.83e−9	1.066e−001	4.95e−01	1.290e−001	0.000e+000
	Mean	8.2610e−2	1.46e−001	4.91e−9	5.410e−002	3.46e−01	4.511e−002	0.000e+000
	Std	6.2418e−2	1.38e−001	1.68e−9	2.628e−002	6.43e−02	2.527e−002	0.000e+000
F8	1st (Min)	2.0813e+1	2.03e+001	2.00e+1	2.011e+001	2.02e+01	2.019e+001	2.000e+001
	7th	2.0964e+1	2.04e+001	2.00e+1	2.018e+001	2.03e+01	2.032e+001	2.000e+001
	13th (Median)	2.1010e+1	2.04e+001	2.00e+1	2.023e+001	2.04e+01	2.037e+001	2.000e+001
	19th	2.1025e+1	2.05e+001	2.00e+1	2.028e+001	2.04e+01	2.040e+001	2.119e+001
	25th (Max)	2.1069e+1	2.05e+001	2.00e+1	2.040e+001	2.05e+01	2.051e+001	2.275e+001
	Mean	2.0991e+1	2.04e+001	2.00e+1	2.023e+001	2.04e+01	2.036e+001	2.063e+001
	Std	5.7946e−2	7.58e−002	0.00e+0	7.649e−002	6.52e−02	7.160e−002	9.600e−004
F9	1st (Min)	9.9496e−1	0.00e+000	1.69e+1	0.000e+000	2.02e+01	0.000e+000	0.000e+000
	7th	2.9849e+0	0.00e+000	3.58e+1	9.950e−001	2.03e+01	0.000e+000	0.000e+000
	13th (Median)	3.9798e+0	9.95e−001	4.78e+1	2.019e+000	2.04e+01	0.000e+000	0.000e+000
	19th	4.9748e+0	1.99e+000	5.27e+1	3.257e+000	2.04e+01	2.042e−010	2.985e−014
	25th (Max)	1.1940e+1	2.98e+000	6.27e+1	4.975e+000	2.05e+01	8.213e−005	3.980e−008
	Mean	4.0196e+0	9.55e−001	4.49e+1	2.238e+000	2.04e+01	3.465e−006	1.950e−016
	Std	2.2703e+0	9.73e−001	1.36e+1	1.340e+000	6.52e−02	1.608e−005	8.107e−007
F10	1st (Min)	1.9899e+0	3.98e+000	6.96e+0	1.293e+000	1.47e+01	1.131e+001	0.000e+000
	7th	3.9798e+0	5.97e+000	1.39e+1	4.375e+000	1.91e+01	1.493e+001	9.950e−001
	13th (Median)	5.9698e+0	9.95e+000	2.29e+1	6.165e+000	2.27e+01	1.802e+001	9.950e−001
	19th	8.9546e+0	1.49e+001	7.59e+1	7.048e+000	2.70e+01	2.441e+001	2.985e+000
	25th (Max)	2.7390e+1	3.83e+001	1.04e+2	1.034e+001	2.96e+01	3.312e+001	3.980e+000
	Mean	7.3044e+0	1.25e+001	4.08e+1	5.763e+000	2.27e+01	2.003e+001	1.632e+000
	Std	5.2116e+0	7.96e+000	3.35e+1	2.076e+000	4.24e+00	6.132e+000	1.223e+000
F11	1st (Min)	2.9952e−4	2.32e−004	1.30e−1	2.862e+000	4.37e+00	3.780e+000	0.000e+000
	7th	1.1150e+0	1.70e−003	2.56e+0	5.007e+000	5.69e+00	5.659e+000	0.000e+000
	13th (Median)	1.6479e+0	1.46e−002	3.42e+0	5.541e+000	6.12e+00	5.971e+000	1.500e+000
	19th	2.7251e+0	1.50e+000	4.80e+0	5.835e+000	6.66e+00	6.418e+000	2.427e+000
	25th (Max)	4.4945e+0	5.95e+000	6.77e+0	6.443e+000	7.20e+00	6.937e+000	1.084e+001
	Mean	1.9098e+0	8.47e−001	3.65e+0	5.328e+000	6.13e+00	5.781e+000	1.962e+000
	Std	1.1598e+0	1.40e+000	1.66e+0	8.864e−001	6.65e−01	8.658e−001	9.648e−001
F12	1st (Min)	3.9090e+0	0.00e+000	1.88e−9	2.153e−002	6.01e+01	0.000e+000	0.000e+000
	7th	1.4991e+01	0.00e+000	5.12e−9	3.174e+000	2.97e+02	9.993e−003	0.000e+000
	13th (Median)	3.0657e+1	2.27e−013	1.00e+1	1.001e+001	3.43e+02	8.388e+000	1.000e+001
	19th	2.1249e+2	1.44e−011	3.55e+1	1.352e+001	4.99e+02	3.054e+001	1.000e+001

Table 3 continued

Function		GA	DE	CMAES	PSO	ABC	CS	IMSS
	25th (Max)	1.5583e+3	7.12e+002	1.56e+3	1.558e+003	1.02e+03	1.350e+003	1.694e+003
	Mean	2.5951e+2	3.17e+001	2.09e+2	1.781e+002	3.99e+02	7.856e+001	1.010e+002
	Std	4.8933e+2	1.42e+002	4.69e+2	4.359e+002	2.11e+02	2.644e+002	3.534e+002
F13	1st (Min)	3.4913e−1	1.39e−001	1.88e−1	3.611e−001	1.19e−01	3.265e−001	3.631e−002
	7th	7.0643e−1	6.96e−001	4.16e−1	7.289e−001	3.79e−01	7.462e−001	1.879e−001
	13th (Median)	8.1781e−1	9.63e−001	4.79e−1	8.118e−001	5.03e−01	8.793e−001	3.335e−001
	19th	1.0153e+0	1.22e+000	5.60e−1	9.052e−001	1.92e−01	1.068e+000	4.626e−001
	25th (Max)	1.3242e+0	2.44e+000	8.17e−1	1.478e+000	8.43e−01	1.796e+000	9.603e−001
	Mean	8.3793e−1	9.77e−001	4.94e−1	8.332e−001	4.90e−01	8.920e−001	3.435e−001
	Std	2.6913e−1	4.67e−001	1.38e−1	2.096e−001	1.92e−01	2.876e−001	2.308e−001
F14	1st (Min)	1.3864e+0	2.10e+000	3.36e+0	2.189e+000	3.07e+00	2.458e+000	3.702e+000
	7th	2.9682e+0	3.24e+000	3.67e+0	2.780e+000	3.43e+00	3.252e+000	4.896e+000
	13th (Median)	3.0898e+0	3.60e+000	4.05e+0	2.966e+000	3.51e+00	3.454e+000	4.897e+000
	19th	3.2952e+0	3.77e+000	4.25e+0	3.101e+000	3.65e+00	3.556e+000	2.369e+002
	25th (Max)	3.6135e+0	3.89e+000	4.43e+0	3.234e+000	3.78e+00	3.684e+000	2.996e+002
	Mean	3.0456e+0	3.45e+000	4.01e+0	2.913e+000	3.51e+00	3.362e+000	8.917e+001
	Std	4.3662e−1	4.40e−001	3.14e−1	2.532e−001	1.55e−01	2.584e−001	1.236e+002

dimensions and also the results at 1e+05 checkpoint for 50 dimension are available in online supplement 1 (Tables 1, 2, 3, and 4). In these tables, cases in which the algorithm stopped the run before reaching the maximum number of evaluations have been marked with "T" character.

Results for 10-dimensional unimodal benchmark problems F1–F5 are presented in Table 2. As given in Table 2, the IMSS performs better or equally on all the five unimodal problems compared to the other metaheuristic algorithms. More precisely, for functions F1, F2, and F5 the PSO, DE, ABC, CS, and IMSS algorithms could find the global optimal solution. For noisy function F4, results show that IMSS and ABC statistically provide better solution. As is evident from the results on function F3, our proposed method outperforms any other competitor in a significant manner. The unimodal problem F3 is an ill-conditioned and non-separable convex quadratic function and is very difficult to solve using traditional optimization algorithms. However, the proposed IMSS can still find the global optimum on function F3 due to the fact that IMSS is specifically designed to fully make use of the both local and global search information.

Results for 10-dimensional multimodal problems with many local minima, namely F6–F14, are given in Table 3. As shown in this table, the IMSS algorithm provided statistically the best final accuracy on test functions F6 and F7, while all the other algorithms fall into local minima. In these problems, the global optimum lies in a narrow basin (F6) and beyond the initial bounds (F7) that cause difficulty in the optimization process. Nevertheless, the IMSS was the best performing algorithm for these functions. This is due to the fact the incorporated multiple search strategy provides more accurate information about the search space and is able to diverse the search behavior of IMSS. Furthermore, the Q-learning mechanism of IMSS makes it possible to choose the most suitable strategy on such complex fitness landscapes. In the case of function F8, Shifted Rotated Ackley's, all the algorithms fail to locate the global optimum. The main problem here is to detect a one narrow on the flat search region which can lead most algorithms to divert away from the global solution. For test problems F9 (shifted Rastrigin's) and F10 (shifted rotated Rastrigin's), the IMSS algorithm performs better than the other algorithms. Upon examination of Table 3, one can also observe that the IMSS is not very sensitive to rotation of test problems which can be explained by the invariance of CMAES against orthogonal transformations [24]. For test problems F11 and F12, CS and DE provide better solutions, respectively. Also, the IMSS showed to be the winner among all the other algorithms for the expanded function F13. Finally, it can be seen that PSO was able to converge closer to zero much faster than other algorithms for problem F14.

In order to test the scalability of IMSS, we also investigate the performance of IMSS on 30-dimensional problems in Tables 4 and 5. From Table 4, we can note that all algorithms could find the global optimal solution on function F1. In the case of function F2, ABC and the proposed IMSS algorithm show better performance. Table 4 also reveals that optimal solution obtained by the IMSS on function F3 outperforms other optimization algorithms. The results for F4 show again that the IMSS achieved better overall performance, whereas

Table 4 Error values for the 30-dimensional functions F1–F5 at 3e+05 (*FEs* function evaluations)

Function		GA	DE	CMAES	PSO	ABC	CS	IMSS
F1	1st (Min)	8.5813e−9T	0.00e+000	2.98e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	7th	9.0664e−9T	0.00e+000	4.81e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	13th (Median)	9.6205e−9T	0.00e+000	5.40e−9	0.000e+000	0.00e+00	0.000e+000	0.000e+000
	19th	9.6205e−9T	0.00e+000	5.82e−9	5.684e−014	0.00e+00	0.000e+000	0.000e+000
	25th (Max)	9.9339e−9T	0.00e+000	7.06e−9	5.684e−014	0.00e+00	0.000e+000	0.000e+000
	Mean	9.3524e−9	0.00e+000	5.28e−9	2.274e−014	0.00e+00	0.000e+000	0.000e+000
	Std	4.6327e−10	0.00e+000	9.82e−10	2.785e−014	0.00e+00	0.000e+000	0.000e+000
F2	1st (Min)	1.1045e−8	9.84e−004	4.90e−9	7.125e−002	0.00e+00	2.785e−008	0.000e+000
	7th	7.1554e−8	7.98e−003	6.49e−9	9.442e−002	0.00e+00	1.942e−007	0.000e+000
	13th (Median)	1.6077e−7	2.22e−002	6.91e−9	2.066e−001	0.00e+00	4.938e−007	0.000e+000
	19th	4.7296e−7	3.00e−002	7.24e−9	2.946e−001	0.00e+00	1.368e−006	0.000e+000
	25th (Max)	7.1024e−6	2.42e−001	8.77e−9	7.939e−001	0.00e+00	8.916e−006	0.000e+000
	Mean	6.9482e−7	3.33e−002	6.93e−9	2.541e−001	0.00e+00	1.458e−006	0.000e+000
	Std	1.4911e−6	4.90e−002	8.27e−10	1.842e−001	0.00e+00	2.202e−006	0.000e+000
F3	1st (Min)	4.1124e+5	2.85e+005	2.98e−9	8.421e+005	1.76e+05	3.476e+005	0.000e+000
	7th	7.6192e+5	4.91e+005	4.55e−9	1.279e+006	1.99e+05	1.021e+006	0.000e+000
	13th (Median)	1.1241e+6	7.29e+005	5.24e−9	1.957e+006	2.16e+05	1.324e+006	0.000e+000
	19th	1.4380e+6	8.46e+005	5.74e−9	4.097e+006	2.39e+05	1.803e+006	0.000e+000
	25th (Max)	2.0430e+6	9.48e+005	7.57e−9	7.606e+006	2.72e+05	3.195e+006	0.000e+000
	Mean	1.1020e+6	6.92e+005	5.18e−9	2.809e+006	2.20e+05	1.495e+006	0.000e+000
	Std	4.2081e+5	2.04e+005	1.03e−9	1.977e+006	2.53e+04	6.969e+005	0.000e+000
F4	1st (Min)	1.2931e−8	8.87e−001	2.31e+1	8.310e+001	1.76e+05	1.089e+002	0.000e+000
	7th	8.2507e−8	2.99e+000	1.85e+5	1.590e+002	1.99e+05	8.122e+002	0.000e+000
	13th (Median)	1.8822e−7	6.68e+000	7.84e+5	2.805e+002	2.16e+05	1.147e+003	0.000e+000
	19th	5.5370e−7	1.87e+001	3.36e+6	4.206e+002	2.39e+05	2.144e+003	0.000e+000
	25th (Max)	8.3149e−6	7.25e+001	4.48e+8	1.529e+003	2.72e+05	6.698e+003	0.000e+000
	Mean	8.1320e−7	1.52e+001	9.26e+7	3.436e+002	2.20e+05	1.752e+003	0.000e+000
	Std	1.7457e−6	1.81e+001	1.68e+8	3.012e+002	2.53e+04	1.671e+003	0.000e+000
F5	1st (Min)	2.1916e+3	2.08e+001	5.23e−9	1.024e+003	4.53e+03	1.758e+003	5.093e−011
	7th	3.2448e+3	8.13e+001	7.39e−9	1.782e+003	5.54e+03	2.539e+003	6.185e−011
	13th (Median)	4.1725e+3	9.35e+001	8.65e−9	2.055e+003	6.13e+03	2.970e+003	7.094e−011
	19th	4.9204e+3	1.77e+002	9.34e−9	2.268e+003	6.43e+03	4.166e+003	7.640e−011
	25th (Max)	7.0869e+3	8.06e+002	9.99e−9	3.150e+003	7.68e+03	5.928e+003	9.459e−011
	Mean	4.2374e+3	1.70e+002	8.30e−9	2.075e+003	6.02e+03	3.261e+003	6.975e−011
	Std	1.3752e+3	1.84e+002	1.38e−9	5.544e+002	7.16e+02	1.039e+003	1.025e−011

