Soft Computing (2020) 24:9121–9141
https://doi.org/10.1007/s00500-019-04443-z

METHODOLOGIES AND APPLICATION

A novel life choice-based optimizer

Abhishek Khatri[1] · Akash Gaba[1] · K. P. S. Rana[1] · Vineet Kumar[1]

Published online: 6 November 2019

Abstract
This paper presents a novel metaheuristic algorithm named as life choice-based optimizer (LCBO) developed on the typical decision-making ability of humans to attain their goals while learning from fellow members. LCBO is investigated on 29 popular benchmark functions which included six CEC-2005 functions, and its performance has been benchmarked against seven optimization techniques including recent ones. Further, different abilities of LCBO optimization algorithm such as exploitation, exploration and local minima avoidance were also investigated and have been reported. In addition to this, scalability is tested for several benchmark functions where dimensions have been varied till 200. Furthermore, two engineering optimization benchmark problems, namely pressure vessel design and cantilever beam design, were also optimized using LCBO and the results have been compared with recently reported other algorithms. The obtained comparative results in all the above-mentioned experimentations revealed the clear superiority of LCBO over the other considered metaheuristic optimization algorithms. Therefore, based on the presented investigations, it is concluded that LCBO is a potential optimizer for engineering problems.

Keywords Optimization · Optimization techniques · Metaheuristic algorithm · Metaheuristics-constrained optimization · Life choice-based optimizer

1 Introduction

Optimization is the process of obtaining optimum results for a given problem while satisfying certain constraints. Several requirements in different fields like science, mathematics, engineering and finance can be framed as optimization problems. Some of the applications of optimization are training neural networks, tuning of controllers, designing digital filters, etc. Though many classical optimization algorithms do exist, they are problem dependent and require gradient information to reach an optimum solution. Further, in some cases, classical methods may fail to attain global optima as they get stuck around local optima making the algorithm unsuitable for that particular problem.

The present fast-paced engineering world is the result of continuous improvements over several centuries. Particularly, this has been achieved through inspiration from numerous intelligent processes that exist in nature. In fact, understanding and modelling of such processes has led to the development of many optimization techniques. These techniques have always been the driving force in solving large number of complex problems involving several variables.

For the past few decades, metaheuristic algorithms have been introduced to solve many complex optimization problems and such algorithms are gaining popularity because of the following key reasons:

- *Simplicity* Most optimization algorithms are based on simple phenomenon which can be represented and described by simple mathematical expressions and methods.

Communicated by V. Loia.

✉ K. P. S. Rana
kpsrana1@gmail.com
https://www.nsut.ac.in

Abhishek Khatri
abhishekkhatri.nsit@gmail.com

Akash Gaba
aakashgaba@gmail.com

Vineet Kumar
vineetkumar27@gmail.com

[1] Department of Instrumentation and Control Engineering, Netaji Subhas University of Technology, Sector-3, Dwarka, New Delhi 110078, India

- *Independency from gradient* Unlike traditional optimization methods like gradient descent, metaheuristic algorithms do not use gradient for their implementation. This feature has been very helpful when the function under consideration either does not have gradient or it is difficult to obtain.
- *Local optima avoidance* Due to randomness and exploration factor of optimization algorithms, they have an inherent capability to avoid local optima.
- *Problem independence* Most optimization algorithms consider the problems as black boxes and therefore are treated as universal algorithms.

Further, for an optimizer, the two most desired features are *exploitation* and *exploration.* The overall capability of an optimization algorithm is highly dependent on these two features. *Exploitation* refers to rigorously searching the promising search space for global optima while *exploration* refers to searching for new promising search space. It is noteworthy that improved exploitation leads to fast convergence to optimal solution and improved exploration leads to avoidance of local optima. On the other hand, it may also be noted that very high exploitation leads to convergence towards local optima before the solutions could reach near global optima and very high exploration may lead to slow convergence of solution towards the global optima. Therefore, for a good optimization algorithm, there must exist a balance between exploitation and exploration.

2 Related works

As mentioned above, understanding and modelling of many natural processes and phenomena has led to the creation of several optimization techniques which have been very helpful in solving complex scientific problems. These algorithms can be broadly divided into following four categories:

Evolutionary algorithms This category is based on evolutionary processes present in the nature. In this class of algorithms, firstly, random population is generated and their fitness is calculated. Following this, new generation is evolved based on the stated rules of evolutionary algorithm. Genetic algorithm (GA) (Holland 1992) is the most popular in this category. In this algorithm, new population is generated by the process of crossover, cloning and mutation. Further, many algorithms have been developed in this category. Some of them are Evolutionary Strategy (François 1998), Genetic Programming (Koza 1994), Population-Based Incremental Learning (Baluja 1994), Fast Evolutionary Programming (Yao and Liu 1996), Differential Evolution (Storn and Price 1997), Grammatical Evolution (Ryan and Collins 1998), Enhanced GA (Coello and Montes 2002), Gene Expression Programming (Ferreira 2006), Co-Evolutionary Differential Evolution (CEDE) (Huang et al. 2007), Biogeography-Based Optimizer (Simons 2008), Asexual Reproduction Optimization (Farasat et al. 2010), States of Matter (Cuevas et al. 2014), Adaptive Dimensional Search (Hasançebi and Azad 2015), Stochastic Fractal Search (SFS) (Salimi 2015) and Multi-Verse Optimizer (MVO) (Mirjalili et al. 2016).

Swarm-based optimization algorithms This class of algorithms are based on social behaviour of animals. Collectively these are called swarms and are inspired from how swarms interact with each other in order to get their food. Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995) is the most popular algorithm in this category. In PSO, each particle changes its position based on personal best, global best, previous velocity and inertia. Following this, there had been several algorithms which make use of swarm-based optimization algorithm. Some of the examples are Ant Colony Optimization (Dorigo and Di Caro 1999), Marriage in Honey Bees Optimization Algorithm (Abbass 2002), Wasp Swarm Optimization (Pinto et al. 2005), Bees Algorithm (Pham et al. 2006), Cat Swarm Optimization (Chu et al. 2006), Co-Evolutionary Particle Swarm Optimization (CEPSO) (Krohling and Dos santos coelho 2006), Glow-Worms Optimization (Krishnanand and Ghose 2006), Artificial Bee Colony (Karaboga and Basturk 2007), Monkey Search Algorithm (Zhao and Tang 2008), Bee Collecting Pollen (Lu and Zhou 2008), Dolphin Partner Optimization (Yang et al. 2009), Group Search Optimizer (He et al. 2009), Cuckoo Search Algorithm (CSA) (Yang 2009b), Termite Colony Optimization (Hedayatzadeh et al. 2010), Firefly Algorithm (Yang 2009a), Bat Algorithm (BA) (Yang 2010), Hunting search (Oftadeh et al. 2010), Enhanced PSO (Gao and Hailu 2010), Krill Herd Algorithm (Gandomi and Alavi 2012), Migrating Birds Optimization (Duman et al. 2012), Fruit Fly Algorithm (Pan 2012), Flower Pollination Algorithm (Yang 2012), Enhanced CSA (Gandomi et al. 2013), Dolphin Echolocation Algorithm (Kaveh and Farhoudi 2013), Social Spider Optimization (Cuevas et al. 2013), Symbiotic Organisms Search (Cheng and Prayogo 2014), Grey Wolf Optimizer (Mirjalili et al. 2014), Bird Mating Optimizer (Askarzadeh 2014), Animal Migration Optimization (Li et al. 2014), Chicken Swarm Optimization (Meng et al. 2014), Firework Algorithm (Tan and Zhu 2015), Moth Flame Optimization (Mirjalili 2015a), Ant Lion Optimizer (Mirjalili 2015b), Elephant Herding Optimization (Wang et al. 2016a), Monarch Butterfly Optimization (Wang et al. 2016b), Dragon-Fly Algorithm (Mirjalili 2016a, b), Whale Optimization Algorithm (Mirjalili and Lewis 2016), Lion Optimization Algorithm (2016) (Yazdani and Jolai 2016),

Spotted Hyena Optimizer (SHO) (Dhiman and Kumar 2017) and Salp Swarm Algorithm (Mirjalili et al. 2017).

Physics-inspired optimization These methods draw inspiration from physical processes present in the nature. One of the oldest algorithms in this category is Simulated Annealing (SA) (Van Laarhoven and Aarts 1987). Annealing means heating a solid and then slowly letting it to cool down. In SA, the process of annealing is used mathematically to solve problems. Other examples of physics-based optimization algorithm are as follows. Harmony search (Woo Geem et al. 2001), Big Bang–Big Crunch (Erol and Eksin 2006), Colonizing Weeds (Mehrabian and Lucas 2006), Gravitational Search Algorithm (Rashedi et al. 2009), Intelligent Water Drops (Hosseini 2009), Charged System Search (Kaveh and Talatahari 2010), Grenade Explosion Method (Ahrari and Atai 2010), Chemical-Reaction-Inspired Metaheuristic (Lam and Li 2010), Artificial Chemical Reaction Optimization Algorithm (Alatas 2011), Galaxy-Based Search Algorithm (Hosseini 2011), Curved Space Optimization (Moghaddam et al. 2012),Water Cycle Algorithm (Eskandar et al. 2012), Black Hole Algorithm (Hatamlou 2013), Mine Blast Algorithm (Sadollah et al. 2013), Colliding Bodies Optimization (Kaveh and Mahdavi 2014), Forest Optimization Algorithm (Ghaemi and Feizi-Derakhshi 2014), Optics Inspired Optimization (Husseinzadeh Kashan 2014), Ecogeographic-Based Optimization (Zheng et al. 2014), Ray Optimization Algorithm (Kaveh 2014b), Tree Seed Algorithm (Kiran 2015), Water Wave Optimization (Zheng 2015), Lightning Search Algorithm (Shareef et al. 2015), Ions Motion Algorithm (Hatamlou et al. 2015), Runner-Root Algorithm (Merrikh-Bayat 2015), Electromagnetic Field Optimization (Abedinpourshotorban et al. 2016), Water Evaporation Optimization (Kaveh and Bakhshpoori 2016), Vibrating Particles System (Kaveh and Ilchi Ghazaan 2017) and Thermal Exchange Optimization (Kaveh and Dadras 2017).

