Engineering with Computers
https://doi.org/10.1007/s00366-017-0569-z

ORIGINAL ARTICLE

Multi-level cross entropy optimizer (MCEO): an evolutionary optimization algorithm for engineering problems

Farid MiarNaeimi[1] · Gholamreza Azizyan[1] · Mohsen Rashki[2]

Received: 16 July 2017 / Accepted: 27 November 2017

Abstract
This work proposes a new meta-heuristic optimization algorithm called multi-level cross entropy Optimizer (MCEO). This algorithm is conducted by combination of a group of cross entropy operators. Situations, with a low probability for optimal point are searched with high speed, and also, locations with a high probability for existence of optimal point are investigated with a low speed and high accuracy. The algorithm is then benchmarked on 13 well-known test functions in high dimension spaces (100 dimensions), and the answers are verified by a comparative study with thermal exchange optimization, selfish herds optimization, water evaporation optimization, Moth-Flame optimization, Flower Pollination Algorithm, states of matter search, and gray wolf optimizer. The results indicate that the MCEO algorithm can provide very competitive results in comparison to these well-known meta-heuristics in a similar condition (in term of NFEs). The paper also considers solving three classical engineering design problems (tension/compression spring, welded beam, and pressure vessel designs) and presents a genuine application of the proposed method to the field of dam engineering. The results of the classical engineering design problems and the real application validate that the proposed algorithm is applicable to challenging difficulties with unknown search spaces.

Keywords Optimization · Meta-heuristic · Cross-entropy · Evolution

1 Introduction

Optimization is the technique of finding the optimum possible solution(s) for a particular problem. Modern optimizers known as meta-heuristic algorithms are becoming more and more popular. The term 'heuristic' has its origin in the earlier Greek word 'heuriskein', meaning the ability of discovering new strategies to solve problems. The suffix 'meta', also is another Greek word, which means 'upper-level methodology'. The definition of meta-heuristic was created by Glover [1].

✉ Gholamreza Azizyan
g.azizyan@eng.usb.ac.ir

Farid MiarNaeimi
Farid.Naeimi@pgs.usb.ac.ir

Mohsen Rashki
mrashki@eng.usb.ac.ir

1 Civil Engineering Department, University of Sistan and Baluchestan, Zahedan 98155-987, Iran

2 Department of Architecture Engineering, University of Sistan and Baluchestan, Zahedan, Iran

Meta-heuristic optimization algorithms can be divided into three main classes: Evolutionary Algorithms (EA), Physics-based algorithms, and Swarm Intelligence (SI) algorithms. The first class of the classification is inspired by the idea of mutation and evolution in nature. The most popular algorithm in this branch is GA [2]. Furthermore, differential evolution (DE) algorithm is also placed in this class of meta-heuristic algorithms which was introduced in 1995 by Storn and Price [3]. Some other EAs include Genetic Programming (GP) [4], Evolution Strategy (ES) [5], Evolutionary Programing (EP) [6], biogeography-based optimizer (BBO) [7], Flower Pollination Algorithm (FPA) [8], and state of matter search (SMS) [9].

The second main group of meta-heuristics is physics-based techniques. They are inspired by physic laws of motion, gravitational forces, heat transfer, and other phenomena. Some of the most popular algorithms in this category include: Gravitational Search Algorithm (GSA) [10], Gravitational Local Search Algorithm (GLSA) [11], charged system search (CSS) [12], central force optimization (CFO) [13], chemical reaction optimization (CRO) [14], black hole (BH) [15], Small-World Optimization Algorithm (SWOA)

Published online: 06 December 2017

[16], curved space optimization (CSO) [17], ideal gas molecular movement [18], water evaporation optimization (WEO) [19], and thermal exchange optimization (TEO) [20].

Another subcategory of meta-heuristics is the SI methods. All these algorithms mainly imitate the societal habits of colonies, herds, flocks, or the community of creatures in nature. The most popular SI technique is PSO proposed by Kennedy and Eberhart [21]. Other population-based algorithms can be named as the following: ant colony optimizer (ACO) [22], artificial bees colony (ABC) [23], Artificial Fish-Swarm Algorithm (AFSA) [24], Termite Algorithm [25], Monkey search (MS) [26], Cuckoo Search (CS) [27], Dolphin partner optimization (DPO) [28], Firefly Algorithm (FA) [29], ray optimization [30], Dolphin echolocation [31], gray wolf optimizer (GWO) [32], Moth-Flame optimizer (MFO) [33], vibrating particles system (VPS) [34], selfish herd optimizer (SHO) [35], and Cyclical Parthenogenesis Algorithm (CPA) [36].

The literature of the optimization algorithms shows that there are many SI and EAs techniques proposed to date, many of them inspired by life pattern, hunting and search behaviors. Nevertheless, there is no EA technique in the literature that employs a multi-level of cross entropy (CE) method to solve engineering optimization problems. This motivated our attempt to mathematically model the CE components (Search Agents and Averages) in hyper spaces, propose a new EA algorithm, and investigate its abilities to solve benchmark and real problems. The presented method throughout this study is an evolution-based meta-heuristic method for unconstrained and constrained optimization problems. This research includes examples of unconstrained and constrained single-objective optimization problem and they are solved by a penalty function.

2 Cross entropy optimization

The common cross entropy (CE) methodology is a recently developed optimization method based on a general Monte-Carlo approach to combinatorial and continuous multi-external optimization and importance sampling. This work shows the application of this optimization method to optimize the test benchmark functions and engineering problems.

Cross-Entropy is a method that derives its name from the Kullback–Leibler distance, also known as Cross-Entropy distance, which is a fundamental concept of modern information theory.

The CE method provides a unifying approach to simulate and optimize [37]. Several applications demonstrate the power of the CE method like in [38] for power system reliability evaluation, in [39] for the selection of the right antenna in communication tasks, and in [40] for motion planning.

It was developed as an efficient method for the estimation of rare-event probabilities. The CE method has been successfully applied to a number of difficult combinatorial optimization problems. This optimization method was employed in a set of applications in which most of them deal with static and noisy combinatorial and continuous global optimization problems.

The cross-entropy method can be used to gradually change the sampling distribution of the random search so that the rare-event is more likely to occur. For this purpose, using the CE distance, the method estimates a sequence of sampling distributions that converges to a distribution with probability mass concentrated in a region of near-optimal solutions.

To date, the CE method has been successfully applied to mixed integer nonlinear programming [41] continuous optimal control problems [42]; continuous multi-extreme optimization [43]; multidimensional independent component analysis [44]; optimal policy search [45]; clustering [46]; signal detection [47]; DNA sequence alignment [48]; noisy optimization problems such as optimal buffer allocation [49]; resource allocation in stochastic systems [50]; network reliability optimization [51]; vehicle routing optimization with stochastic demands [52]; power system combinatorial optimization problems [53]; and neural and reinforcement learning [54].

The CE method can be applied to two types of problems:

1. *Estimation* Estimate $\ell = \mathbb{E}[H(\boldsymbol{X})]$, where $\boldsymbol{X}$ is a random object taking values in some set $\mathcal{X}$ and H is a function on $\mathcal{X}$. An important special case is the estimation of probability $\ell = \mathbb{P}[S(\boldsymbol{X}) \geqslant \gamma]$, where S is another function on $\mathcal{X}$.
2. *Optimization* Optimize (that is, maximize or minimize) a given objective function $S(\boldsymbol{x})$ over all $\boldsymbol{x} \in \mathcal{X}$. S can either be a known or a *noisy* function. In the latter case the objective function needs to be estimated, e.g., via simulation.

We review the CE method as an estimation algorithm which naturally leads to a simple optimization tool.

To see how the CE algorithm can be used for optimization purposes, assuming that the goal is to find the minimum of $S(\boldsymbol{x})$ over a given set $\mathcal{X}$. For simplicity assume that there is only one minimizer $\boldsymbol{x}^*$. Denote the minimum by γ^*, so that,

$$S(\boldsymbol{x}^*) = \gamma^* = \min_{x\in\mathcal{X}} S(\boldsymbol{x}). \tag{1}$$

We can now associate with the above optimization problem the estimation of the probability $\ell = \mathbb{P}[S(\boldsymbol{X}) \leqslant \gamma]$, where $\boldsymbol{X}$ has some probability density $f(\boldsymbol{x};\boldsymbol{u})$ on $\mathcal{X}$ (for example, corresponding to the normal distribution on $\mathcal{X}$) and γ is close to the unknown γ^*. Typically, ℓ is a rare-event probability.

Sampling from such a distribution thus produces optimal or near-optimal states. Note that, in contrast to the rare-event simulation setting, the final level $\gamma = \gamma^*$ is generally not known in advance, but the CE method for optimization produces a sequence of levels $\{\hat{\gamma}_t\}$ and reference parameters $\{\hat{\boldsymbol{v}}_t\}$ such that ideally the former tends to the optimal γ^* and the latter to the optimal reference vector $\boldsymbol{v}^*$ corresponding to the point mass at $\boldsymbol{x}^*$; see, e.g., [55].

The common cross-entropy optimization method uses an initial mean and standard deviation in a normal distribution, as mentioned earlier. The best particle in the best position will be appointed as the next mean in the next iteration and the standard deviation (SD) of Search Agents will be decreased. A simple explanation of this method is described in Fig. 1.

3 Multi-level cross entropy optimizer

This section describes the proposed algorithm, and then demonstrates mathematical modeling of the MCEO. Initialization of the algorithm is presented at the beginning, and then the characteristics of a successful optimization algorithm are provided. Afterward, several solutions have been proposed to enhance the exploration and exploitation phase of MCEO.

3.1 Initialization

The multi-level cross entropy (MCEO) methodology presented in this study uses several mean points simultaneously at each iteration. MCEO uses the basic concept of entropy. It repeats the process to achieve stable entropy. The mean points are named as 'Averages' for simplicity. The Averages are generated uniformly in the search space to create the initial condition.

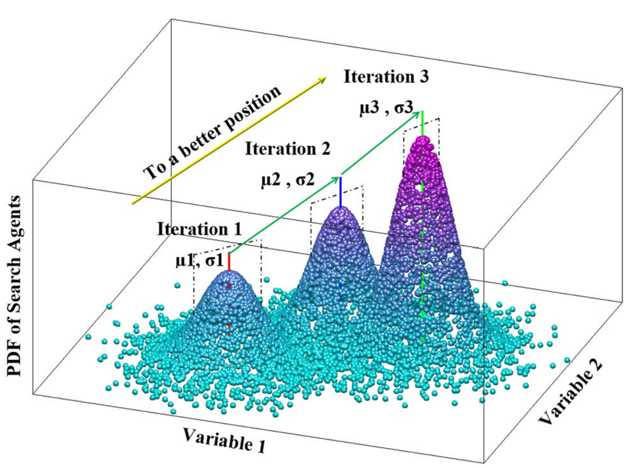

Fig. 1 Common CE optimization steps in three iterations

$$\text{Averages} = {}_1^n\left[p\,(H_\text{b} - L_\text{b})\right]. \tag{2}$$

In which Averages indicates the matrix of mean points in n-dimensional space with a uniform distribution in the first iteration. p is a random number between 0 and 1. L_b and H_b are minimum and maximum bound of search space, respectively. We also named the search points as 'Search Agents'. Subsequently, the search points are produced around Averages with normal distribution and a primary standard deviation. MCEO tries to cover the entire search space with Search Agents in the first iteration. Therefore, the approximate space for each Average is obtained by dividing the length of each bound by the number of Averages. It is suggested that this amount needs to be doubled, to ensure coverage of the entire search space by Averages, in the primary iterations. Thus, the Eq. 3 is obtained for SD of Search Agents around Averages, as stated.

$$\sigma_\text{Initial} = 2 \times \frac{(H_\text{b} - L_\text{b})}{n_\text{Averages}}. \tag{3}$$

n_Average and σ_Initial are the number of Averages and standard deviation of Search Agents, respectively. These steps are shown in Fig. 2.