ABC falls into the local minima in the 30-dimensional cases. Based on this observation, it may be inferred that introduction of noise to test functions does not affect the performance of the IMSS algorithm on function F4 regardless of the functions dimensionality. Finally, one can say that the IMSS converges closer to the optimum solution for function F5.

Table 5 presents the results for the 30-dimensional version of functions F6 to F14. As shown in this table, the IMSS approach alone achieved the best final accuracy on F6 and F7 in a statistically meaningful way. Besides, all algorithms have unacceptable results on benchmark F8. For test problems F9 and F10, the IMSS provides satisfactory solutions and outperforms other algorithms. In the case of function F11, the CMAES was the best performing algorithm. Also, the mean best fitness from IMSS is better than other algorithms on test function F12. For F13, it can be seen that the CMAES results are superior to those obtained by the competitor methods. Finally, one can observe that PSO yields best overall results for F14. In general, Table 5 shows that the IMSS outperforms other six state−of-the−art algorithms on the majority of the considered test functions. Furthermore, examination of Tables 2, 3, 4, and 5 indicates that the IMSS

Table 5 Error values for the 30-dimensional functions F6–F14 at 3e+05 (*FEs* function evaluations)

Function		GA	DE	CMAES	PSO	ABC	CS	IMSS
F6	1st (Min)	8.1387e−2	4.47e−001	4.28e−9	7.305e+000	3.35e+01	9.128e−008	0.000e+000
	7th	1.1869e+1	9.96e+000	5.55e−9	1.804e+001	9.37e+01	3.987e+000	0.000e+000
	13th (Median)	1.9260e+1	1.29e+001	6.35e−9	2.197e+001	1.43e+02	4.025e+000	0.000e+000
	19th	6.9430e+2	1.83e+001	7.38e−9	8.920e+001	1.82e+02	1.100e+001	0.000e+000
	25th (Max)	5.0325e+3	1.06e+002	8.44e−9	1.496e+002	2.60e+02	7.137e+001	0.000e+000
	Mean	6.1978e+2	2.51e+001	6.31e−9	5.117e+001	1.38e+02	9.439e+000	0.000e+000
	Std	1.2010e+3	2.90e+001	1.14e−9	4.307e+001	5.81e+01	1.491e+001	0.000e+000
F7	1st (Min)	1.3911e−6	3.54e−010	4.46e−9	7.396e−003	2.68e−08	2.842e−014	0.000e+000
	7th	7.4778e−3	2.60e−009	5.56e−9	9.857e−003	3.69e−01	1.138e−004	0.000e+000
	13th (Median)	9.8946e−3	1.20e−008	6.44e−9	9.857e−003	1.17e+02	9.858e−003	0.000e+000
	19th	1.4809e−2	6.69e−008	7.13e−9	9.865e−003	1.72e+02	1.970e−002	0.000e+000
	25th (Max)	5.1686e−2	1.72e−002	8.79e−9	4.427e−002	2.60e+02	4.672e−002	0.000e+000
	Mean	1.4598e−2	2.96e−003	6.48e−9	1.409e−002	1.05e+02	1.301e−002	0.000e+000
	Std	1.2391e−2	5.55e−003	1.14e−9	9.702e−003	8.20e+01	1.304e−002	0.000e+000
F8	1st (Min)	2.0837e+1	2.08e+001	2.00e+1	2.071e+001	2.08e+01	2.081e+001	2.000e+001
	7th	2.0908e+1	2.09e+001	2.00e+1	2.077e+001	2.09e+01	2.090e+001	2.000e+001
	13th (Median)	2.0925e+1	2.10e+001	2.00e+1	2.081e+001	2.10e+01	2.095e+001	2.125e+001
	19th	2.0952e+1	2.10e+001	2.00e+1	2.083e+001	2.10e+01	2.098e+001	2.136e+001
	25th (Max)	2.1030e+1	2.10e+001	2.00e+1	2.090e+001	2.10e+01	2.104e+001	2.142e+001
	Mean	2.0932e+1	2.10e+001	2.00e+1	2.081e+001	2.09e+01	2.094e+001	2.077e+001
	Std	4.5876e−2	5.11e−002	9.62e−15	4.489e−002	5.63e−02	6.107e−002	6.500e−001
F9	1st (Min)	1.0945e+1	9.95e+000	2.43e+2	1.371e+001	5.49e+01	4.975e+000	3.980e+000
	7th	1.9899e+1	1.49e+001	2.60e+2	2.368e+001	6.15e+01	1.094e+001	7.960e+000
	13th (Median)	2.3879e+1	1.81e+001	2.86e+2	2.718e+001	6.53e+01	1.492e+001	8.955e+000
	19th	2.6864e+1	2.09e+001	3.13e+2	3.184e+001	7.27e+01	1.990e+001	1.094e+001
	25th (Max)	3.6813e+1	3.28e+001	3.66e+2	5.254e+001	7.95e+01	2.487e+001	1.293e+001
	Mean	2.3934e+1	1.85e+001	2.91e+2	2.851e+001	6.60e+01	1.536e+001	8.915e+000
	Std	6.2477e+0	5.20e+000	3.54e+1	9.332e+000	6.74e+00	6.043e+000	2.295e+000
F10	1st (Min)	1.5919e+1	2.15e+001	4.20e+1	2.965e+001	1.60e+02	1.364e+002	2.985e+000
	7th	3.5819e+1	2.98e+001	5.80e+2	4.208e+001	1.99e+02	1.904e+002	6.965e+000
	13th (Median)	5.1738e+1	4.88e+001	6.56e+2	5.192e+001	2.02e+02	2.048e+002	8.955e+000
	19th	6.1687e+1	2.07e+002	7.42e+2	5.525e+001	2.10e+02	2.498e+002	1.094e+001
	25th (Max)	1.8395e+2	2.19e+002	7.92e+2	7.281e+001	2.24e+02	2.842e+002	1.492e+001
	Mean	6.0297e+1	9.69e+001	5.63e+2	5.027e+001	2.01e+02	2.127e+002	9.313e+000
	Std	4.0576e+1	8.23e+001	2.48e+2	1.210e+001	1.44e+01	3.965e+001	2.840e+000
F11	1st (Min)	9.8234e+0	7.39e+000	6.85e+0	2.761e+001	3.36e+01	2.670e+001	1.421e−014
	7th	1.4715e+1	3.76e+001	1.31e+1	2.955e+001	3.48e+01	2.846e+001	3.595e+000
	13th (Median)	1.8224e+1	3.95e+001	1.55e+1	3.085e+001	3.59e+01	2.960e+001	1.241e+001
	19th	2.1543e+1	4.00e+001	1.75e+1	3.183e+001	3.64e+01	3.023e+001	2.064e+001
	25th (Max)	2.7456e+1	4.13e+001	2.26e+1	3.513e+001	3.68e+01	3.245e+001	5.248e+001
	Mean	1.8091e+1	3.42e+001	1.52e+1	3.073e+001	3.56e+01	2.946e+001	1.891e+001
	Std	4.4454e+0	1.03e+001	3.51e+0	1.669e+000	8.84e−01	1.471e+000	1.865e+001
F12	1st (Min)	1.4259e+3	9.56e+000	1.29e+0	8.073e+001	5.09e+04	1.281e+003	0.000e+000
	7th	6.1867e+3	4.02e+002	3.67e+3	2.888e+003	8.65e+04	1.363e+004	8.303e+001
	13th (Median)	9.9526e+3	1.78e+003	9.91e+3	5.119e+003	9.52e+04	2.110e+004	4.416e+002
	19th	1.6045e+4	3.79e+003	1.87e+4	1.050e+004	1.05e+05	2.404e+004	1.559e+003

Table 5 continued

Function		GA	DE	CMAES	PSO	ABC	CS	IMSS
	25th (Max)	6.8650e+4	1.13e+004	3.50e+4	2.976e+004	1.24e+05	4.755e+004	1.507e+004
	Mean	1.3134e+4	2.75e+003	1.32e+4	7.590e+003	9.55e+04	2.040e+004	1.486e+003
	Std	1.3346e+4	3.22e+003	1.15e+4	7.359e+003	1.75e+04	9.570e+003	2.999e+003
F13	1st (Min)	1.8774e+0	1.83e+000	1.48e+0	2.118e+000	8.21e+00	3.089e+000	2.004e+000
	7th	2.6942e+0	2.63e+000	2.10e+0	3.342e+000	1.02e+01	5.470e+000	2.500e+000
	13th (Median)	3.4020e+0	3.18e+000	2.24e+0	3.902e+000	1.08e+01	6.936e+000	2.878e+000
	19th	4.3430e+0	3.68e+000	2.52e+0	4.783e+000	1.14e+01	7.882e+000	3.176e+000
	25th (Max)	6.4165e+0	4.97e+000	2.98e+0	8.173e+000	1.21e+01	8.833e+000	4.304e+000
	Mean	3.5881e+0	3.23e+000	2.32e+0	4.104e+000	1.07e+01	6.744e+000	2.932e+000
	Std	1.0857e+0	8.23e−001	3.46e−1	1.167e+000	9.32e−01	1.435e+000	5.142e−001
F14	1st (Min)	1.2270e+1	1.30e+001	1.28e+1	1.235e+001	1.41e−01	1.235e+001	1.196e+001
	7th	1.3093e+1	1.34e+001	1.36e+1	1.264e+001	1.28e+01	1.281e+001	1.460e+001
	13th (Median)	1.3152e+1	1.34e+001	1.40e+1	1.277e+001	1.32e+01	1.303e+001	1.461e+001
	19th	1.3265e+1	1.35e+001	1.43e+1	1.293e+001	1.34e+01	1.312e+001	1.461e+001
	25th (Max)	1.3481e+1	1.36e+001	1.45e+1	1.305e+001	1.34e+01	1.342e+001	1.463e+001
	Mean	1.3131e+1	1.34e+001	1.40e+1	1.275e+001	1.36e+01	1.298e+001	1.450e+001
	Std	2.6887e−1	1.41e−001	4.04e−1	1.948e−001	1.33e+01	2.323e−001	5.181e−001

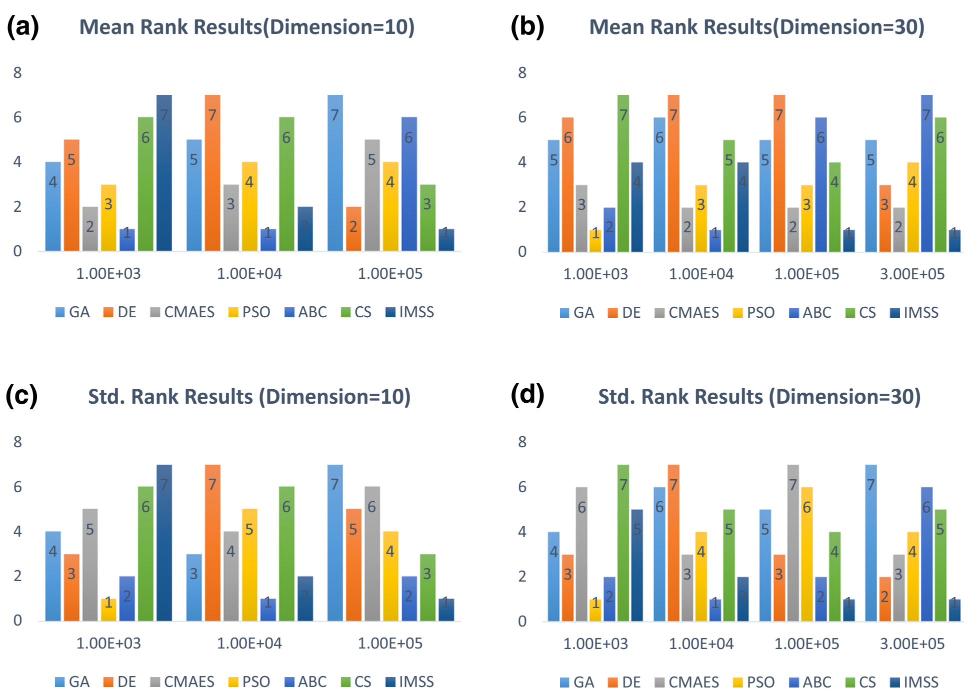

Fig. 5 Histogram of ranks of algorithms based on the mean and standard deviation for the 10- and 30-dimensional problems at different iterations

is not influenced by the problem dimensions and has a good scalability.