Human-based optimization This optimization class draws inspiration from behaviour and activities performed by humans. It may be noted that humans are the most intelligent species in this world, and this very fact offers good inspiration for developing optimization algorithms. One of the recent algorithms in this class is Jaya algorithm (Venkata 2016) which takes inspiration from human behaviour of following best and avoiding worst. Other examples of human-based algorithm are as follows. Tabu Search (Glover 1989), Seeker-Based optimization (Dai et al. 2006), Imperialist Competitive Algorithm (Atashpaz-Gargari and Lucas 2007), Teaching Learning-Based Optimization (Rao et al. 2007), Interior Search (Gandomi 2014), Soccer League Competition Algorithm (Moosavian and Kasaee Roodsari 2014), Exchange Market Algorithm (Ghorbani and Babaei 2014), Group Counselling Optimization Algorithm (Eita and Fahmy 2014), Tug of War Optimization (Kaveh and Zolghadr 2016), Most Valuable Player Algorithm (Bouchekara 2017) and Volleyball Premier League Algorithm (Moghdani and Salimifard 2018).

In addition to the above major classes, there have been several algorithms which are inspired from mathematics concepts like geometry, algebra, etc. The Method of Moving Asymptotes (Svanberg 1987), Nonlinear Integer and Discrete Programming (NIDP) (Sandgren 1990), Generalized Convex Approximation (Chickermane and Gea 1996) and Sin Cosine Algorithm (Mirjalili 2016b) are such algorithms.

From the above-presented survey, one can easily infer that different metaheuristic algorithms have been developed to target different problems. Therefore, in the interest of technical development, there is always a need for a new algorithm to be developed and evaluated for particular problem so as to obtain superior results than the existing algorithms. The new algorithm introduced in this paper, life choice-based optimizer (LCBO), comes under the category of human-based algorithm. It is based on how a person makes a decision in life to attain his/her goal. Generally, a person makes decision based on different parameters which are dependent upon his colleagues. This very fact has been the key motivation of this work. Further, according to no free lunch (NFL) theory (Wolpert and Macready 1997), no algorithm performs best for all problems. Though several optimization algorithms, as mentioned above, already exist, NFL says that no algorithm is uniformly perfect and therefore there is always a need to develop superior methods. Furthermore, superior techniques are always required to be developed and tested for different scientific problems as they will save the time and effort of the scientific community, thereby making a significant contribution to the domain.

The paper is organized into five sections. Following introduction and survey in Sect. 1 and 2, respectively, in Sect. 3, inspiration, mathematical formulation and representation of LCBO are presented. In Sect. 4, the details of the used 29 benchmark functions, which consist of unimodal, multimodal and six CEC-2005 composite functions (Liang et al. 2005; Suganthan et al. 2005), are provided. In Sect. 5, LCBO is tested on optimization of benchmark functions and the comparative study of the obtained results with other recently reported popular algorithms has been presented in this section. Further, this section also includes the investigations of scalability and convergence tests for enhanced dimensions. LCBO is also investigated for solving two engineering benchmark problems, namely, pressure vessel design (PVD) and cantilever beam design (CBD), and the comparative performance results have been reported in this section. Finally, Sect. 6 draws the

conclusion and presents the future scope of research for LCBO. The mathematical descriptions of the investigated engineering problems are given in "Appendix" section.

3 Life choice-based optimizer

In this section, the inspiration and mathematical details of the LCBO algorithm are presented. Mathematical modelling of the LCBO algorithm has been presented highlighting all the background formulations and their expressions.

3.1 Inspiration

The LCBO algorithm is inspired by carefully observing the life cycle of an human being and his work ethics during active life where a person is motivated and has several different aims and targets to achieve. It is noteworthy that human is truly the most intellectual species and is thus far smarter and strategic. Humans always took inspiration from nature and thereby learnt new things. For example, certain Yogasanas like Gomukhasana (Cow-Face Pose) and Simhasana (Lion Pose) are practised for healthy lifestyle world over. The ability to learn from our fellow creatures and species has always been a crucial factor that has helped humans to emerge as far more superior than any other species. Humans have understood the significance of food chain and lifecycle that nature has enforced upon all species. Humans are able to realize the significance of each species and roles played by them for sustenance of life, so instead of focusing on complete extinction of other species, they have considered animals and plants as a part of a big family and focused on mutual survival. Humans have also built restricted zones for animals, created wildlife reserve throughout and are highly resolute to protect the endangered species from extinction. They have tamed animals, adopted them as pets and hence focused on mutual survival. Thus, humans have the capability to understand things better than any other species; that's why a lot of focus and investment has been made for creating machines that are able to think and act like humans, for example recently built humanoid robots. Recently, Sophia, a humanoid robot, became the first robot to receive citizenship in a country (Saudi Arabia) and was also named the United Nations Development Program Innovation Champion, also the first humanoid to hold a United Nations title (https://www.hansonrobotics.com/sophia/). Therefore, there exists lot of scope to develop new and future technologies which are based on human behaviour and thus the novel algorithm LCBO is also inspired from the choices and thinking pattern of humans to accomplish a target.

Inspiration from Jaya optimization technique The algorithm proposed in this work is also inspired from the already established recent algorithm Jaya which makes use of selective influence. It may be noted that in Jaya (Venkata 2016), only the best and worst search agent affects the current search agent, whereas in the proposed optimizer, Eq. 6, which is only a particular/optional branch in the proposed algorithm according to random number generation, the best and better search agents (explained later in Sect. 3.2.2) also affect the current search agent resulting in better exploitation.

3.2 LCBO algorithm

In the proposed LCBO, the following three concepts can be used to completely describe it. These are presented in the following subsections.

3.2.1 Learning from the common best group

Human is always inspired by one thing or the other, whether it is his/her senior, some celebrity or fellow mates. When a person has some target in sight, he/she ponders and studies about how the best people in that field work to create a strategy in order to achieve targets. He/she always tries to take something resourceful from the best in the fields to achieve the target and derive a pattern or parameter by observing the superior person's efficiency and work on it so that he/she can develop some skills to achieve the target or solve the problem under consideration. For a given population X with sorted fitness values/cost functions, Eq. 1 represents the learning from the best feature of the LCBO algorithm:

$$X'_j = \sum_{k=1}^{n} [rand(k) * X_k]/n \tag{1}$$

Here, in summation, k varies from 1 to n, where n is a parameter in the algorithm and is equal to the ceil of the square root of the population considered to solve the problem. Parameter X_j is the jth or the current search agent in process, and X'_j represents that X_j will be updated only if X'_j has better fitness than X_j. Figure 1 depicts this feature of the algorithm. The search agent in the centre of the circles represents the current search agent in the process. The search agent is affected only by the position of the common best n search agents, and the level of influence is decided by the random numbers as shown in Fig. 1 by arrows of variable lengths.

3.2.2 Knowing very next best

Everyone wants to achieve his/her target, like achieving the dream job or purchasing a dream car but to accomplish large targets or dream, it takes lot of time and

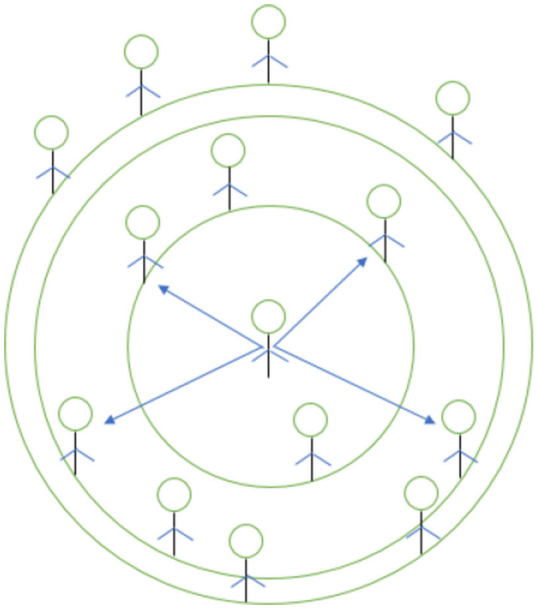

Fig. 1 Learning from the common best group

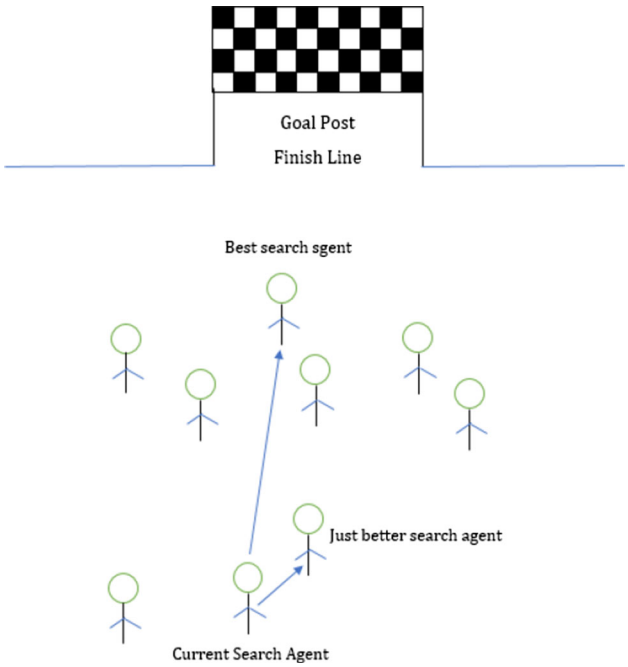

Fig. 2 Knowing presently best

perseverance. Instead of completely focusing onto massive targets, one must be able to realize the current position and the very nearest target in sight. Further, one also needs to understand how to move to a better position from the current position. So, the current target should also be prioritized. Therefore, there is a requirement to focus both on final destination as mentioned above and on the very next destination to achieve future goals. This operation is implemented with the help of the following algorithm:

$$f1 = 1 - (currentChances - 1)/(numberOfChances - 1) \tag{2}$$

$$f2 = 1 - f1 \tag{3}$$

$$bestDiff = f1 * r1 * \left(X_1 - X_j\right) \tag{4}$$

$$betterDiff = f2 * r1 * \left(X_{j-1} - X_j\right) \tag{5}$$

$$X_j^{'} = X_j + rand() * betterDiff + rand() * bestDiff \tag{6}$$

Here, $f1$ and $f2$ vary linearly from 0 to 1 and 1 to 0, respectively. The value of $r1$ is constant 2.35, and X_{j-1} refers to the position of the search agent whose fitness was just better than current search agent till the previous iteration. Further, X_1 refers to the best position of search agent that has been achieved till the previous iteration. The position of X_j will only be updated to X_j' if X_j' has better fitness than X_j. From Fig. 2, one can see that the current search agent is only affected by the search agents which have the best fitness value and agent that has just better fitness value and the level of influence is determined by Eqs. 3–6.

