Information Sciences 258 (2014) 54–79

Contents lists available at ScienceDirect

Information Sciences

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

CAPSO: Centripetal accelerated particle swarm optimization

Zahra Beheshti, Siti Mariyam Hj. Shamsuddin *

Soft Computing Research Group, Faculty of Computer Science and Information System, Universiti Teknologi Malaysia, Skudai, 81310 Johor, Malaysia

ARTICLE INFO

Article history:
Received 23 December 2011
Received in revised form 12 June 2013
Accepted 9 August 2013
Available online 19 August 2013

Keywords:
Optimization
Meta-heuristic search algorithms
Swarm intelligence
Particle swarm optimization
Centripetal accelerated particle swarm optimization
Binary centripetal accelerated particle swarm optimization

ABSTRACT

Meta-heuristic search algorithms are developed to solve optimization problems. Such algorithms are appropriate for global searches because of their global exploration and local exploitation abilities. Swarm intelligence (SI) algorithms comprise a branch of meta-heuristic algorithms that imitate the behavior of insects, birds, fishes, and other natural phenomena to find solutions for complex optimization problems. In this study, an improved particle swarm optimization (PSO) scheme combined with Newton's laws of motion, the centripetal accelerated particle swarm optimization (CAPSO) scheme, is introduced. CAPSO accelerates the learning and convergence of optimization problems. In addition, the binary mode of the proposed algorithm, binary centripetal accelerated particle swarm optimization (BCAPSO), is introduced for binary search spaces. These algorithms are evaluated using nonlinear benchmark functions, and the results are compared with the gravitational search algorithm (GSA) and PSO in both the real and the binary search spaces. Moreover, the performance of CAPSO in solving the functions is compared with some well-known PSO algorithms in the literature. The experimental results showed that the proposed methods enhance the performance of PSO in terms of convergence speed, solution accuracy and global optimality.

1. Introduction

Exact optimization algorithms do not efficiently solve optimization problems that have a high-dimensional search space because the search space grows exponentially with the problem size, making an exhaustive search impractical [38]. In recent decades, many meta-heuristic algorithms have been designed to solve problems with high dimensions. A branch of these algorithms that mimics the behavior of natural phenomena is known as swarm intelligence (SI).

Ant colony optimization (ACO) [15], artificial bee colony (ABC) [24], particle swarm optimization (PSO) [26,52], stochastic diffusion search (SDS) [6], artificial immune system (AIS) [17], gravitational search algorithm (GSA) [45], intelligent water drops (IWD) [51], river formation dynamics (RFD) [44] and charged system search (CSS) [25] are in the class of SI algorithms. These algorithms have been analysed over time by researchers in various areas [5,10,16,35]. They have shown good performance in a wide range of problems, such as neural network learning [42,43], combinatorial problems [4,48], image and video processing [13,14,19,59], function optimization [3,30], data clustering [18,54], and pattern recognition [9,65].

Although the aforementioned algorithms have obtained satisfactory results, there are still some disadvantages in their utilization. For example, PSO can sometimes find local optima or exhibit slow convergence speed; GSA and CSS take long computational time to obtain results. Furthermore, some of the algorithms require several parameters to tune, and setting the parameters can be challenging for some optimization problems. None of the meta-heuristic search algorithms are capable of offering adequately high performance to solve all optimization problems in comparison with other alternatives.

In addition, many optimization problems are expressed in a binary representation. That is, some solutions are encoded in binary form or some problems are binary in nature. For example, cell formation [31,58], data compression [34,55], feature

* Corresponding author. Tel.: +60 123710679; fax: +60 7 5532215.
E-mail addresses: bzahra2@live.utm.my (Z. Beheshti), mariyam@utm.my (S.M. Hj. Shamsuddin).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.08.015

selection and dimensionality reduction [1,12], unit commitment [49,62] and image compression [11,37] are in the former group; problems such as max-ones and royal-road are in the latter group. Therefore, an optimizer that operates on two-valued functions, i.e., in both the real and the binary search spaces, would be beneficial.

In this study, a new optimization algorithm is proposed based on Newton's laws of motion and the PSO algorithm, the centripetal accelerated particle swarm optimization (CAPSO) algorithm, for both the real and binary high-dimensional search spaces. The proposed methods are compared with PSO and GSA in both search spaces and with some well-known PSO algorithms in the real search space.

The remainder of this paper is organized as follows. A brief review of related works is presented in Section 2. Details of the proposed method and the binary form of CAPSO are presented in Section 3. In Section 4, the experimental results for the nonlinear benchmark functions are presented in both the real and the binary search spaces. Section 5 contains the summary and concluding remarks.

2. Related works

In recent decades, the use of SI algorithms in the sciences has dramatically increased because of their capability to solve a variety of problems. These algorithms are composed of simple particles or agents interacting locally with each other and with their environment [7]. Every particle follows one or several rules without any centralized structure to control its behavior. Consequently, local and random interactions among the particles lead to an intelligent global behavior. These algorithms are in the class of population-based meta-heuristic search algorithms that apply two approaches to obtain good performance [56]: global exploration and local exploitation.

Exploration represents the power of expanding search space, whereas exploitation is the ability to find the best solution around a good solution. Because of the insufficient knowledge of the search space in the initial steps, more exploration is performed by the swarm to avoid entrapment by local optima. In contrast, more exploitation is required as the steps proceed so that the algorithm can tune itself around semi-optimal points [45]. In other words, an appropriate trade-off between exploration and exploitation is necessary to achieve high performance searching. To realize the concepts of exploration and exploitation, particles pass three phases inspired from nature in every step, i.e., self-adaptation, cooperation and competition. In the self-adaptation phase, every particle enhances its performance. Particles collaborate by transferring information in the cooperation phase and, finally, they compete to survive in the competition phase. These concepts direct an algorithm to find good solutions.

Regarding these concepts, Farmer et al. [17] proposed the artificial immune system (AIS), inspired by the structure and function of the biological immune system, to solve computational problems. In 1995, particle swarm optimization (PSO) [26,52] was suggested by Kennedy and Eberhart. PSO simulates a flock of birds or insect motion. Another SI algorithm is the ant colony optimization (ACO) [15] algorithm, introduced in 1996. This algorithm models the behavior of foraging ants and is applied to find the shortest path on a graph. In 2005, Karaboga presented the artificial bee colony (ABC) algorithm [24] that mimics the intelligent behavior of honey bees to find a good solution for optimization problems. Shah-Hosseini [51] proposed the intelligent water drops (IWD) algorithm, inspired by the behavior of water drops in natural rivers, to find the shortest path. River formation dynamics (RFD) [44] is another nature-inspired optimization algorithm and is a gradient version of ant colony optimization (ACO). The algorithm copies the behavior of water in forming rivers. The gravitational search algorithm (GSA) [45], introduced by Rashedi et al. in 2009, was motivated by Newton's law of gravity and mass interactions. In the algorithm, agents attract each other by a gravitational force and move towards agents with larger masses (better solutions) to obtain good results. Additionally, based on Newton's laws and the governing Coulomb and Gauss laws from electrostatics, a new SI algorithm, known as charged system search (CSS) [25], was developed by Kaveh and Talatahari in 2010 to solve optimization problems.

Although these algorithms solve different optimization problems, according to the theorem of "no free lunch" [57], no algorithm is able to perform better than the others to solve all problems. Hence, the search to find new meta-heuristic optimization algorithms is still an open problem.

In this study, the goal is to establish an SI algorithm based on Newton's laws of motion and the PSO algorithm in both the real and the binary search spaces. Hence, a brief overview of PSO and GSA in the search spaces is provided in the next sections to provide a background for the proposed methods.

2.1. Meta-heuristic algorithms in the real search space

The majority of meta-heuristic algorithms have been designed for search spaces containing real valued vectors. In this section, PSO and GSA are described in the real search space.

2.1.1. PSO in the real search space

PSO is a global optimization algorithm in which every feasible solution can be represented as a point or surface in a multi-dimensional search space. In the algorithm, the birds or insects, i.e., particles, are evaluated by their fitness values. They move towards those particles that have better fitness values to obtain good solutions.

The algorithm is initialised by creating a swarm, i.e., population of particles, with random positions and velocities [26,52]. Every particle is described by a group of vectors denoted as $(\overrightarrow{X}_i, \overrightarrow{V}_i, \overrightarrow{P}_i)$ in a *d-dimensional* search space, where $\overrightarrow{X}_i$ and $\overrightarrow{V}_i$ are the position and velocity of the *i*th particle defined as follows:

$$\overrightarrow{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id}) \quad \text{for } i = 1, 2, \ldots, N \text{ and} \tag{1}$$

$$\overrightarrow{V}_i = (v_{i1}, v_{i2}, \ldots, v_{id}) \quad \text{for } i = 1, 2, \ldots, N. \tag{2}$$

$\overrightarrow{P}_i$ is the personal best position found by the *i*th particle and is defined as follows:

$$\overrightarrow{P}_i = (p_{i1}, p_{i2}, \ldots, p_{id}) \quad \text{for } i = 1, 2, \ldots, N. \tag{3}$$

The best position obtained by the entire population $(\overrightarrow{P}_g)$ is computed to update the particle velocity as follows:

$$\overrightarrow{P}_g = (p_{g1}, p_{g2}, \ldots, p_{gd}). \tag{4}$$

From $\overrightarrow{P}_i$ and $\overrightarrow{P}_g$, the next velocity and position of the *i*th particle are updated using Eqs. (5) and (6) as follows:

$$v_{id}(t+1) = w(t) \times v_{id}(t) + C_1 \times rand \times (p_{id}(t) - x_{id}(t)) + C_2 \times rand \times (p_{gd}(t) - x_{id}(t)) \text{ and} \tag{5}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \tag{6}$$

where $v_{id}(t+1)$ and $v_{id}(t)$ are the next and current velocity of the *i*th particle, respectively, w is the inertial weight, C_1 and C_2 are acceleration coefficients, *rand* is a uniformly random number in the interval [0,1], and N is the number of particles. $x_{id}(t+1)$ and $x_{id}(t)$ are the next and current position of the *ith* particle. Additionally, $|v_{id}(t+1)| < v_{max}$ and v_{max} is set to a constant, based on the bounds of the solution space set by the user.