3.2 Search parameters

Applied motion to the particles in each iteration loop must include five main factors: (a) attraction to global best: the desire to move on to the best position; (b) cohesion: movement of particles should be dependent and in touch with each other; (c) distraction from worst: avoidance from worst position; (d) separation: the ability to separate the particles from each other, to search for an unknown location; and (e)

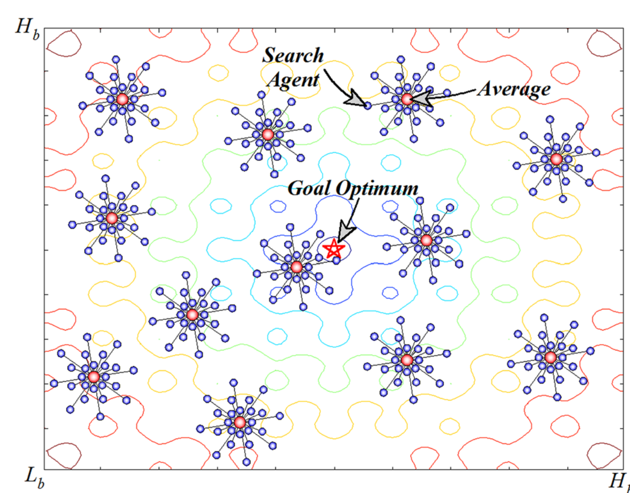

Fig. 2 Initial implementation of Averages and Search Agents

alignment: arrangement of particles in motion; these five factors are presented in Fig. 3.

The steps below are presented based on allocation of these five factors in the MCEO algorithm. A comprehensive matrix of Search Agents and answers of any position should be formed after preliminary exploration. This matrix must be sorted based on the last column (answers), such that the first row of the global matrix should be the best Search Agent with best answer. n_{Average} numbers of the best positions should be selected as new Averages, from top of the comprehensive matrix. The best Search Agents should be selected as new Search Agents and the first next Averages, to avoid removing this position (under the effect of normal distribution of Search Agents). This makes the Search Agents to move toward the global optimum. So, at this stage *a* condition of Fig. 3 (i.e., attraction to global best) is established.

3.3 MCEO's computational complexity

Meta-heuristics are already employed to obtain global or near-global optimum solutions for complex problems, instead of the traditional mathematical approaches. Computation complexity of an algorithm can be an important metric for assessment of its run time that is defined using the implementation of the algorithm's parameters. The computational complexity of the MCEO algorithm is determined by the number of Averages, length of search space, number of problem's dimension, and maximum number of iterations. Considering that the sorting mechanism is used for Search Agent's matrix, to create the new averages. Here are several solutions for speeding up the algorithm and avoidance of local minimum. The first strategy is high speed search in situations with low probability of optimal point and search with high accuracy in locations with high possibility of optimum point.

Stages *b* and *c* of Fig. 3 (Cohesion and Distraction from worst) are considered to be applied in SD selection of Search Agents. Thus, the SD of Search Agents should be decreased in bad situations and be increased in suitable locations. This can be introduced by $\left(\frac{i}{n_{\text{Averages}}}\right)$, in which i is the counter of sorted Averages. It is proposed to apply the cube of mentioned factor to achieve the goal (decrease and increase the SD in good and bad situations, respectively), and parallelizing multiplex cross entropies. φ is also conducted to apply the effects of dimensions of the problem on distribution of Search Agents along the bound of variables. Hence, the following function has been suggested for standard deviation of Search Agents.

$$\sigma_i = \left(\frac{i}{n_{\text{Averages}}}\right)^3 \left(\frac{H_b - L_b}{\varphi}\right), \quad \varphi = \frac{n\text{Var}^2}{n_{\text{Averages}}}. \tag{4}$$

It is also proposed to remove 20% of the worst Averages, and produce new Averages randomly in search space. It allows the algorithm to have the separation ability, mentioned in Fig. 3d. These steps are presented in Fig. 4a, b.

A linear function is also proposed to arrange the motion of Search Agents. It is called Sigma Reduction Factor (SRF), and considered as a factor of standard deviation. It causes a decrease in standard deviation of Search Agents in good positions, and also increases the SD of Search Agents in the bad positions. This phase of the proposed algorithm is depicted in Fig. 5. In this figure, ten Averages are used as a simple example. SRF ranges between 0.95 and 1.05. SD of the first Search Agents in the best position is reduced in each loop using a multiplication factor of 0.95. This trend will keep going until the SD of the worst Average (No. 10) will be multiplied in a factor of 1.05, and it will be increased. It makes the Search Agents to have a unique destination, and their motions will be purposeful. Therefore, the alignment factor in Fig. 3 e will also be satisfied.

Application of the updated SD scatters Search Agents in search space with a specified pattern. The best Averages are exposed to low SRF and produce the Search Agents with a low SD, thus they explore their surrounding space

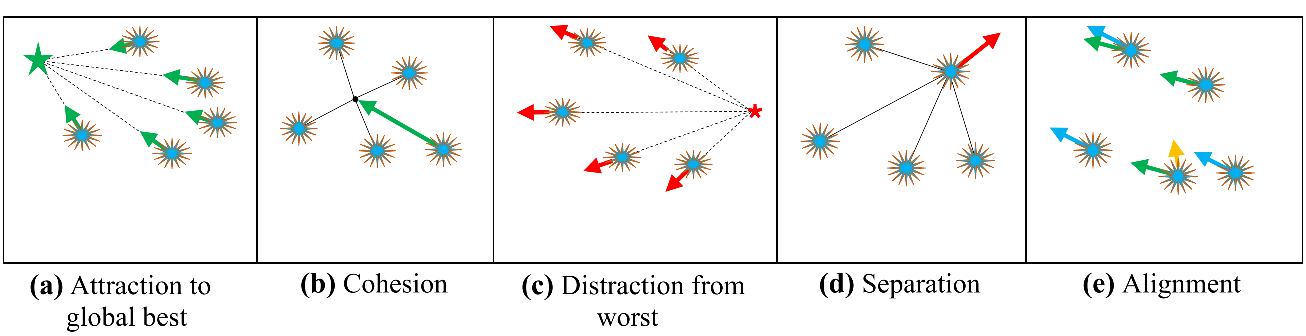

Fig. 3 Fundamentals of particle movements in meta-heuristic algorithms

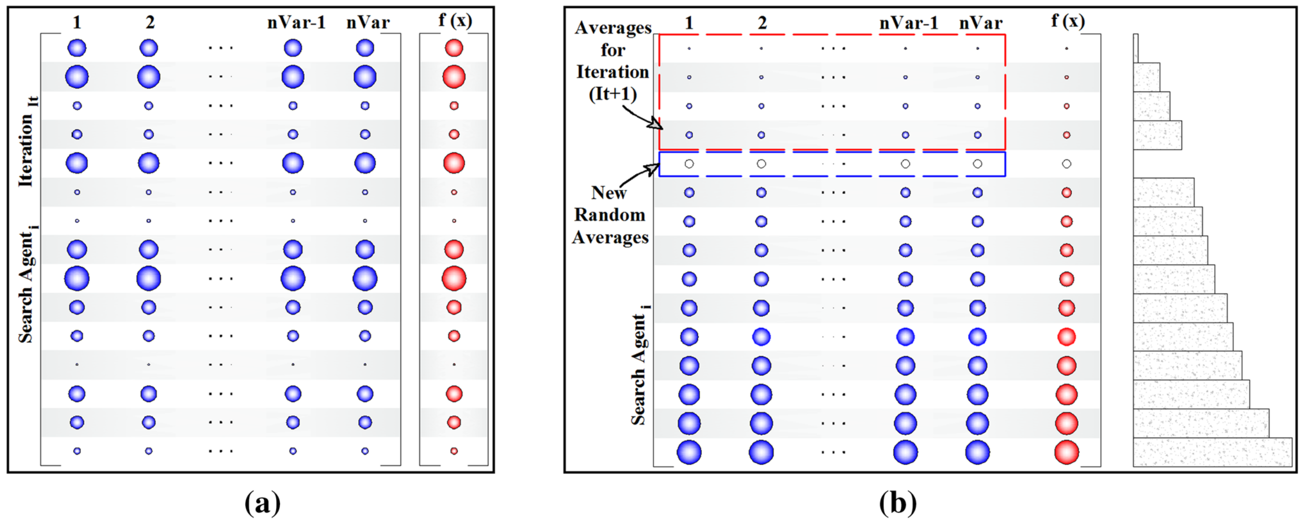

Fig. 4 **a** Response of each Search Agents and **b** sorting of global matrix and new Averages selection

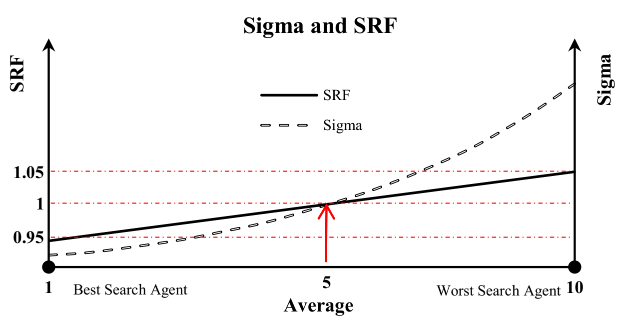

Fig. 5 Standard deviation of Averages and SRF in each iteration

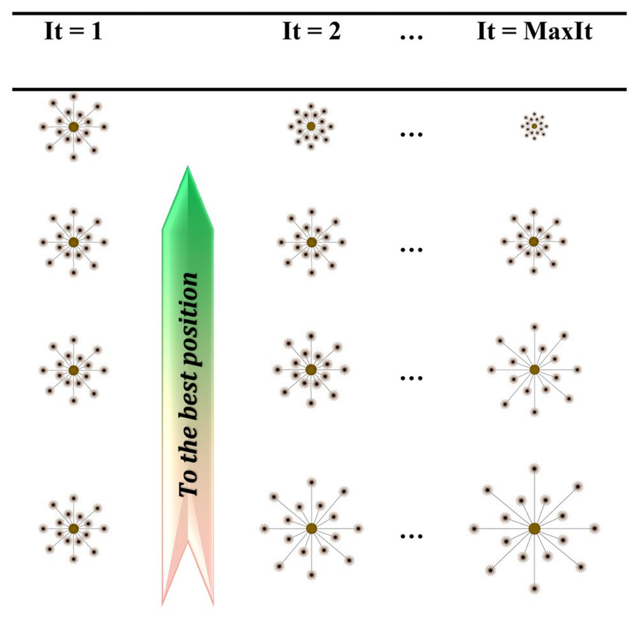

Fig. 6 SRF effects: increase and decrease in standard deviation

more accurately. Averages in the locations with a high SRF produces Search Agents with a high SD, and they find local optimal. The creation of the new Search Agents in each cycle is shown in Fig. 6.

According to Fig. 7, the SD decreased near or around the optimum, and it increased in faraway locations from the optimum. The rate of increase and decrease becomes more in each iteration using SRF.

Allocation of a percentage of the number of Averages to generate random new position is another technique to escape from local optima, which has been done in MCEO. In addition, some of the Search Agents went out of search space due to the high amount of SD. These Search Agents can be mirrored on the border of the search space and returned to the authorized area. Exploration of the feasible space from several perspectives is the main novelty of MCEO. It finds very close positions to Global Best (GB) for the problems with one optimum (uni-modal benchmark functions for examining the exploration ability) by a few primary iterations. Thus, it provides the exact GB with a few loops iterations.