3.2.3 Reviewing mistakes

If humans are stuck somewhere or the technique they have been using to solve the problem under consideration is not working, they have the natural intelligence to review things and do proper analysis of the technique which is being used to solve the problem and try alternate methods. They are also capable of doing things in reverse to evaluate and approach the problem in a completely different manner, and it also increases the exploration part in the algorithm by trying to look at things from a completely different perspective.

$$X_j' = Xmax - (X_j - Xmin) * rand() \tag{7}$$

The technique described by Eq. 7 is named as Avi escape technique and has been used as generalized technique to increase the exploration of algorithm. Here, in the algorithm, *Xmax* and *Xmin* are the upper and lower bound values, respectively. It is similar to GA (Holland 1992) and CSA (Yang and Deb 2009) where new agents are created to solve the problem by using upper bound and lower bound values. Here, X_j is the current agent in the process of evaluation.

As usual with all the optimization methods, the LCBO will start with population size, lower and upper bounds and number of iterations. For the first iteration, population is generated and corresponding fitness is evaluated and ordered along with the agents. Now, the positions of the agents and their fitness are updated iteratively till the target fitness has been obtained or the number of iterations is

exhausted. It may be noted that only one of the three operations as described above will be executed for updating of the agents depending on the value of random number. The pseudocode presented in the next section describes the operations in an orderly manner.

3.3 Pseudocode

The following is the pseudocode of the LCBO.

Initialize the human population X_j (j = 1, 2, 3, ... population)
Initialize $r1$, *currentChance*
Compute n
Calculate the fitness values and sort the population

While *(currentChance < numberOfChances)*

For *each Search Agent* ***do***
$z = rand()$

If $(z > 0.875)$
Update current search agent using Eq. 1
Else if $(z < 0.70)$
Update $f1$, $f2$ *using Eqs. 2 and 3*
Update current search agent using Eqs. 4-6
Else
Update current search agent using Eq. 7
End if

Evaluate fitness value of the search agent

If *(new fitness value is better than previous fitness value)*
Update search agent position and fitness value
End if

End for

$X = sort(X)$
currentChance = currentChance + 1

End while
return X_1

4 Details of the used benchmark functions

In order to determine the optimization efficiency of an algorithm, its critical testing is required. The importance of exploration and exploitation has already been stated in Introduction section, and thus for checking the overall performance of the algorithm, the benchmark test functions have been carefully chosen and have been presented in the following subsections. For systematic evaluation of the LCBO algorithm, the 29 chosen functions are divided into following three parts.

4.1 Unimodal functions (functions 1 to 7)

In the chosen unimodal functions (1 to 7), there exists only a single local optima value and hence it is the global minimum value of the respective function. These functions are used to test the exploitation affinity of the algorithm. Algorithms which are able to optimize these functions have great exploitation ability. As there is only a single minimum value, the LCBO algorithm should be able to reach quickly towards the global minima. The details of these functions are presented in Table 1.

4.2 Multimodal benchmark functions (functions 8 to 23)

These functions consist of many local optima and therefore are difficult to solve than unimodal functions. The search agents sometimes get stuck in the local optima and are unable to escape. It is noteworthy that functions 8 to 13 are of variable dimension and 14 to 23 are of fixed-dimension multimodal benchmark functions. The mathematical details of these functions are tabulated in Tables 2 and 3. It may be noted that the difficulty level of these functions increases with the search area, number of local optima and number of dimensions. The ability to explore new search region plays a vital role in evaluation of these functions, and hence they are good for determining the exploration ability of the algorithm.

Table 1 Unimodal benchmark functions

Function no.	Function	Dimension	Range	Optimal minima
1.	$F_1(x) = \sum_{i=1}^{N} x_i^2$	30, 200	[− 100, 100]	0
2.	$F_2(x) = \sum_{i=1}^{N} \lvert x_i \rvert + \prod_{i=1}^{N} \lvert x_i \rvert$	30, 200	[− 10, 10]	0
3.	$F_3(x) = \sum_{i=1}^{N} \left(\sum_{j=1}^{i} x_i \right)^2$	30, 200	[− 100, 100]	0
4.	$F_4(x) = \max(\lvert x_i \rvert, 1 \le i \le N)$	30, 200	[− 100, 100]	0
5.	$F_5(x) = \sum_{i=1}^{N} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2 \right]$	30, 200	[− 30, 30]	0
6.	$F_6(x) = \sum_{i=1}^{N} (\lvert x_i + 0.5 \rvert)^2$	30, 200	[− 100, 100]	0
7.	$F_7(x) = \sum_{i=1}^{N} i x_i^4 + random[0, 1)$	30, 200	[− 1.28, 1.28]	0

Table 2 Multimodal benchmark functions

Function no.	Function	Dimension	Range	Optimal Minima
8.	$F_8(x) = \sum_{i=1}^{N} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30, 200	[− 500, 500]	− 418.982 × *Dim*
9.	$F_9(x) = \sum_{i=1}^{N} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30, 200	[− 5.12, 5.12]	0
10.	$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}\right) - \exp\left(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\right) + 20 + e$	30, 200	[− 32, 32]	0
11.	$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30, 200	[− 600, 600]	0
12.	$F_{12}(x) = \frac{\pi}{N}\left\{10\sin(\pi z_1) + \sum_{i=1}^{N-1}(z_i - 1)^2\left[1 + 10\sin^2(\pi z_{i+1})\right] + (z_n - 1)^2\right\} + \sum_{i=1}^{N} u(x_i, 10, 100, 4) z_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30, 200	[− 50, 50]	0
13.	$F_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{N}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{N} u(x_i, 5, 100, 4)$	30, 200	[− 50, 50]	0

4.3 CEC composite benchmark functions (functions 24 to 29)

These are six composite benchmark functions taken from CEC-2005. These are rotated and shifted classical variants of standard functions and are of greatest complexity among all benchmark functions in terms of difficulty. They have a large number of local optima values which are very difficult to escape from. The functions are available in Table 4. The detailed equations and expression of the benchmark function are available in the CEC-2005 technical reports (Liang et al. 2005; Suganthan et al. 2005).

5 Results and discussions

In this section, the experimental setup used to conduct various tests and the details regarding the evaluation of tests and obtained results of the used benchmark functions are presented. In subsection 5.1, experimental setup arrangement is presented which includes details regarding function testing such as population, iteration and system software version. In subsection 5.2, the study of the results of LCBO and other algorithms has been carried out. In subsection 5.3, the analysis of results of functions 1 to 13 having very high dimension (200) shows the adaptability of LCBO for dealing with high-complexity problems. In subsection 5.4, the convergence curve patterns are analysed. Engineering problem solving is an important component for the testing of any proposed optimization method. Therefore, two important design benchmark problems, namely PVD and CBD, are investigated for LCBO in subsection 5.5.

5.1 Experimental setup

For investigating the optimization capability of LCBO, each of the functions described earlier was optimized 30 times independently and the results in terms of average fitness value of 30 runs along with the standard deviations for each function or application have been recorded. The optimization technique offering least average fitness and deviation is considered as the winning technique. The software used for all the investigation was MATLAB™ on Windows 10 and 64 bits i-5 Processor 7th Generation, 2.5 GHz and 8 GB RAM.

5.2 Benchmark functions' testing results

In this section, comparative study of LCBO algorithm with the other popular and latest algorithms has been presented. The presentation has been organized into three different

Table 3 Fixed-dimension multimodal benchmark functions

Function no.	Function	Dimension	Range	Optimal minima
14.	$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\right)^{-1}$	2	[− 65, 65]	1
15.	$F_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[− 5, 5]	0.0003
16.	$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[− 5, 5]	− 1.0316
17.	$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[− 5, 5]	0.398
18.	$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	[− 2, 2]	3
19.	$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	[1, 3]	− 3.86
20.	$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	[0, 1]	− 3.32
21.	$F_{21}(x) = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	− 10.1532
22.	$F_{22}(x) = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	− 10.4028
23.	$F_{23}(x) = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	− 10.536

subsections: Sects. 5.2.1, 5.2.2 and 5.2.3, for detailed and complete analysis of the performance of LCBO algorithm. For comparative performance analysis, optimization of 29 benchmark functions using seven potential reported optimization techniques, namely SHO, GWO, PSO, MFO, MVO, SCA and GA, has been used. It may be noted that SHO, GWO, MFO, MVO and SCA are the most recent ones as they were reported in 2017, 2014, 2014, 2016 and 2016, respectively. The results and parameters of these algorithms were already reported in SHO (Dhiman and Kumar 2017) for the above-mentioned benchmark functions. It is noteworthy that the population and iteration used for benchmark function optimization in (Dhiman and Kumar 2017) were 30 and 1000, respectively, for each algorithm. In order to offer a fair competition, same population and iteration values were chosen for LCBO algorithm.

5.2.1 Functions 1–7 (unimodal)

Table 5 presents the obtained results in terms of the average fitness values and the standard deviations for the optimized unimodal functions. As mentioned above along with the LCBO, the results of seven other optimization methods as reported by Dhiman and Kumar (2017) are also presented. Based on the average fitness values and the deviations obtained, one can clearly infer that LCBO offered least values. Therefore, for optimization of unimodal benchmark functions, it is concluded that LCBO is a superior optimization method as compared to the seven other methods. LCBO algorithm offers the best results for functions 1 to 5 and second best results for functions 6 and 7.

5.2.2 Functions 8–23 (multimodal)

In line with the unimodal function optimization, functions 8 to 23 were investigated under multimodal function optimization. Table 6 presents the obtained results wherein it can be inferred that for functions 8, 9, 10, 11, 13, 15, 18, 19, 20, 21 and 23, LCBO is the clear winner. On the other hand, for functions 12, 14, 16, 17 and 22, it is the second or third best. Therefore, for optimization of multimodal benchmark functions also, it can be concluded that LCBO is a superior optimization method as compared to the seven other methods.

5.2.3 Functions 24–29 (composite CEC benchmark functions)

In order to further test the capability of the LCBO, the next experiment was to test the complex function optimization. For the same, six composite benchmark functions were taken from CEC 2005. These are rotated and shifted classical variants of standard functions and therefore offer greatest complexity among all the benchmark functions. They also have multiple local optima values, and it is usually difficult to escape from these local optima. From the results given in Table 7, one can clearly observe that LCBO algorithm gives the best result for four out of the six functions (24, 25, 26 and 28). This confirms the LCBO's ability to easily escape local minima and move towards global minima. It also ensures superior balance between exploration and exploitation ability as exhibited by LCBO.