In the following, we conduct a rank-based analysis in order to provide an overall comparison among the algorithms. In this study, the compared algorithms are ranked according to their mean fitness and standard deviation values at different check points. The first comparison is helpful to validate the overall convergence behavior and the second one benchmarks the robustness of algorithms. The final mean fitness and standard deviation rank results for 10 and 30 dimensions are presented in Fig. 5. More details about the mathematical calculations and the average ranks are given in Tables 6, 7, 8, and 9. Regarding the convergence rate, it can be observed from Fig. 5a, b that the IMSS is not much promising at ini-

Table 6 Average rank results (ARR) of algorithms based on the mean values for 10-dimensional functions F1–F14 at 1e+03, 1e+04 and 1e+05 iterations

FEs	Algorithms	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	ARR
1e+3	GA	3	3	3	2	4	3	7	7	6	4	4	6	6	4	4.4286
	DE	5	4	5	3	5	5	6	5	3	3	6	4	7	5	4.7143
	CMAES	1	2	2	7	1	1	1	1.5	7	7	5	2	1	6	3.1786
	PSO	4	5	4	4	3	4	4	3	2	2	3	3	3	1	3.2143
	ABC	2	1	1	1	2	2	2	1.5	1	1	1	1	2	2	1.4643
	CS	7	6	6	5	6	7	5	4	4	6	2	7	4	3	5.1429
	IMSS	6	7	7	6	7	6	3	6	5	5	7	5	5	7	5.8571
1e+4	GA	3	4	6	3	5	4	7	7	6	5	1	6	2	3	4.4286
	DE	7	7	5	5	7	6	6	5	5	6	6	4	7	5	5.7857
	CMAES	2	2	1	7	1	1	2	1	7	7	5	7	1	6	3.5714
	PSO	6	6	7	4	3	5	5	2	4	2	4	2	6	1	4.0714
	ABC	1	1	3	1	4	3	4	3	2	4	3	3	3	2	2.6429
	CS	5	5	4	6	6	7	3	4	3	3	2	5	5	4	4.4286
	IMSS	4	3	2	2	2	2	1	6	1	1	7	1	4	7	3.0714
1e+5	GA	7	7	6	6	7	7	5	7	5	3	2	6	5	2	5.3571
	DE	3	3	3	4	3	3	6	4.5	3	4	1	1	7	4	3.5357
	CMAES	6	6	2	7	6	2	2	1	7	7	4	5	3	6	4.5714
	PSO	3	3	7	5	3	5	4	2	4	2	5	4	4	1	3.7143
	ABC	3	3	5	1.5	3	6	7	4.5	6	6	7	7	2	5	4.7143
	CS	3	3	4	3	3	4	3	3	2	5	6	2	6	3	3.5714
	IMSS	3	3	1	1.5	3	1	1	6	1	1	3	3	1	7	2.5357

Table 7 Average rank results (ARR) of algorithms based on the standard deviation values for 10-dimensional functions F1–F14 at 1e+03, 1e+04 and 1e+05 iterations

FEs	Algorithms	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	ARR
1e+3	GA	3	5	2	3	3	3	7	2	4	5	6	4	6	5	4.1429
	DE	5	3	5	4	4	5	5	3	2	1	5	3	7	4	4.0000
	CMAES	1	2	4	7	1	1	1	6	7	7	7	7	1	7	4.2143
	PSO	4	4	3	2	2	4	3	1	3	2	1	2	2	3	2.5714
	ABC	2	1	1	1	5	2	2	5	1	3	4	1	3	6	2.6429
	CS	7	7	7	5	7	7	6	4	5	4	2	6	4	2	5.2143
	IMSS	6	6	6	6	6	6	4	7	6	6	3	5	5	1	5.2143
1e+4	GA	3	4	6	3	6	4	7	1	4	1	5	6	5	1	4.0000
	DE	7	6	5	5	5	6	6	2	6	6	4	4	7	2	5.0714
	CMAES	2	2	1	7	1	1	2	7	7	7	7	7	1	7	4.2143
	PSO	6	7	7	4	2	5	5	4	3	2	3	3	4	6	4.3571
	ABC	1	1	3	1	4	3	4	5	2	4	2	2	2	5	2.7857
	CS	5	5	4	6	7	7	3	3	5	5	1	5	3	3	4.4286
	IMSS	4	3	2	2	3	2	1	6	1	3	6	1	6	4	3.1429
1e+5	GA	6	6	6	6	6	7	5	3	6	4	5	7	5	5	5.5000
	DE	3	3	3	4	3	3	7	6	4	6	6	1	7	6	4.4286
	CMAES	7	7	2	7	7	2	2	1	7	7	7	6	1	4	4.7857
	PSO	3	3	7	5	3	6	4	7	5	2	3	5	3	2	4.1429
	ABC	3	3	5	1.5	3	5	6	4	3	3	1	2	2	1	3.0357
	CS	3	3	4	3	3	4	3	5	2	5	2	3	6	3	3.5000
	IMSS	3	3	1	1.5	3	1	1	2	1	1	4	4	4	7	2.6071

Table 8 Average rank results (ARR) of algorithms based on the mean values for 30-dimensional functions F1–F14 at 1e+03, 1e+04, 1e+05 and 3e+05 iterations

FEs	Algorithms	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	ARR
1e+3	GA	3	2	4	1	7	7	7	1	7	5	5	5	7	4.5	4.6786
	DE	6	6	7	4	5	5	6	4	4	4	6	6	6	4.5	5.2500
	CMAES	1	3	2	7	1	1	1	4	6	7	1	1	5	7	3.3571
	PSO	5	1	5	2	2	3	3	6.5	2	2	4	4	1	1	2.9643
	ABC	2	7	1	6	3	2	2	4	1	1	2	2	3	6	3.0000
	CS	7	5	6	5	6	6	5	6.5	5	6	3	7	4	3	5.3214
	IMSS	4	4	3	3	4	4	4	2	3	3	7	3	2	2	3.4286
1e+4	GA	4	4	4	2	7	5	6	2	6	4	5	6	6	3	4.5714
	DE	7	7	7	6	6	7	7	4	5	5	6	7	7	4	6.0714
	CMAES	1	1	2	7	1	1	1	4	7	7	1	2	1	6	3.0000
	PSO	5	6	6	4	3	4	4	1	3	1	3	4	4	1	3.5000
	ABC	2	2	1	1	5	3	3	4	1	3	4	3	2	5	2.7857
	CS	6	5	5	5	4	6	5	6	2	6	2	5	3	2	4.4286
	IMSS	3	3	3	3	2	2	2	7	4	2	7	1	5	7	3.6429
1e+5	GA	6	4	4	2	6	7	7	5	6	3	1	4	3	3	4.3571
	DE	7	7	6	3	3	5	6	5	2	5	7	2	7	4	4.9286
	CMAES	5	3	2	7	1	1	2	1	7	7	2	5	1	6	3.5714
	PSO	3	6	7	4	4	4	4	2	3	2	5	3	4	1	3.7143
	ABC	1.5	1.5	3	6	7	6	5	5	5	4	6	7	6	5	4.8571
	CS	4	5	5	5	5	3	3	3	4	6	4	6	5	2	4.2857
	IMSS	1.5	1.5	1	1	2	2	1	7	1	1	3	1	2	7	2.2857
3e+5	GA	7	4	5	2	6	7	6	5	4	3	2	4	4	3	4.4286
	DE	2.5	6	4	3	3	4	3	7	3	4	6	2	3	4	3.8929
	CMAES	6	3	2	7	2	2	2	1	7	7	1	5	1	6	3.7143
	PSO	5	7	7	4	4	5	5	3	5	2	5	3	5	1	4.3571
	ABC	2.5	1.5	3	6	7	6	7	4	6	5	7	7	7	5	5.2857
	CS	2.5	5	6	5	5	3	4	6	2	6	4	6	6	2	4.4643
	IMSS	2.5	1.5	1	1	1	1	1	2	1	1	3	1	2	7	1.8571

tial check points (1e+03 and 1e+04). The main reason for this behavior is that Q-learning uses trial and error approach for learning purpose and is not able to choose the most suitable strategy at early iterations. Considering this fact, we can see that after a learning period the IMSS approach outperforms all the other algorithms. Conversely, optimization algorithms such as ABC and PSO perform marginally better at early function evaluations, while at later iterations converge slowly and suffer from the local minima problem. The same results are shown in Fig. 5c, d where the IMSS robustness increases as optimization process goes on.

6.2 Comparison Between IMSS and State-of-the-Art Extensions

For further validation, several numerical experiments are conducted to benchmark the IMSS algorithm performance. For this purpose, the IMSS is compared with seven state-of-the-art DE extensions, thirteen state-of-the-art PSO extensions, five well-known CS extensions, and three well-known CMAES extensions. Experiments are performed on CEC 2005 and CEC 2013 [55] (Table 10) optimization problems with dimensions 30. In order to make a fair comparison, all the results are taken from the literature. For all algorithms, the number of function evaluations and number of runs are the same as those in the original references. The parameters configuration of CS and IMSS are explained in Sect. 6.1.

First, a performance comparison is made among the IMSS and DE extensions. We use DE with self-adapted parameters (jDE) [56], DE with adapted mutation strategies and parameters (SaDE) [57], DE with "current-to-pbest/1" mutation strategy and adaptive parameters (JADE) [58], DE with ensemble of mutation strategies and parameters (EPSDE) [59], DE with success-history-based parameter adaptation (SHADE) [60], DE with composition of multiple strategies and parameter settings (CoDE) [61], and DE with multi-

Table 9 Average rank results (ARR) of algorithms based on the standard deviation values for 30-dimensional functions F1–F14 at 1e+03, 1e+04, 1e+05 and 3e+05 iterations

FEs	Algorithms	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	ARR
1e+3	GA	3	5	3	3	5	6	4	6	5	2	2	6	7	1	4.1429
	DE	5	1	5	2	1	5	7	3	2.5	4	1	7	6	5	3.8929
	CMAES	1	6	2	7	6	1	1	5	7	7	7	3	5	6	4.5714
	PSO	4	2.5	6	1	2	3	3	2	1	1	3	4	1	2	2.5357
	ABC	2	7	1	6	3	2	2	1	2.5	3	4	1	3	7	3.1786
	CS	7	4	7	4	7	7	6	4	6	6	6	5	4	4	5.5000
	IMSS	6	2.5	4	5	4	4	5	7	4	5	5	2	2	3	4.1786
1e+4	GA	4	6	5	2	7	5	6	4	5	2	6	4	6	2	4.5714
	DE	7	7	7	5	6	7	7	2	4	5	1	7	7	4	5.4286
	CMAES	1	1	2	7	2	1	1	6	7	7	7	6	1	6	3.9286
	PSO	5	5	6	3	3	4	4	5	3	3	2	5	4	5	4.0714
	ABC	2	2	1	1	5	3	3	3	1	1	3	3	2	7	2.6429
	CS	6	4	4	6	4	6	5	1	6	6	4	2	5	3	4.4286
	IMSS	3	3	3	4	1	2	2	7	2	4	5	1	3	1	2.9286
1e+5	GA	5	4	4	2	7	7	5	2	6	5	5	5	3	1	4.3571
	DE	7	6	6	3	3	5	6	1	5	3	3	2	7	2	4.2143
	CMAES	6	3	2	7	1	1	2	7	7	7	6	7	1	6	4.5000
	PSO	3	7	7	4	4	6	3	5	4	4	1	3	6	4	4.3571
	ABC	1.5	1.5	3	6	5	4	7	4	2	2	2	6	4	7	3.9286
	CS	4	5	5	5	6	3	4	3	3	6	4	4	5	3	4.2857
	IMSS	1.5	1.5	1	1	2	2	1	6	1	1	7	1	2	5	2.3571
3e+5	GA	6	4	5	2	7	7	5	3	4	5	5	6	5	4	4.8571
	DE	2.5	6	4	3	3	4	3	4	2	6	6	2	3	1	3.5357
	CMAES	7	3	2	7	2	2	2	1	7	7	4	5	1	5	3.9286
	PSO	5	7	7	4	4	5	4	2	6	2	3	3	6	2	4.2857
	ABC	2.5	1.5	3	6	5	6	7	5	5	3	1	7	4	7	4.5000
	CS	2.5	5	6	5	6	3	6	6	3	4	2	4	7	3	4.4643
	IMSS	2.5	1.5	1	1	1	1	1	7	1	1	7	1	2	6	2.4286

population-based ensemble of mutation strategies (MPDED) [62]. The aforementioned DE variants are very efficient and frequently cited in the literature for comparison purposes. Table 11 presents the mean error and standard deviation results of the IMSS and DE extensions from 30 independent runs on the F1–F14 CEC 2005 benchmark problems. The experimental results for DE variants are taken from [62] study. In this table, "+", "=" and "−" denote whether the obtained mean value for IMSS is better than, equal to, or worse than the mean value for a compete algorithm, respectively. It is evident from Table 11 that all algorithms exhibit equal performance on unimodal function F1. Remarkably, the IMSS alone achieves the best final accurate solutions on benchmark functions F2–F7 and F10. Conversely, all DE variants present better performance than the IMSS on function F8. For functions F9 and F12, also it can be seen that CoDE performs better than the other algorithms. Furthermore, the final results of MPDED and SHADE are better than others on functions F12 and F13. In the case of function F14, CoDE, MPDED, and JADE offer more accurate solutions. Altogether, the IMSS shows better or equal results for 8 out of 14 functions and outperforms different DE algorithms.