The main performance of MCEO has also becomes apparent when it is employed to solve multi-modal problems with several local optimums. MCEO is capable of finding the exact optimum with a multi-directional performance within the shortest possible time and with a few NFEs in such cases. It is worth mentioning that since the proposed MCEO is a Monte Carlo based optimizer, it has the capability of taking the advantages of posterior distribution approach [56], variance reduction methods (i.e., importance sampling) [57] and low discrepancy sampling [58] (i.e., Latin hyper cube

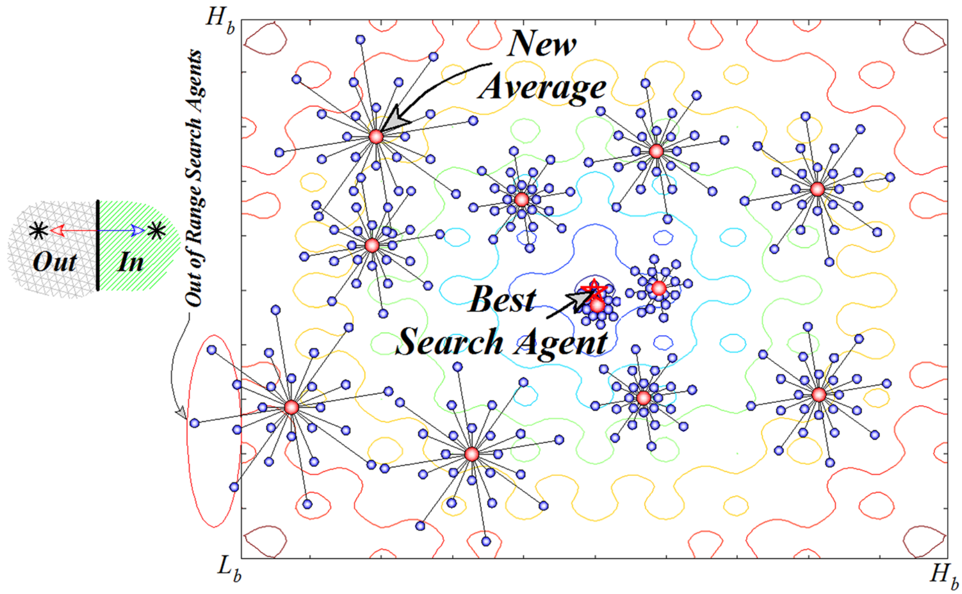

Fig. 7 Movement and standard deviation of Averages

sampling, Halton sequence, etc.) during the optimization process (Figs. 8, 9).

4 Results and discussion

It is necessary to statistically validate the obtained responses and compare them with the results of existing algorithms, to prove the lack of chance in finding the optimum due to the random nature of meta-heuristics. MCEO algorithm is benchmarked on 13 known benchmark functions. These benchmark functions are stated in Tables 1 and 2 where dim indicates the function's dimension, range is the boundary of the function's search space, and $f_{\min}$ is the optimum. The MCEO algorithm had been run 30 times on each benchmark function. The statistical results (average and standard deviation) of uni-modal functions are reported in Table 3, and the p values of

Fig. 8 Pseudo code of MCEO

Initialize Averages and Search Agents population

Sort Search Agents based on cost

for Iteration=1:MaxIteration

    Spread Search Agents around each Average with normal distribution and $\sigma_{Iteration}$

    Calculate the cost of each Search Agents

    Sort Search Agents based on cost

    Select '$n_{Averages}$**(1-α)' numbers of Search Agents from the top of sorted matrix*

    Select '$n_{Averages}$**(α)' numbers of Averages with uniform random distribution*

    Update $\sigma_{Iteration+1}$ *using SRF*

end for

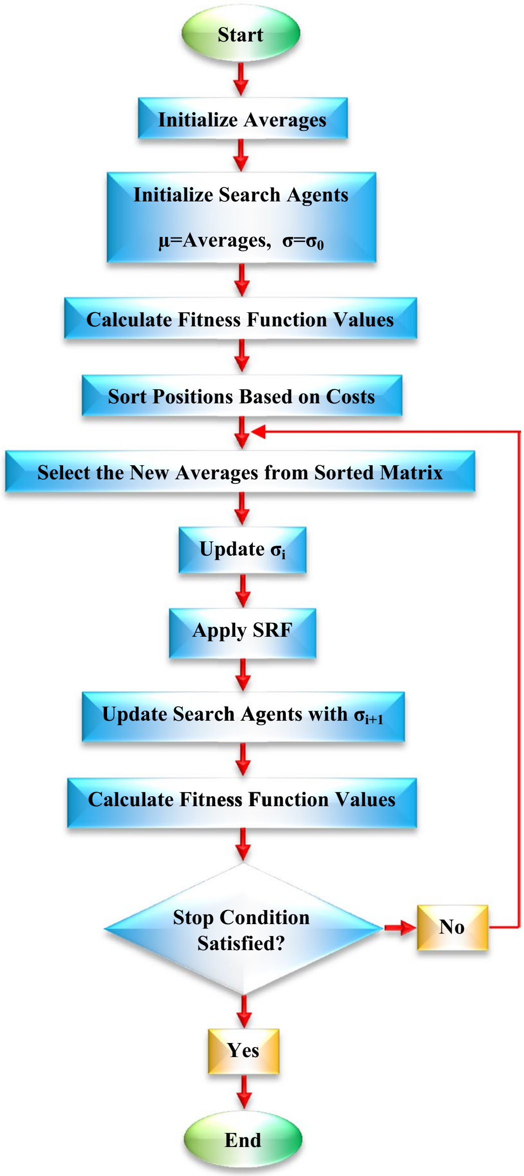

Fig. 9 Flowchart of MCEO

Wilcoxon test are shown in Table 4. These results are presented in Tables 5 and 6 for multi-modal functions. The MCEO algorithm is compared to TEO, SHO, WEO, MFO, FPA, SMS, and GWO in order to verify the results. Tension/compression spring design, welded beam design and pressure vessel design are also conducted for testing the ability of MCEO in semi-real constrained problem solving. Moreover, design and geometry optimization of SHAFAROUD concrete gravity dam has been done and the responses have been compared to PSO and GA algorithms.

Table 1 Uni-modal benchmark functions

Function	Dim	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	100	[− 100, 100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	100	[− 10, 10]	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	100	[−100, 100]	0
$F_4(x) = \max(\lvert x_i \rvert)$	100	[−100, 100]	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2 \right]$	100	[− 30, 30]	0
$F_6(x) = \sum_{i=1}^{n} \left(\left[x_i + 0.5 \right] \right)^2$	100	[−100, 100]	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{rand}[0, 1)$	100	[− 1.28,1.28]	0

4.1 Unconstrained benchmark test functions

Uni-modal test functions are suitable means of demonstrating the algorithm's ability in exploration phase. They are listed in Table 1.

Multi-modal benchmark functions are also a proper criterion for the exploitation phase, and are listed in Table 2.

According to Table 3, the MCEO algorithm has the best answers in uni-modal benchmark functions compared to MFO, FPA, GWO, and SMS in all cases. It represents the ability of the MCEO algorithm in finding the optimal response and exploitation phase. The non-parametric statistical significance proof called the Wilcoxon's rank sum test for independent samples was conducted over the most effective fitness values discovered by each of the compared method on 30 independent test. Table 4 reports the p values produced from the Wilcoxon's test for the uni-modal function and Table 6 shows this for the multi-modal test functions, by considering a 5% significance level. All values for MCEO are under 0.05, and they also satisfy the 5% significance level criteria, as shown with the reported p values. Such evidence indicates that the proposed method's answers are statistically significant, and also they had not occurred by coincidence because of common noise within the process. It should be mentioned that 50 Search Agents and 1000 iterations are utilized for all of the algorithms to conduct a fair comparison. The number of Averages and Search Agents in MCEO are 5 and 10, respectively, for this purpose.

The results presented in Table 5, show that MCEO has a better performance compared to most of the powerful available algorithms. MCEO in both exploration and also in the exploitation has a very good balance, despite the immense local minimum in evaluated functions. Finding a very close

Table 2 Multi-modal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_8(x)=\sum_{i=1}^{n}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	100	[− 500, 500]	− 418.9829*n
$F_9(x)=\sum_{i=1}^{n}\left(x_i^2-10\cos\left(2\pi x_i\right)+10\right)$	100	[−5.12, 5.12]	0
$F_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{\sum x_i^2}{2}}\right)-\exp\left(\frac{\sum\cos\left(2\pi x_i\right)}{2}\right)+20+e$	100	[− 32, 32]	0
$F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^{\,2}-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	100	[− 600, 600]	0
$F_{12}(x)=\frac{\pi}{n}\left\{10\sin\left(\pi y_1\right)+\sum_{i=1}^{n-i}\left(y_i-1\right)^2\left[1+10\sin^2\left(\pi y_{i+1}\right)\right]+\left(y_n-1\right)^2\right\}+\sum_{i=1}^{n}u\left(x_i,10,100,4\right)$	100	[−50, 50]	0
$y_i=1+\frac{x_i+1}{4},\quad u\left(x_i,a,k,m\right)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$			
$F_{13}(x)=0.1\left\{\sin^2\left(3\pi x_1\right)+\sum_{i=1}^{n}\left(x_i-1\right)^2\left[1+\sin^2\left(3\pi x_i+1\right)\right]+\left(x_n-1\right)^2\left[1+\sin^2\left(2\pi x_n\right)\right]\right\}+\sum_{i=1}^{n}u\left(x_i,5,100,4\right)$	100	[− 50, 50]	0

Table 3 Results of uni-modal benchmark functions

F	MCEO		TEO [20]		SHO [35]		WEO [19]	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F1	**2.55E−39**	**1.91E−40**	8.03E−34	5.11E−35	5.32E−19	4.86E−21	3.42E−26	2.10E−29
F2	**4.01E−20**	**7.07E−22**	4.42E−15	9.30E−17	8.92 E−09	4.01E−10	5.83E−11	3.99E−13
F3	**1.63E−38**	**1.32E−38**	8.12E−37	4.21E−38	6.74 E−23	7.32E−25	4.44E−35	1.06E−36
F4	**1.30E−20**	**1.89E−22**	5.46E−20	3.77E−21	5.94 E−15	1.02E−16	5.02E−17	4.80E−19
F5	**9.74E−04**	**7.07E−04**	6.28E−03	2.43E−04	3.62 E−02	7.12E−03	1.03E−03	9.91E−04
F6	**0**	**0**	0	7.39E−49	9.10 E−22	5.84E−23	0	10.2E−46
F7	**2.76E−04**	**1.93E−04**	6.91E−03	3.07E−04	6.42 E−01	7.56E−02	8.80E−03	5.42E−04
F	MFO [33]		FPA [8]		SMS [9]		GWO [32]	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F1	1.17E−04	1.50E−04	2.03E+02	7.83E+01	1.20E+02	0	2.62E−34	2.29E−34
F2	6.39E−04	8.77E−04	1.11E+01	2.91E+00	2.05E−02	4.71E−03	6.10E−20	6.96E−21
F3	6.96E+2	1.88E+02	2.37E+02	1.36E+02	3.78E+04	0	1.76E−01	1.48E−01
F4	7.06E+1	5.27E+00	1.25E+01	2.29E+00	6.91E+01	3.87E+00	4.75E−05	4.26E−05
F5	139.1487	1.20E+02	1.09E+04	1.20E+04	6.38E+06	7.29E+05	9.79E+01	6.56E−01
F6	1.13E−04	9.87E−05	1.75E+02	6.34E+01	4.14E+04	3.29E+03	8.48E+00	3.93E−01
F7	9.11E−02	4.64E−02	1.35E−01	6.12E−02	4.95E−02	2.40E−02	1.11E−03	5.56E−04

Bold indicates the best answers obtained by MCEO

Table 4 p values of the Wilcoxon test for uni-modal functions ($p \geq 0.05$ have been bolded)