Table 4 Composite benchmark functions

Function no.	Function	Dimension	Range	Optimal minima
24.	F_{24} : $f1,f2,f3,\ldots,f10 =$ Sphere Function $[\alpha1,\alpha2,\alpha3,\ldots,\alpha10]$=$[1,1,1,\ldots,1]$ $[\beta1,\beta2,\beta3,\ldots,\beta10] = [5/100,5/100,5/100,\ldots,5/100]$	10	[− 5, 5]	0
25.	F_{25} : $f1,f2,f3,\ldots,f10 =$ Griewank function $[\alpha1,\alpha2,\alpha3,\ldots,\alpha10]$=$[1,1,1,\ldots,1]$ $[\beta1,\beta2,\beta3,\ldots,\beta10] = [5/100,5/100,5/100,\ldots,5/100]$	10	[− 5, 5]	0
26.	F_{26} : $f1,f2,f3,\ldots,f10 =$ Griewank function $[\alpha1,\alpha2,\alpha3,\ldots,\alpha10] = [1,1,1,\ldots,1]$ $[\beta1,\beta2,\beta3,\ldots,\beta10] = [1,1,1,\ldots,1]$	10	[− 5, 5]	0
27.	F_{27} : $f1,f2 =$ Ackley function $f3,f4 =$ Rastrigin function $f5,f6 =$ Weierstrass function $f7,f8 =$ Griewank function $f9,f10 =$ Sphere Function $[\alpha1,\alpha2,\alpha3,\ldots,\alpha10] = [1,1,1,\ldots,1]$ $[\beta1,\beta2,\beta3,\ldots,\beta10] = [5/32,5/32,1,1,5/0.5,5/0.5,5/100,5/100,5/100,5/100]$	10	[− 5, 5]	0
28.	F_{28} : $f1,f2 =$ Rastrigin function $f3,f4 =$ Weierstrass function $f5,f6 =$ Griewank function $f7,f8 =$ Ackley function $f9,f10 =$ Sphere Function $[\alpha1,\alpha2,\alpha3,\ldots,\alpha10] = [1,1,1,\ldots,1]$ $[\beta1,\beta2,\beta3,\ldots,\beta10] = [1/5,1/5,5/0.5,5/0.5,5/100,5/100,5/32,5/32,5/100,5/100]$	10	[− 5, 5]	0
29.	F_{29} : $f1,f2 =$ Rastrigin function $f3,f4 =$ Weierstrass function $f5,f6 =$ Griewank function $f7,f8 =$ Ackley function $f9,f10 =$ Sphere Function $[\alpha1,\alpha2,\alpha3,\ldots,\alpha10] = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1]$ $[\beta1,\beta2,\beta3,\ldots,\beta10] = \left[0.1\times\frac{1}{5},0.2\times\frac{1}{5},0.3\times\frac{5}{0.5},0.4\times\frac{5}{0.5},0.5\times\frac{5}{100},0.6\times\frac{5}{100},0.7\times\frac{5}{32},0.8\times\frac{5}{32},0.9\times\frac{5}{100},1\times\frac{5}{100}\right]$	10	[− 5, 5]	0

Table 5 Result of unimodal benchmark functions

Method → Function ↓	LCBO		SHO		GWO		PSO	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
1.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.69E−59	7.30E−59	4.98E−09	1.40E−08
2.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.20E−34	1.30E−34	7.29E−04	1.84E−03
3.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E−14	4.10E−14	1.40E+01	7.13E+00
4.	4.26E−307	0.00E+00	7.78E−12	8.96E−12	2.02E−14	2.43E−14	6.00E−01	1.72E−01
5.	2.66E+00	7.9641E+00	8.59E+00	5.53E−01	2.79E+01	1.84E+00	4.93E+01	3.89E+01
6.	1.13E−06	3.44E−06	2.46E−01	1.78E−01	6.58E−01	3.38E−01	9.23E−09	1.78E−08
7.	1.15E−04	1.22E−04	3.29E−05	2.43E−05	7.80E−04	3.85E−04	6.92E−02	2.87E−02
Method → Function ↓	MFO		MVO		SCA		GA	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
1.	3.15E−04	5.99E−04	2.81E−01	1.11E−01	3.55E−02	1.06E−01	1.95E−12	2.01E−11
2.	3.71E+01	2.16E+01	3.96E−01	1.41E−01	3.23E−05	8.57E−05	6.53E−18	5.10E−17
3.	4.42E+03	3.71E+03	4.31E+01	8.97E+00	4.91E+03	3.89E+03	7.70E−10	7.36E−09
4.	6.70E+01	1.06E+01	8.80E−01	2.50E−01	1.87E+01	8.21E+00	9.17E+01	5.67E+01
5.	3.50E+03	3.98E+03	1.18E+02	1.43E+02	7.37E+02	1.98E+03	5.57E+02	4.16E+01
6.	1.66E−04	2.01E−04	3.15E−01	9.98E−02	4.88E+00	9.75E−01	3.15E−01	9.98E−02
7.	3.22E−01	2.93E−01	2.02E−02	7.43E−03	3.88E−02	5.79E−02	6.79E−04	3.29E−03

5.3 Scalability test

Scalability test is an important component of evaluation of any optimization technique. It assesses the ability of an optimization technique to handle higher dimensions' test functions. The complexity of an optimization problem increases exponentially with increase in the number of dimensions, and solving any problem set with large number of unknown variables is always a challenge. In this test, functions 1 to 13, defined in Tables 1 and 2, were used wherein the dimensions were increased from 30 to 200. For scalability performance, population and iterations were kept as 30 and 1000, respectively. In line with previous subsection, 30 independent trials were executed and the results in terms of average and standard deviation were recorded. For comparison purpose, scalability results of seven techniques, namely ALO, PSO, SMS, BA, FPA, CSA and GA, as reported in (Mirjalili 2015b) were used. It may be noted that in this work the population and iteration values of 100 and 5000 were used by competing methods against the 30 and 1000 for LCBO. The setting of these parameters is a real challenge to LCBO. The results of scalability test are presented in Table 8 wherein it can be confirmed that LCBO algorithm gives the best result for all functions from 1 to 13 except function 6. This proves the dominance of the LCBO algorithm in dealing with functions with large dimensions and clearly shows that LCBO is superior as it gives better results in significantly lower number of function calls. The performance of other algorithms is considerably poorer than LCBO. Dull performance of the rest of the algorithms also highlights the fact that large dimensions' problem solving is quite difficult.

5.4 Convergence analysis

Having dealt with the scalability analysis, the next activity was to study the convergence. Convergence pattern is useful for understanding the exploration and exploitation ability of an optimization algorithm. For the same, in this section, a total of 16 scalable functions were tested and their convergences were recorded for varying dimensions. The investigated dimensions were 30, 50, 80, 100 and 200. Figures 3 and 4 depict the convergence plots of LCBO for the 16 functions considered for varying dimensions. It may be noted that Figs. 3 and 4 represent the plots for unimodal and multimodal functions, respectively. As evident from these plots, LCBO survives the test of scalability and passes it with flying colours.

Figure 5 shows the convergence of the LCBO for CEC test functions. Again in all the presented cases, LCBO convergence can be clearly observed.

Based on the above-presented results, the following remarks about LCBO can be inferred.

Table 6 Result of multimodal benchmark functions

Method → Function ↓	LCBO		SHO		GWO		PSO	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
8.	− 1.25E+04	9.11E−06	− 1.16E+03	2.72E+02	− 6.14E+03	9.32E+02	− 6.01E+03	1.30E+03
9.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.34E−01	1.66E+00	4.72E+01	1.03E+01
10.	2.66E−15	1.81E−15	2.48E+00	1.41E+00	1.63E−14	3.14E−15	3.86E−02	2.11E−01
11.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.29E−03	5.24E−03	5.50E−03	7.39E−03
12.	2.95E−08	8.18E−08	3.68E−02	1.15E−02	3.93E−02	2.42E−02	1.05E−10	2.06E−10
13.	2.60E−03	4.70E−03	9.29E−01	9.52E−02	4.75E−01	2.38E−01	4.03E−03	5.39E−03
14.	2.40E+00	3.35E+00	9.68E+00	3.29E+00	3.71E+00	3.86E+00	2.77E+00	2.32E+00
15.	3.37E−04	5.39E−05	9.01E−04	1.06E−04	3.66E−03	7.60E−03	9.09E−04	2.38E −04
16.	− 1.03E+00	6.71E−16	− 1.06E+01	2.86E−011	− 1.03E+00	7.02E−09	− 1.03E+00	0.00E+00
17.	3.98E−01	1.39E−07	3.97E−01	2.46E−01	3.98E−01	7.00E−07	3.97E−01	9.03E−16
18.	3.00E+00	1.67E−15	3.00E+00	9.05E+00	3.00E+00	7.16E−06	3.00E+00	6.59E−05
19.	− 3.86E+00	2.71E−15	− 3.75E+00	4.39E−01	− 3.86E+00	1.57E−03	3.90E+00	3.37E−15
20.	− 3.32E+00	1.36E−15	− 1.44E+00	5.47E−01	− 3.27E+00	7.27E−02	− 3.32E+00	2.66E−01
21.	− 1.02E+01	1.10E−03	− 2.08E+00	3.80E−01	− 9.65E+00	1.54E+00	− 7.54E+00	2.77E+00
22.	− 1.04E+01	1.10E−02	− 1.61E+01	2.04E−04	− 1.04E+01	2.73E−04	− 8.55E+00	3.08E+00
23.	− 1.05E+01	3.78E−01	− 1.68E+00	2.64E−01	− 1.05E+01	1.81E−04	− 9.19E+00	2.52E+00
Method → Function ↓	MFO		MVO		SCA		GA	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
8.	− 8.04E+03	8.80E+02	− 6.92E+03	9.19E+02	− 3.81E+03	2.83E+02	− 5.11E+03	4.37E+02
9.	1.63E++02	3.74E+01	1.01E+02	1.89E+01	2.23E+01	3.25E+01	1.23E−01	4.11E+01
10.	1.60E+01	6.18E+00	1.15E+00	7.87E−01	1.55E+01	8.11E+00	5.31E−11	1.11E−10
11.	5.03E−02	1.74E−01	5.74E−01	1.12E−01	3.01E−01	2.89E−01	3.31E−06	4.23E−05
12.	1.26E+00	1.83E+00	1.27E+00	1.02E+00	5.21E+01	2.47E+02	9.16E−08	4.88E−07
13.	7.24E−01	1.48E+00	6.60E−02	4.33E−02	2.81E+02	8.63E+02	6.39E−02	4.49E−02
14.	2.21E+00	1.80E+00	9.98E−01	9.14E−12	1.26E+00	6.86E−01	4.39E+00	4.41E−02
15.	1.58E−03	3.50E−03	7.15E−03	1.26E−02	1.01E−03	3.75E−04	7.36E−03	2.39E−04
16.	− 1.03E+00	0.00E+00	− 1.03E+00	4.74E−08	− 1.03E+00	3.23E−05	− 1.04E+00	4.19E−07
17.	3.98E−01	1.13E−16	3.98E−01	1.15E−07	3.99E−01	7.61E−04	3.98E−01	3.71E−17
18.	3.00E+00	4.25E−15	5.70E+00	1.48E+01	3.00E+00	2.25E−05	3.01E+00	6.33E−07
19.	− 3.86E+00	3.16E−15	− 3.86E+00	3.53E−07	− 3.86E+00	2.55E−03	− 3.30E+00	4.37E−10
20.	− 3.23E+00	6.65E−02	− 3.23E+00	5.37E−02	− 2.84E+00	3.71E−01	− 2.39E+00	4.37E−01
21.	− 6.20E+00	3.52E+00	− 7.38E+00	2.91E+00	− 2.28E+00	1.80E+00	− 5.19E+00	2.34E+00
22.	− 7.95E+00	3.20E+00	− 8.50E+00	3.02E+00	− 3.99E+00	1.99E+00	− 2.97E+00	1.37E−02
23.	− 7.50E+00	3.68E+00	− 8.41E+00	3.13E+00	− 4.49E+00	1.96E+00	− 3.10E+00	2.37E+00