Next, the performance of IMSS is compared with PSO exertions. More precisely, the IMSS is evaluated against PSO with inertia weight (In-PSO) [63], PSO with constriction factor (Co-PSO) [64], Gaussian PSO (GPSO) [65], Gaussian bare bones PSO (GBBPSO) [66], PSO with exponential distribution (PSO-E) [67], Lévy PSO (LPSO) [68], comprehensive learning PSO (CLPSO) [69], dynamic multiple swarm PSO (DMS-PSO) [70], fully informed particle swarm (FIPS) [71], quantum-behaved PSO (QPSO) [72], Gaussian probability-based QPSO (GAQPSO) [73], QPSO-Type I, and QPSO-Type II extensions. Table 12 reports mean error values and standard deviations out of 30 runs of each algorithm on CEC 2005 benchmark problems F1 to F12. The experimental results for PSO extensions are taken from [73] study. These

Table 10 Benchmark functions

No.	Name	Properties	Search range	Optimum solution
G1	Sphere Function	U, SE	[−100,100]	−1400
G2	Rotated High Conditioned Elliptic Function	U, R, NS	[−100,100]	−1300
G3	Rotated Bent Cigar Function	U, R, NS	[−100,100]	−1200
G4	Rotated Discus Function	U, R, NS, A	[−100,100]	−1100
G5	Different Powers Function	U, SE	[−100,100]	−1000
G6	Rotated Rosenbrock's Function	M, R, NS	[−100,100]	−900
G7	Rotated Schaffers F7 Function	M, R, NS, A	[−100,100]	−800
G8	Rotated Ackley's Function	M, R, NS, A	[−100,100]	−700
G9	Rotated Weierstrass Function	M, R, NS, A	[−100,100]	−600
G10	Rotated Griewank's Function	M, R, NS	[−100,100]	−500
G11	Rastrigin's Function	M, SE, A	[−100,100]	−400
G12	Rotated Rastrigin's Function	M, R, NS, A	[−100,100]	−300
G13	Non−Continuous Rotated Rastrigin's Function	M, R, NS, A, NC	[−100,100]	−200
G14	Schwefel's Function	M, R, NS, A	[−100,100]	−100
G15	Rotated Schwefel's Function	M, R, NS, A	[−100,100]	100
G16	Rotated Katsuura Function	M, R, NS, A	[−100,100]	200
G17	Lunacek Bi_Rastrigin Function	M, NS	[−100,100]	300
G18	Rotated Lunacek Bi_Rastrigin Function	M, R, NS, A	[−100,100]	400
G19	Expanded Griewank's plus Rosenbrock's Function	M, R, NS	[−100,100]	500
G20	Expanded Scaffer's F6 Function	M, R, NS, A	[−100,100]	600
G21	Composition Function 1	M, NS, A	[−100,100]	700
G22	Composition Function 2	M, R, SE, A	[−100,100]	800
G23	Composition Function 3	M, NS, A	[−100,100]	900
G24	Composition Function 4	M, NS, A	[−100,100]	1000
G25	Composition Function 5	M, NS, A	[−100,100]	1100
G26	Composition Function 6	M, NS, A	[−100,100]	1200
G27	Composition Function 7	M, NS, A	[−100,100]	1300
G28	Composition Function 8	M, NS, A	[−100,100]	1400

U unimodal, *M* multimodal, *A* asymmetrical, *R* rotated, *SE* separable, *NS* non-separable, *NC* non-continuous

results demonstrate the effectiveness of the incorporated multiple search strategy and reinforcement learning technique.

Thereafter, performance of the IMSS is compared with several well-known CS extensions in order to provide a more comprehensive study. As the results for CEC 2005 benchmark problems are not reported in the literature, 28 benchmark functions proposed in CEC 2013 special session are used for this purpose (please see Table 10). In this test suite, there are 5 unimodal functions (G1–G5), 15 basic multimodal functions (G6–G20), and 8 composition functions G21–G28 (the detailed descriptions of these benchmark functions can be found in the original paper). For the purpose of comparison, improved CS (ICS) [21], hybrid CS and PSO (CSPSO) [74], CS algorithm with dimension by dimension improvement (DDICS) [75], CS with varied scaling factor (VCS) [29], VCS-LMH, VCS-HML, VCS-L, VCS-H, VICS, VICSPSO, and VDDICS are used. Mean function error obtained by different CS algorithms [29] for the 30-dimensional CEC 2013 benchmark functions is presented in Table 13. According to this table, we can say that IMSS provides better results for the CEC 2013 benchmark problems in comparison with other state-of-the-art CS variants.

Finally, the proposed optimization algorithm is compared with three CMAES extensions on the CEC 2005 benchmarks. Table 14 provides a comparison of the IMSS with a pure local restart CMAES (LR-CMA-ES), a CMAES with wide initial sample distribution and iteratively increasing population size (IPOP-CMA-ES), and particle swarm CMA-ES (PS-CMA-ES). To make the comparison fair, performance results of the CMAES algorithms are taken from the literature [47]. From the results of Table 14, it can be seen that the mean function error of IMSS algorithm is approximately equal to or better than the other CMAES extensions. Remarkably, the IMSS is the only algorithm that is able to find the true global minimum for the test functions F1–F4, F6, and F7. In general, the results of Tables 11, 12, 13, and 14

Table 11 Comparison results between IMSS and DE variants for the 30-dimensional functions F1–F14

Functions		JADE	jDE	SaDE	EPSDE	CoDE	SHADE	MPEDE	IMSS
F1	Mean	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+000
	Std.	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+000
F2	Mean	1.26e−28	3.45e−06	2.77e−06	8.32e−26	6.77e−15	4.51e−29	1.01e−26	0.00e+000
	Std.	1.22e−28	2.76e−06	8.52e−06	2.66e−26	3.44e−15	7.28e−29	2.05e−26	0.00e+000
F3	Mean	8.42e+03	2.44e+05	5.33e+05	6.34e+05	5.65e+05	6.20e+03	1.01e+01	0.00e+000
	Std.	6.58e+03	3.22e+05	4.34e+05	3.44e+06	5.66e+04	5.14e+03	8.32e+00	0.00e+000
F4	Mean	4.13e−16	4.78e−02	1.93e+02	3.88e+02	6.21e−03	7.03e−16	6.61e−16	0.00e+000
	Std.	3.45e−16	2.12e−01	3.22e+02	3.13e+03	4.67e−02	1.01e−15	5.68e−16	0.00e+000
F5	Mean	7.59e−08	5.56e+02	3.76e+03	1.38e+03	3.16e+02	3.15e−10	7.21e−06	6.98e−011
	Std.	5.65e−07	5.62e+02	6.12e+02	7.43e+02	3.62e+02	6.91e−10	5.12e−06	1.03e−011
F6	Mean	1.16e+01	2.65e+01	5.28e+01	6.44e−01	2.32e−01	2.64e−27	9.65e+00	0.00e+000
	Std.	3.16e+01	2.32e+01	4.15e+01	1.24e+00	6.57e−01	1.32e−26	4.65e+00	0.00e+000
F7	Mean	8.27e−03	1.14e−02	1.65e−02	1.58e−02	7.39e−03	2.17e−03	2.36e−03	0.00e+000
	Std.	8.22e−03	7.28e−03	1.58e−02	2.54e−02	6.45e−03	4.29e−03	1.15e−03	0.00e+000
F8	Mean	2.09e+01	2.09e+01	2.09e+01	2.09e+01	2.01e+01	2.05e+01	2.09e+01	2.08e+001
	Std.	1.68e−01	4.54e−01	3.54e−01	2.84e−01	1.25e−01	3.39e−01	5.87e−01	6.50e−001
F9	Mean	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	8.91e+000
	Std.	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	2.29e+000
F10	Mean	2.42e+01	5.46e+01	4.76e+01	5.24e+01	4.21e+01	1.62e+01	1.52e+01	9.31e+000
	Std.	5.44e+00	8.85e+00	1.26e+01	4.64e+01	2.84e+01	3.35e+00	2.98e+00	2.84e+000
F11	Mean	2.57e+01	2.88e+01	1.68e+01	3.77e+01	1.24e+01	2.71e+01	2.58e+01	1.89e+001
	Std.	2.21e+00	2.61e+00	1.64e+00	6.22e+00	3.55e+00	1.57e+00	3.11e+00	1.86e+001
F12	Mean	6.45e+03	8.23e+03	3.44e+03	3.67e+04	3.21e+03	2.90e+03	1.17e+03	1.49e+003
	Std.	2.89e+03	8.54e+03	4.42e+03	5.66e+03	4.48e+03	3.11e+03	8.66e+02	3.00e+003
F13	Mean	1.47e+00	1.67e+00	3.84e+00	2.04e+00	1.66e+00	1.15e+00	2.92e+00	2.93e+000
	Std.	1.15e−01	1.56e−01	2.66e−01	2.12e−01	3.25e−01	9.33e−02	6.33e−01	5.14e−001
F14	Mean	1.23e+01	1.30e+01	1.26e+01	1.35e+01	1.23e+01	1.25e+01	1.23e+01	1.45e+001
	Std.	3.21e−01	2.23e−01	2.83e−01	2.35e−01	3.56e−01	3.67e−01	4.22e−01	5.18e−001
	+/=/−	10/1/3	10/1/3	10/1/3	10/1/3	8/1/5	9/1/4	9/1/4	

reveal that the IMSS performance is comparable with or even better than other state-of-the-art variants of DE, PSO, CS, and CMAES.

6.3 Real-World Optimization Problems

In this section, two constrained engineering problems taken from the optimization literature have been used to investigate the performance of IMSS on real-world applications. The considered test cases are welded beam design and spring design. The obtained results for both problems are compared with the previously published algorithms in the literature. Indeed, our approach is tested against GA, DE, random search (RS), PSO, geometric programming (GP), evolutionary strategy (ES), simulated annealing (SA), socio-behavioral model (SMB), society and civilization algorithm (SCA), HS, unified PSO (UPSO), unified PSO with mutation (UPSOm), SA-GA, hybrid GA artificial immune system (GA-AIS), bacterial foraging optimization (BFO), HS-based sequential quadratic programming (HS-SQP), Nelder–Mead PSO (NM-PSO), T-cell algorithm (TCA), charged system search (CSS), firefly algorithm (FA), bat algorithm (BA), interior search algorithm (ISA), enhanced bat algorithm (EBA), cultural algorithm (CA), and fuzzy proportional-derivative controller (FPC). For a fair comparison, efficiency of these algorithms has been measured based on the number of function evaluations. For IMSS, values of the n (population size) and μ parameters are set to 20 and 5, respectively. Configuration of the other parameters is the same as those expressed in Sect. 6.1.