A larger value of w encourages global exploration (searching new areas); a smaller inertial weight promotes local exploitation [52]. In the algorithm, the two models applied to choose $\overrightarrow{P}_g$ are known as *gbest* (for global topology) and *lbest* (for local topology) models. In the global model, the position of each particle is influenced by the best-fitness particles of the entire population in the search space; in the local model, each particle is affected by the best-fitness particles in its neighborhood. According to Bratton and Kennedy [8], the *lbest* model can return better results than the *gbest* model for many problems; however; it may have a lower convergence rate than the *gbest* model. In this study, the PSO algorithm that utilizes the *lbest* model, LPSO, is used. In Eq. (5), the second term is the cognition term and the third term is the social term.

Although PSO is easy to implement, it may converge prematurely, becoming easily trapped into local optima because of its poor exploration when solving complex multimodal problems. Hence, many studies have been performed through parameter selection [22,47], integration of its self-adaptation [23,60] and swarm topology to improve the performance of PSO [2].

Kennedy and Mendes proposed a ring topological structure PSO (LPSO) [28] and a Von Neumann topological structure PSO (VPSO) [29] to avoid premature convergence when solving multimodal problems. Mendes et al. suggested the fully informed particle swarm (FIPS) algorithm [36], which applied information from the entire neighborhood to guide the particles. Dynamic multi-swarm PSO (DMS-PSO) [33] was introduced by Liang and Suganthan to dynamically enhance the topological structure. The quadratic interpolation PSO (QIPSO) algorithm [40] introduced a quadratic crossover operator to improve numerical results. Ratnaweera et al. [47] proposed the HPSO-TVAC algorithm, which used linearly time-varying acceleration coefficients, using a larger C_1 and a smaller C_2 set at the beginning, gradually reversing their relationship throughout the search. In another study, an adaptive fuzzy particle swarm optimization (AFPSO) [23] was proposed to utilize fuzzy inferences that adaptively adjust acceleration coefficients rather than fixed constants. Additionally, the quadratic crossover operator [40] has been used in the proposed AFPSO algorithm (AFPSO-QI) [23] to solve the multimodal functions algorithm. Parsopoulos and Vrahatis [41] integrated exploration and exploitation to form a unified particle swarm optimization (UPSO). Moreover, Liang et al. [32] presented comprehensive learning particle swarm optimization (CLPSO) that focused on avoiding the local optima by encouraging each particle to learn its behavior from other particles in different dimensions.

An adaptive particle swarm optimization (APSO) [63] was also proposed by Zhan et al., in which the algorithm applied a real-time evolutionary state estimation procedure and an elitist learning strategy. Zhan et al. [64] introduced the orthogonal learning particle swarm optimization (OLPSO) algorithm. The OL strategy guides a particle to discover useful information from its personal best position and from its neighborhood particles' best positions to move in better directions. In another study, Gao et al. [20] used PSO with chaotic opposition-based population initialisation and a stochastic search technique to solve complex multimodal problems. The algorithm, CSPSO, finds new solutions in the neighborhoods of the previous best positions to escape from local optima in multimodal functions. Neri et al. [39] proposed compact particle swarm optimization (cPSO) to address some real-world optimization problems that arise from limited hardware availability. The algorithm applies the search logic typical of PSO plus a probabilistic representation of the swarm's behavior. This representation allows a modest memory usage for the entire algorithmic calculation.

2.1.2. GSA in the real search space

GSA is a meta-heuristic algorithm based on Newton's laws of gravity and motion. According to the law of gravity, objects attract each other by gravitational force [21,50]. This force depends directly on the product of both objects' masses and is inversely proportional to the square of the distance between them [50].

Table 1
Unimodal test functions.

Test function	[Range]^n	F_{opt}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100,100]^n$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10,10]^n$	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100,100]^n$	0
$F_4(x) = \max_i\{\lvert x_i \rvert, 1 \leqslant i \leqslant n\}$	$[-100,100]^n$	0
$F_5(x) = \sum_{i=1}^{n-1} [(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30,30]^n$	0
$F_6(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	$[-100,100]^n$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	$[-1.28,1.28]^n$	0

Table 2
Multimodal high-dimensional test functions.

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

Table 3
Multimodal low-dimensional test functions.

Test function	[Range]^n	F_{opt}
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	$[-65.535,65.535]^2$	1
$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	$[-5,5]^2$	0.0003
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 + 4x_2^2 + 4x_2^4$	$[-5,5]^2$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	$[-5,10] \times [0,15]$	0.398
$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	$[-2,2]^2$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	$[0,1]^3$	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	$[0,1]^6$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	$[0,10]^4$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	$[0,10]^4$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	$[0,10]^4$	-10.5363

Table 4
Maximization function (max-ones) in the binary search space.

Test function	[Range]^n	F_{opt}
$F_{24}(x) = \sum_{i=1}^{n} x_i^2$	$[\{0,1\}]^n$	n

In GSA, the objects of the real world are considered agents. Agents are a collection of masses that interact with each other based on Newton's laws. A heavier mass is a more efficient agent. Each mass (agent) presents the solution of the problem in a search space. As time lapses, masses are attracted by the heaviest mass, which has a better solution. Based on this law, GSA defines a system with N agents in a *d-dimensional* search space and the position of the ith agent is as follows:

Table 5
Minimization results for the unimodal functions in Table 1 in the real search space (maximum iteration = 1000 and $n = 30$).

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
F1					
Avg. best solution	$\mathbf{3.46 \times 10^{-38}}$	2.12×10^{-17}	1.22×10^{-8}	3.96×10^{-14}	0.126
SD	$\mathbf{9.97 \times 10^{-38}}$	7.06×10^{-18}	5.47×10^{-8}	2.57×10^{-14}	0.078
Median Best solution	6.11×10^{-38}	1.97×10^{-17}	4.69×10^{-10}	3.37×10^{-14}	0.124
Avg. search time	1.5	7.8	1.7	1.9	2.1
F2					
Avg. best solution	$\mathbf{5.02 \times 10^{-20}}$	2.27×10^{-8}	10.33	9.28×10^{-9}	0.037
SD	$\mathbf{6.20 \times 10^{-20}}$	3.06×10^{-9}	8.51	3.28×10^{-9}	0.016
Median best solution	2.15×10^{-20}	2.22×10^{-8}	10.00	8.98×10^{-9}	0.035
Avg. search time	1.6	7.5	1.7	1.9	2.1
F3					
Avg. best solution	$\mathbf{9.34 \times 10^{-16}}$	238.99	$8.45 \times 10^{+3}$	3.80×10^{-3}	$2.99 \times 10^{+4}$
SD	$\mathbf{1.70 \times 10^{-15}}$	88.24	$6.38 \times 10^{+3}$	5.12×10^{-3}	$6.40 \times 10^{+3}$
Median best solution	3.9×10^{-16}	208.8	$6.90 \times 10^{+3}$	2.20×10^{-3}	$2.95 \times 10^{+4}$
Avg. search time	4.9	13.7	5.0	6.2	6.3
F4					
Avg. best solution	$\mathbf{5.06 \times 10^{-15}}$	3.20×10^{-9}	0.468	5.06×10^{-4}	8.90
SD	$\mathbf{5.28 \times 10^{-15}}$	5.24×10^{-10}	0.233	2.02×10^{-4}	1.97
Median best solution	2.86×10^{-15}	3.34×10^{-9}	0.392	4.07×10^{-4}	8.66
Avg. search time	1.5	7.6	1.7	1.9	2.0
F5					
Avg. best solution	**25.90**	26.13	$6.24 \times 10^{+3}$	26.82	174.53
SD	**0.56**	0.195	$2.28 \times 10^{+4}$	0.381	108.97
Median best solution	25.82	26.12	94.33	26.81	153.86
Avg. search time	1.8	8.0	2.1	4.3	4.4
F6					
Avg. best solution	3.77	**0**	**0**	**0**	**0**
SD	1.50	**0**	**0**	**0**	**0**
Median best solution	4	0	0	0	0
Avg. search time	0.8	2.4	1.3	1.4	3.8
F7					
Avg. best solution	**0.012**	0.026	0.212	0.048	0.224
SD	$\mathbf{3.34 \times 10^{-3}}$	0.011	0.678	0.016	0.08
Median best solution	0.012	0.023	0.036	0.046	0.21
Avg. search time	1.8	3.0	2.0	2.1	2.2
Avg. rank	**1.1**	2.1	3.4	2.4	4.1
Final rank	**1**	2	4	3	5

$$\vec{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id}) \quad \text{for } i = 1, 2, \ldots, N. \tag{7}$$

At the beginning, these positions are randomly initialised. Then, the force of gravity from mass j on mass i at a specific time t is computed as follows:

$$F_{ij,d}(t) = G(t)\frac{M_i(t) \times M_j(t)}{R_{ij}(t) + \varepsilon}(x_{jd}(t) - x_{id}(t)), \tag{8}$$

where M_i and M_j are the masses of agent i and agent j, respectively, $\in$ is a small constant and R_{ij}(t) is the Euclidean distance between probes i and j at time t as follows:

$$R_{ij}(t) = \|x_i(t), x_j(t)\|_2. \tag{9}$$

$G(t)$ is a gravitational constant that is initialised at the start and decreases with time, t, to control the search accuracy. That is, G is a function of the initial value, G_0, and time, t, as follows:

$$G(t) = G(G_0, t). \tag{10}$$

In Eq. (8), $M_i(t)$ is calculated as follows:

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \text{ and} \tag{11}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \tag{12}$$

where $fit_i(t)$ is the fitness value of the agent i at time t and $best(t)$ and $worst(t)$ are the best and the worst values of the fitness function at time t.