- The exploitation ability of LCBO algorithm is very impressive as can be seen from results of unimodal functions optimization.
- The exploration ability of LCBO algorithm is great as can be seen from its superior result than the other algorithms. In none of the multimodal functions, it offered unsatisfactory result and was always in top 3 in about 95% of the benchmark functions.
- In multimodal composite CEC functions, which are extremely difficult to handle, it gave best result in four out of six functions.
- LCBO has a very good balance between exploration and exploitation and thus has a very wide scope for modification and future work.

Further, the optimization techniques are usually applied for real-life problem solving and they are expected to be a

Table 7 Result of composite benchmark functions

Method → Function ↓	LCBO		SHO		GWO		PSO	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
24.	36.6667	10.358	2.30E+02	1.37E+02	8.39E+01	8.42E+01	6.00E+01	8.94E+01
25.	78.9951	28.0369	4.08E+02	9.36E+01	1.48E+02	3.78E+01	2.44E+02	1.73E+02
26.	153.2114	36.5364	3.39E+02	3.14E+01	3.53E+02	5.88E+01	3.39E+02	8.36E+01
27.	710.1354	183.2439	7.26E+02	1.21E+02	4.23E+02	1.14E+02	4.49E+02	1.42E+02
28.	6.1171	1.0227	1.06E+02	1.38E+01	1.36E+02	2.13E+02	2.40E+02	4.25E+02
29.	877.5392	27.6731	5.97E+02	4.98E+00	8.26E+02	1.74E+02	8.22E+02	1.80E+02
Method → Function ↓	MFO		MVO		SCA		GA	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
24.	1.18E+02	7.40E+01	1.40E+02	1.52E+02	1.20E+02	3.11E+01	5.97E+02	1.34E+02
25.	9.20E+01	1.36E+02	2.50E+02	1.44E+02	1.14E+02	1.84E+00	4.09E+02	2.10E+01
26.	4.19E+02	1.15E+02	4.05E+02	1.67E+02	3.89E+02	5.41E+01	9.30E+02	8.31E+01
27.	3.31E+02	2.09E+01	3.77E+02	1.28E+02	4.31E+02	2.94E+01	4.97E+02	3.24E+01
28.	1.13E+02	9.27E+01	2.45E+02	9.96E+01	1.56E+02	8.30E+01	1.90E+02	5.03E+01
29.	8.92E+02	2.41E+01	8.33E+02	1.68E+02	6.06E+02	1.66E+02	6.65E+02	3.37E+02

strong tool in this aspect also. For the same, two benchmark engineering problems have been taken up as described in the following section.

5.5 Engineering applications

Engineering problems are generally constraint based, and the optimization algorithms are required to be modified accordingly so as to apply in these applications. Different types of penalty functions are used for handling constraints. The basic idea behind using these penalty functions is that when the search agents go out of range or violate given constraints, then some form of penalty is imposed to the cost function so that these agents are modified. The following are the popular types of penalty functions.

- *Static penalty* This type of penalty function is completely independent of the number of iterations, and this type of penalty varies with the square of magnitude of amount of violation.
- *Dynamic penalty* In this type of penalty function, the penalty value varies with time and may increase or decrease with the current iteration value. Usually, it increases with time.
- *Annealing penalty* In this type of penalty function, the penalty coefficients are changed with iteration whenever the algorithm gets stuck in the local optima and only the active constraints are considered in each iteration that is generally increased with iteration.
- *Death penalty* Whenever any constraint is violated by the search agent, it is assigned zero fitness and there is no need to compute extent of violation of constraints.

In this work, death penalty has been imposed for both the following design problems. This was done by assigning zero fitness to the search agents violating the constraints.

5.5.1 Pressure vessel design

In this problem, it is required to reduce the cost of fabrication of the vessel. The mathematical description of the PVD problem has been taken the same as in (Salimi 2015), and mathematical expressions are provided in "Appendix" section. There are four constraint conditions apart from cost function minimization. The population and iteration, for optimizing PVD, were kept as 30 and 400, respectively. The performance of LCBO algorithm has been compared with other algorithms, namely GA, CEPSO, CEDE, PSO, NIDP and SFS, as reported in (Salimi 2015). Table 9 presents the best results obtained out of 30 trials of LCBO and compares the results reported in (Salimi 2015). From Table 9, it can be concluded that LCBO offers the least cost function and therefore, is the most suitable technique for presser vessel design. It is able to maintain all the given constraints while leading to the optimal solution.

In addition to the above, statistical analysis of the 30 cost function values was also performed and the results are presented in Table 10. From Table 10, one can infer that

Table 8 Scalability results for 200-dimensions

Method → Function ↓	LCBO		ALO		PSO		SMS	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
1.	0.00E+00	0.00E+00	7.89E−07	1.10E−07	23.799	11.721	1039.213	0.4243
2.	2.7687E−313	0.00E+00	530.82	222.67	237.87	22.432	1832.44	0.0122
3.	0.00E+00	0.00E+00	2331.4	507.18	4693.34	503.57	2034.88	0.3780
4.	1.31E−300	0.00E+00	30.58	1.1446	40.111	0.5879	300.265	0.0023
5.	1.0677	0.2054	167.04	49.746	911.2342	95.245	3863.53	0.5329
6.	1.33E+01	0.4279	7.60E−07	7.39E−08	43.421	14.206	2494.43	0.0003
7.	9.20E−05	1.87E−05	0.050546	0.014407	17.321	4.0133	28.359	1.99E−05
8.	− 8.38E+04	0.0681	− 44,426	1442.5	− 18,136	4962.4	− 35,969	0.8765
9.	0.00E+00	0.00E+00	613.89	66.795	748.58	24.301	480.01	0.2365
10.	3.85E−15	1.06E−15	2.3058	0.25542	15.183	0.57627	17.293	0.0974
11.	0.00E+00	0.00E+00	0.007424	0.00651	3241.2	137.49	4801.5	0.8532
12.	9.56E−02	0.0059	5.3982	0.59591	4.07E+05	4.77E+05	1.00E+08	1.99E−05
13.	1.09E−02	0.0032	0.13915	0.22199	1.24E+06	5.82E+05	1.00E+08	1.99E−05
Method → Function ↓	BA		FPA		CSA		GA	
	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation	Average value	Standard deviation
1.	1117.34	20731	55.989	32. 678	3.80E−05	1.85E−05	227.75	186.56
2.	3842.82	468.28	280.6	6.9384	400.10	0.8656	6322.6	1092.7
3.	1090.75	475.06	24,219	8540	12,957	633.75	11,206	3986.1
4.	65.667	2.8293	37.689	2.4572	30.936	1.6877	101.54	2.5321
5.	1410.80	591.07	3150.7	1490.6	332.67	159.88	964.49	748.76
6.	51.2056	12.005	166.99	41.109	8.17E−05	4.55E−05	482.56	278.61
7.	2.4344	0.12756	4.8391	1.5354	0.40131	0.008707	116.56	60.161
8.	− 25,632	869.47	− 45,771	3097.8	− 52,600	156.04	− 28,660	1011
9.	723.38	100.96	702.95	69.653	541.58	41.889	1645.8	37.155
10.	18.159	0.067775	17.544	0.16684	17.654	2.982	20.361	0.14256
11.	4937	268.42	180.74	36.084	0.001191	0.001148	3306.8	113.3
12.	1.69E+09	4.28E+08	4.37E+07	3.22E+07	1.00E+10	0.0045	8.14E+09	9.54E+08
13.	2.25E+09	8.85E+08	9.87E+07	3.80E+07	1.00E+10	0.0568	1.38E+10	1.45E+09

the best, mean and worst values of cost function of LCBO are the least among the investigated methods. The number of function evaluations (FE) gives the computation load of a given method. As seen from Table 10, LCBO makes use of very small number of FE and therefore is a very light optimization technique.

5.5.2 Cantilever beam design

CBD is one of the most widely tackled engineering problems. The mathematical description of the CBD problem has been the same as in (Wolpert and Macready 1997), and brief expressions are given in "Appendix" section. In this problem, one needs to find the optimum values of the five parameters of the beam within the given bounds. The designed parametric values should be so as to yield the minimum cost function while obeying the given constraint.

The population and iteration, for optimizing CBD, were kept as 30 and 1000, respectively. The performance of LCBO algorithm has been compared with other algorithms such as MFO, MMA, GCA_1, GCA_2, CSA and SOS which is the same as considered in (Mirjalili 2015a). Table 11 presents the best results obtained of 30 trials of LCBO and compares them with the results reported in (Mirjalili 2015a). From Table 11, it can be concluded that LCBO offers the least cost function and hence is the most suitable technique for CBD. It is able to maintain the given constraint while leading to the optimal solution.