6.3.1 Constraint Handling

The employed engineering problems have inequality constraints and are not solvable with unconstrained optimization methods such as IMSS. One common approach to cope with

Table 12 Comparison results of mean fitness values and standard deviations between IMSS and different PSO algorithms for the 30-dimensional functions F1–F12

Functions		In-PSO	Co-PSO	GPSO	GBBPSO	PSO-E	LPSO	CLPSO	DMS-PSO	FIPS	QPSO	GAQPSO	GAQPSO-Type I	GAQPSO-Type II	IMSS
F1	Mean	3.88e−013	1.57e−026	7.37e−026	1.79e−025	5.25e−024	1.19e−024	3.55e−008	7.25e−006	3.32e−027	1.27e−027	8.54e−027	7.68e−027	4.89e−012	0.00e+000
	Std.	1.61e−012	1.44e−025	5.92e−025	8.46e−025	2.24e−023	1.15e−023	2.24e−008	2.21e−005	2.57e−028	3.71e−028	1.67e−027	1.92e−027	2.29e−011	0.00e+000
F2	Mean	7.85e+002	1.27e−001	9.88e−002	1.69e+001	2.03e+001	3.70e+001	5.34e+003	8.45e+002	7.55e+001	1.21e+002	1.53e+001	1.13e+002	2.58e+003	1.03e+000
	Std.	6.61e+002	3.80e−001	3.36e−001	1.62e+001	1.52e+001	2.91e+001	1.22e+003	3.50e+002	7.61e+001	6.22e+001	1.70e+001	8.32e+001	1.94e+003	5.54e+000
F3	Mean	3.97e+007	8.65e+006	1.17e+007	7.79e+006	6.29e+006	1.74e+007	5.14e+007	1.28e+007	1.04e+007	4.43e+006	4.17e+006	1.12e+007	3.68e+007	1.85e−005
	Std.	4.64e+007	9.12e+006	2.52e+007	4.32e+006	2.80e+006	1.90e+007	1.35e+007	4.97e+006	4.48e+006	2.33e+006	1.87e+006	6.32e+006	3.16e+007	4.02e−005
F4	Mean	1.12e+004	1.32e+004	2.40e+004	1.14e+004	8.27e+003	7.48e+003	1.61e+004	2.71e+003	1.05e+004	4.00e+003	2.35e+003	3.07e+003	7.71e+003	1.26e+002
	Std.	5.44e+003	6.09e+003	1.25e+004	6.77e+003	3.63e+003	6.66e+003	3.48e+003	9.73e+002	3.85e+003	2.72e+003	1.86e+003	2.00e+003	1.51e+003	1.25e+003
F5	Mean	6.05e+003	7.69e+003	8.03e+003	9.58e+003	7.26e+003	8.25e+003	5.50e+003	2.92e+003	4.35e+003	3.37e+003	2.89e+003	2.53e+003	2.62e+003	2.32e+001
	Std.	2.03e+003	2.39e+003	2.37e+003	3.02e+003	1.87e+003	2.23e+003	8.89e+002	8.12e+002	9.79e+002	9.76e+002	7.32e+002	9.48e+002	9.36e+002	1.95e+001
F6	Mean	2.64e+002	1.23e+002	1.51e+002	1.44e+002	1.90e+002	1.34e+002	1.17e+002	2.96e+002	1.89e+002	8.80e+001	5.66e+001	7.24e+001	1.17e+002	3.25e+001
	Std.	4.37e+002	2.66e+002	3.03e+002	1.65e+002	3.76e+002	2.94e+002	5.49e+001	3.47e+002	2.94e+002	1.60e+002	9.07e+001	1.11e+002	1.48e+002	3.36e+001
F7	Mean	9.91e−001	2.55e−002	2.24e−002	2.05e−002	4.93e−002	4.46e−002	2.42e+000	3.99e−001	3.30e−002	2.08e−002	1.61e−002	1.52e−002	7.00e−002	0.00e+000
	Std.	4.78e+000	3.27e−002	1.78e−002	2.08e−002	5.38e−002	1.18e−001	7.53e−001	2.50e−001	4.64e−002	1.30e−002	1.41e−002	1.25e−002	5.71e−002	0.00e+000
F8	Mean	4.14e−002	5.11e+000	2.77e+000	3.55e+000	3.59e+000	2.22e+000	1.16e−004	1.21e−001	3.84e−001	2.10e−014	1.56e−014	1.99e−014	5.71e−007	2.13e+001
	Std.	2.39e−001	4.57e+000	1.46e+000	6.19e+000	5.53e+000	1.36e+000	6.79e−005	3.72e−001	5.71e−001	1.91e−014	3.11e−015	5.53e−015	1.22e−006	7.25e−002
F9	Mean	3.96e+001	9.67e+001	1.04e+002	8.09e+001	6.65e+001	7.40e+001	6.99e−001	4.00e+001	6.46e+001	2.99e+001	2.55e+001	9.79e+001	1.25e+002	8.86e+000
	Std.	1.62e+001	2.81e+001	2.86e+001	2.21e+001	2.10e+001	2.17e+001	7.98e−001	1.02e+001	1.46e+001	1.06e+001	2.10e+001	4.62e+001	4.58e+001	2.52e+000
F10	Mean	2.40e+002	1.72e+002	1.84e+002	1.64e+002	1.64e+002	1.54e+002	1.51e+002	1.13e+002	1.98e+002	1.18e+002	1.70e+002	2.02e+002	2.15e+002	8.51e+000
	Std.	7.23e+001	5.86e+001	5.74e+001	7.29e+001	5.51e+001	7.63e+001	2.35e+001	7.13e+001	2.18e+001	5.30e+001	3.28e+001	1.37e+001	1.91e+001	2.57e+000
F11	Mean	4.11e+001	3.60e+001	3.35e+001	2.98e+001	2.93e+001	2.90e+001	3.09e+001	2.59e+001	3.55e+001	2.82e+001	3.38e+001	4.07e+001	4.12e+001	2.93e+001
	Std.	6.03e+000	7.27e+000	6.58e+000	3.27e+000	3.21e+000	5.02e+000	1.67e+000	3.15e+000	2.72e+000	6.22e+000	7.64e+000	1.40e+000	1.16e+000	2.32e+001
F12	Mean	3.68e+004	9.96e+003	NA	3.43e+004	1.72e+004	1.63e+004	5.44e+004	1.37e+004	4.63e+004	1.29e+004	6.87e+003	2.47e+004	3.50e+004	1.27e+003
	Std.	4.09e+004	1.62e+004	6.56e+004	6.24e+004	1.09e+004	2.52e+004	1.25e+004	8.90e+003	2.47e+004	1.38e+004	6.32e+003	2.18e+004	3.34e+004	2.15e+003
	+/=/−	11/0/1	10/0/2	10/0/2	11/0/1	10/0/2	10/0/2	10/0/2	10/0/2	11/0/1	10/0/2	11/0/1	11/0/1	11/0/1	

Table 13 Mean function error obtained by CS algorithms for the 30-dimensional CEC 2013 benchmark functions

Functions	CS	ICS-I	VICS-I	ICS-II	VICS-II	CSPSO	VCSPSO	DDICS	VDDICS	VCS-LMH	VCS-HML	VCS-L	VCS-H	VCS	IMSS
G1	0.00e+000	0.00e+000	0.00e+000	2.58e−018	8.96e−017	3.79e−010	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000
G2	2.14e+006	1.27e+007	1.20e+007	3.79e+007	3.00e+007	3.20e+006	8.62e+005	8.16e+006	8.89e+006	5.02e+006	5.00e+006	4.03e+006	4.76e+006	4.98e+006	0.00e+000
G3	1.43e+007	5.44e+007	1.28e+008	1.21e+008	7.06e+008	3.04e+008	1.27e+007	4.06e+008	4.54e+008	6.86e+006	2.01e+006	8.38e+006	3.24e+006	3.94e+006	6.00e+007
G4	7.23e+003	2.19e+004	2.31e+004	3.73e+004	3.84e+004	1.14e+003	3.50e+002	9.51e+004	8.94e+004	1.88e+004	1.57e+004	1.89e+004	1.81e+004	1.67e+004	0.00e+000
G5	3.18e−014	0.00e+000	0.00e+000	4.72e−010	1.39e−009	2.15e−005	1.07e−027	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	0.00e+000	1.00e−014
G6	1.52e+001	1.48e+001	1.41e+001	3.39e+001	3.72e+001	3.61e+001	1.93e+001	1.22e+001	1.34e+001	1.21e+001	1.21e+001	1.12e+001	1.17e+001	1.20e+001	0.00e+000
G7	8.15e+001	6.66e+001	7.38e+001	6.54e+001	8.58e+001	9.66e+001	7.42e+001	1.18e+002	1.05e+002	7.04e+001	4.83e+001	7.55e+001	4.82e+001	6.32e+001	8.04e+000
G8	2.09e+001	2.10e+001	2.10e+001	2.10e+001	2.10e+001	2.10e+001	2.09e+001	2.09e+001	2.10e+001	2.09e+001	2.09e+001	2.10e+001	2.10e+001	2.09e+001	2.13e+001
G9	2.89e+001	2.86e+001	2.85e+001	3.44e+001	3.11e+001	2.62e+001	2.21e+001	3.10e+001	3.15e+001	2.92e+001	2.86e+001	2.85e+001	2.97e+001	2.85e+001	6.36e+000
G10	9.65e−002	2.40e−002	2.80e−002	1.60e+000	2.77e+000	4.69e−001	2.68e−002	2.88e−001	4.54e−001	2.51e−002	1.70e−002	1.46e−002	1.82e−002	1.52e−002	0.00e+000
G11	2.82e+001	3.64e+001	3.19e+001	7.48e+001	8.43e+001	1.30e+002	2.10e+001	0.00e+000	0.00e+000	2.79e+001	1.48e+001	2.31e+001	2.10e+001	2.03e+001	8.88e+000
G12	1.46e+002	1.09e+002	1.05e+002	2.00e+002	1.58e+002	2.08e+002	1.49e+002	2.95e+002	2.51e+002	1.23e+002	7.97e+001	1.10e+002	9.79e+001	9.76e+001	8.24e+000
G13	1.87e+002	1.41e+002	1.54e+002	2.07e+002	1.91e+002	2.41e+002	2.01e+002	3.48e+002	2.94e+002	1.51e+002	1.23e+002	1.55e+002	1.33e+002	1.34e+002	1.31e+001
G14	1.02e+003	2.90e+003	2.89e+003	4.71e+003	4.10e+003	2.14e+003	1.84e+003	1.37e+000	1.71e+000	3.00e+003	2.29e+003	2.58e+003	2.78e+003	2.56e+003	3.77e+003
G15	5.25e+003	5.05e+003	4.84e+003	7.02e+003	5.83e+003	4.37e+003	3.43e+003	3.72e+003	3.90e+003	5.10e+003	5.04e+003	4.72e+003	5.33e+003	5.00e+003	2.74e+003
G16	2.01e+000	1.91e+000	1.87e+000	2.55e+000	2.29e+000	2.37e+000	1.81e+000	9.17e−001	9.21e−001	1.99e+000	2.00e+000	1.84e+000	2.02e+000	1.94e+000	4.89e−003
G17	7.53e+001	1.14e+002	1.13e+002	1.89e+002	1.86e+002	1.72e+002	1.09e+002	3.04e+001	3.04e+001	1.14e+002	9.89e+001	1.10e+002	1.11e+002	1.13e+002	3.69e+001
G18	2.10e+002	1.83e+002	1.80e+002	2.49e+002	2.39e+002	2.20e+002	1.52e+002	3.26e+002	2.98e+002	1.91e+002	1.58e+002	1.66e+002	1.78e+002	1.65e+002	1.32e+002
G19	7.74e+000	8.99e+000	8.97e+000	1.36e+001	1.21e+001	8.80e+000	5.46e+000	3.12e−001	3.55e−001	9.01e+000	6.42e+000	7.37e+000	8.85e+000	7.85e+000	3.25e+000
G20	1.23e+001	1.21e+001	1.21e+001	1.28e+001	1.26e+001	1.34e+001	1.20e+001	1.48e+001	1.43e+001	1.22e+001	1.19e+001	1.22e+001	1.20e+001	1.20e+001	1.47e+001
G21	2.89e+002	2.54e+002	2.29e+002	2.94e+002	3.14e+002	3.05e+002	3.22e+002	1.69e+002	1.70e+002	2.45e+002	2.67e+002	2.20e+002	2.59e+002	2.55e+002	3.00e+002
G22	1.49e+003	3.55e+003	3.09e+003	4.52e+003	4.28e+003	2.44e+003	1.05e+003	2.01e+001	2.00e+001	3.24e+003	2.24e+003	2.87e+003	2.86e+003	2.78e+003	1.59e+003
G23	5.04e+003	5.52e+003	5.40e+003	7.34e+003	6.31e+003	4.92e+003	4.17e+003	5.03e+003	4.90e+003	5.65e+003	5.37e+003	5.41e+003	5.68e+003	5.42e+003	4.07e+003
G24	2.76e+002	2.65e+002	2.72e+002	2.61e+002	2.76e+002	2.78e+002	2.62e+002	2.93e+002	2.86e+002	2.65e+002	2.44e+002	2.62e+002	2.45e+002	2.59e+002	2.17e+002
G25	2.95e+002	2.92e+002	2.96e+002	2.96e+002	3.00e+002	3.06e+002	2.84e+002	3.20e+002	3.13e+002	3.02e+002	2.91e+002	2.97e+002	2.96e+002	2.94e+002	2.91e+002
G26	2.33e+002	2.01e+002	2.01e+002	2.02e+002	2.01e+002	2.19e+002	2.18e+002	2.01e+002	2.00e+002	2.00e+002	2.00e+002	2.00e+002	2.00e+002	2.00e+002	3.15e+002
G27	1.01e+003	1.05e+003	1.03e+003	1.11e+003	1.11e+003	9.70e+002	8.47e+002	8.74e+002	7.95e+002	1.06e+003	1.03e+003	1.02e+003	1.04e+003	1.03e+003	1.02e+003
G28	3.00e+002	3.00e+002	3.00e+002	3.00e+002	3.00e+002	1.22e+003	3.78e+002	3.35e+002	2.55e+002	3.00e+002	3.00e+002	3.00e+002	3.00e+002	3.00e+002	3.00e+002
+/=/−	18/2/8	19/2/7	20/2/6	23/1/4	24/1/3	23/0/5	18/1/9	17/1/10	12/1/7	16/1/11	19/2/7	19/2/7	19/2/7	19/2/7	