Table 6
Minimization results for the multimodal high-dimensional functions in Table 2 for the real search space (maximum iteration = 1000 and $n = 30$).

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
F8					
Avg. best solution	−7046	−2733.12	−9068.92	−7603.31	**−9419.74**
SD	416.77	391.82	843.08	438.70	501.00
Median best solution	−6950.93	−2696.14	−8871.66	−7632.04	−9410.88
Avg. search time	2.1	0.1	2.1	4.1	4.6
F9					
Avg. best solution	98.85	**16.48**	63.26	118.70	80.09
SD	28.90	**5.50**	20.15	20.41	17.68
Median best solution	94.00	15.42	61.23	119.62	76.89
Avg. search time	2.3	0.65	2.4	4.1	4.4
F10					
Avg. best solution	$\mathbf{2.56 \times 10^{-14}}$	3.20×10^{-9}	8.56×10^{-8}	2.23×10^{-8}	0.013
SD	$\mathbf{6.36 \times 10^{-15}}$	4.62×10^{-10}	1.11×10^{-9}	2.09×10^{-8}	6.77×10^{-3}
Median best solution	2.22×10^{-14}	3.19×10^{-9}	3.75×10^{-8}	1.78×10^{-8}	0.012
Avg. search time	1.9	10.6	2.4	4.5	4.3
F11					
Avg. best solution	0.011	3.99	0.014	$\mathbf{1.96 \times 10^{-3}}$	0.188
SD	0.012	1.73	0.015	$\mathbf{3.97 \times 10^{-3}}$	0.099
Median best solution	7.54×10^{-3}	3.5	7.40×10^{-3}	1.85×10^{-6}	0.177
Avg. search time	2.4	11.0	2.8	4.9	5.2
F12					
Avg. best solution	0.499	0.032	0.077	$\mathbf{8.99 \times 10^{-3}}$	3.31
SD	1.126	0.072	0.150	$\mathbf{2.61 \times 10^{-3}}$	1.45
Median best solution	0.064	1.43×10^{-19}	1.32×10^{-6}	8.56×10^{-3}	3.35
Avg. search time	4.1	12.7	4.3	7.9	8.2
F13					
Avg. best solution	0.190	$\mathbf{7.33 \times 10^{-4}}$	6.23×10^{-3}	0.116	8.74
SD	0.120	$\mathbf{2.79 \times 10^{-3}}$	0.018	0.017	4.61
Median best solution	0.168	2.12×10^{-18}	1.42×10^{-7}	0.111	8.61
Avg. search time	4.2	12.5	4.3	7.9	8.3
Avg. rank	3.2	**2.7**	**2.7**	**2.7**	3.8
Final rank	2	**1**	**1**	**1**	3

The total force acting on agent i at time t in dimension d is as follows:

$$F_{id}(t) = \sum_{j \in Kbest, j \neq i}^{N} rand_j \times F_{ij,d}(t), \tag{13}$$

where $rand_j$ is a random number in the range $[0,1]$ and $Kbest$ is the set of the first K agents with the best fitness value and the biggest mass. At the beginning, $Kbest$ is initialised at K_0 and linearly reduced step-by-step as time lapses. Regarding the law of motion, the force accelerates the agent i as follows:

$$a_{id}(t) = \frac{F_{id}(t)}{M_i(t)}. \tag{14}$$

This acceleration moves the agent from one position to another. Hence, the next velocity of agent i in dimension d is computed as its current velocity plus its acceleration as follows:

$$v_{id}(t+1) = v_{id}(t) + rand_i \times a_{id}(t). \tag{15}$$

The next position is calculated as follows:

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \tag{16}$$

2.2. Meta-heuristic algorithms in the binary search space

A binary search space can be represented as a hypercube within which a particle or an agent can move to its nearer and farther corners by flipping various numbers of the bits [27]. Each dimension has only a binary value of '0' or '1'. Hence, the particle velocity is the number of bits changed per iteration, or the Hamming distance between the particles at time t and $t + l$. The particle position is updated when a switch between '0' and '1' occurs. Based on these rules, binary versions of PSO and GSA have been proposed by Kennedy et al. [27] and Rashedi et al. [46], respectively. To provide a background of optimization in the binary search space, a brief overview of these algorithms is presented in the following subsections.

Table 7
Minimization results for the multimodal low-dimensional functions in Table 3 for the real search space (maximum iteration = 500).

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
F14					
Avg. best solution	**0.998**	4.61	**0.998**	**0.998**	**0.998**
SD	**0**	3.18	**0**	**0**	**0**
Median best solution	0.998	3.76	0.998	0.998	0.998
Avg. search time	1.9	0.6	2.1	2.4	2.6
F15					
Avg. best solution	4.53×10^{-3}	6.34×10^{-3}	1.01×10^{-2}	$\mathbf{4.02 \times 10^{-4}}$	1.07×10^{-3}
SD	8.06×10^{-3}	2.30×10^{-3}	9.93×10^{-3}	$\mathbf{1.87 \times 10^{-4}}$	1.39×10^{-3}
Median best solution	3.08×10^{-4}	6.66×10^{-3}	1.66×10^{-3}	3.22×10^{-4}	7.83×10^{-4}
Avg. search time	1.1	1.8	1.2	2.9	3.0
F16					
Avg. best solution	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
SD	1.33×10^{-8}	$\mathbf{5.38 \times 10^{-16}}$	6.71×10^{-16}	1.85×10^{-9}	9.98×10^{-9}
Median best solution	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
Avg. search time	0.4	1.7	0.4	0.5	0.6
F17					
Avg. best solution	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**
SD	2.13×10^{-6}	**0**	**0**	2.15×10^{-5}	2.17×10^{-13}
Median best solution	0.398	0.398	0.398	0.398	0.398
Avg. search time	0.2	1.6	0.2	0.4	0.4
F18					
Avg. best solution	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
SD	3.74×10^{-8}	$\mathbf{3.06 \times 10^{-16}}$	1.21×10^{-15}	1.31×10^{-5}	4.62×10^{-15}
Median best solution	3.00	3.00	3.00	3.00	3.00
Avg. search time	0.5	1.8	0.5	0.6	0.6
F19					
Avg. best solution	**−3.86**	−3.64	**−3.86**	**−3.86**	**−3.86**
SD	3.2×10^{-6}	0.264	2.7×10^{-15}	5.99×10^{-5}	$\mathbf{2.36 \times 10^{-15}}$
Median best solution	−3.86	−3.82	−3.86	−3.86	−3.86
Avg. search time	0.5	1.2	0.6	0.8	0.8
F20					
Avg. best solution	**−3.2957**	−2.0540	−3.2185	−3.233	−3.2296
SD	**0.054**	0.589	0.137	0.097	0.094
Median best solution	−3.3218	−1.9777	−3.2626	−3.3183	−3.2031
Avg. search time	0.5	0.6	0.5	0.8	0.9
F21					
Avg. best solution	−6.9663	−5.0552	−5.8768	**−10.0858**	−9.9848
SD	3.5276	8.73×10^{-16}	3.02	0.123	0.922
Median best solution	−10.1146	−5.0552	−5.0780	−10.1318	−10.1532
Avg. search time	1.0	5.0	0.9	2.0	2.4
F22					
Avg. best solution	−6.8884	−7.5681	−8.1332	−10.3275	**−10.4028**
SD	3.65	2.70	3.10	0.092	$\mathbf{8.08 \times 10^{-16}}$
Median best solution	−7.7447	−5.0877	−10.4029	−10.3706	−10.4028
Avg. search time	0.8	4.8	0.9	2.1	2.5
F23					
Avg. best solution	−7.2685	−10.0858	−8.4866	−10.2636	**−10.3561**
SD	3.82	1.75	3.27	1.22	0.987
Median best solution	−10.5083	−10.5364	−10.5364	−10.5090	−10.5364
Avg. search time	1.2	5.2	1.0	2.6	3.0
Avg. rank	2.2	2.8	2.5	**1.3**	1.4
Final rank	3	5	4	**1**	2