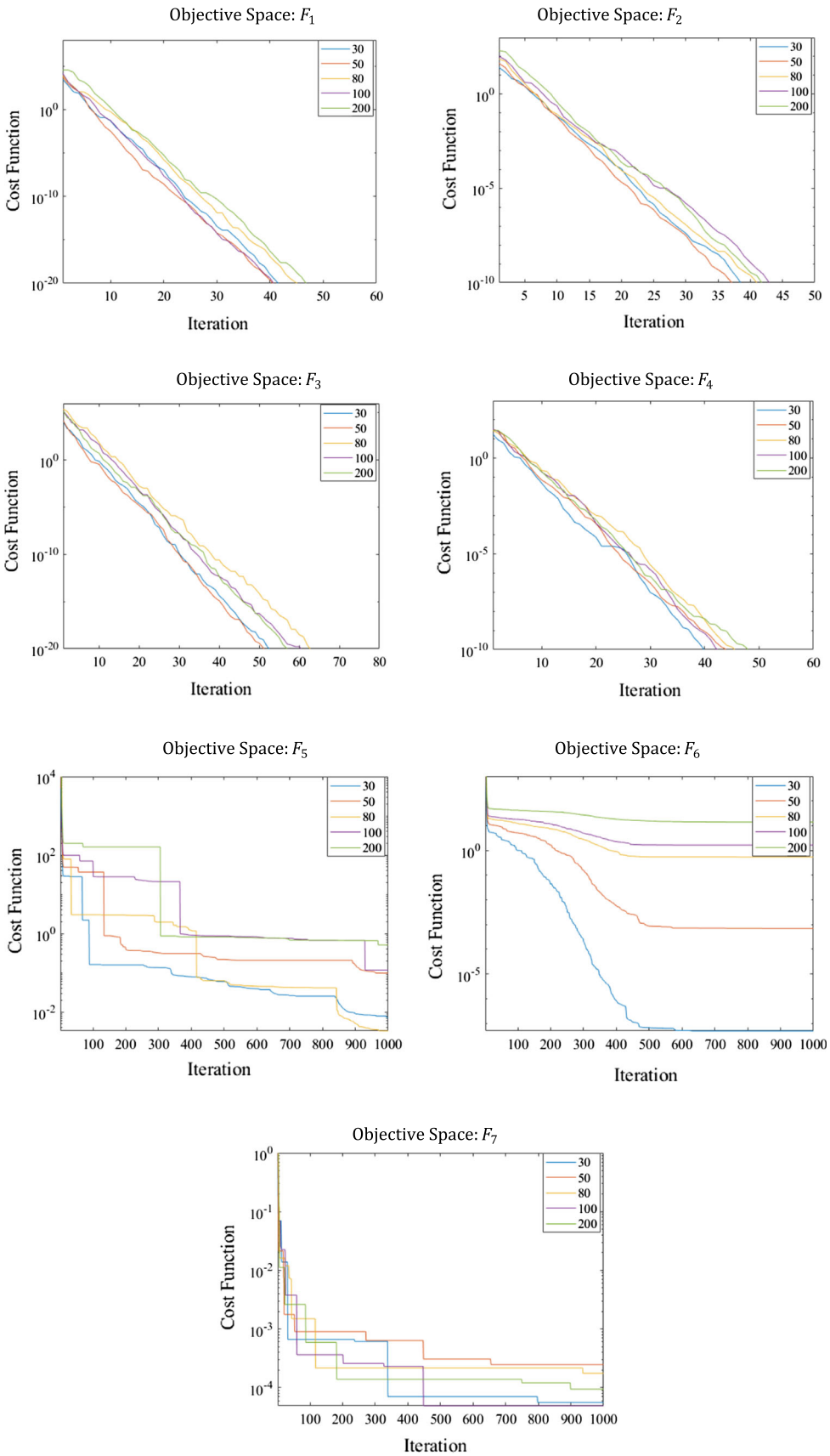

Fig. 3 Convergence plots of unimodal test functions under varying dimension

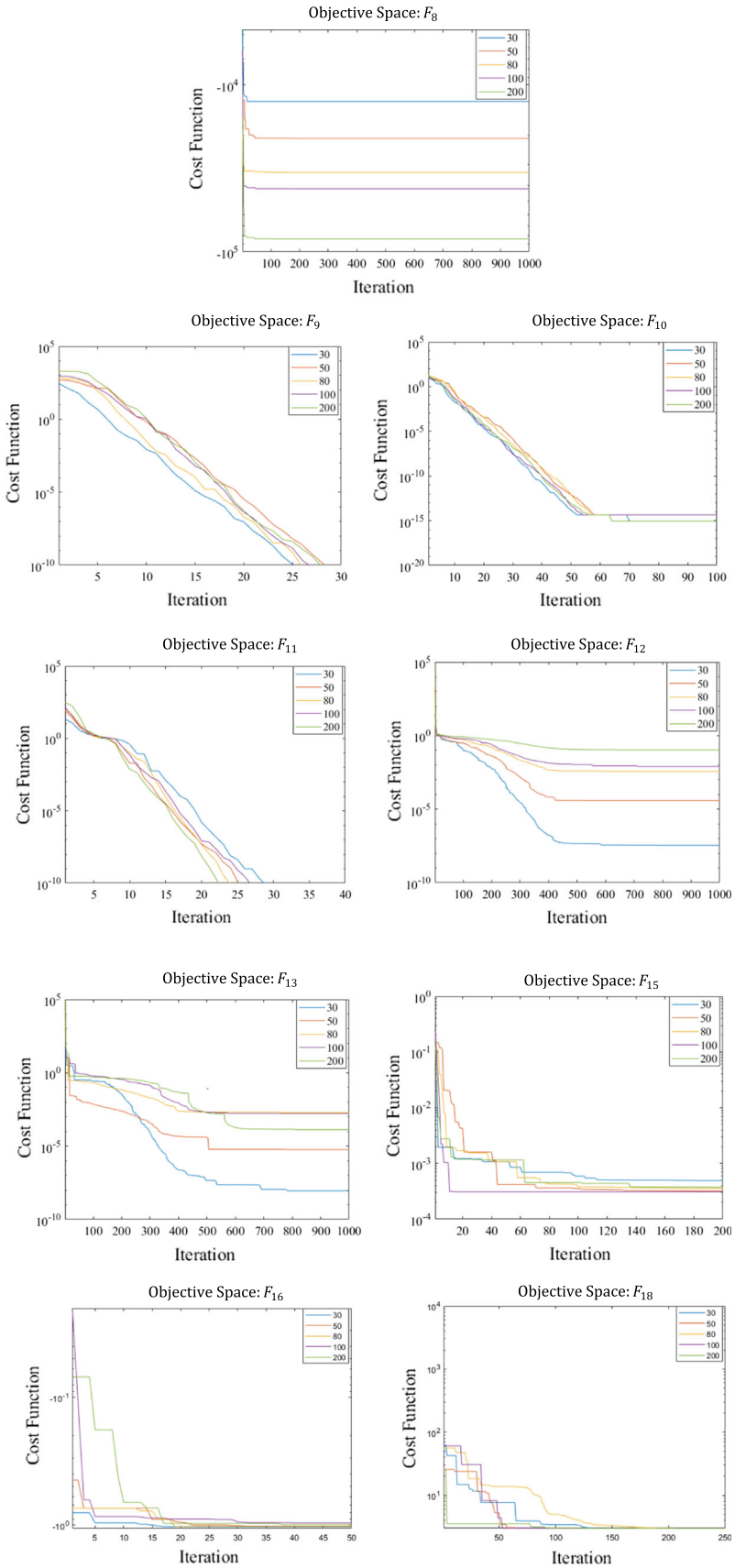

Fig. 4 Convergence plots of multimodal test functions under varying dimension

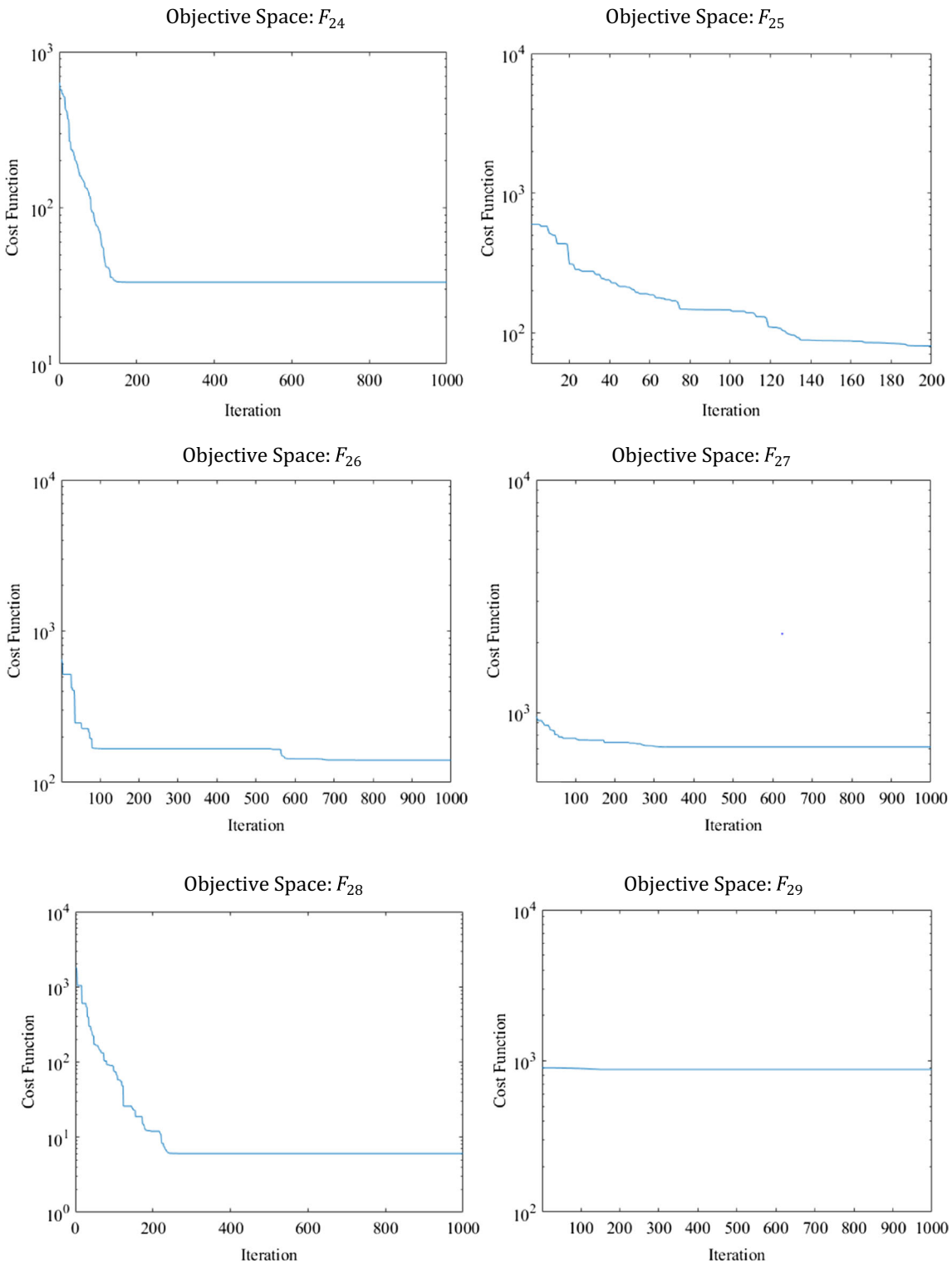

Fig. 5 Convergence plots of composite benchmark functions

Table 9 Comparison of the best solution for pressure vessel design with other algorithms