Table 14 Mean function error obtained by IMSS and CMAES variants for the 30-dimensional functions F1–F14

Functions	PS-CMA-ES	LR-CMA-ES	IPOP-CMA-ES	IMSS
F1	8.79e−09	5.28e−09	5.42e−09	0.00e+000
F2	9.26e−09	6.93e−09	6.22e−09	0.00e+000
F3	8.00e+04	5.18e−09	5.55e−09	0.00e+000
F4	8.47e−04	9.26e+07	1.11e+04	0.00e+000
F5	3.98e+02	8.30e−09	8.62e−09	6.98e−011
F6	1.35e+01	6.31e−09	5.90e−09	0.00e+000
F7	9.33e−09	6.48e−09	5.31e−09	0.00e+000
F8	2.10e+01	2.00e+01	2.01e+01	2.08e+001
F9	8.85e−09	2.91e+02	9.38e−01	8.91e+000
F10	8.98e−09	5.63e+02	1.65e+00	9.31e+000
F11	3.91e+00	1.52e+01	5.48e+00	1.89e+001
F12	7.89e+01	1.32e+04	4.43e+04	1.49e+003
F13	2.11e+00	2.32e+00	2.49e+00	2.93e+000
F14	1.29e+01	1.40e+01	1.29e+01	1.45e+001
+/=/−	8/0/6	10/0/4	8/0/6	

this issue is to introduce a penalty function to the given problem. In this way, a constrained optimization problem can be transformed into an unconstrained one. The concept of penalty function has often been used with the quadratic loss function as below:

$$\psi(x) = f(x) + C \times \sum_{i=1}^{m} \max(0, g_i(x))^2 \tag{19}$$

where x is a solution vector, $f(x)$ is a cost function, C is the coefficients of penalty, g_i represents ith constrain, and finally, $\psi(x)$ is the penalized cost function. The incorporated parameter C should be large enough, and its value depends on the optimization problem. In this equation, if no constraint is violated, then $\psi(x) = 0$, otherwise $\psi(x) > 0$.

6.3.2 Welded Beam Design

This problem involves minimization of the overall fabrication cost for a welded beam depicted in Fig. 6. The thickness of the weld (x_1), the length of the welded joint (x_2), the width of the beam (x_3), and the thickness of the beam (x_4) are design variables. Furthermore, it also contains shear stress (τ), bending stress in the beam (σ), buckling load on the bar (P_c), deflection of the beam (δ), and side constraints [76]. The ranges of the variables are $0.125 \leq x_1 \leq 5$ and $0.1 \leq x_2, x_3, x_4 \leq 10$. The x_1 and x_2 are discrete variables and integer multiples of 0.0065 in.

The cost function of the welded beam problem is stated as below [76]:

$$\min_x f(x) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) \tag{20}$$

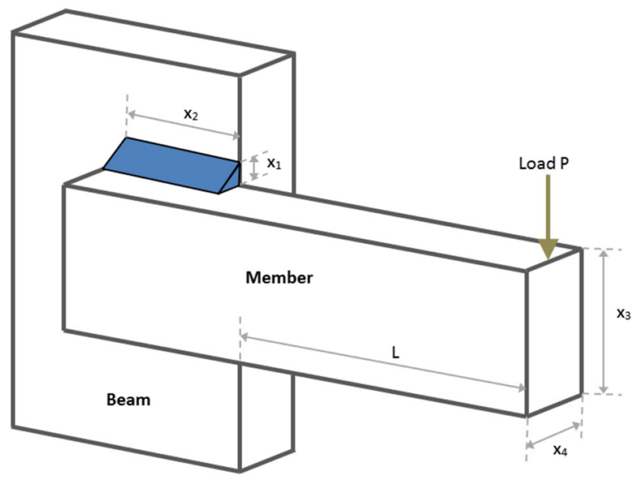

Fig. 6 Decision variables of welded beam problem

Subject to:

$$g_1(x) : \tau(x) - \tau_{\max} \leq 0, \tag{21}$$
$$g_2(x) : \sigma(x) - \sigma_{\max} \leq 0, \tag{22}$$
$$g_3(x) : x_1 - x_4 \leq 0, \tag{23}$$
$$g_4(x) : 0.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leq 0, \tag{24}$$
$$g_5(x) : 0.125 - x_1 \leq 0, \tag{25}$$
$$g_6(x) : \delta(x) - \delta_{\max} \leq 0, \tag{26}$$
$$g_7(x) : P - P_c(x) \leq 0, \tag{27}$$

where

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2}, \tag{28}$$

Table 15 Best solution of the welded beam design problem (*FEs* function evaluations, *NA* not available)

References	Algorithm	X_1	X_2	X_3	X_4	Cost	FEs
[79]	RS	0.2444	6.2819	8.2915	0.2444	2.3815	NA
[80]	GP	0.2536	7.1410	7.1044	0.2536	2.3398	NA
[81]	GA	0.2489	6.1730	8.1789	0.2533	2.4331	1350
[82]	GA	0.2489	6.1097	8.2484	0.2485	2.4000	50,000
[83]	GA	0.2088	3.4205	8.9975	0.2100	1.7483	900,000
[84]	EA	NA	NA	NA	NA	1.8245	5,000
[76]	GA	NA	NA	NA	NA	2.38	40,080
[85]	SA	0.2471	6.1451	8.2721	0.2495	2.4148	NA
[86]	SBM	0.2407	6.4851	8.2399	0.2497	2.4426	19,259
[87]	GA	0.2442	6.2231	8.2915	0.2444	2.3814	320,000
[88]	SCA	0.2444	6.2380	8.2886	0.2446	2.3854	33,095
[89]	GA	0.2443	6.2117	8.3015	0.2443	2.3816	320,000
[90]	PSO	0.2444	6.2175	8.2915	0.2444	2.3810	30,000
[91]	HS	0.2442	6.2231	8.2915	0.2443	2.3807	110,000
[92]	SA	0.2444	6.2175	8.2915	0.2444	2.3810	NA
[93]	UPSO	NA	NA	NA	NA	1.9220	100,000
[93]	UPSOm	NA	NA	NA	NA	1.7656	100,000
[94]	SA	0.2444	6.2158	8.2939	0.2444	2.3811	56,243
[95]	SA-GA	0.2231	1.5815	12.8468	0.2245	2.2500	26,466
[96]	GA-AIS	0.2443	6.2202	8.2915	0.2444	2.3812	320,000
[77]	HS	0.2057	3.4705	9.0366	0.2057	1.7248	200,000
[97]	GA-AIS	0.2444	6.2183	8.2912	0.2444	2.3812	320,000
[98]	BFO	0.2057	3.4711	9.0367	0.2057	2.3868	48,000
[99]	HS-SQP	0.2057	3.4706	9.0368	0.2057	1.7248	90,000
[100]	DE	0.2444	6.2175	8.2915	0.2444	2.3810	24,000
[101]	EA	0.2443	6.2201	8.2940	0.2444	2.3816	28,897
[102]	NM-PSO	0.206	3.468	9.037	0.206	1.7248	80,000
[103]	TCA	0.2444	6.2186	8.2915	0.2444	2.3811	320,000
[104]	CSS	0.2058	3.4681	9.0380	0.2057	1.7249	NA
[105]	FA	0.2015	3.562	9.0414	0.2057	1.7312	50,000
[106]	BA	0.2015	3.5620	9.0414	0.2057	1.7312	50,000
[107]	ISA	0.2443	6.2199	8.2915	0.2443	2.3812	30,000
[108]	EBA	0.2015	3.5620	9.0414	0.2057	1.7312	40,000
Present study	IMSS	0.2015	3.5620	9.0414	0.2057	1.7312	25,000

$$\tau' = \frac{P}{\sqrt{2}x_1x_2}, \tag{29}$$

$$\tau'' = \frac{MR}{J}, \tag{30}$$

$$M = P\left(l + \frac{x_2}{2}\right), \tag{31}$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \tag{32}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \tag{33}$$

$$\sigma(x) = \frac{6PL}{x_4x_3^2}, \tag{34}$$

$$\delta(x) = \frac{4PL^3}{Ex_3^3x_4}, \tag{35}$$

$$P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \tag{36}$$

$$P = 6000\text{lb},\ L = 14\,\text{in},\ E = 30 \times 10^6\,\text{psi},\ G = 12 \times 10^6\,\text{psi},\ \tau_{\max} = 13{,}600\,\text{psi},\ \sigma_{\max} = 30{,}000\,\text{psi},\ \delta_{\max} = 0.25\,\text{in}. \tag{37}$$

Table 15 compares the results obtained by IMSS and other optimization algorithms in the literature. Although it seems fair to say that HS [77], HS-SQP, NM-PSO, and CSS algo-

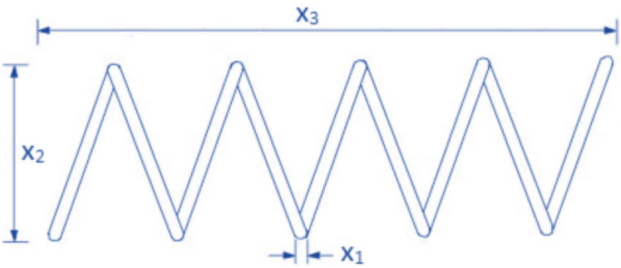

Fig. 7 Decision variables of Tension–compression spring design problem

rithms found the best cost value, it should be noted that they ignored the discrete variables and treated them as continuous. Taking this fact into consideration, we can say that FA, BA, EBA, and the present study have achieved the best solutions without violating any of the constraints. It is also worth to point out that the IMSS requires only 25,000 function evaluations to find the best cost value. More precisely, our proposed algorithm is 2 times faster than FA and BA. It is also 1.6 times faster than EBA algorithm.

6.3.3 Spring Design

The spring design is another well-known engineering optimization problem which consists of the minimization of the weight of the spring depicted in Fig. 7. In this problem, the wire diameter (x_1), the mean coil diameter (x_2), and the number of active coils (x_3) are design variables. Furthermore, it is also subject to four constraints on the minimum deflection, shear stress, surge frequency, and diameter. The ranges of the variables are $0.05 \le x_1 \le 1, 0.25 \le x_2 \le 1.3$, and $2 \le x_3 \le 15$.