2.2.1. PSO in the binary search space (BPSO)

BPSO [27] operates on discrete binary variables; therefore, the main difference between binary PSO and real PSO is that the particle position has two possible values, '0' or '1'. Therefore, the velocity of a particle in BPSO is calculated as in the PSO algorithm (Eq. (5)) and is transferred into a probability function in the interval of [0,1] to update the particle position using Eqs. (17) and (18) as follows:

$$S(v_{id}(t+1)) = \frac{1}{1+e^{-v_{id}(t+1)}}, \tag{17}$$

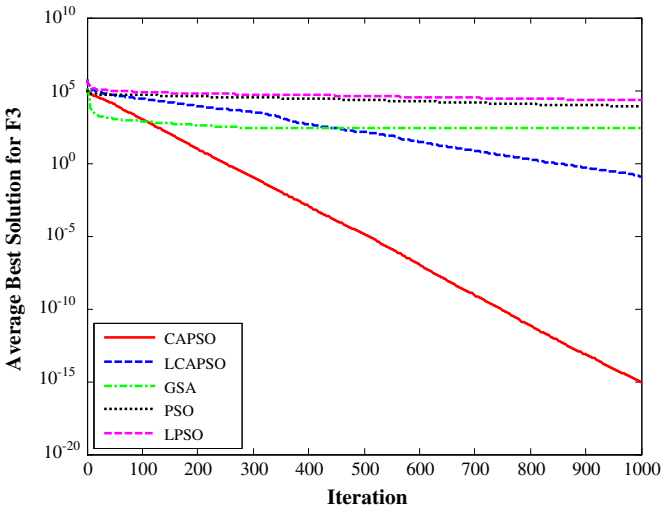


Fig. 1. Convergence performance of CAPSO, LCAPSO, GSA, PSO and LPSO for F_3 with $n = 30$.

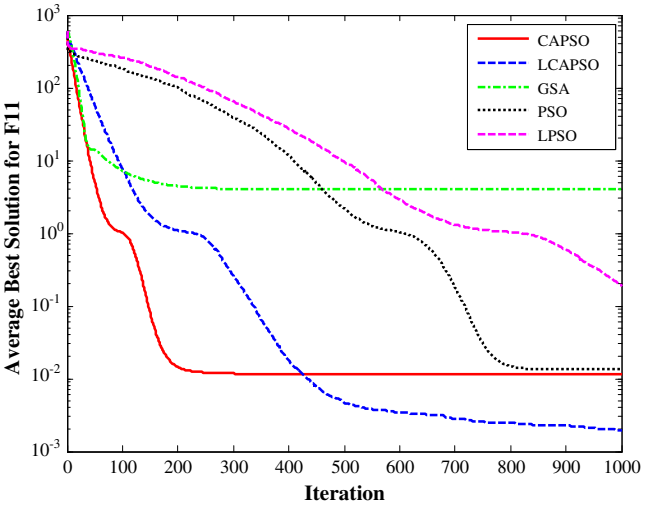


Fig. 2. Convergence performance of CAPSO, LCAPSO, GSA, PSO and LPSO for F_{11} with $n = 30$.

$$\begin{aligned} &\text{if} \quad rand < S(v_{id}(t+1)) \quad \text{then} \quad x_{id}(t+1) = 1 \\ &\quad \text{else } x_{id}(t+1) = 0 \quad \text{for} \quad i = 1, 2, \ldots, N, \end{aligned} \tag{18}$$

where $\overrightarrow{V}_i$ and $\overrightarrow{X}_i$ are the velocity and position of the ith particle, respectively, and N is the number of particles. Additionally, $|v_{id}(t+1)| < v_{max}$ and v_{max} is equal to 6 for the better convergence rate.

2.2.2. GSA in the binary search space (BGSA)

In the binary mode of GSA, the force, acceleration and velocity are updated using a method similar to that used for the GSA algorithm (Eqs. (8)–(12), (12), (14), (15)).The variable $R_{ij}(t)$ in Eq. (8) is computed as the Hamming distance between two agents, i and j, and G is a linear decreasing function as follows:

$$G(t) = G_0\left(1 - \frac{t}{T}\right), \tag{19}$$

where G_0 is a constant, t is the current iteration and T is the total number of iterations.

The position is switched between '0' and '1' based on the velocity. A probability function is defined to map the value of velocity to the range of [0, 1]. That is, BGSA modifies the velocity according to Eq. (15) and the new position changes to either '0' or '1' based on the probability of the function in Eqs. (20) and (21) as follows:

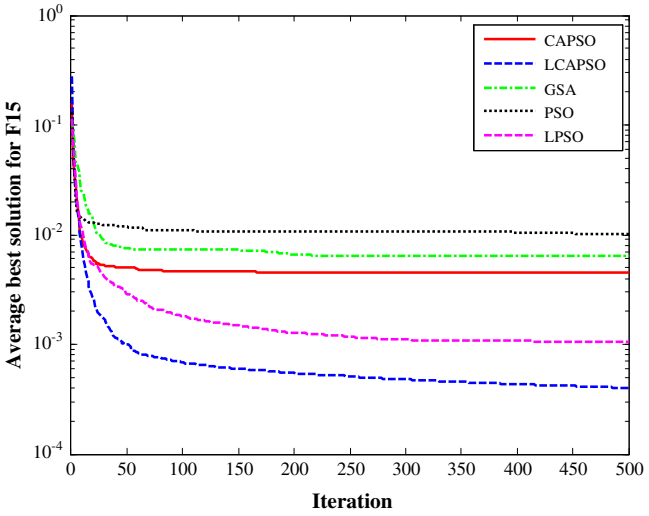


Fig. 3. Convergence performance of CAPSO, LCAPSO, GSA, PSO and LPSO for F_{15} with $n = 4$.

$$S(v_{id}(t+1)) = |\tanh(v_{id}(t+1))|, \tag{20}$$

$$\begin{aligned} &\text{if } rand < S(v_{id}(t+1)) \quad \text{then} \quad x_{id}(t+1) = complement(x_{id}(t)) \\ &\text{else } x_{id}(t+1) = x_{id}(t) \quad \text{for} \quad i = 1, 2, \ldots, N. \end{aligned} \tag{21}$$

Similar to BPSO, in this context, $|v_{id}(t+1)| < v_{max}$ and $v_{max} = 6$.

3. CAPSO – the proposed algorithm

In this section, an efficient optimization algorithm, the centripetal accelerated particle swarm optimization (CAPSO) is introduced, utilizing the laws of mechanics and the PSO algorithm for both the real and the binary search spaces.

Classical or Newtonian mechanics describes the motion of objects [21]. During the time step Δt, an object with initial velocity v_1 moves from position x_1 to position x_2. If the velocity modifies v_2 during the time step, the object will be accelerated. The acceleration is defined as follows:

$$a = \frac{v_2 - v_1}{\Delta t}. \tag{22}$$

According to Eq. (22), the new velocity is obtained by Eq. (23) as follows:

$$v_2 = v_1 + a \cdot \Delta t. \tag{23}$$

Then, the distance travelled by the object is computed as follows:

$$x_2 = x_1 + \frac{1}{2} \cdot \Delta t^2 + v_1 \cdot \Delta t. \tag{24}$$

CAPSO overcomes the drawbacks of PSO by accelerating the convergence speed and avoiding local optima. In PSO, at times, if a particle falls into a local optimum, it will not be able to get out of it. That is, if $\overrightarrow{P}_g$ obtained through the population lies in a local optimum while the current position and the personal best position of particle i are in the same local optimum, the second and third terms of Eq. (5) tend to zero and w decreases linearly to near zero. Consequently, the next velocity of particle i tends to zero, and its next position in Eq. (6) does not change; thus, the particle remains in the local optimum.

CAPSO applies the laws of motion in mechanics and some other terms to the PSO equations to overcome the aforementioned problems. In this algorithm, each particle has four specifications: position, velocity, acceleration, and centripetal acceleration. The position and velocity of the ith particle in a *d-dimensional* search space are defined as follows:

$$\overrightarrow{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id}) \quad \text{for} \quad i = 1, 2, \ldots, N. \tag{25}$$

$$\overrightarrow{V}_i = (v_{i1}, v_{i2}, \ldots, v_{id}) \quad \text{for} \quad i = 1, 2, \ldots, N. \tag{26}$$

Additionally, the personal best position of the ith particle, $\overrightarrow{P}_i$, and the best position explored so far by all particles, $\overrightarrow{P}_g$, are defined according to Eqs. (27) and (28) as follows:

Table 8
Minimization results of the benchmark functions from Tables 1 and 2 for the real search space (Maximum iteration = 2000 and $n = 100$).