Method → Parameter ↓	LCBO	GA	CEPSO	CEDE	PSO	NIDP	SFS
T_s	1.2569	0.8125	0.8125	0.8125	0.8125	1.125	0.7781
T_h	0.6187	0.4375	0.4375	0.4375	0.4375	0.625	0.3846
R	65.1248	42.097	42.0913	42.0984	42.0984	48.3807	40.319
L	10.433	176.65	176.74	176.74	176.636	11.744	199.99
$g1$	− 3.21E−06	− 2.00E−05	− 1.37E−06	− 6.67E−07	− 8.80E−07	− 0.1913	− 1.11E−16
$g2$	− 7.82E−07	− 0.035891	− 3.59E−04	− 3.58E−02	− 0.0359	− 0.1634	− 1.11E−16
$g3$	− 3.9067	− 27.88607	− 118.7687	− 3.70512	3.1227	− 75.875	0.00E+00
$g4$	− 229.5667	− 63.34595	− 63.2535	− 63.3623	− 63.3634	128.255	− 40.00
Cost function	5.32E+03	6059.9463	6061.0777	6059.734	6059.7143	8048.619	5885.3327

Table 10 Statistical analysis of cost function values of various algorithms for pressure vessel design problem

Method → Parameter ↓	LCBO	GA	CEPSO	CEDE	PSO	NIDP	SFS
Best	5.32E+03	6059.9463	6061.0777	6059.734	6059.7143	8048.619	5885.332
Mean	5.35E+03	6177.2533	6147.1332	6085.2303	6066.0311	NA	5885.332
Worst	5.48E+03	6469.322	6363.8041	6371.0455	NA	NA	5885.332
Standard deviation	37.3703	130.9297	86.45	43.013	12.2718	NA	0
FE	12,000	80,000	240,000	27,500	60,000	NA	24,000

Table 11 Comparison of the LCBO solution for cantilever beam design with other algorithms

Parameter → Optimization Technique ↓	x_1	x_2	x_3	x_4	x_5	Optimum weight
LCBO	6.02367049	5.30083455	4.49769073	3.4892855	2.1558503	1.339959
MFO	5.9848	5.3167	4.4973	3.5136	2.1616	1.339988
MMA	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA_1	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA_2	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
CSA	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
SOS	6.01878	5.30344	4.4958	3.4989	2.1556	1.33996

Based on the detailed investigations presented in both the case studies reported above, it can be clearly concluded that LCBO has performed extremely well.

6 Conclusion

In this work, a life choice-based optimizer (LCBO) has been proposed and investigated. LCBO essentially makes use of the fundamental choices humans make in life to sort priorities and always move ahead to improve and achieve life objectives. The proposed LCBO algorithm has been described, and its performance has been assessed for exploration and exploitation on several benchmark functions. The functions used included varieties such as unimodal, multimodal and composite CEC-2005 benchmark functions. Detailed investigations on scalability and convergence were conducted and presented. Additionally, application of the LCBO algorithm on two important practical engineering problems was also investigated. The performance comparison between LCBO and other popular algorithms clearly revealed the superiority of LCBO over other algorithms in dealing with different optimization problems.

Overall, based on the presented investigations, it is concluded that LCBO is a competent algorithm which can compete with recent algorithms such as Spotted Hyena Optimizer, Moth Flame Optimizer, The Ant Lion Optimizer and Grey Wolf Optimizer as well as the standard algorithms like Particle Swarm Optimization and Genetic

Algorithm. For future research work, several development options are available such as multiobjective form of LCBO and binary version of the algorithm which are a big possibility and application of this algorithm for various different fields related to optimization and parameter determination could be looked into.

Acknowledgements The author thanks the anonymous reviewers who contributed to improving the quality and clarity of this paper with their comments during the revision process. MATLAB code of LCBO may be provided to the researchers and developers on request.

Compliance with ethical standards

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

Appendix

The engineering problems used in this paper are pressure vessel design and cantilever beam design. The mathematical details of these engineering problems have been presented. The mathematical equations of constraints, range space and cost function to be minimized are given below.

Pressure vessel design

The objective of this problem is to minimize the total cost consisting of material, forming and welding of a cylindrical vessel as in Fig. 6. Both ends of the vessel are capped, and the head has a hemispherical shape. There are four variables in this problem, namely thickness of the shell (T_s), thickness of the head (T_h), inner radius (R) and length of the cylindrical section without considering the head (L). The function $f(T_s, T_h, R, L)$ is to be minimized subjected to the following four constraints $g1$, $g2$, $g3$ and $g4$ and variable ranges:

$$f(T_s, T_h, R, L) = 0.6224\ T_s RL + 1.7781\ T_h R^2 + 3.1661\ \mathrm{T}_s^2 L + T_h + 19.84\ T_h^2 L$$

$$g1 = -T_h + 0.0193\ R \leq 0$$

$$g2 = -T_h + 0.0095\ R \leq 0$$

$$g3 = -\pi R^2 L - \frac{4}{3}\pi R^3 + 1296000 \leq 0$$

$$g4 = L - 240 \leq 0$$

$$1 * 0.0625 \leq T_s, T_h \leq 99 * 0.0625 \text{ and } 10 \leq R,\ L \leq 200$$

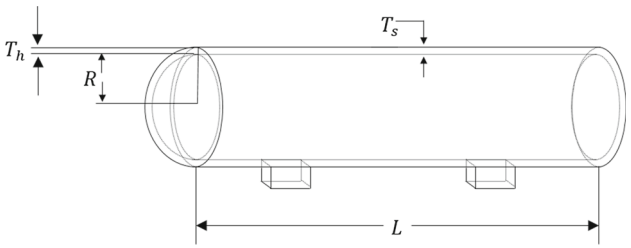

Fig. 6 Pressure vessel design problem

Cantilever beam design

The cantilever beam shown in Fig. 7 is made of five elements, each having a hollow cross section with constant thickness. There is external force acting at the free end of the cantilever. The weight of the beam is to be minimized while assigning an upper limit on the vertical displacement of the free end. The design variables are the heights (or widths) x_i of the cross section of each element. Another interesting requirement is the lower bounds on these design variables are very small and the upper bounds very large so they do not become active in the problem. The problem is formulated using classical beam theory as follows:

$$f(x) = 0.0624 * (x_1 + x_2 + x_3 + x_4 + x_5)$$

subjected to the following constraint and range of variables:

$$g(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0$$

$$0.01 \leq \mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, x_4, x_5 \leq 100$$

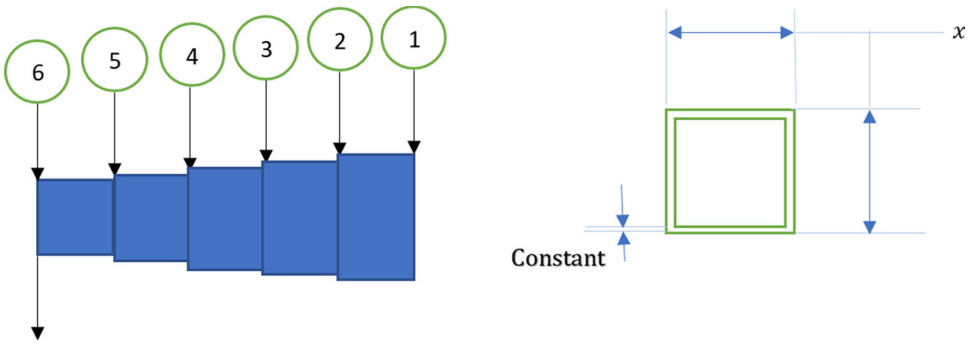

Fig. 7 Cantilever beam design problem

References

Abbass H (2002) MBO: marriage in honey bees optimization-a Haplometrosis polygynous swarming approach. Inst Electr Electron Eng 1:207–214

Abedinpourshotorban H, Mariyam Shamsuddin S, Beheshti Z, Jawawi D (2016) Electromagnetic field optimization: a physics-inspired metaheuristic optimization algorithm. Swarm Evol Comput 26:8–22

Ahrari A, Atai A (2010) Grenade explosion method—a novel tool for optimization of multimodal functions. Appl Soft Comput J 10(4):1132–1140

Alatas B (2011) ACROA: artificial chemical reaction optimization algorithm for global optimization. Expert Syst Appl 38(10):13170–13180

Askarzadeh A (2014) Bird mating optimizer: an optimization algorithm inspired by bird mating strategies. Commun Nonlinear Sci Numer Simul 19(4):1213–1228

Atashpaz-Gargari E, Lucas C (2007) Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: IEEE congress on evolutionary computation (CEC 2007), pp 4661–4667

Baluja S (1994) Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning. https://www.ri.cmu.edu/pub_files/pub1/baluja_shumeet_1994_2/baluja_shumeet_1994_2.pdf. Accessed 2 June 1994

Bouchekara H (2017) Most valuable player algorithm: a novel optimization algorithm inspired from sport. Oper Res 1–57

Cheng M, Prayogo D (2014) Symbiotic organisms search: a new metaheuristic optimization algorithm. Comput Struct 139:98–112

Chickermane H, Gea H (1996) Structural optimization using a new local approximation method. Int J Numer Methods Eng 39(5):829–846

Chu S-C, Tsai P-W, Pan J-S (2006) Cat swarm optimization. PRICAI 2006: trends in artificial intelligence. PRICAI 2006. Lect Notes Comput Sci 4099:854–858

Coello C, Montes M (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Inform 16(3):193–203

Cuevas E, Cienfuegos M, Zaldívar D, Pérez-Cisneros M (2013) A swarm optimization algorithm inspired in the behavior of the social-spider. Expert Syst Appl 40(16):6374–6384

Cuevas E, Echavarría A, Ramírez-Ortegón M (2014) An optimization algorithm inspired by the states of matter that improves the balance between exploration and exploitation. Appl Intell 40(2):256–272

Dai C, Zhu Y, Cheng W (2006) Seeker optimization algorithm. Computational intelligence and security. CIS 2006. Lect Notes Comput Sci 4456:167–176

Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70

Dorigo M, Di Caro G (1999) Ant colony optimization: A new metaheuristic. In: Proceedings of the 1999 congress on evolutionary computation-CEC99, pp 1470–1477