F	MCEO	TEO [20]	SHO [35]	WEO [19]	MFO [33]	FPA [8]	SMS [9]	GWO [32]
F1	N/A	4.89E − 15	5.22E − 12	8.44E − 14	5.83E − 04	1.83E − 04	6.39E − 05	3.22E − 05
F2	N/A	4.89E − 15	3.02E − 11	8.44E − 14	5.83E − 04	1.83E − 04	1.83E − 04	**2.76E − 01**
F3	N/A	4.89E − 15	2.89E − 11	4.82E − 14	5.83E − 04	1.83E − 04	6.39E − 05	4.85E − 05
F4	N/A	4.89E − 15	3.02E − 11	3.21E − 14	1.83E − 04	1.83E − 04	1.83E − 04	7.28E − 03
F5	N/A	4.89E − 15	3.01E − 11	8.44E − 14	5.83E − 04	1.83E − 04	1.83E − 04	**1.48E − 01**
F6	N/A	9.61E − 15	8.15E − 11	8.44E − 14	5.83E − 04	1.83E − 04	1.83E − 04	8.90E − 04
F7	N/A	5.44E − 15	6.68E − 11	2.19E − 14	1.40E − 02	1.83E − 04	1.83E − 04	4.15E − 02

Table 5 Results of multi-modal benchmark functions

F	MCEO		TEO [20]		SHO [35]		WEO [19]	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F8	**−41898.2887**	**0**	−41898.26	2.31E−22	−40207.3	4.78E−14	−41898.71	5.49E−19
F9	**0**	**0**	1.02E−46	4.79E−49	7.28E−38	1.47E−40	7.39E−43	4.53E−45
F10	**2.07E−15**	**0**	5.99E−13	6.01E−15	5.80E−08	3.61E−09	1.62E−12	7.22E−13
F11	**0**	**0**	8.73E−40	4.56E−42	4.18E−28	5.07E−29	4.90E−36	3.47E−37
F12	**1.71E−02**	**4.96E−05**	5.13E−02	7.30E−04	7.04E−01	4.15E−02	5.84E−02	6.14E−04
F13	**9.57E−02**	**1.77E−03**	1.40E−01	2.27E−02	3.72E+00	4.23E−01	9.12E−01	4.20E−02
F	MFO [33]		FPA [8]		SMS [9]		GWO [32]	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F8	−8496.78	7.25E+02	−8086.74	1.55E+02	−3942.82	4.04E+02	−1395.621	7.03E+02
F9	8.46E+01	1.61E+01	9.26E+01	1.42E+01	1.52E+02	1.85E+01	1.42E−13	5.68E−14
F10	1.26E+00	7.29E−01	6.84E+00	1.24E+00	1.91E+01	2.38E−01	7.55E−14	0
F11	1.90E−02	2.17E−01	2.71E+00	7.27E−01	4.20E+02	2.52E+01	0	0
F12	8.94E−01	8.81E−01	4.10E+00	1.04E+00	8.74E+06	1.40E+06	1.78E−01	2.03E−02
F13	1.15E−01	1.93E−01	6.23E+01	9.48E+01	1.00E+08	0	5.52	3.84E−01

Bold indicates the best answers obtained by MCEO

Table 6 *p* values of the Wilcoxon test for muli-modal functions ($p \geq 0.05$ have been bolded)

F	MCEO	TEO [20]	SHO [35]	WEO [19]	MFO [33]	FPA [8]	SMS [9]	GWO [32]
F8	N/A	5.82E − 15	2.96E − 11	7.44E − 14	0.000181	**0.161972**	0.000183	**5.17E − 01**
F9	N/A	9.43E − 15	3.01E − 11	6.82E − 14	0.000181	0.000181	0.000181	3.74E − 03
F10	N/A	5.82E − 15	1.01E − 09	6.82E − 14	0.000181	0.000183	0.000183	7.69E − 04
F11	N/A	5.82E − 15	2.97E − 11	5.73E − 14	0.000181	0.000183	0.000183	9.32E − 04
F12	N/A	5.82E − 15	9.92E − 11	6.82E − 14	**0.472676**	0.000183	0.000182	**7.16E − 01**
F13	N/A	5.82E − 15	2.71E − 11	6.82E − 14	0.000181	0.000183	6.39E − 05	**6.72E − 01**

answer to the optimum at the first iteration is the key distinguishing feature of this algorithm.

SCA shows the minimum values for both *ave* and *SD*, proving that this algorithm reliably outperforms others in total. The *p* values in …, demonstrate that the superiority of the MCEO algorithm is statistically significant. MCEO offers very competitive results compared to TEO, SHO, WEO, MFO, FPA, SMS, and GWO on all uni-modal functions F1–F7.

The ultra-high potential of MCEO is obvious compared to TEO, SHO, WEO, MFO, FPA, and SMS, and GWO optimization algorithms. These are the strongest available algorithms to optimize the functions with many local minima. MCEO has been managed to find the optimal point in just a few primary loops, almost in all cases. This is due to the disordered nature of CE dispersion in the search area through the first loop. Search area is covered with a few numbers of Averages, thereafter; the Search Agents are normally distributed around each Average to search for optimum. This process leads to the discovery of a global optimum after a few loops (less than 10 loops), hence MCEO has an ultra-high speed for solving optimization problems. The statistical results of the algorithms on multi-modal test function are presented in Tables 5 and 6 It could be seen that MCEO highly outperforms other algorithms on all multi-modal functions, and this superiority is statistically significant. The answers of MCEO on these multi-modal test functions strongly prove that high exploration of the MCEO algorithm is beneficial for avoiding local solutions. Considering that the multi-modal functions have an exponential number of local solutions, their results show that the MCEO algorithm is able to explore the search space extensively and discover optimum regions of the search space. Furthermore, high local optima avoidance with this algorithm is another finding which can be inferred from these results.

4.2 Constrained engineering problems

The provided benchmark instances include objective functions and constraints of various types and nature (quadratic, cubic, polynomial, and nonlinear) with several numbers of design variables (continuous, mixed, discrete, and integer). The mathematical formulations of the test problems are also provided. The obtained optimization results have been compared with other well-known optimizers. Results were compared in terms of statistical results and the number of function evaluations (NFEs). In this paper, the computational cost which is considered as the best NFEs corresponding to the obtained best answer is calculated by the product of the number of Search Agents and the maximum number of iterations (i.e., $\text{NFEs}_{\text{MCEO}} = n_{\text{Averages}} \times n_{\text{Search Agents}} \times \text{It}_{\max}$). The proposed algorithm was coded in MATLAB programming software and the simulations and numerical solutions were run on an Intel(R) Core(TM) i7-4500U CPU @ 1.80 GHz with 4 GB Random Access Memory (RAM).

A selection of 20–50 number of Search Agents may be sufficient for complex optimization problems [59]. So, n_{Averages} *and* $n_{\text{Search Agents}}$ are considered as 5 and 10, respectively, in which the total number of Search Agents ($n_{\text{Averages}} \times n_{\text{Search Agents}}$) will be equal to 50. The maximum number of iterations $\text{It}_{\max}$ is dependent on the complexity of the optimization problems. The number of function evaluations NFEs ($n_{\text{Averages}} \times n_{\text{Search Agents}} \times \text{It}_{\max}$) is reported for all of the engineering problems in their relevant tables.

The penalty function method [39] is adopted in MCEO, to take into account constraints violation. This method is easy to implement and has a simple principle, especially for continuous constrained problems.

Typically, a constrained optimization problem is identified as follows:

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{Subject to :} \quad g_k(x) \leqslant 0, \quad k = 1, \ldots, m \end{aligned} \tag{5}$$

where f is the objective function and g_k is the kth inequality constraint. Integration of penalty functions into the objective function will transform the above constrained problem to an unconstrained one. The penalized objective function f_p is then written as follows:

$$f_p(x) = f(x) + \lambda \sum_{k=1}^{m} \delta_k \left[g_k(x)\right]^2 \tag{6}$$

In which $\lambda > 0$ (e.g. $\lambda = 10^{15}$) is a penalty factor and $\left\{ \begin{array}{l} \delta_k = 1 \text{ if constraint } g_k \text{ is violated} \\ \delta_k = 0 \text{ if constraint } g_k \text{ is satisfied} \end{array} \right\}$ [59].

It must be noted that the statistical answers are not presented anymore because it had been proved that MCEO can outperform other algorithm within a statistically significant manner. Additionally, the key objective of solving a real issue is to obtain the global optimum with minimal possible computational cost. Therefore, this section only presents the worst, mean and best obtained design, and also standard deviation (SD) of results, and number of function evaluations (NFEs).

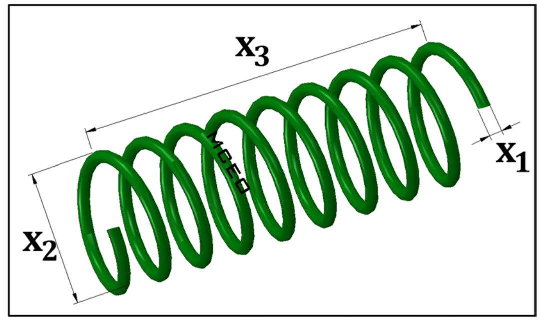

Fig. 10 Tension/compression spring design problem

4.2.1 Tension/compression spring design

The goal of this issue is to minimize the weight of a tension/compression spring, illustrated in Fig. 10. The minimization course of action is governed by several limitations, including shear stress, frequency wave, and minimum displacement. There are three variables for this problem: wire diameter ($x1$), mean coil diameter ($x2$), and the number of active coils ($x3$). The mathematical formulation of this problem is expressed as in Eq. 5. The current issue is discussed repeatedly by using precise mathematical methods and also meta-heuristic optimization algorithms. Wang and He solved this problem using PSO algorithm [60]. The other algorithms such as Evolutionary Strategy [61], GA [62], Search Harmony [63] and Differential Evolution [64] were employed to solve this problem.

Objective function: Minimize $f(\vec{x}) = (x_3 + 2)x_2 x_1^2$ (7)

Subject to
$$\begin{aligned} g_1(\vec{x}) &= 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0, \\ g_2(\vec{x}) &= \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} \leq 0, \\ g_3(\vec{x}) &= 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0, \\ g_4(\vec{x}) &= \frac{x_1 + x_2}{1.5} - 1 \leq 0. \end{aligned}$$

Parameters domain:
$0.05 \leq x_1 \leq 2.00,$
$0.25 \leq x_2 \leq 1.30,$
$2.00 \leq x_3 \leq 15.0$

The comparisons of results of several techniques and MCEO are provided in Table 7. Note that we use a similar

Table 7 Statistical comparison of results for tension/compression spring design problem

Method	Worst	Mean	Best	SD	NFEs
GA1 [61]	0.012822	0.012769	0.012704	3.94E−05	900,000
GA2 [65]	0.012973	0.012742	0.012681	5.90E−05	80,000
CAEP [66]	0.015116	0.013568	0.012721	8.42E −04	50,020
CPSO [60]	0.012924	0.01273	0.012674	5.20E−04	240,000
HPSO [67]	0.012719	0.012707	0.012665	1.58E −05	81,000
NM–PSO [68]	0.012633	0.012631	0.01263	8.47E−07	80,000
G-QPSO [69]	0.017759	0.013524	0.012665	0.001268	2000
QPSO [69]	0.018127	0.013854	0.012669	0.001341	2000
PSO [69]	0.071802	0.019555	0.012857	0.011662	2000
DE [70]	0.01279	0.012703	0.01267	2.7E −05	204,800
DELC [71]	0.012665	0.012665	0.012665	1.3E−07	20,000
DEDS [72]	0.012738	0.012669	0.012665	1.3E−05	24,000
HEAA [73]	0.012665	0.012665	0.012665	1.4E−09	24,000
PSO–DE [74]	0.012665	0.012665	0.012665	1.2E−08	24,950
SC [75]	0.016717	0.012922	0.012669	5.9E −04	25,167
(μ + λ)-ES [76]	NA	0.013165	0.012689	3.9E−04	30,000
ABC [77]	NA	0.012709	0.012665	1.28E−02	30,000
LCA [78]	0.01266667	0.0126654	0.01266523	3.88E−07	15,000
WCA [79]	0.012952	0.012746	0.012665	8.06E−05	11,750
IGMM [18]	0.0135125	0.0128657	0.01266523	2.56E−04	4000
APSO [80]	0.014937	0.013297	0.0127	6.85E−04	120,000
MCEO	**0.01350901**	**0.0127196**	**0.01266051**	**3.79E−05**	**2000**

Bold indicates the best answers obtained by MCEO

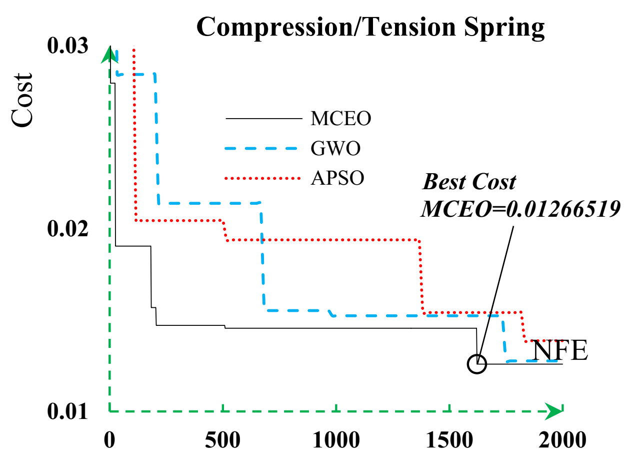

Fig. 11 Comparison of convergence curves of MCEO and literature algorithms obtained in tension/compression spring design problem

penalty function for MCEO to perform a fair comparison. Table 7 suggests that MCEO finds a design with the minimum weight for this problem.