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
F1					
Avg. best solution	$\mathbf{8.09 \times 10^{-38}}$	2.48×10^{-16}	$1.60 \times 10^{+4}$	3.45×10^{-16}	117.22
SD	$\mathbf{9.83 \times 10^{-38}}$	6.68×10^{-17}	$1.22 \times 10^{+4}$	2.25×10^{-16}	47.30
F2					
Avg. best solution	$\mathbf{1.81 \times 10^{-20}}$	0.183	165.6	9.10×10^{-9}	188.99
SD	$\mathbf{9.38 \times 10^{-21}}$	0.373	37.72	5.17×10^{-9}	101.89
F3					
Avg. best solution	$\mathbf{1.57 \times 10^{-9}}$	$3.12 \times 10^{+3}$	$1.91 \times 10^{+5}$	251.91	$3.37 \times 10^{+5}$
SD	$\mathbf{8.54 \times 10^{-9}}$	571.74	$2.99 \times 10^{+4}$	246.95	$4.02 \times 10^{+4}$
F4					
Avg. best solution	$\mathbf{1.22 \times 10^{-17}}$	7.64	35.80	9.68×10^{-6}	73.00
SD	$\mathbf{3.31 \times 10^{-17}}$	1.42	5.07	8.58×10^{-6}	7.14
F5					
Avg. best solution	**96.26**	194.14	$8.02 \times 10^{+6}$	96.68	$7.86 \times 10^{+5}$
SD	**0.72**	82.76	$2.44 \times 10^{+7}$	0.134	$5.48 \times 10^{+5}$
F6					
Avg. best solution	4.70	26.26	$1.01 \times 10^{+4}$	**0**	268.17
SD	2.96	38.28	$9.46 \times 10^{+3}$	**0**	59.67
F7					
Avg. best solution	**0.014**	0.578	77.96	0.041	4.58
SD	$\mathbf{1.56 \times 10^{-3}}$	0.316	48.36	6.25×10^{-3}	2.18
F8					
Avg. best solution	−13679.60	−5135.05	**−22798.80**	−16016.50	−22477.70
SD	753.54	812.25	$1.61 \times 10^{+3}$	$1.68 \times 10^{+3}$	$1.20 \times 10^{+3}$
F9					
Avg. best solution	820.71	**96.51**	595.24	788.98	845.06
SD	66.36	**11.17**	77.2	64.90	67.37
F10					
Avg. best solution	$\mathbf{5.99 \times 10^{-14}}$	0.045	12.62	0.151	19.96
SD	$\mathbf{9.5 \times 10^{-15}}$	0.195	4.01	0.631	1.27×10^{-3}
F11					
Avg. best solution	$\mathbf{8.88 \times 10^{-17}}$	24.79	126.86	3.28×10^{-6}	6.33
SD	$\mathbf{9.84 \times 10^{-17}}$	5.61	77.05	1.63×10^{-5}	3.10
F12					
Avg. best solution	0.165	0.667	$8.53 \times 10^{+6}$	**0.075**	$8.89 \times 10^{+6}$
SD	0.054	0.338	$4.67 \times 10^{+7}$	**0.021**	$6.21 \times 10^{+6}$
F13					
Avg. best solution	5.94	20.32	$4.1 \times 10^{+7}$	**2.75**	$1.57 \times 10^{+7}$
SD	0.884	8.73	$1.25 \times 10^{+8}$	**0.229**	$7.51 \times 10^{+6}$
Avg. rank	**1.7**	2.9	4.1	2.1	4.2
Final rank	**1**	3	4	2	5

$$\overrightarrow{P}_i = (p_{i1}, p_{i2}, \ldots, p_{id}) \quad \text{for} \quad i = 1, 2, \ldots, N. \tag{27}$$

$$\overrightarrow{P}_g = (p_{g1}, p_{g2}, \ldots, p_{gd}). \tag{28}$$

As originally proposed for PSO, there are two approaches for choosing $\overrightarrow{P}_g$, the *gbest* and the *lbest* (LCAPSO) models. Each particle updates its velocity based on the current values of velocity, acceleration and centripetal acceleration, as follows:

$$v_{id}(t+1) = v_{id}(t) + a_{id}(t) + A_{id}(t), \tag{29}$$

where $v_{id}(t+1)$ and $v_{id}(t)$ are the next and current velocity, respectively, and $a_{id}(t)$ is the acceleration obtained using Eqs. (5) and (22), as follows:

$$a_{id}(t) = rand \times (p_{id}(t) - x_{id}(t)) + rand \times (p_{gd}(t) - x_{id}(t)), \tag{30}$$

where *rand* is a random number with uniform distribution between 0 and 1, $x_{id}(t)$ is the current position, $p_{id}(t)$ is the personal best position of the ith particle and *pgd (t)* is the global best position explored so far by the population in the dth dimension.

Another term in Eq. (29) is the centripetal acceleration, $A_{id}(t)$, which increases the convergence speed of the algorithm and helps it to escape from local optima. It is calculated as follows:

$$A_{id}(t) = E_i(t) \times rand \times (p_{id}(t) - p_{med,d}(t) - x_{id}(t)), \tag{31}$$

Table 9
Minimization results for benchmark functions in Tables 1 and 2 for the real search space (maximum iteration = 3000 and $n = 200$).

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
F1					
Avg. best solution	$\mathbf{3.46 \times 10^{-46}}$	63.1	$8.19 \times 10^{+4}$	9.36×10^{-21}	$6.34 \times 10^{+3}$
SD	$\mathbf{3.1 \times 10^{-46}}$	47.09	$2.89 \times 10^{+4}$	9.99×10^{-21}	$2.50 \times 10^{+3}$
F2					
Avg. best solution	$\mathbf{2.31 \times 10^{-24}}$	4.71	686.2	1.01×10^{-10}	937.45
SD	$\mathbf{1.27 \times 10^{-24}}$	1.86	54.38	7.34×10^{-11}	43.51
F3					
Avg. best solution	**0.069**	$1.07 \times 10^{+4}$	$6.61 \times 10^{+5}$	116.37	$3.48 \times 10^{+5}$
SD	**0.152**	$1.70 \times 10^{+3}$	$7.77 \times 10^{+4}$	157.11	$2.98 \times 10^{+4}$
F4					
Avg. best solution	$\mathbf{2.24 \times 10^{-15}}$	11.96	89.17	2.09	97.32
SD	$\mathbf{2.67 \times 10^{-15}}$	1.11	5.42	2.05	1.31
F5					
Avg. best solution	**196.08**	$4.46 \times 10^{+3}$	$1.9 \times 10^{+8}$	196.40	$2.27 \times 10^{+7}$
SD	0.896	$4.96 \times 10^{+3}$	$1.04 \times 10^{+8}$	0.174	$5.93 \times 10^{+8}$
F6					
Avg. best solution	4.57	616.43	$7.44 \times 10^{+4}$	**0**	$6.34 \times 10^{+3}$
SD	3.26	254.45	$2.07 \times 10^{+4}$	**0**	933.95
F7					
Avg. best solution	$\mathbf{5.66 \times 10^{-3}}$	12.25	$1.19 \times 10^{+3}$	0.043	528.17
SD	$\mathbf{1.09 \times 10^{-3}}$	10.51	535.96	5.80×10^{-3}	155.75
F8					
Avg. best solution	−19334.6	−7395.64	−35803.8	−26221.4	**− 36,975**
SD	$1.33 \times 10^{+3}$	$1.19 \times 10^{+3}$	$3.68 \times 10^{+3}$	$2.59 \times 10^{+3}$	$2.46 \times 10^{+3}$
F9					
Avg. best solution	$1.97 \times 10^{+3}$	**260.13**	$1.38 \times 10^{+3}$	$1.84 \times 10^{+3}$	$2.00 \times 10^{+3}$
SD	104.89	**26.76**	119.27	114.63	108.43
F10					
Avg. best solution	$\mathbf{7.28 \times 10^{-14}}$	1.66	18.76	0.812	19.96
SD	$\mathbf{7.62 \times 10^{-15}}$	0.359	0.516	1.33	8.44×10^{-4}
F11					
Avg. best solution	$\mathbf{1.69 \times 10^{-16}}$	66.72	688.99	1.77×10^{-8}	139.81
SD	$\mathbf{1.01 \times 10^{-16}}$	9.88	252.41	7.01×10^{-8}	47.13
F12					
Avg. best solution	0.350	1.66	$5.05 \times 10^{+8}$	**0.169**	$7.21 \times 10^{+8}$
SD	0.040	0.367	$2.81 \times 10^{+8}$	**0.012**	$2.99 \times 10^{+8}$
F13					
Avg. best solution	17.58	118.43	$9.32 \times 10^{+8}$	**11.39**	$1.10 \times 10^{+9}$
SD	1.07	19.06	$4.92 \times 10^{+8}$	**0.546**	$4.58 \times 10^{+8}$
Avg. rank	**1.7**	3	4.1	1.9	4.2
Final rank	**1**	3	4	2	5

where $p_{med,d}(t)$ is the current median position of particles in dimension d and $E_i(t)$ is the acceleration coefficient given by Eq. (33) as follows:

$$e_i(t) = fit_i(t) - GWfit(t), \tag{32}$$

$$E_i(t) = \frac{e_i(t)}{\sum_{j=1}^{N} e_j(t)}, \tag{33}$$

where $fit_i(t)$ is the fitness value of the particle i and $GWfit(t)$ is the worst fitness value explored so far by the swarm.