Duman E, Uysal M, Alkaya A (2012) Migrating birds optimization: a new metaheuristic approach and its performance on quadratic assignment problem. Inf Sci 217:65–77

Eita M, Fahmy M (2014) Group counseling optimization. Appl Soft Comput J 22:585–604

Erol O, Eksin I (2006) A new optimization method: big bang–big crunch. Adv Eng Softw 37(2):106–111

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 110–111:151–166

Farasat A, Menhaj M, Mansouri T, Moghadam M (2010) ARO: a new model-free optimization algorithm inspired from asexual reproduction. Appl Soft Comput J 10(4):1284–1292

Ferreira C (2006) Gene expression programming: mathematical modeling by an artificial intelligence. Studies in Computational Intelligence, vol. 21, Springer, Berlin

François O (1998) An evolutionary strategy for global minimization and its Markov chain analysis. IEEE Trans Evol Comput 2(3):77–90

Gandomi A (2014) Interior search algorithm (ISA): a novel approach for global optimization. ISA Trans 53(4):1168–1183

Gandomi A, Alavi A (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845

Gandomi A, Yang X, Alavi A (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29(1):17–35

Gao L, Hailu A (2010) Comprehensive learning particle swarm optimizer for constrained mixed-variable optimization problems. Int J Comput Intell Syst 3(6):832–842

Ghaemi M, Feizi-Derakhshi M (2014) Forest optimization algorithm. Expert Syst Appl 41(15):6676–6687

Ghorbani N, Babaei E (2014) Exchange market algorithm. Appl Soft Comput J 19:177–187

Glover F (1989) Tabu search—part I. ORSA J Comput 1(3):190–206

Hasançebi O, Azad S (2015) Adaptive dimensional search: a new metaheuristic algorithm for discrete truss sizing optimization. Comput Struct 154:1–16

Hatamlou A (2013) Black hole: a new heuristic optimization approach for data clustering. Inf Sci 222:175–184

Hatamlou A, Javidy B, Mirjalili S (2015) Ions motion algorithm for solving optimization problems. Appl Soft Comput 32:72–79

He S, Wu Q, Saunders J (2009) Group search optimizer: an optimization algorithm inspired by animal searching behavior. IEEE Trans Evol Comput 13(5):973–990

Hedayatzadeh R, Salmassi F, Keshtgari M, Akbari R, Ziarati K (2010) Termite colony optimization: a novel approach for optimizing continuous problems. In: Proceedings—2010 18th Iranian conference on electrical engineering, ICEE, pp 553–558

Holland J (1992) Adaptation in natural and artificial systems. The MIT Press, Cambridge

Hosseini H (2009) The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. Int J Bio Inspired Comput 1(1/2):71–79

Hosseini H (2011) Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation. Int J Comput Sci Eng 6(1/2):132–140

Huang F, Wang L, He Q (2007) An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 186(1):340–356

Husseinzadeh Kashan A (2014) A new metaheuristic for optimization: optics inspired optimization (OIO). Comput Oper Res 55:99–125

Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39(3):459–471

Kaveh A (2014) Advances in metaheuristic algorithms for optimal design of structures. Springer, Berlin

Kaveh A, Bakhshpoori T (2016) Water evaporation optimization: a novel physically inspired optimization algorithm. Comput Struct 167:69–85

Kaveh A, Dadras A (2017) A novel meta-heuristic optimization algorithm: thermal exchange optimization. Adv Eng Softw 110:69–84
Kaveh A, Farhoudi N (2013) A new optimization method: dolphin echolocation. Adv Eng Softw 59:53–70
Kaveh A, Ilchi Ghazaan M (2017) Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints. Acta Mech 228(1):307–322
Kaveh A, Mahdavi V (2014) Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 139:18–27
Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3–4):267–289
Kaveh A, Zolghadr A (2016) A novel meta-heuristic algorithm: tug of war optimization. Int Journal of Optim Civil Eng 6(4):469–492
Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95—international conference on neural networks, pp 1942–1948
Kiran M (2015) TSA: tree-seed algorithm for continuous optimization. Expert Syst Appl 42(19):6686–6698
Koza JR (1994) Genetic programming II: automatic discovery of reusable subprograms. MIT Press, Cambridge
Krishnanand KN, Ghose D (2006) Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. Multiagent and Grid Systems 2(3):209–222
Krohling R, Dos Santos Coelho L (2006) Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems. IEEE Trans Syst Man Cybern B Cybern 36(6):1407–1416
Lam A, Li V (2010) Chemical-reaction-inspired metaheuristic for optimization. IEEE Trans Evol Comput 14(3):381–399
Li X, Zhang J, Yin M (2014) Animal migration optimization: an optimization algorithm inspired by animal migration behavior. Neural Comput Appl 24(7–8):1867–1877
Liang J, Suganthan P, Deb K (2005) Novel composition test functions for numerical global optimization. Swarm Intell Symp 2005:68–75
Lu X, Zhou Y (2008) A novel global convergence algorithm: bee collecting pollen algorithm. Advanced intelligent computing theories and applications. With aspects of artificial intelligence. ICIC 2008. Lect Notes Comput Sci 5227:518–525
Mehrabian A, Lucas C (2006) A novel numerical optimization algorithm inspired from weed colonization. Ecol Inform 1(4):355–366
Meng X, Liu Y, Gao X, Zhang H (2014) A new bio-inspired algorithm: chicken swarm optimization. Advances in swarm intelligence. ICSI 2014. Lect Notes Comput Sci 8794:86–94
Merrikh-Bayat F (2015) The runner-root algorithm: a metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature. Appl Soft Comput J 33:292–303
Mirjalili S (2015a) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 89:228–249
Mirjalili S (2015b) The ant lion optimizer. Adv Eng Softw 83:80–98
Mirjalili S (2016a) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073
Mirjalili S (2016b) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133
Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
Mirjalili S, Mirjalili S, Lewis A (2014) Grey Wolf Optimizer. Adv Eng Softw 69:46–61
Mirjalili S, Mirjalili S, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513
Mirjalili S, Gandomi A, Mirjalili S, Saremi S, Faris H, Mirjalili S (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
Moghaddam F, Moghaddam R, Cheriet M (2012) Curved space optimization: a random search based on general relativity theory
Moghdani R, Salimifard K (2018) Volleyball premier league algorithm. Appl Soft Comput J 64:161–185
Moosavian N, Kasaee Roodsari B (2014) Soccer league competition algorithm: a novel meta-heuristic algorithm for optimal design of water distribution networks. Swarm Evol Comput 17:14–24
Oftadeh R, Mahjoob M, Shariatpanahi M (2010) A novel meta-heuristic optimization algorithm inspired by group hunting of animals: hunting search. Comput Math Appl 60(7):2087–2098
Pan W-T (2012) A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowl Based Syst 26:69–74
Pham D, Eldukhri E, Soroka A, Ghanbarzadeh A, Kog E, Otri S, Zaidi M (2006) The bees algorithm-a novel tool for complex optimisation problems. In: Intelligent production machines and systems, pp 454–459
Pinto P, Runkler TA, Sousa JM (2005) Wasp swarm optimization of logistic systems. In: Ribeiro B, Albrecht RF, Dobnikar A, Pearson DW, Steele NC (eds) Adaptive and natural computing algorithms. Springer, Vienna, pp 264–267
Rao RV, Vimal JS, Vakharia DP (2007) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315
Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248
Ryan C, Collins J (1998) Grammatical evolution: evolving programs for an arbitrary language. Genet Program Lect Notes Comput Sci 1391:83–96
Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput J 13(5):2592–2612
Salimi H (2015) Stochastic fractal search: a powerful metaheuristic algorithm. Knowl-Based Syst 75:1–18
Sandgren E (1990) Nonlinear integer and discrete programming in mechanical design optimization. J Mech Des 112(2):223–229
Shareef H, Ibrahim A, Mutlag A (2015) Lightning search algorithm. Appl Soft Comput J 36:315–333
Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–349
Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL report, 2005005
Svanberg K (1987) The method of moving asymptotes—a new method for structural optimization. Int J Numer Methods Eng 24:359–373
Tan Y, Zhu Y (2015) Fireworks algorithm for optimization. Advances in swarm intelligence. ICSI 2010. Lect Notes Comput Sci 6145:355–364
Van Laarhoven PJM, Aarts EHL (1987) Simulated annealing. In: Simulated annealing: theory and applications. Springer, Dordrecht, pp 7–15
Venkata RR (2016) Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int J Ind Eng Comput 7(1):19–34

Wang G, Deb S, Coelho L (2016a) Elephant herding optimization. In: Proceedings—2015 3rd international symposium on computational and business intelligence, ISCBI 2015, pp 1–5

Wang G, Zhao X, Deb S (2016b) A novel monarch butterfly optimization with greedy strategy and self-adaptive. In: Proceedings—2015 2nd international conference on soft computing and machine intelligence, ISCMI 2015, pp 45–50

Wolpert D, Macready W (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82

Woo Geem Z, Hoon Kim J, Loganathan G (2001) A new heuristic optimization algorithm: harmony search. Simul Trans Soc Model Simul Int 76(2):60–68

Sophia Robot. https://www.hansonrobotics.com/sophia/

Yang X-S (2009) Firefly algorithms for multimodal optimization. Stochastic algorithms: foundations and applications. SAGA 2009. Lect Notes Comput Sci 5792:169–178

Yang X-S (2010) A new metaheuristic Bat-inspired Algorithm. Stud Comput Intell 284:65–74

Yang X-S (2012) Flower pollination algorithm for global optimization. Unconventional computation and natural computation. UCNC 2012. Lect Notes Comput Sci 7445:240–249

Yang X-S, Deb S (2009) Cuckoo search via levy flights. In: 2009 World congress on nature & biologically inspired computing (NaBIC), pp 210–214

Yang S, Jiang J, Yan G (2009) A dolphin partner optimization. In: Proceedings of the 2009 WRI global congress on intelligent systems, GCIS 2009, vol. 1, pp. 124–128

Yao X, Liu Y (1996) Fast evolutionary programming. Computational intelligence and intelligent systems. ISICA 2010. Commun Comput Inf Sci 107:79–86

Yazdani M, Jolai F (2016) Lion optimization algorithm (LOA): a nature-inspired metaheuristic algorithm. J Comput Des Eng 3(1):24–36

Zhao R, Tang W (2008) Monkey algorithm for global numerical optimization. J Uncertain Syst 2(3):165–176

Zheng Y (2015) Water wave optimization: a new nature-inspired metaheuristic. Comput Oper Res 55:1–11

Zheng Y, Ling H, Xue J (2014) Ecogeography-based optimization: enhancing biogeography-based optimization with ecogeographic barriers and differentiations. Comput Oper Res 50:115–127