Reduction in the cost of tension/compression spring, compared to the best answer obtained by available powerful algorithms, shows the high ability of MCEO in solving constrained engineering problems. Table 7 implies that the MCEO algorithm outperforms other algorithms in terms of NFEs and final cost. The cost of spring design is reported 0.01266051 by MCEO and has not been obtained up till date. It should be noted that NM-PSO has violated the first constraint and it cannot be an acceptable measure for comparison (violated constraints are underlined according to

Table 8). This shows the high performance of MCEO in calculating the global optimum for this issue. The maximum NFEs in Table 8 and Fig 11 also demonstrates that MCEO determines the global optimum for this problem with less NFEs than the others. Furthermore, it is observed that MCEO is able to identify design parameters with the optimal weight compared to others with regard to the constraints. However, this algorithm requires 2000 NFEs to outperform all of the other algorithms where the constraints are satisfied.

4.2.2 Welded beam design

The aim of this problem is to minimize the construction cost of a welded beam as displayed in Fig. 12. The constraints are: shear stress ($\boldsymbol{\tau}$); bending stress in the beam ($\boldsymbol{\sigma}$); buckling load of the bar (P_c); end deflection of the beam ($\boldsymbol{\delta}$) and side constraints. This problem has four variables such as thickness of weld ($x1$), length of attached part of bar ($x2$), the height of the bar ($x3$) and thickness of the bar ($x4$).

The particular mathematical description is as follows:

Objective function

$$\text{Minimize} \quad f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \tag{8}$$

$$\begin{aligned} \text{Subject to} \quad & g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0, \\ & g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0, \\ & g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0, \\ & g_4(\vec{x}) = x_1 - x_4 \le 0, \\ & g_5(\vec{x}) = P - P_c(\vec{x}) \le 0, \\ & g_6(\vec{x}) = 0.125 - x_1 \le 0, \\ & g_7(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5 \le 0. \end{aligned}$$

Parameters domain

$$\begin{aligned} & 0.10 \le x_1 \le 2.00, \\ & 0.10 \le x_2 \le 10.0, \\ & 0.10 \le x_3 \le 10.0, \\ & 0.10 \le x_4 \le 2.00. \end{aligned}$$

Dependent parameters

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \quad \tau' = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J},$$

$$M = P\left(L + \frac{x_2}{2}\right), \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \quad \sigma(\vec{x}) = \frac{6PL}{x_3^2x_4}, \quad \delta(\vec{x}) = \frac{6PL^3}{Ex_3^2x_4},$$

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

Fixed parameters

$$P = 6000\text{ lb}, \quad L = 14\text{ in}, \quad \tau_{\max} = 13600\text{ psi}, \quad \sigma_{\max} = 30000\text{ psi},$$
$$\delta_{\max} = 0.25\text{ in}, \quad E = 30 \times 10^6\text{ psi}, \quad G = 12 \times 10^6\text{ psi}.$$

Table 8 Comparison of the best solutions for tension/compression spring design problem

DV	DEDS [72] *	HEAA [73]	NM–PSO [68] *	DELC [71] *	WCA [79]	LCA [78]	IGMM [18]	APSO [80]	GWO [32]	MCEO
x1	0.051689	0.051689	0.05162	0.051689	0.05168	0.051689	0.051718	0.052588	0.05169	**0.05168**
x2	0.356717	0.356729	0.355498	0.356717	0.356522	0.356718	0.357415	0.378343	0.356737	**0.356728**
x3	11.288965	11.288293	11.333272	11.288965	11.30041	11.28896	11.2482	10.13886	11.28885	**11.288293**
g1(x)	*1.56E−06*	−3.98E−5	*1.01E−03*	*1.56E−06*	−6.78E−5	−6.40E−06	−1.75E−6	−1.51E−04	−7.9E−05	**−7.28E−04**
g2(x)	−0.134273	−0.134246	−0.13367	−0.1342	−0.134278	−0.134371	−0.13409	−0.1298652	−0.13427	**−0.1338036**
g3(x)	−4.053800	−4.053808	−4.0618597	−4.0538	−4.053326	−4.05377	−4.055159	−4.08918	−4.053383	**−4.052910**
g4(x)	−0.727729	−0.72772	−0.728588	−0.72772	−0.727865	−0.72772	−0.727244	−0.71271	−0.7277153	**−0.7277280**
f(x)	0.0126651	0.01266496	0.01263019	0.0126651	0.0126647	0.0126652	0.0126652	0.012700	0.012666	**0.01266051**

Bold indicates the best answers obtained by MCEO

Italic values are violated constraints, and are placed in infeasible domain

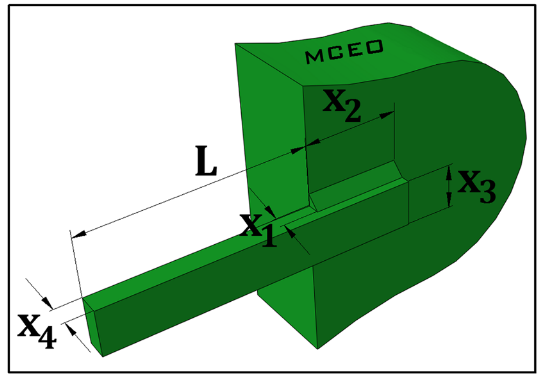

Fig. 12 Welded beam design problem

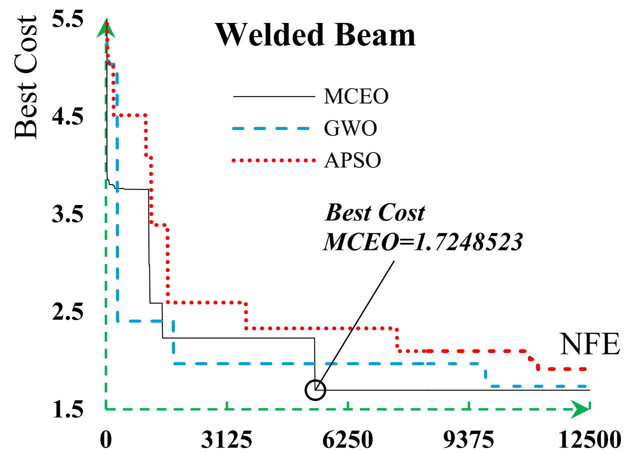

Fig. 13 Comparison of convergence curves of MCEO and literature algorithms obtained in welded beam design problem

Coello and Deb [59] used the GA to solve this problem. Lee and Geem [63] applied HS algorithm to this problem. Ragsdell and Philips [81] exerted Richardson random method, Simplex method, Davidon–Fletcher–Powel method and Griffith and Stewart linear approach to mathematically solve this problem.

MCEO provides an acceptable level of answers in welded beam design optimization problem and the final result is similar to IGMM algorithm's answer (Tables 9, 10). The mentioned algorithm had the best responses up to date. It should be observed that some constraint are violated in the CAEP, HGA, and NM–PSO algorithms. These algorithms are marked with a superscripted star. MCEO stably finds the best known answer achieved by LCA as well as clearly dominates APSO. Additionally, MCEO provided improved answers using a lesser number of function evaluations (NFEs) in comparison to all other employed algorithms, with regard to statistical responses. Even so, the answer of MCEO is obtained with a SD value greater than that given

Table 9 Statistical comparison of results for the welded beam design problem

Method	Worst	Mean	Best	SD	NFEs
GA1 [61]	1.785835	1.771973	1.748309	0.0112	900,000
GA2 [65]	1.993408	1.792654	1.728224	0.0747	80,000
CAEP [66]	3.179709	1.971809	1.724852	0.443	50,020
CPSO [60]	1.782143	1.748831	1.7314849	0.0129	240,000
HPSO [67]	1.814295	1.74904	1.724852	0.0401	81,000
PSO–DE [74]	1.724852	1.724852	1.724852	6.7E−16	66,600
NM–PSO [68]	1.733393	1.726373	1.724720	0.0035	80,000
SC [75]	6.399678	3.002588	2.385434	0.96	33,095
DE [70]	1.824105	1.768158	1.733461	0.0221	204,800
WCA [79]	1.744697	1.726427	1.724856	0.00429	46,450
LCA [78]	1.7248523	1.7248523	1.7248523	7.11E−15	15,000
IGMM [18]	1.74769	1.732152	1.724855	7.14E−03	8000
APSO [80]	1.993999	1.877851	1.736193	0.076118	50,000
MCEO	**1.7248732**	**1.7248621**	**1.7248523**	**1.02E−05**	**12,500**

Bold indicates the best answers obtained by MCEO

by LCA. It was noticed that MCEO provided an excellent converging behavior, according to Fig 13, because of the effectiveness of the searching techniques and also the perfect balance in exploration and exploitation throughout the optimization process.

4.2.3 Pressure vessel design

The intention of this issue is to minimize the total cost, which consists of material, building, and welding of a cylindrical vessel as shown in Fig. 14. Both ends of the vessel are capped, and the head has a hemi-spherical shape. There are four variables in this problem: thickness of the shell ($x1$); thickness of the head ($x2$); inner radius ($x3$) and length of the cylindrical section without considering the head ($x4$).

This problem is subject to four constraints. These constraints and the problem are formulated as follow:

This issue has been discussed several times by different researchers. Meta-heuristic algorithms (PSO [60], GA [65], ES [61], DE [64] and ACO [82]) were used to solving this problem. Mathematical methods such as Augmented Lagrangian-Multiplier [83] and branch-and-bound method [84] were also employed to solve this problem.