According to Eqs. (24), (29) and (30), the next position is found using Eq. (34) as follows:

$$x_{id}(t+1) = x_{id}(t) + \frac{1}{2} \times a_{id}(t) + v_{id}(t+1). \tag{34}$$

Then, the necessary modifications are introduced to apply CAPSO to problems in the binary search space. The concepts of BCAPSO are similar to the original CAPSO, except that, to update the position, the position value is switched between '0' and '1', and each dimension can only have a value of '0' or '1'. Therefore, Eqs. (25)–(33) are used in BCAPSO and the next position of the particle is modified based on Eq. (35)–(37) as follows:

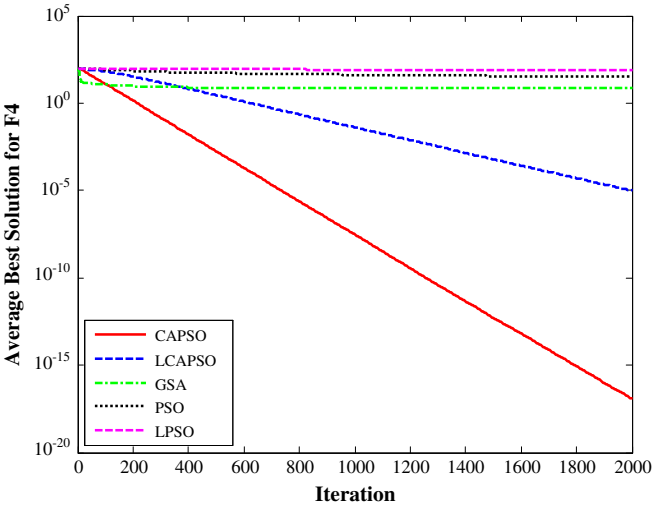


Fig. 4. Convergence performance of CAPSO, LCAPSO, GSA, PSO and LPSO for F_4 with $n = 100$.

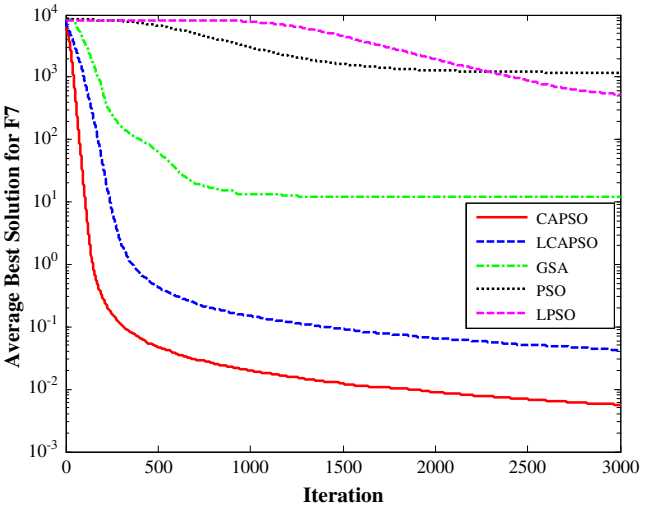


Fig. 5. Convergence performance of CAPSO, LCAPSO, GSA, PSO and LPSO for F_7 with $n = 200$.

Table 10
Seven PSO algorithms in the literature.

Algorithm	Topology	Parameter settings
GPSO	Global Star	ω: 0.9–0.4, $C_1 = C_2 = 2.0$
LPSO	Local Ring	ω: 0.9–0.4, $C_1 = C_2 = 2.0$
HPSO-TVAC	Global Star	ω: 0.9–0.4, C_1: 2.5–0.5, C_2: 0.5–2.5
DMS-PSO	Dynamic Multi-swarm	ω: 0.9–0.2, $C_1 = C_2 = 2.0$, $m = 3$, $R = 5$
VPSO	Local Von Neumann	ω: 0.9–0.4, $C_1 = C_2 = 2.0$
CLPSO	Comprehensive Learning	ω: 0.9–0.4, $C = 1.49445$, $m = 7$
APSO	Global Star	ω: 0.9, $C_1 = C_2 = 2.0$, δ: random in [0.05, 0.1], σ: 1–0.1

$$y_{id}(t+1) = \frac{1}{2} \times a_{id}(t) + v_{id}(t+1), \tag{35}$$

$$S(y_{id}(t+1)) = \frac{0.5}{0.5 + e^{-y_{id}(t+1)}}, \tag{36}$$

Table 11
Comparative results of seven PSO Algorithms [63] with CAPSO and LCAPSO on eleven benchmark functions in Tables 1 and 2 (Maximum iteration = 200,000 and $n = 30$).

Function	GPSO	LPSO	HPSO-TVAC	DMS-PSO	VPSO	CLPSO	APSO	CAPSO	LCAPSO
F1									
Best	1.98×10^{-53}	4.77×10^{-29}	3.38×10^{-41}	3.85×10^{-54}	5.11×10^{-38}	1.89×10^{-19}	1.45×10^{-150}	**0**	**0**
SD	7.08×10^{-53}	1.13×10^{-28}	8.50×10^{-41}	1.75×10^{-53}	1.91×10^{-37}	1.49×10^{-19}	5.73×10^{-150}	**0**	**0**
F2									
Best	2.51×10^{-34}	2.03×10^{-20}	6.9×10^{-23}	2.61×10^{-29}	6.29×10^{-27}	1.01×10^{-13}	5.15×10^{-84}	**0**	**0**
SD	5.84×10^{-34}	2.89×10^{-20}	6.89×10^{-23}	6.6×10^{-29}	8.68×10^{-27}	6.51×10^{-14}	1.44×10^{-83}	**0**	**0**
F3									
Best	6.45×10^{-2}	18.60	2.89×10^{-7}	47.5	1.44	395	1.0×10^{-10}	**0**	**0**
SD	9.46×10^{-2}	30.71	2.97×10^{-7}	56.4	1.55	142	2.13×10^{-10}	**0**	**0**
F5									
Best	28.1	21.8627	13	32.3	37.6469	11	**2.84**	26.60	26.45
SD	24.6	11.1593	16.5	24.1	24.9378	14.5	3.27	0.576	0.349
F6									
Best	0	0	0	0	0	0	0	0	0
SD	0	0	0	0	0	0	0	0	0
F7									
Best	7.77×10^{-3}	1.49×10^{-2}	5.54×10^{-2}	1.1×10^{-2}	1.08×10^{-2}	3.92×10^{-3}	4.66×10^{-3}	$\mathbf{3.56 \times 10^{-5}}$	9.22×10^{-5}
SD	2.42×10^{-3}	5.66×10^{-3}	2.08×10^{-2}	3.94×10^{-3}	3.24×10^{-3}	1.14×10^{-3}	1.7×10^{-3}	$\mathbf{8.56 \times 10^{-6}}$	2.23×10^{-5}
F8									
Best	−10090.16	−9628.35	−10868.57	−9593.33	−9845.27	− 12557.65	**−12569.5**	−6565.6	−7657.34
SD	495	456.54	289	441	588.87	36.2	$\mathbf{5.22 \times 10^{-11}}$	463.88	558.84
F9									
Best	30.7	34.90	2.39	28.1	34.09	2.57×10^{-11}	$\mathbf{5.8 \times 10^{-15}}$	52.92	63.71
SD	8.68	7.25	3.71	6.42	8.07	6.64×10^{-11}	$\mathbf{1.01 \times 10^{-14}}$	14.47	25.39
F10									
Best	1.15×10^{-14}	1.85×10^{-14}	2.06×10^{-10}	8.52×10^{-15}	1.14×10^{-14}	2.01×10^{-12}	1.11×10^{-14}	1.46×10^{-14}	$\mathbf{7.88 \times 10^{-15}}$
SD	2.27×10^{-15}	4.80×10^{-15}	9.45×10^{-10}	1.79×10^{-15}	3.48×10^{-15}	9.22×10^{-13}	3.55×10^{-15}	3.7×10^{-15}	$\mathbf{1.74 \times 10^{-15}}$
F11									
Best	2.37×10^{-2}	1.10×10^{-2}	1.07×10^{-2}	1.31×10^{-2}	1.31×10^{-2}	6.45×10^{-13}	1.67×10^{-2}	9.29×10^{-3}	**0**
SD	2.57×10^{-15}	1.60×10^{-2}	1.14×10^{-2}	1.73×10^{-2}	1.35×10^{-2}	2.07×10^{-12}	2.41×10^{-2}	1.09×10^{-2}	**0**
F12									
Best	1.04×10^{-2}	2.18×10^{-30}	7.07×10^{-30}	$\mathbf{2.05 \times 10^{-32}}$	3.46×10^{-3}	1.59×10^{-21}	3.76×10^{-31}	5.27×10^{-2}	2×10^{-3}
SD	3.16×10^{-2}	5.14×10^{-30}	4.05×10^{-30}	$\mathbf{8.12 \times 10^{-33}}$	1.89×10^{-2}	1.93×10^{-21}	1.2×10^{-30}	4.85×10^{-2}	1.09×10^{-3}

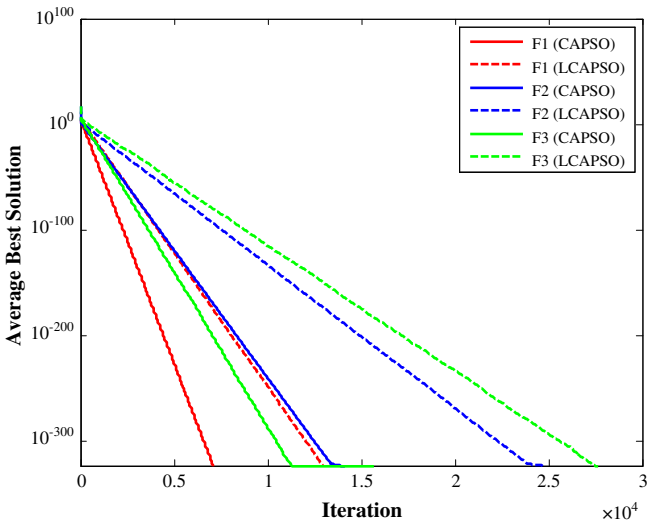


Fig. 6. Convergence performance of CAPSO and LCAPSO for the functions F_1, F_2 and F_3 with $n = 30$.