The spring design problem is formulated as [78]:

$$\min_x f(x) = (x_3 + 2)\, x_2 x_1^2 \tag{38}$$

Subject to:

$$g_1(x) : 1 - \frac{x_2^3 x_3}{71{,}785 x_1^4} \le 0, \tag{39}$$

$$g_2(x) : \frac{4x_2^2 - x_1 x_2}{12{,}566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \le 0, \tag{40}$$

$$g_3(x) : 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0, \tag{41}$$

$$g_4(x) : \frac{x_1 + x_2}{1.5} - 1 \le 0, \tag{42}$$

The compared results for spring problem are given in Table 16. In this table, violated sets are indicated with bold style. As can be seen, the IMSS converges faster to the

Table 16 Best solution of the spring design problem (*FEs* function evaluations, *NA* not available)

References	Algorithm	X_1	X_2	X_3	G_1	G_2	G_3	G_4	cost	FEs
[83]	GA	0.05148	0.35166	11.6322	−0.0033	−0.0001	−4.0263	−0.7312	0.0127	900,000
[109]	PSO	0.05042	0.32153	13.9799	−0.0019	−0.1556	−3.8994	−0.752	0.01306	1,291
[110]	PSO	0.05147	0.35138	11.6087	−3e−05	**2e−05**	−4.0431	−0.7314	0.01267	200,000
[88]	SCA	0.05216	0.36816	10.6484	0	−0.1314	−4.0758	−0.7198	0.01267	25,167
[111]	CA	0.05	0.3174	14.0318	0	−8e−05	−3.968	−0.7551	0.01272	50,000
[90]	PSO	0.05169	0.35675	11.2871	−4e−05	**2e−05**	−4.0538	−0.7277	0.01267	15,000
[93]	UPSO	NA	NA	NA	NA	NA	NA	NA	0.01312	100,000
[112]	PSO	0.05173	0.35764	11.2445	−0.0009	−1e−05	−4.0513	−0.7271	0.01267	200,000
[94]	SA	0.05174	0.358	11.2139	0	0	−4.0563	−0.7268	0.01267	49,531
[96]	GA-AIS	0.05166	0.35603	11.3296	**1e−05**	−2e−05	−4.0524	−0.7282	0.01267	36,000
[113]	DE	0.05161	0.35471	11.4108	−4e−05	−0.0002	−4.0486	−0.7291	0.01267	204,800
[114]	FPC	0.05236	0.37315	10.3649	**0.002**	**8e−05**	−0.8038	−0.7162	0.01265	NA
[97]	GA-AIS	0.05143	0.35053	11.6612	0	0	−4.0414	−0.732	0.01267	36,000
[115]	ES	0.05164	0.35536	11.3979	−0.0017	−0.0006	−4.0393	−0.7287	0.0127	25,000
[98]	BFO	0.05183	0.35994	11.1071	−0.0002	−0.0002	−4.0584	−0.7255	0.01267	48,000
[100]	DE	0.05169	0.35672	11.289	0	0	−4.0538	−0.7277	0.01267	4,800
[103]	TCA	0.05162	0.35511	11.3845	−3e−05	0	−4.0504	−0.7289	0.01267	36,000
[106]	BA	0.05169	0.35673	11.2885	0	0	−4.0538	−0.7277	0.01267	5,000
[107]	ISA	NA	NA	NA	NA	NA	NA	NA	0.01267	8,000
[108]	EBA	0.0519	0.3620	10.980	NA	NA	NA	NA	0.01267	5,000
Present study	IMSS	0.0516	0.3545	11.4226	−3.01e−4	−8.05e−5	−4.0475	−1.0939	0.01267	2,500

violated sets are in bold

optimum cost value without violating the constraints of the problem. To sum up, upon examination of Tables 15 and 16 we can say that our proposed approach interestingly outperforms other existing methods on the two described engineering problems.

7 Discussion and Conclusion

This paper proposes the IMSS algorithm as a new cuckoo extension to tackle optimization problems more efficiently and effectively. The proposed IMSS algorithm is based on a multiple search strategy and a reinforcement learning technique. The multiple search procedure integrates the exploration capability of CS with the exploitation behavior offered by CMAES. The reinforcement learning technique is another interesting characteristic of the IMSS which helps the algorithm to maintain a good balance between exploration and exploitation search behaviors. To verify the performance of IMSS, its results for solving CEC 2005 and CEC 2013 benchmark problems are compared with the state-of-the-art algorithms in the literature. The results in Tables 2, 3, and 4 clearly show that IMSS has an acceptable performance in the case of 30- and 50- dimensional benchmark functions and thus has proved to be scalable. The obtained results in Tables 2 and 3 also reveal the fact that the introduced IMSS can be more compatible with noisy conditions than other competitive algorithms. Furthermore, we compared the proposed algorithm with the extensions of the DE, PSO, CS and CMAES algorithms and reported the results in Tables 11, 12, 13, and 14. These experiments show that the IMSS outperforms other extensions on the CEC 2005 and CEC 2013 numerical problems. Thereafter, the performance of IMSS was also investigated on two real-world constrained engineering problems. According to the obtained results in Tables 15 and 16, we can say that the IMSS algorithm converges faster than other algorithms to the best solution and consequently reduces the computation time. In general, experimental studies state that the considered modifications promisingly improve the performance of the CS over the discussed optimization problems. It should be point out that main feature of the IMSS is the introduced Q-learning method which enables the IMSS to dynamically adapt its exploration and exploitation behaviors according to the performance requirements of the problem at hand. The obtained results in Fig. 5 are in agreement with this fact. This figure explains that after some learning periods the Q-learning approach enhances both the convergence rate and robustness of the IMSS.

References

1. Mohan, B.C.; Baskaran, R.: A survey: ant colony optimization based recent research and implementation on several engineering domain. Exp. Syst. Appl. **39**(4), 4618–4627 (2012)
2. Qiu, J.; Chen, R.-B.; Wang, W.; Wong, W.K.: Using animal instincts to design efficient biomedical studies via particle swarm optimization. Swarm Evol. Comput. **18**, 1–10 (2014). doi:10.1016/j.swevo.2014.06.003
3. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
4. Geem, Z.W.; Kim, J.H.; Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. Simulation **76**(2), 60–68 (2001). doi:10.1177/003754970107600201
5. Kennedy, J.; Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks, 1995. Proceedings, Nov/Dec 1995, vol. 1944, pp. 1942–1948 (1995)
6. Storn, R.; Price, K.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. **11**(4), 341–359 (1997)
7. Karaboga, D.; Basturk, B.: A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Glob. Optim. **39**(3), 459–471 (2007). doi:10.1007/s10898-007-9149-x
8. Yang, X.-S.; Deb, S.: Cuckoo search via levy flights. In: Nature and Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, 9–11 Dec. 2009, pp. 210-214 (2009). doi:10.1109/NABIC.2009.5393690
9. Yang, X.-S.; Deb, S.: Cuckoo search: recent advances and applications. Neural. Comput. Appl. **24**(1), 169–174 (2014)
10. Civicioglu, P.; Besdok, E.: A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. Artif. Intell. Rev. **39**(4), 315–346 (2013)
11. Ko, Y.-D.; Moon, P.; Kim, C.E.; Ham, M.-H.; Myoung, J.-M.; Yun, I.: Modeling and optimization of the growth rate for ZnO thin films using neural networks and genetic algorithms. Exp. Syst. Appl. **36**(2), 4061–4066 (2009)
12. Durgun, I.; Yildiz, A.R.: Structural design optimization of vehicle components using cuckoo search algorithm. Mater Test **54**(3), 185–188 (2012)
13. Kashan, A.H.; Kashan, M.H.; Karimiyan, S.: A particle swarm optimizer for grouping problems. Inf. Sci. **252**, 81–95 (2013)
14. Gupta, V.; Lehal, G.S.: A survey of text mining techniques and applications. J. Emerg. Technol. Web Intell. **1**(1), 60–76 (2009). doi:10.4304/jetwi.1.1.60-76
15. Yazdani, D.; Nasiri, B.; Sepas-Moghaddam, A.; Meybodi, M.R.: A novel multi-swarm algorithm for optimization in dynamic environments based on particle swarm optimization. Appl. Soft Comput. **13**(4), 2144–2158 (2013)
16. Walton, S.; Hassan, O.; Morgan, K.; Brown, M.: Modified cuckoo search: a new gradient free optimisation algorithm. Chaos Solition Fract **44**(9), 710–718 (2011)
17. Layeb, A.: A novel quantum inspired cuckoo search for knapsack problems. Int. J. Bio Inspir. Comput. **3**(5), 297–305 (2011)
18. Chakraverty, S.; Kumar, A.: A Fuzzy cuckoo-search driven methodology for design space exploration of distributed multiprocessor embedded systems. In: Embedded and Real Time System Development: A Software Engineering Perspective, pp. 131–150. Springer (2014). doi:10.1007/978-3-642-40888-5_5
19. Li, X.; Wang, J.; Yin, M.: Enhancing the performance of cuckoo search algorithm using orthogonal learning method. Neural Comput. Appl. **24**(6), 1233–1247 (2014). doi:10.1007/s00521-013-1354-6

20. Zhang, Y.; Wang, L.; Wu, Q.: Modified adaptive cuckoo search (MACS) algorithm and formal description for global optimisation. Int. J. Comput. Appl. Technol. **44**(2), 73–79 (2012)
21. Valian, E.; Tavakoli, S.; Mohanna, S.; Haghi, A.: Improved cuckoo search for reliability optimization problems. Comput. Ind. Eng. **64**(1), 459–468 (2013). doi:10.1016/j.cie.2012.07.011
22. Kanagaraj, G.; Ponnambalam, S.G.; Jawahar, N.: A hybrid cuckoo search and genetic algorithm for reliability–redundancy allocation problems. Comput. Ind. Eng. **66**(4), 1115–1124 (2013). doi:10.1016/j.cie.2013.08.003
23. Wolpert, D.H.; Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)
24. Hansen, N.; Müller, S.D.; Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol. Comput. **11**(1), 1–18 (2003)
25. Watkins, C.J.; Dayan, P.: Q-learning. Mach. Learn. **8**(3–4), 279–292 (1992)
26. Yang, X.-S.; Deb, S.: Multiobjective cuckoo search for design optimization. Comput. Oper. Res. **40**(6), 1616–1624 (2013)
27. Wang, G.-G.; Gandomi, A.H.; Yang, X.-S.; Alavi, A.H.: A new hybrid method based on krill herd and cuckoo search for global optimization tasks. Int. J. Bio Inspir. Comput. (2012)
28. Rakhshani, H.; Rahati, A.; Dehghanian, E.: Cuckoo search algorithm and its application for secondary protein structure prediction. In: 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), IEEE, pp. 412–417 (2015). doi:10.1109/KBEI.2015.7436080
29. Wang, L.; Yin, Y.; Zhong, Y.: Cuckoo search with varied scaling factor. Front. Comput. Sci. **9**(4), 623–635 (2015)
30. Ilunga-Mbuyamba, E.; Cruz-Duarte, J.M.; Avina-Cervantes, J.G.; Correa-Cely, C.R.; Lindner, D.; Chalopin, C.: Active contours driven by Cuckoo Search strategy for brain tumour images segmentation. Exp. Syst. Appl. **56**, 59–68 (2016)
31. Araghi, S.; Khosravi, A.; Creighton, D.: Intelligent cuckoo search optimized traffic signal controllers for multi-intersection network. Exp. Syst. Appl. **42**(9), 4422–4431 (2015)
32. Naumann, D.; Evans, B.; Walton, S.; Hassan, O.: A novel implementation of computational aerodynamic shape optimisation using Modified Cuckoo Search. Appl. Math. Model. (2015). doi:10.1016/j.apm.2015.11.023
33. Suresh, S.; Lal, S.: An efficient cuckoo search algorithm based multilevel thresholding for segmentation of satellite images using different objective functions. Exp. Syst. Appl. **58**, 184–209 (2016)
34. Wang, J.; Jiang, H.; Wu, Y.; Dong, Y.: Forecasting solar radiation using an optimized hybrid model by Cuckoo Search algorithm. Energy **81**, 627–644 (2015)
35. Huang, L.; Ding, S.; Yu, S.; Wang, J.; Lu, K.: Chaos-enhanced Cuckoo search optimization algorithms for global optimization. Appl. Math. Model. (2015). doi:10.1016/j.apm.2015.10.052
36. Liu, X.; Fu, M.: Cuckoo search algorithm based on frog leaping local search and chaos theory. Appl. Math. Comput. **266**, 1083–1092 (2015)
37. Nguyen, T.T.; Vo, D.N.: The application of one rank cuckoo search algorithm for solving economic load dispatch problems. Appl. Soft Comput. **37**, 763–773 (2015)
38. Huang, J.; Gao, L.; Li, X.: An effective teaching-learning-based cuckoo search algorithm for parameter optimization problems in structure designing and machining processes. Appl. Soft Comput. **36**, 349–356 (2015)
39. Cobos, C.; Muñoz-Collazos, H.; Urbano-Muñoz, R.; Mendoza, M.; León, E.; Herrera-Viedma, E.: Clustering of web search results based on the cuckoo search algorithm and balanced Bayesian information criterion. Inf. Sci. **281**, 248–264 (2014)
40. Li, X.; Yin, M.: Modified cuckoo search algorithm with self adaptive parameter method. Inf. Sci. **298**, 80–97 (2015)
41. AlRashidi, M.; El-Naggar, K.; AlHajri, M.: Convex and non-convex heat curve parameters estimation using cuckoo search. Arab. J. Sci. Eng. **40**(3), 873–882 (2015)
42. Hansen, N.; Niederberger, A.S.; Guzzella, L.; Koumoutsakos, P.: A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Trans. Evol. Comput. **13**(1), 180–197 (2009)
43. Fukagata, K.; Kern, S.; Chatelain, P.; Koumoutsakos, P.; Kasagi, N.: Evolutionary optimization of an anisotropic compliant surface for turbulent friction drag reduction. J. Turbul. **9**, N35 (2008). doi:10.1080/14685240802441126
44. Hansen, N.: The CMA evolution strategy: a comparing review. In: Towards a New Evolutionary Computation, pp. 75–102. Springer (2006). doi:10.1007/3-540-32494-1_4
45. Hansen, N.; Kern, S. (2004) Evaluating the CMA evolution strategy on multimodal test functions. In: Parallel Problem Solving from Nature-PPSN VIII, pp. 282–291. Springer. doi:10.1007/978-3-540-30217-9_29
46. Hansen, N.; Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**(2), 159–195 (2001)
47. Muller, C.; Baumgartner, B.; Sbalzarini, I.F.: Particle swarm CMA evolution strategy for the optimization of multi-funnel landscapes. In: Evolutionary Computation. CEC'09. IEEE Congress on 2009, IEEE, pp. 2685–2692 (2009). doi:10.1109/CEC.2009.4983279
48. BoussaïD, I.; Lepagnot, J.; Siarry, P.: A survey on optimization metaheuristics. Inf. Sci. **237**, 82–117 (2013)
49. Suganthan, P.N.; Hansen, N.; Liang, J.J.; Deb, K.; Chen, Y.-P.; Auger, A.; Tiwari, S.: Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL report 2005005 (2005)
50. Ballester, P.J.; Stephenson, J.; Carter, J.N.; Gallagher, K.: Real-parameter optimization performance study on the CEC-2005 benchmark with SPC-PNX. In: Congress on Evolutionary Computation, pp. 498–505 (2005)
51. Ronkkonen, J.; Kukkonen, S.; Price, K.V.: Real-parameter optimization with differential evolution. In: Proc. IEEE CEC, pp. 506–513 (2005)
52. Auger, A.; Hansen, N.: Performance evaluation of an advanced local search evolutionary algorithm. In: Evolutionary Computation. The 2005 IEEE Congress on 2005, IEEE, pp. 1777–1784 (2005). doi:10.1109/CEC.2005.1554903
53. Akay, B.; Karaboga, D.: A modified artificial bee colony algorithm for real-parameter optimization. Inf. Sci. **192**, 120–142 (2012)
54. Karaboga, D.; Akay, B.: A comparative study of artificial bee colony algorithm. Appl. Math. Comput. **214**(1), 108–132 (2009)
55. Liang, J.; Qu, B.; Suganthan, P.; Hernández-Díaz, A.G.: Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report **201212** (2013)
56. Brest, J.; Greiner, S.; Bošković, B.; Mernik, M.; Zumer, V.: Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE. Trans. Evol. Comput. **10**(6), 646–657 (2006)
57. Qin, A.K.; Huang, V.L.; Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE. Trans. Evol. Comput. **13**(2), 398–417 (2009)
58. Zhang, J.; Sanderson, A.C.: JADE: adaptive differential evolution with optional external archive. IEEE. Trans. Evol. Comput. **13**(5), 945–958 (2009)
59. Mallipeddi, R.; Suganthan, P.N.; Pan, Q.-K.; Tasgetiren, M.F.: Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl. Soft. Comput. **11**(2), 1679–1696 (2011)