Results of pressure vessels design by the various algorithms are demonstrated in Tables 11 and 12. MCEO has significantly reduced the fabrication costs of the pressure vessel when compared with the values obtained from the other algorithms. Power and accuracy of the presented algorithm are clearly demonstrated in solving the constrained engineering problems in mixed integer-continuous spaces. It should be mentioned that results of NM-PSO, WCA and GWO are not placed in the feasible domain, because the design variables x1 and x2 have no discrete values (a multiplier of 0.0625) according to Table 12. All the violated constraints and results in infeasible domain are underlined. Even so, IGMM indicates the lowest SD,

Objective function

$$\text{Minimize} \quad f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{9}$$

$$\begin{aligned} \text{Subject to} \quad & g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0, \\ & g_2(\vec{x}) = -x_2 + 0.00954x_3 \le 0, \\ & g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0, \\ & g_4(\vec{x}) = x_4 - 240 \le 0. \end{aligned}$$

Parameters domain

$$\begin{aligned} & 1 * 0.0625 \le x_1, x_2 \le 99 * 0.0625, \\ & 10 \le x_3, x_4 \le 200. \end{aligned}$$

whereas MCEO is converged to the optimal answer with the lower NFEs compared to IGMM. Final solutions show that MCEO surpasses all other algorithms in terms of the NFEs and SD. In conclusion, it is obvious that MCEO is the most effective algorithm to optimize the design variables of a pressure vessel. Figure 15 demonstrates the function values according to NFEs for the pressure vessel issue. MCEO indicates an excellent converging tendencies, due to the effectiveness of search model that is caused by the five mentioned search parameters in Sect. 3.

4.2.4 SHAFAROUD concrete gravity dam design

SHAFAROUD irrigation project is located in the North-Western part of Iran by the Caspian Sea. At present, traditional water courses are irrigating about 7150 ha of paddy field. Although, there is water shortages during low river runoff and peak growing season, it is decided to extend the area under paddy to about 12,300 ha in net. However, this is impossible without taking full advantage from river runoff, and preventing the flow of large volume of water to the Caspian Sea annually. Therefore, it is anticipated to construct a reservoir dam on one of the four rivers existing within the area, and three diversion dams on three others. The objective can be accomplished by taking full advantage from river runoff through diversion dams and provision of irrigation water deficiency from the reservoir dam. SHAFAROUD dam is used as a practical example. This problem includes nine variables, $x1$ to $x9$ shown in Fig. 16. The objective function is a minimization of concrete volume used to build up the SHAFAROUD concrete gravity dam. The constraints of this problem included: sliding and overturning safety factors, tension, compression stresses on the downstream and upstream of the dam.

Objective function

$$\text{Minimize} \quad f(\vec{x}) = x_1 H + x_3(x_4 + x_6) + x_5 x_6 + \tfrac{1}{2}\left[x_2 x_3 + x_4 x_5 + x_6 x_7 + x_8 x_9\right] \tag{10}$$

Constrains

$$\begin{aligned}
\text{Subject to} \quad & g_1(\vec{x}) = 4 - \mathrm{SFS} \leqslant 0\,,\\
& g_2(\vec{x}) = 1.5 - \mathrm{SFO} \leqslant 0\,,\\
& g_3(\vec{x}) = -\sigma_U \leqslant 0\,,\\
& g_4(\vec{x}) = -\sigma_D \leqslant 0\,,\\
& g_5(\vec{x}) = \sigma_U - \sigma_{\max} \leqslant 0\,,\\
& g_6(\vec{x}) = \sigma_D - \sigma_{\max} \leqslant 0\,.
\end{aligned}$$

Parameters domain

$$\begin{aligned}
& 0.10 \leqslant x_1 \leqslant 20.0\,,\\
& 1.00 \leqslant x_2 \leqslant 50.0\,,\\
& 1.00 \leqslant x_3 \leqslant 20.0\,,\\
& 5.00 \leqslant x_4, x_5, x_6, x_7 \leqslant 50.0\,,\\
& 50.0 \leqslant x_8, x_9 \leqslant 150.0.
\end{aligned}$$

Dependent parameters

$$\mathrm{SFS} = \frac{f' \sum F_{\mathrm{V}} + \sigma(b)}{\sum F_{\mathrm{H}}}, \quad \mathrm{SFO} = \frac{\sum M_{\mathrm{R}}}{\sum M_{\mathrm{O}}}, \quad \sigma_{\mathrm{U}} = \frac{\sum F_{\mathrm{V}}}{b} - \frac{6\sum M_{\mathrm{O}}}{b^2},$$

$$\sigma_{\mathrm{D}} = \frac{\sum F_{\mathrm{V}}}{b} + \frac{6\sum M_{\mathrm{O}}}{b^2},$$

$$\sum F_{\mathrm{H}} = Fh_{\mathrm{H}} + P_{\mathrm{SH}}, \quad \sum F_{\mathrm{V}} = P_{\mathrm{SV}} + W + Fh_{\mathrm{V}} - F_{\mathrm{Uplift}} - F_{\mathrm{E}},$$

$$b = x_1 + x_3 + x_5 + x_7 + x_9, \quad Fh_{\mathrm{H}} = \frac{1}{2}\gamma_w h^2, \quad Fh_{\mathrm{V}} = \gamma_w V_w, \quad F_{\mathrm{Uplift}} = \frac{1}{2}\gamma_w h,$$

$$F_{\mathrm{E}} = \alpha W, \quad P_{\mathrm{SH}} = \frac{1}{2}\gamma_s h_s^2 \frac{1 - \sin\varphi}{1 + \sin\varphi}, \quad P_{\mathrm{SV}} = \frac{1}{2}\gamma_s h_s^2 \tan\theta, \quad \theta = \frac{x_6}{x_7}\,.$$

Fixed parameters

$$\gamma_w = 9.81\frac{KN}{m^3}, \quad \gamma_c = 23\frac{KN}{m^3}, \quad \gamma_s = 15\frac{KN}{m^3}, \quad \sigma_{\max} = 25000\,\frac{KN}{m^2},$$

$$\sigma = 3500\,\frac{KN}{m^2},$$

$$H = 150\,\mathrm{m}, \quad h = 145\,\mathrm{m}, \quad h_s = 7\,\mathrm{m}, \quad \varphi = 50^\circ, \quad \alpha = 0.05 \quad f' = 0.70.$$

Table 10 Comparison of the best solutions for the welded beam design problem

DV	GA2 [65]	CPSO [60]	CAEP [66]*	HGA [59]*	NM-PSO [68]*	WCA [79]	IGMM [18]	APSO [80]	GWO [32]	MCEO
*x*1	0.205986	0.202369	0.2057	0.2057	0.20583	0.205728	0.205729	0.202701	0.205676	**0.20572964**
*x*2	3.471328	3.544214	3.4705	3.4705	3.468338	3.470522	3.470496	3.574272	3.478377	**3.47048866**
*x*3	9.020224	9.04821	9.0366	9.0366	9.036624	9.03662	9.036625	9.040209	9.03681	**9.03662391**
*x*4	0.20648	0.205723	0.2057	0.2057	0.20573	0.205729	0.205730	0.2059215	0.205778	**0.20572964**
g1(*x*)	−774.540	−811.4181	−769.3402	−769.3402	−770.3698	−0.034128	−771.187	−117.467062	−794.279	**−771.2021**
g2(*x*)	−0.23174	−75.81407	*4.481548*	*4.481548*	−0.053122	−3.49E−05	−0.05976	−51.712981	−8.2855887	**−2.88E−05**
g3(*x*)	−0.22827	−0.228392	−0.228307	−0.228307	−0.228310	−1.19e−6	−0.228310	−0.003221	−0.2283169	**−0.2283104**
g4(*x*)	−4.93E−4	−0.003354	0	0	*1.00E−04*	−3.43298	−1e−6	−3.421741	−1.02E−04	**0**
g5(*x*)	−58.646	−4.472858	*2.603347*	*2.603347*	−0.031555	−0.080728	−0.0319920	−0.077701	−4.313487	**−1.85E−05**
g6(*x*)	−0.08098	−0.077369	−0.08070	−0.08070	−0.080830	−0.23554	−0.08072	−0.235571	−0.0806	**−0.0807296**
g7(*x*)	−3.4300	−3.424572	−3.43321	−3.43321	−3.43316	−0.013503	−3.4329802	−18.367012	−3.389578	**−3.432983**
f(*x*)	1.728224	1.7314849	1.724577	1.724577	1.724720	1.724856	1.7248552	1.736193	1.72624	**1.7248523**

Bold indicates the best answers obtained by MCEO

Italic values are violated constraints, and are placed in infeasible domain

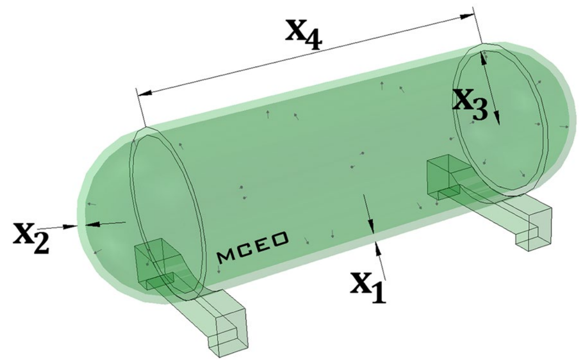

Fig. 14 Pressure vessel design problem

The unusual loading on the dam has been considered under this section. The applied loads are dead load (weight of the dam); hydrostatic load in the full reservoir state; earthquake and uplift [85]. Important researches in the field of optimization of concrete dams can be mentioned as: Kaveh, and Zakian [86], Akbari et al. [87], Mahani et al. [88], Cai et al. [89], Ahmed et al. [90], Gandomi et al. [91], and Gholizadeh and Seyedpoor [92]. The total number of Search Agents is adjusted to 50 ($n_{\text{Mean}} = 5$ and $n_{\text{Search Agent}} = 10$) to solve this optimization problem, and allowed to calculate the optimal shape for the gravity dam per 1000 iterations. The algorithm is run 30 times and the best results are presented in Table 13

The results in Table 13 show a proper amount of constraints, near the bound of constrain, compared to PSO and GA. It caused a significant reduction in the cost of dam construction, and it is less than the existing dam plan. Figure 17 obviously implies that the MCEO algorithm helps the initial Search Agents to minimize the volume of concrete of the dam. The improvement is absolutely considerable, in which concrete volume was reduced from 17,000 to 7449 m^3, in a section with a width of 1 m. These results remarkably demonstrate that MCEO can solve real issues with unknown, challenging, and constrained search spaces. Sections obtained by each algorithm in which the reduction in the area of dam section is clearly visible are shown in Fig. 18

5 Conclusion

The proposed optimization method has modeled the multi-level CE in hyper spaces. 13 test functions were employed in order to benchmark the performance of the proposed algorithm in terms of exploration, exploitation and local optima avoidance. The results showed that MCEO could provide highly competitive results compared to well-known heuristics such as TEO, SHO, WEO, MFO, FPA, SMS, and

Table 11 Statistical comparison of results for the pressure vessel design problem

Method	Worst	Mean	Best	SD	NFEs
GA1 [61]	6308.497	6293.8432	6288.7445	7.4133	900,000
GA2 [65]	6469.322	6177.2533	6059.9463	130.9297	80,000
CPSO [60]	6363.8041	6147.1332	6061.0777	86.45	240,000
HPSO [67]	6288.677	6099.9323	6059.7143	86.2	81,000
NM–PSO [68]	5960.0557	5946.7901	5930.3137	9.161	80,000
G-QPSO [69]	7544.4925	6440.3786	6059.7208	448.4711	8000
QPSO [69]	8017.2816	6440.3786	6059.7209	479.2671	8000
PSO [69]	14076.324	8756.6803	6693.7212	1492.567	8000
CDE [64]	6371.0455	6085.2303	6059.734	43.013	204,800
WCA [79]	6590.2129	6198.6172	5885.3327	213.049	27,500
LCA [78]	6090.6114	6070.5884	6059.8553	11.37534	24,000
IGMM [18]	6061.2868	6060.1598	6059.7143	0.5421	8000
APSO [80]	7544.49272	6470.71568	6059.7242	326.9688	200,000
MCEO	**6060.3096**	**6060.0315**	**6059.7143**	**1.2532**	**7500**

Bold indicates the best answers obtained by MCEO

GWO. The results on the uni-modal functions revealed the superior exploitation of the MCEO algorithm and also the exploration ability of MCEO was confirmed by the results on multi-modal functions. Furthermore, the results of the engineering design problems demonstrate that the MCEO algorithm has a high performance in unknown and challenging search spaces. The MCEO algorithm was finally applied to a real problem in concrete gravity dam engineering. The outcomes of this issue showed a considerable improvement of dam section in comparison to current approaches, which exhibits the applicability of the suggested algorithm in dealing with the real challenges. It can be mentioned that the results on semi-real and real issues also demonstrated that MCEO can show powerful performance not only on unconstrained problems but also on constrained problems.