Table 12
Minimization results of shifted functions from Tables 1 and 2 in the real search space.

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
n = 30, maximum iteration = 1000					
Shifted F4					
Avg. best solution	**−393.99**	−380.44	−392.47	−385.15	−393.81
SD	12.12	2.99	10.51	6.24	4.85
Median Best solution	−394.50	−380.29	−392.76	−385.79	−393.24
Avg. search time	0.6	5.4	0.7	2.7	2.5
Shifted F6					
Avg. best solution	−440.80	**−449.20**	222.50	−444.67	−443.93
SD	2.94	0.484	2537.94	1.21	1.3
Median best solution	−441.00	−449.00	−444.50	−445.00	−444.00
Avg. search time	1.0	3.6	1.0	2.4	3.3
Shifted F10					
Avg. best solution	**−118.98**	−118.90	−118.97	−118.89	−118.91
SD	0.067	0.053	0.049	0.060	0.048
Median best solution	−118.98	−118.89	−118.96	−118.89	−118.91
Avg. search time	0.9	1.8	1.0	2.1	2.1
Shifted F11					
Avg. best solution	−177.53	−175.94	−164.57	**−177.80**	−177.62
SD	0.353	1.11	30.50	0.217	0.216
Median best solution	−177.58	−176.00	−177.91	−177.84	−177.63
Avg. search time	0.8	7.9	1.3	3.4	4.3
n = 100, maximum iteration = 2000					
Shifted F4					
Avg. best solution	−351.73	**−359.87**	−344.00	−347.30	−334.36
SD	10.40	**1.53**	15.78	4.69	13.02
Median Best solution	−349.97	−359.64	−344.00	−347.28	−332.00
Avg. search time	1.9	25.8	2.0	6.2	6.6
Shifted F6					
Avg. best solution	−399.33	**−423.97**	$1.45 \times 10^{+4}$	−413.30	22.03
SD	8.93	17.26	$1.21 \times 10^{+4}$	2.62	88.94
Median best solution	−401.00	−430.50	$9.88 \times 10^{+3}$	−413.00	18.00
Avg. search time	2.0	21.6	4.2	7.4	8.7
Shifted F10					
Avg. best solution	**−118.65**	−118.60	−118.64	−118.61	−118.61
SD	**0.023**	0.030	0.025	0.023	0.024
Median best solution	−118.65	−118.60	−118.63	−118.61	−118.60
Avg. search time	3.2	5.3	2.4	4.9	5.6
Shifted F11					
Avg. best solution	−167.46	−150.78	53.11	**−170.09**	−122.02
SD	1.81	5.54	127.93	1.02	11.27
Median best solution	−168.15	−150.94	26.49	−170.09	−121.25
Avg. search time	4.0	29.9	5.0	9.3	11.6
n = 200, maximum iteration = 3000					
Shifted F4					
Avg. best solution	−329.76	**−352.56**	−311.14	−333.25	−312.27
SD	10.15	**0.690**	9.50	6.26	23.90
Median Best solution	−328.45	−352.42	−311.24	−332.27	−299.18
Avg. search time	2.6	62.5	3.7	12.0	7.8
Shifted F6					
Avg. best solution	−338.60	41.53	$5.66 \times 10^{+4}$	**−361.57**	$9.80 \times 10^{+3}$
SD	14.35	199.55	$2.70 \times 10^{+4}$	**5.89**	$1.75 \times 10^{+3}$
Median best solution	−340.00	5.50	$5.46 \times 10^{+4}$	−362.00	$9.58 \times 10^{+3}$
Avg. search time	5.4	74.8	7.8	12.3	14.9
Shifted F10					
Avg. best solution	**−118.53**	−118.51	**−118.53**	−118.52	−118.52
SD	0.020	0.015	**0.017**	0.018	0.022
Median best solution	−118.52	−118.51	−118.53	−118.51	−118.51
Avg. search time	4.2	13.0	4.2	11.2	9.6
Shifted F11					
Avg. best solution	−149.61	−97.59	$1.28 \times 10^{+3}$	**−153.76**	515.47
SD	2.37	8.26	333.32	**2.00**	113.40
Median best solution	−149.96	−97.33	$1.219 \times 10^{+3}$	−153.72	527.88

(*continued on next page*)

Table 12 (*continued*)

Functions	CAPSO	GSA	PSO	LCAPSO	LPSO
Avg. search time	7.2	77.0	10.1	18.1	21.3
Avg. rank	**2.1**	2.8	3.9	2.3	3.3
Final rank	**1**	3	5	2	4

Table 13
Minimization results for the unimodal functions in Table 1 in the binary search space (maximum iteration = 500, n = 5 and dim = 75).

Functions	BCAPSO	BGSA	BPSO	LBCAPSO	LBPSO
F1					
Avg. best solution	$\mathbf{4.65 \times 10^{-5}}$	2.3×10^{-3}	21.23	$\mathbf{4.65 \times 10^{-5}}$	45.00
SD	$\mathbf{2.7 \times 10^{-20}}$	9.16×10^{-3}	11.44	$\mathbf{2.7 \times 10^{-20}}$	23.66
Median best solution	4.65×10^{-5}	4.65×10^{-5}	18.84	4.65×10^{-5}	40.09
Avg. search time	0.5	12.5	0.5	1.7	1.2
F2					
Avg. best solution	$\mathbf{1.53 \times 10^{-3}}$	$\mathbf{1.53 \times 10^{-3}}$	0.669	$\mathbf{1.53 \times 10^{-3}}$	1.04
SD	$\mathbf{4.41 \times 10^{-19}}$	$\mathbf{4.41 \times 10^{-19}}$	0.177	$\mathbf{4.41 \times 10^{-19}}$	0.298
Median best solution	1.53×10^{-3}	1.53×10^{-3}	0.673	1.53×10^{-3}	1.013
Avg. search time	0.5	13.2	0.5	1.6	1.1
F3					
Avg. best solution	0.746	36.06	27.93	**0.539**	76.91
SD	3.57	98.47	16.16	**1.28**	37.99
Median best solution	6.52×10^{-5}	0.345	21.99	0.02	72.86
Avg. search time	1.0	13.6	0.6	3.9	1.5
F4					
Avg. best solution	$\mathbf{7.32 \times 10^{-3}}$	0.121	3.92	8.34×10^{-3}	5.86
SD	$\mathbf{9.5 \times 10^{-3}}$	0.315	1.30	0.013	1.35
Median best solution	3.05×10^{-3}	0.015	3.8	3.05×10^{-3}	5.63
Avg. search time	0.8	12.8	0.6	2.8	1.3
F5					
Avg. best solution	24.73	54.32	257.55	**2.88**	667.47
SD	76.47	97.99	234.568	**1.00**	489.23
Median best solution	3.78	4.01	140.541	3.17	587.76
Avg. search time	0.7	7.0	0.442533	3.1	1.3
F6					
Avg. best solution	**0.00**	0.033	15.07	**0.00**	41.10
SD	**0.00**	0.183	11.23	**0.00**	23.12
Median best solution	0.00	0	11	0.00	38.50
Avg. search time	0.3	11.9	0.4	0.8	1.1
F7					
Avg. best solution	$\mathbf{8.22 \times 10^{-4}}$	2.87×10^{-3}	0.016	1.6×10^{-3}	4.8×10^{-3}
SD	$\mathbf{8.67 \times 10^{-4}}$	1.73×10^{-3}	0.010	1.27×10^{-3}	3.3×10^{-3}
Median best solution	4.47×10^{-4}	2.72×10^{-3}	0.013	1.47×10^{-3}	4.55×10^{-3}
Avg. search time	1.5	13.1	0.7	2.6	2.3
Avg. rank	**1.3**	2.6	3.4	**1.3**	4.3
Final rank	**1**	2	3	**1**	4

$$\begin{aligned} &\text{if } rand < S(y_{id}(t+1)) \quad \text{then} \quad x_{id}(t+1) = 1 \\ &\text{else} \quad x_{id}(t+1) = 0 \quad \text{for} \quad i = 1, 2, \ldots, N. \end{aligned} \tag{37}$$

In Eq. (37), $x_{id}(t+1)$, the next position, is changed to '1' if the random number is less than the given probability by the sigmoid function, otherwise, it changes to '0'. Additionally, $|y_{id}(t+1)| < 11$ to provide the best convergence rate. In this paper, the local topology version of BCAPSO is referred to as LBCAPSO.