60. Tanabe, R.; Fukunaga, A.: Success-history based parameter adaptation for differential evolution. In: Evolutionary Computation (CEC), IEEE Congress on 2013, IEEE, pp. 71–78 (2013). doi:10.1109/CEC.2013.6557555
61. Wang, Y.; Cai, Z.; Zhang, Q.: Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. Evol. Comput. **15**(1), 55–66 (2011)
62. Wu, G.; Mallipeddi, R.; Suganthan, P.N.; Wang, R.; Chen, H.: Differential evolution with multi-population based ensemble of mutation strategies. Inf. Sci. **329**, 329–345 (2016). doi:10.1016/j.ins.2015.09.009
63. Shi, Y.; Eberhart, R.: A modified particle swarm optimizer. In: Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on 1998, IEEE, pp. 69–73. (1998)
64. Clerc, M.: The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: Evolutionary Computation. CEC 99. Proceedings of the 1999 Congress on 1999. IEEE (1999)
65. Krohling, R.: Gaussian swarm: a novel particle swarm optimization algorithm. In: Cybernetics and Intelligent Systems, IEEE Conference on 2004, pp. 372–376. IEEE (2004)
66. Kennedy, J.: Bare bones particle swarms. In: Swarm Intelligence Symposium. SIS'03. Proceedings of the 2003 IEEE 2003, pp. 80–87. IEEE (2003)
67. Krohling, R.; Coelho, L.D.S.: PSO-E: Particle swarm with exponential distribution. In: Evolutionary Computation. CEC 2006. IEEE Congress on 2006, pp. 1428–1433. IEEE (2006)
68. Richer, T.J.; Blackwell, T.M.: The Lévy particle swarm. In: Evolutionary Computation. CEC 2006. IEEE Congress on 2006, pp. 808–815. IEEE (2006)
69. Liang, J.J.; Qin, A.K.; Suganthan, P.N.; Baskar, S.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans. Evol. Comput. **10**(3), 281–295 (2006)
70. Liang, J.; Suganthan, P.N.: Dynamic multi-swarm particle swarm optimizer. In: Swarm Intelligence Symposium. SIS 2005. Proceedings 2005 IEEE 2005, pp. 124–129. IEEE (2005)
71. Mendes, R.; Kennedy, J.; Neves, J.: The fully informed particle swarm: simpler, maybe better. IEEE Trans. Evol. Comput. **8**(3), 204–210 (2004)
72. Sun, J.; Xu, W.; Feng, B.: A global search strategy of quantum-behaved particle swarm optimization. In: Cybernetics and Intelligent Systems, IEEE Conference on 2004, pp. 111–116. IEEE (2004)
73. Sun, J.; Fang, W.; Palade, V.; Wu, X.; Xu, W.: Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point. Appl. Math. Comput. **218**(7), 3763–3775 (2011)
74. Wang, F.; Luo, L.; He, X.-s.; Wang, Y.: Hybrid optimization algorithm of PSO and Cuckoo Search (2011)
75. Wang, L.; Yin, Y.; Zhong, Y.: Cuckoo search algorithm with dimension by dimension improvement. J. Softw. **24**(11), 2687–2698 (2013)
76. Deb, K.: An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Eng. **186**(2), 311–338 (2000)
77. Mahdavi, M.; Fesanghary, M.; Damangir, E.: An improved harmony search algorithm for solving optimization problems. Appl. Math. Comput. **188**(2), 1567–1579 (2007)
78. Cagnina, L.C.; Esquivel, S.C.; Coello, C.A.C.: Solving engineering optimization problems with the simple constrained particle swarm optimizer. Informatica **32**, 319–326 (2008)
79. Siddall, J.N.: Analytical Decision-Making in Engineering Design. Prentice Hall, Upper Saddle (1972)
80. Ragsdell, K.M.; Phillips, D.T.: Optimal design of a class of welded structures using geometric programming. J. Eng. Ind. **98**(3), 1021–1025 (1976). doi:10.1115/1.3438995
81. Deb, K.: Optimal design of a welded beam via genetic algorithms. AIAA J. **29**(11), 2013–2015 (1991)
82. Leite, J.P.; Topping, B.H.: Improved genetic operators for structural engineering optimization. Adv. Eng. Softw. **29**(7), 529–562 (1998)
83. Coello, C.A.C.: Self-adaptive penalties for GA-based optimization. In: Evolutionary Computation. CEC 99. Proceedings of the 1999 Congress on 1999. IEEE (1999)
84. Coello Coello, C.C.: Constraint-handling using an evolutionary multiobjective optimization technique. Civil Eng. Syst. **17**(4), 319–346 (2000)
85. Atiqullah, M.M.; Rao, S.: Simulated annealing and parallel processing: an implementation for constrained global design optimization. Eng. Optim.+ A35 **32**(5), 659–685 (2000)
86. Akhtar, S.; Tai, K.; Ray, T.: A socio-behavioural simulation model for engineering design optimization. Eng. Optim. **34**(4), 341–354 (2002)
87. Barbosa, H.J.; Lemonge, A.C.: An Adaptive Penalty Scheme In Genetic Algorithms For Constrained Optimiazation Problems. In: GECCO, pp. 287–294. Citeseer (2002)
88. Ray, T.; Liew, K.M.: Society and civilization: an optimization algorithm based on the simulation of social behavior. IEEE Trans. Evol. Comput. **7**(4), 386–396 (2003)
89. Lemonge, A.C.; Barbosa, H.J.: An adaptive penalty scheme for genetic algorithms in structural optimization. Int. J. Numer. Methods Eng. **59**(5), 703–736 (2004)
90. He, S.; Prempain, E.; Wu, Q.: An improved particle swarm optimizer for mechanical design optimization problems. Eng. Optim. **36**(5), 585–605 (2004)
91. Lee, K.S.; Geem, Z.W.: A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput. Methods Appl. Mech. Eng. **194**(36), 3902–3933 (2005)
92. Liu, J.-L.: Novel orthogonal simulated annealing with fractional factorial analysis to solve global optimization problems. Eng. Optim. **37**(5), 499–519 (2005)
93. Parsopoulos, K.E.; Vrahatis, M.N.: Unified particle swarm optimization for solving constrained engineering optimization problems. In: Advances in natural computation, pp. 582–591. Springer, Berlin (2005)
94. Hedar, A.-R.; Fukushima, M.: Derivative-free filter simulated annealing method for constrained continuous global optimization. J. Glob. Optim. **35**(4), 521–549 (2006)
95. Hwang, S.-F.; He, R.-S.: A hybrid real-parameter genetic algorithm for function optimization. Adv. Eng. Inf. **20**(1), 7–21 (2006)
96. Bernardino, H.S.; Barbosa, H.J.; Lemonge, A.C.: A hybrid genetic algorithm for constrained optimization problems in mechanical engineering. In: 2007 IEEE Congress on Evolutionary Computation (2007)
97. Bernardino, H.S.; Barbosa, H.J.; Lemonge, A.C.; Fonseca, L.: A new hybrid AIS-GA for constrained optimization problems in mechanical engineering. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence) (2008)
98. Mezura-Montes, E.; Hernández-Ocana, B.: Bacterial foraging for engineering design problems: preliminary results. In: Memorias del 4o Congreso Nacional de Computacion Evolutiva (COMCEV'2008), CIMAT, Gto. Mexico (2008)
99. Fesanghary, M.; Mahdavi, M.; Minary-Jolandan, M.; Alizadeh, Y.: Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. Comput. Methods Appl. Mech. Eng. **197**(33), 3080–3091 (2008)

100. Zhang, M.; Luo, W.; Wang, X.: Differential evolution with dynamic stochastic selection for constrained optimization. Inf. Sci. **178**(15), 3043–3074 (2008)
101. Zhang, J.; Liang, C.; Huang, Y.; Wu, J.; Yang, S.: An effective multiagent evolutionary algorithm integrating a novel roulette inversion operator for engineering optimization. Appl. Math. Comput. **211**(2), 392–416 (2009)
102. Zahara, E.; Kao, Y.-T.: Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems. Exp. Syst. Appl. **36**(2), 3880–3886 (2009)
103. Aragón, V.S.; Esquivel, S.C.; Coello, C.A.C.: A modified version of a T-cell algorithm for constrained optimization problems. Int. J. Numer. Methods Eng. **84**(3), 351–378 (2010). doi:10.1002/nme.2904
104. Kaveh, A.; Talatahari, S.: A novel heuristic optimization method: charged system search. Acta Mech. **213**(3–4), 267–289 (2010)
105. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H.: Mixed variable structural optimization using firefly algorithm. Comput. Struct. **89**(23), 2325–2336 (2011)
106. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H.; Talatahari, S.: Bat algorithm for constrained optimization tasks. Neural Comput. Appl. **22**(6), 1239–1255 (2013)
107. Gandomi, A.H.: Interior search algorithm (ISA): a novel approach for global optimization. ISA Trans. **53**(4), 1168–1183 (2014)
108. Yılmaz, S.; Küçüksille, E.U.: A new modification approach on bat algorithm for solving optimization problems. Appl. Soft. Comput. **28**, 259–275 (2015)
109. Ray, T.; Saini, P.: Engineering design optimization using a swarm with an intelligent information sharing among individuals. Eng. Optim. **33**(6), 735–748 (2001)
110. Hu, X.; Eberhart, R.C.; Shi, Y.: Engineering optimization with particle swarm. In: Swarm Intelligence Symposium. SIS’03. Proceedings of the 2003 IEEE 2003, pp. 53–57. IEEE (2003)
111. Coello Coello, C.A.; Becerra, R.L.: Efficient evolutionary optimization through the use of a cultural algorithm. Eng. Optim. **36**(2), 219–236 (2004)
112. He, Q.; Wang, L.: An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng. Appl. Artif. Intell. **20**(1), 89–99 (2007)
113. Huang, F.-Z.; Wang, L.; He, Q.: An effective co-evolutionary differential evolution for constrained optimization. Appl. Math. Comput. **186**(1), 340–356 (2007)
114. Hsu, Y.-L.; Liu, T.-C.: Developing a fuzzy proportional-derivative controller optimization engine for engineering design optimization problems. Eng. Optim. **39**(6), 679–700 (2007)
115. Mezura-Montes, E.; Coello, C.A.C.: An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int. J. Gen. Syst. **37**(4), 443–473 (2008)