This is caused by three major reasons. The first is that MCEO is a parallel cross entropy algorithm; therefore it intrinsically gains advantage from high exploration and local optima avoidance. This allows MCEO to avoid the large number of local optima and search all the important feasible space (with high probability for optimum existence) extensively. The second is that the MCEO algorithm efficiently changes from exploration to exploitation using the flexible system of σ function (proposed in Eq. 4). The linear sigma reduction factor function furthermore leads to local optima avoidance at the primary iterations process of algorithm and fast convergence into the most probable optimum region of the feasible space in the final iterations. Finally, MCEO sorts the solutions based on their costs to replace the positions of the Search Agents toward the global best as the destination point.

Table 12 Comparison of the best solutions for the pressure vessel design problem

DV	CDE [64]	GA1 [61]	CPSO [60]	HPSO [67] *	NM–PSO [68] *	G-QPSO [69]	WCA [79] *	IGMM [18]	APSO [80]	GWO [32] *	MCEO
$x1$	0.8125	0.8125	0.8125	0.8125	*0.8036*	0.8125	*0.7781*	0.8125	0.8125	0.8125	**0.8125**
$x2$	0.4375	0.4375	0.4375	0.4375	*0.3972*	0.4375	*0.3846*	0.4375	0.4375	*0.4345*	**0.4375**
$x3$	42.0984	42.0974	42.0913	42.0984	41.6392	42.0984	40.3196	42.098445	42.0984	42.089181	**42.0984455**
$x4$	176.6376	176.654	176.7465	176.6366	182.412	176.6372	200	176.63659	176.6374	176.758731	**176.636596**
g1(x)	− 6.6E − 7	−2.01E − 3	− 1.3E − 6	− 8.80E − 7	*0.0000365*	−8.79E − 07	*6.828e − 05*	−3.17E − 10	−8.799E − 07	−1.14E − 08	**−1.13E − 10**
g2(x)	− 3.58E − 2	−3.58E − 2	− 0.000359	− 3.76E − 2	*0.0000379*	−0.0358	*4.898e − 05*	−0.03756	−0.0359	−0.03756477	**−0.037564**
g3(x)	− 3.70512	− 24.7593	− 118.7687	*3.1226*	− 1.5914	−0.2179	*1.33120*	−0.00012	−1.3315386	*0.073639*	**−4.73E − 04**
g4(x)	− 63.3623	− 63.346	− 63.2535	− 63.3634	− 57.5879	−63.3628	−40	−63.3634	−63.362	−63.3634	**−63.3634**
$f(x)$	6059.734	6059.9463	6061.0777	6059.7143	5930.3137	6059.7208	5885.3327	6059.7143	6059.72418	6059.714	**6059.7143**

Bold indicates the best answers obtained by MCEO

Italic values are violated constraints, and are placed in infeasible domain

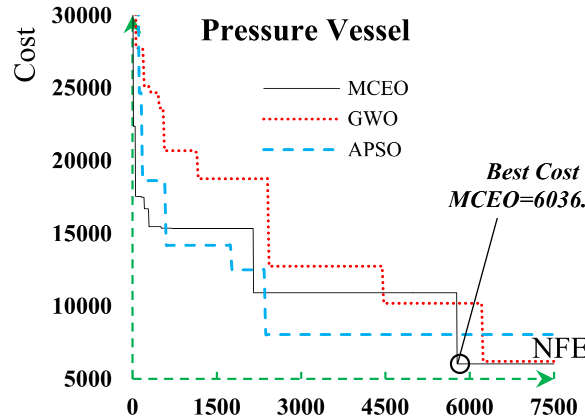

Fig. 15 Comparison of convergence curves of MCEO and literature algorithms obtained in pressure vessel design problem

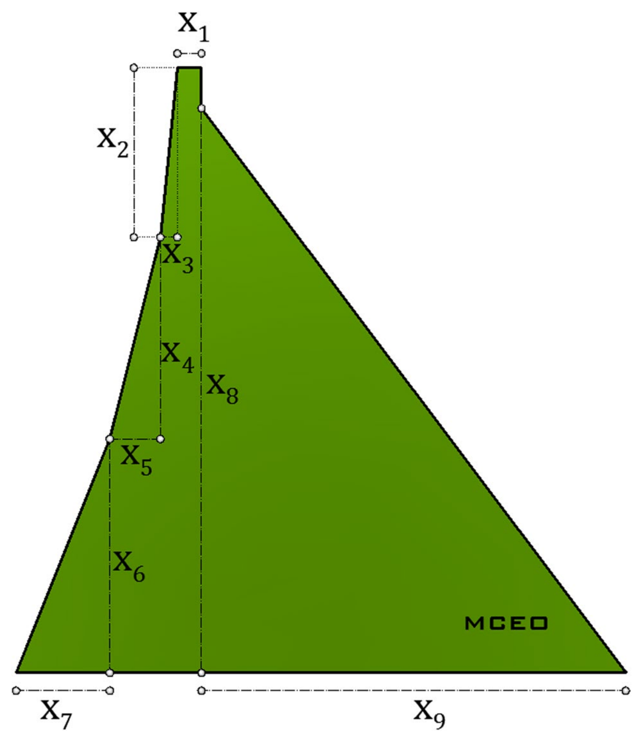

Fig. 16 SHAFAROUD concrete gravity dam

Table 13 Comparison of the best solutions for SHAFAROUD concrete gravity dam design problem

	SHAFAROUD existing dam	PSO	GA	MCEO
NFEs	–	50,000	50,000	**50,000**
x_1	4	6.0323	5.1965	**5.1678**
x_2	42	37.805	31.8972	**25.1978**
x_3	4.2	2.0582	2.3238	**4.9556**
x_4	50	73.2012	75.9014	**55.7811**
x_5	12.5	11.5573	11.0781	**16.0874**
x_6	58	39.02	42.2014	**21.0009**
x_7	23.2	30.6433	28.9231	**29.9036**
x_8	140	125.6454	132.025	**120.169**
x_9	105	98.3741	101.713	**85.382**
$\sum M_R$	23,979,329.67	20,778,650.8	21,584,896.2	**18,520,793**
$\sum M_O$	5,056,164.707	5,055,074.16	5,055,375.01	**5,051,705.95**
$\sum F_H$	103,176.3093	103,176.309	103,176.309	**103,176.309**
$\sum F_V$	262,408.494	234,946.624	242,356.882	**215,538.616**
Fh_H	103,127.625	103,127.625	103,127.625	**103,127.625**
Fh_V	32,730.084	42,455.6199	39,249.451	**53,236.0323**
P_{SH}	48.6843168	48.6843168	48.6843168	**48.6843168**
P_{SV}	918.75	468.382353	536.215499	**258.090638**
σ_U	406.6809	207.527923	262.034703	**9.3743472**
σ_D	3071.231545	2954.60698	2985.96602	**3037.18705**
SFS	6.899218925	6.63488199	6.70668075	**6.26223367**
SFO	4.742592669	4.11045419	4.26969238	**3.66624526**
$g_1(\vec{x})$	− 2.8992	− 2.6349	− 2.7067	**− 2.2622**
$g_2(\vec{x})$	− 3.2426	− 2.6105	− 2.7697	**− 2.1662**
$g_3(\vec{x})$	− 406.6809	− 207.5279	− 262.0347	**− 9.3743**
$g_4(\vec{x})$	− 3.07E+03	− 2.95E+03	− 2.99E+03	**− 3.04E+03**
$g_5(\vec{x})$	− 2.46E+04	− 2.48E+04	− 2.47E+04	**− 2.50E+04**
$g_6(\vec{x})$	− 2.19E+04	− 2.20E+04	− 2.20E+04	**− 2.20E+04**
$f(\vec{x})$	10,502.1	8820.771	9303.5442	**7448.7743**

Bold indicates the best answers obtained by MCEO

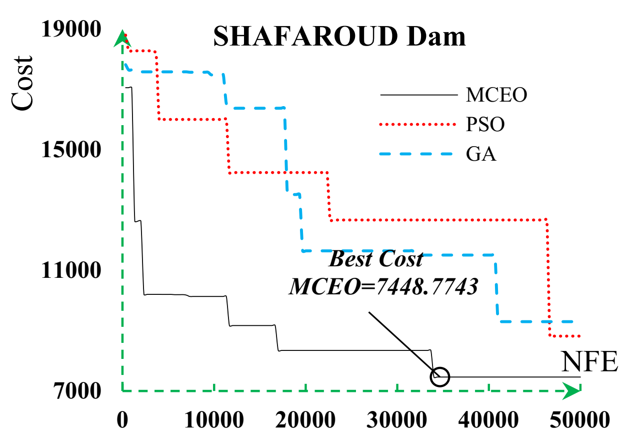

Fig. 17 Comparison of convergence curves of MCEO and literature algorithms obtained in dam design problem

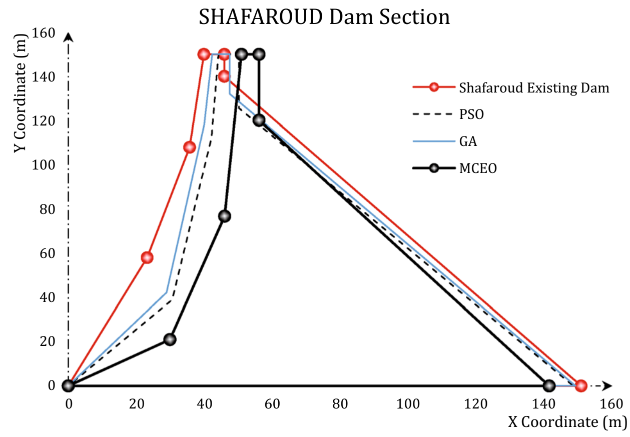

Fig. 18 Comparison of dam section obtained from different optimization engines

References

1. Kaveh A (2014) Advances in Metaheuristic algorithms for optimal design of structures. Springer International Publishing, Basel
2. Goldberg DE, Holland JH (1968) Genetic algorithms and machine learning. Mach Learn 3(2):95–99
3. Storn R, Price K (1997) Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
4. Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection, vol 1. MIT press, Cambridge
5. Rechenberg I (1994) Evolution strategy. Comput Intell 1:147–159
6. Fogel LJ, Owens AJ, Walsh MJ (1966) Artificial intelligence through simulated evolution. Wiley, New York
7. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
8. Yang XS (2012) Flower pollination algorithm for global optimization. Confrence paper. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol 7445. LNCS, pp 240–249. https://doi.org/10.1007/978-3-642-32894-7_27
9. Cuevas E, Echavarría A, Ramírez-Ortegón MA (2014) An optimization algorithm inspired by the States of Matter that improves the balance between exploration and exploitation. Appl Intell 40(2):256–272
10. Banichuk NV, Neittaanmäki PJ (2010) Structural optimization with uncertainties. Solid Mechanics its Applications, vol 162, Springer, Netherlands
11. Webster B, Bernhard PJ (2003) A local search optimization algorithm based on natural principles of gravitation. Technical report. Florida Institute of Technology. #CS-2003-10
12. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3–4):267–289
13. Formato RA (2007) Central force optimization: A new metaheuristic with applications in applied electromagnetics. Progr Electromagn Res PIER 77:425–491
14. Lam AYS, Li VOK (2010) Chemical-reaction-inspired metaheuristic for optimization. IEEE Trans Evol Comput 14(3):381–399

15. Hatamlou A (2013) Black hole: A new heuristic optimization approach for data clustering. Inf Sci 222:175–184
16. Du H, Wu X, Zhuang J (2006) Small-world optimization algorithm for function optimization. Adv Nat Comput Pt 2 4222:264–273
17. Moghaddam FF, Moghaddam RF, Cheriet M (2012) Curved space optimization: a random search based on general relativity theory. arXiv:1208.2214
18. Varaee H, Reza M (2016) Engineering optimization based on ideal gas molecular movement algorithm. Eng Comput 33:71–93
19. Kaveh A, Bakhshpoori T (2016) Water Evaporation Optimization: A novel physically inspired optimization algorithm. Comput Struct 167:69–85
20. Kaveh A, Dadras A (2017) A novel meta-heuristic optimization algorithm: Thermal exchange optimization. Adv Eng Softw 110:69–84
21. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Sixth International Symposium on Micro Machine and Human Science on IEEE. Japan. https://doi.org/10.1109/MHS.1995.494215
22. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization." IEEE Comput Intell Mag 1(4):28–39
23. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization, Technical report, Erciyes university
24. Nesahat M, Sepidnam G, Sargolzaei M, Toosi A N (2014) Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications. Artif Intell Rev. 42(4):965–997
25. Roth M (2006) Termite: A swarm intelligent routing algorithm for mobile wireless ad-hoc networks. Stigmergic Optimization. Springer, Berlin, Heidelberg, pp 155–184
26. Mucherino A, Seref O (2007) Monkey search: A novel metaheuristic search for global optimization. AIP Conf Proc. 953(1):162–173. http://doi.org/10.1063/1.2817338
27. Gandomi AH, Yang X (2013) Cuckoo search algorithm : a metaheuristic approach to solve structural optimization problems. Eng Comput 29:17–35
28. Shiqin Y, Jianjun J, Guangxing Y (2009) A dolphin partner optimization. In: Intelligent Systems. GCIS'09. WRI Global Congress on, vol 1, IEEE, Washington. pp 124–128
29. Yang X-S (2010) Firefly Algorithm, Stochastic Test Functions and Design Optimisation. Int J Bio-Inspired Comput. 2(2):78–84
30. Kaveh A, Khayatazad M (2012) A new meta-heuristic method: Ray Optimization. Comput Struct 112–113:283–294
31. Kaveh A, Farhoudi N (2013) A new optimization method: Dolphin echolocation. Adv Eng Softw 59:53–70
32. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey Wolf Optimizer. Adv Eng Softw 69:46–61
33. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 89:228–249
34. Kaveh A, Ghazaan MI (2017) A new meta-heuristic algorithm : vibrating particles system. Sci Iran 24(2):1–32
35. Fausto F, Cuevas E, Valdivia A, González A (2017) BioSystems A global optimization algorithm inspired in the behavior of selfish herds. BioSystems 160:39–55
36. Kaveh A, Zolghadr A (2017) Cyclical parthenogenesis algorithm for guided modal strain energy based structural damage detection. Appl Soft Comput J 57:250–264
37. Rubinsten RY, Kroese DP (2004) The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning. Springer, New York
38. Belmudes F, Ernst D, Wehenkel L (2008) Cross-entropy based rare-event simulation for the identification of dangerous events in power system. in: Probabilistic Methods Applied to Power Systems, 2008. PMAPS'08. Proceedings of the 10th International Conference on IEEE, Puerto Rico. pp 1–7. Print ISBN: 978-1-9343-2521-6
39. Lindell IV, Sihvola AH, Hänninen I (2006) Realization of perfectly anisotropic impedance boundary. Eur Space Agency. 626:1131–1136. https://doi.org/10.1109/EUCAP.2006.4584717
40. Kobilarov M (2012) Cross-entropy randomized motion planning. In: Robotics: Science and Systems VII. pp 153–160. https://doi.org/10.15607/RSS.2011.VII.022
41. Kothari RP, Kroese DP (2009) Optimal generation expansion planning via the cross-entropy metho. In: Simulation Conference (WSC) on IEEE, Proceeding of the 2009 winter. pp 1482–1491. https://doi.org/10.1109/WSC.2009.5429296
42. Sani A (2009) Stochastic Modelling and Intervention of the Spread of HIV/AIDS. Ph.D. thesis, The University of Queensland, School of Physical Science, Brisbane
43. Kroese DP, Porotsky S, Rubinstein RY (2006) The cross-entropy method for continuous multi-extremal optimization. Methodol Comput Appl Prob 8(3):383–407
44. Szabó Z, Póczos B, Lőrincz A (2006) Cross-Entropy Optimization for Independent Process Analysis. IIn: International Conference on Independent Component Analysis and Signal Separation ICA 2006: Independent Component Analysis and Blind Signal Separation. vol 3889, Springer, Berlin. pp 909–916
45. Busnoniu L, Babuska R, De Schutter B, Ernst D (2010) Reinforcement learning and dynamic programming using function approximators. Taylor & Francis Group, New York
46. Kroese DP, Rubinstein RY, Taimre T (2007) Application of the cross-entropy method to clustering and vector quantization. J Global Optim 37(1):137–157
47. Liu Z, Doucet A, Singh SS (2004) The cross-entropy method for blind multiuser detection. In: Information Theory. ISIT Proceedings. International Symposium on IEEE, p. 510. https://doi.org/10.1109/ISIT.2004.1365547
48. Pihur V, Datta S, Datta S (2007) Weighted rank aggregation of cluster validation measures: a monte carlo cross-entropy approach. Bioinformatics 23(13):1607–1615
49. Alon G, Kroese DP, Raviv T, Rubinstein RY (2005) Application of the cross-entropy method to the buffer allocation problem in a simulation-based environment. Ann Oper Res 134(1):137–151
50. Cohen I, Golany B, Shtub A (2007) Resource allocation in stochastic, finite-capacity, multi-project systems through the cross entropy methodology. J Sched 10(3):181–193
51. Kroese DP, Hui KP, Nariai S (2007) Network reliability optimization via the cross-entropy method." IEEE Trans Reliab 56(2):275–287
52. Fu C, Wang H, Zhu L (2005) Solving the vehicle routing problem with stochastic demands and customers. In: Parallel and distributed computing, applications and technologies, PDCAT Proceedings, vol 2005, pp 736–738
53. Ernst D, Glavic M, Stan G-B, Mannor S, Wehenkel L (2007) The cross-entropy method for power system combinatorial optimization problems. In: Power Tech, IEEE Lausanne, pp 1290–1295. https://doi.org/10.1109/PCT.2007.4538502
54. Lőrincz A, Palotai Z, Szirtes G (2008) Spike-based cross-entropy method for reconstruction. Neurocomputing 71(16):3635–3639
55. Yang X, Press L (2010) Nature-Inspired Metaheuristic Algorithms Second Edition. University of Cambridge, Cambridge
56. Tung YK, Yen BC, Melching CS (2013) Reliability assessment and risk analysis. McGraw-Hill, New York
57. Haldar A, Mahdevan S (2010) Probability, reliability, and statistical methods in engineering design. Wiley, New York
58. Dai H, Wang W (2009) Application of low discrepancy sampling method in structural reliability analysis. Struct Saf 31:55–64
59. Rao SS (2009) Engineering optimization: theory and practice, 4 edn. Wiley, New York
60. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99

61. Mezura-Montes E, Coello CAC (2008) An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int J Gen Syst 37(4):443–473
62. Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41(2):113–127
63. Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. Comput Methods Appl Mech Eng 194(36–38):3902–3933
64. Huang F, Wang L, He Q (2007) An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 186(1):340–356
65. Coello CAC, Montes EM (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Inform 16(3):193–203
66. Coello Coello CA, Becerra RL (2004) Efficient evolutionary optimization through the use of a cultural algorithm. Eng Optim 36(2):219–236
67. He Q, Wang L (2007) A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. Appl Math Comput 186(2):1407–1422
68. Zahara E, Kao Y-T (2009) Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems. Expert Syst Appl 36(2):3880–3886
69. L dos Santos Coelho (2010) Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. Expert Syst Appl 37(2):1676–1683
70. Lampinen J (2002) A constraint handling approach for the differential evolution algorithm. In: Evolutionary Computation, CEC'02. Proceedings of the 2002 Congress, vol 2, pp 1468–1473
71. Wang L, Li L (2010) An effective differential evolution with level comparison for constrained engineering design. Struct Multidiscip Optim 41(6):947–963
72. Zhang M, Luo W, Wang X (2008) Differential evolution with dynamic stochastic selection for constrained optimization. Inf Sci 178(15):3043–3074
73. Wang Y, Cai Z, Zhou Y, Fan Z (2009) Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique. Struct Multidiscip Optim 37(4):395–413
74. Liu H, Cai Z, Wang Y (2010) Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. Appl Soft Comput 10(2):629–640
75. Ray T, Liew KM (2003) Society and civilization: An optimization algorithm based on the simulation of social behavior. IEEE Trans Evol Comput 7(4):386–396
76. Mezura-Montes E, Coello CAC (2005) Useful infeasible solutions in engineering optimization with evolutionary algorithms. In: Mexican International Conference on Artificial Intelligence. vol 3789, Springer, Berlin, pp 652–662
77. Karaboga D, Basturk B (2007) Artificial bee colony (ABC) optimization algorithm for solving constrained optimization. In: International Fuzzy Systems Association World Congress. IFSA 2007: Foundations of fuzzy logic and soft computing, vol 4529. Springer, New York, pp 789–798
78. Kashan AH (2011) An efficient algorithm for constrained global optimization and application to mechanical engineering design: League championship algorithm (LCA), Comput Aided Design 43(12):1769–1792
79. Eskandar H, Sadollah A, Bahreininejad A, Hamdi M (2012) Water cycle algorithm–a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 110:151–166
80. Yang X-S (2010) Engineering optimization: an introduction with metaheuristic applications. Wiley, New York
81. Ragsdell KM, Phillips DT (1976) Optimal design of a class of welded structures using geometric programming. J Eng Ind 98(3):1021–1025
82. Kaveh A, Talatahari S (2010) An improved ant colony optimization for constrained engineering design problems. Eng Comput 27(1):155–182
83. Kannan BK, Kramer SN (1994) An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J Mech Design 116(2):405–411
84. Sandgren E (1990) Nonlinear integer and discrete programming in mechanical design optimization. J Mech Des 112(2):223–229
85. Duncan W, Huntley C, Hokenstrom J, Cudworth A, McDaniel T (1987) Design of small dams. A water resources technical publication. Final report. Engineering and Research Center, Bureau of Reclamation, Denver
86. Kaveh A, Zakian P (2015) "Stability Based Optimum Design of Concrete Gravity Dam Using Css, Cbo and Ecbo Algorithms. Iran Univ Sci Technol 5(4):419–431
87. Akbari J, Ahmadi MT, Moharrami H (2011) Advances in concrete arch dams shape optimization. Appl Math Model 35(7):3316–3333
88. Mahani AS, Shojaee S, Salajegheh E, Khatibinia M (2015) Hybridizing two-stage meta-heuristic optimization model with weighted least squares support vector machine for optimal shape of double-arch dams. Appl Soft Comput 27:205–218
89. Cai X, Wu Y, Yi J, Ming Y (2011) Research on shape optimization of CSG dams. Water Sci Eng 4(4):445–454
90. Ahmed GL, Srivastava DK, Rani D (2013) Optimization–simulation models for yield assessment of a single reservoir system. J Indian Water Res Soc 33(4):9–16
91. Gandomi AH, Yang X-S, Talatahari S, Alavi AH (2013) Metaheuristic applications in structures and infrastructures. Newnes. Book. Elsevier. ISBN: 9780123983640
92. Gholizadeh S, Seyedpoor SM (2011) Shape optimization of arch dams by metaheuristics and neural networks for frequency constraints." Sci Iran 18(5):1020–1027