4. Experimental results and discussion

To evaluate the performance of CAPSO, LCAPSO, BCAPSO and LBCAPSO, some numerical examples [45,46,61] are considered in both the real and the binary search spaces. Twenty-three (23) minimization benchmark functions and one maximization binary function (shown in Tables 1–4) are selected to compare the proposed algorithms with GSA and PSO. Additionally, a comparison of CAPSO and LCAPSO to well-known PSO algorithms is performed on the unimodal and multimodal high-dimensional benchmark functions shown in Tables 1 and 2. Some comparisons among the results of the pro-

Table 14
Minimization results of the multimodal high-dimensional functions in Table 2 for the binary search space (maximum iteration = 500, n = 5 and dim = 75).

Functions	BCAPSO	BGSA	BPSO	LBCAPSO	LBPSO
F8					
Avg. best solution	−2053.41	−2077.51	−2042.58	**−2093.59**	−2000.57
SD	47.94	31.18	40.97	**6.30**	37.07
Median best solution	−2060.34	−2094.53	−2061.61	−2094.8	−1997.52
Avg. search time	1.0	12.8	0.7	3.7	1.2
F9					
Avg. best solution	1.14	3.74	7.37	**0.896**	8.53
SD	0.912	1.22	2.06	0.951	1.85
Median best solution	1.24	3.62	7.38	0.995	8.04
Avg. search time	1.0	13.18	0.7	3.0	1.3
F10					
Avg. best solution	$\mathbf{3.95 \times 10^{-3}}$	$\mathbf{3.95 \times 10^{-3}}$	3.79	$\mathbf{3.95 \times 10^{-3}}$	4.96
SD	**0**	**0**	0.67	**0**	1.04
Median best solution	3.95×10^{-3}	3.95×10^{-3}	3.69	3.95×10^{-3}	5.16
Avg. search time	0.6	12.7	0.5	2.15	1.0
F11					
Avg. best solution	0.061	0.047	0.970	**0.039**	1.42
SD	0.038	0.031	0.232	0.035	0.257
Median best solution	0.060	0.034	0.979	0.026	1.41
Avg. search time	0.8	8.1	0.5	2.6	1.1
F12					
Avg. best solution	0.832	1.97	2.67	**0.153**	3.41
SD	0.878	3.89	1.83	**0.168**	1.82
Median best solution	0.703	0.685	2.42	0.079	3.41
Avg. search time	0.9	12.87	0.8	4.21	2.0
F13					
Avg. best solution	0.253	0.258	1.58	**0.076**	2.50
SD	0.134	0.174	0.79	**0.060**	1.27
Median best solution	0.213	0.211	1.35	0.073	2.40
Avg. search time	1.0	13.62	0.9	4.2	1.9
Avg. rank	2.3	2.5	3.3	**1**	4.7
Final rank	2	3	4	**1**	5

posed methods and the standard GSA and PSO in both search spaces are performed to evaluate the efficiency of the proposed algorithms. The functions in Table 1, F_1 to F_7, are unimodal functions. Six functions of Table 2 and ten functions of Table 3 are multimodal functions. For the unimodal functions, the convergence rate of the search algorithm is more important than the final results because other methods have been designed to optimize these functions.

In multimodal functions, finding an optimal (or a good near-global optimal) solution is important. These functions are more difficult to optimize because of the number of local optima. Functions F_8 to F_{13} (Table 2) have many local minima and the number of local minima exponentially increases as the dimension increases. F_{14} *to* F_{23} (Table 3) have fixed dimensions with a few local minima. In these cases, the search algorithms should not become trapped in a local optimum and should be able to obtain good solutions. The function in Table 4 is a discrete binary function that should be maximized. It is applied only in a binary search space; the other functions are applied in both search spaces.

In Tables 1–4, *Range* is the feasible bound, n is the dimension of the function and F_{opt} is the optimum value of the function in the optimum position (X_{opt}). For all of the functions in Tables 1 and 2, F_{opt} is zero and X_{opt} is $[0]^n$, for F_8, F_{opt} is $-418.9829 \times n$ and X_{opt} is $[420.96]^n$, and for F_5,F_{12} and F_{13}, X_{opt} is $[1]^n$. The values of X_{opt} and F_{opt} in Table 3 are given in Appendix A. In Table 4, the maximum value of the function, F_{opt}, depends on its dimension. For example, F_{opt} will be 100 if n is equal to 100.

4.1. Analysis of CAPSO and LCAPSO in the real search space

Five algorithms, CAPSO, LCAPSO, GSA, PSO and LPSO, are randomly initialised and run on each benchmark function in Tables 1–3. Several parameters must be set to initial values. The population size is set to 50 (N = 50), and the number of maximum iterations is set at 1000 for functions in Tables 1 and 2, and at 500 for those in Table 3. The function dimension for Tables 1 and 2 is 30 (n = 30), and for Table 3, it is fixed for each function. In PSO and LPSO, C_1 and C_2 are set to 2, and w decreases linearly from 0.9 to 0.2. The ring topology is used as the neighborhood structure in the *lbest* model for both the LCAPSO and the LPSO algorithms and the number of neighbors is 2. The parameter *Kbest* in Eq. (13) in GSA is initialised with 50 (population size) and then decreased linearly to 1. G in Eq. (8) is computed using Eq. (38) as follows:

$$G(t) = G_0 e^{\frac{-\alpha t}{T}}, \tag{38}$$

where G_0 is set to 100, α is set to 20 and T is the maximum number of iterations.

Table 15
Minimization results of the multimodal low-dimensional functions in Table 3 for the binary search space (maximum iteration = 500).

Functions	BCAPSO	BGSA	BPSO	LBCAPSO	LBPSO
F14					
Avg. best solution	1.02	1.01	1.04	**0.998**	**0.998**
SD	0.084	0.031	0.084	$\mathbf{1.5 \times 10^{-10}}$	1.4×10^{-7}
Median best solution	0.998	0.999	0.998	0.998	0.998
Avg. search time	0.6	7.2	1.94	1.6	3.1
F15					
Avg. best solution	3.76×10^{-3}	2.08×10^{-3}	1.35×10^{-3}	$\mathbf{7.85 \times 10^{-4}}$	9.4×10^{-4}
SD	6.67×10^{-3}	1.68×10^{-3}	1.66×10^{-3}	$\mathbf{1.95 \times 10^{-4}}$	1.51×10^{-4}
Median	1.24×10^{-3}	1.49×10^{-3}	8.49×10^{-4}	7.77×10^{-4}	9.07×10^{-4}
Avg. search time	0.6	7.8	0.9	2.0	2.3
F16					
Avg. best solution	−1.0201	−1.0287	−1.0284	**−1.0316**	**− 1.0316**
SD	0.017	5.39×10^{-3}	8.59×10^{-3}	9.93×10^{-5}	$\mathbf{6.99 \times 10^{-5}}$
Median best solution	− 1.031	−1.0316	−1.0316	−1.0316	−1.0316
Avg. search time	0.1	5.4	0.4	0.7	0.8
F17					
Avg. best solution	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**
SD	1.02×10^{-3}	2.16×10^{-5}	2.99×10^{-4}	$\mathbf{1.79 \times 10^{-5}}$	2.71×10^{-5}
Median best solution	0.398	0.398	0.398	0.398	0.398
Avg. search time	0.2	5.6	0.3	1.0	0.8
F18					
Avg. best solution	**3.00**	3.03	**3.00**	**3.00**	**3.00**
SD	1.5×10^{-6}	0.055	1.1×10^{-5}	$\mathbf{1.5 \times 10^{-7}}$	2.60×10^{-4}
Median best solution	3.00	3.00	3.0	3	3.00
Avg. search time	0.1	5.6	0.3	0.3	1.2
F19					
Avg. best solution	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**
SD	7.46×10^{-4}	1.23×10^{-4}	2.98×10^{-4}	$\mathbf{1.76 \times 10^{-5}}$	9.16×10^{-5}
Median best solution	−3.86	−3.86	−3.86	−3.86	−3.86
Avg. search time	0.3	8.5	0.6	2.5	1.2
F20					
Avg. best solution	−3.2679	−3.3161	−3.2389	**−3.3212**	−3.2574
SD	0.056	0.022	0.05	$\mathbf{1.45 \times 10^{-3}}$	0.031
Median best solution	−3.2986	−3.322	−3.1993	−3.3219	−3.2636
Avg. search time	1.2	15.3	0.5	3.4	2.4
F21					
Avg. best solution	−5.0663	−3.3248	−6.3991	**−7.5827**	−6.8460
SD	2.67	1.43	2.69	2.14	1.80
Median best solution	−4.8436	−2.6829	−4.9871	−6.3011	−7.1668
Avg. search time	0.6	7.6	0.9	2.7	1.3
F22					
Avg. best solution	−5.4226	−5.3539	−6.7974	**−8.5957**	−7.4833
SD	2.6037	3.03	2.87	2.2	1.68
Median best solution	−4.9875	−4.1884	−6.3521	−9.9810	−7.8952
Avg. search time	0.6	7.4	0.9	3.0	1.42
F23					
Avg. best solution	−5.5236	−3.8068	−6.1412	**−7.4303**	−7.4024
SD	2.89	2.32	3.0	2.45	1.36
Median best solution	−5.1134	−3.0550	−5.0073	−6.6032	−7.5449
Avg. search time	0.8	5.6	1.1	2.2	1.8
Avg. rank	3.0	2.9	2.7	**1.0**	1.7
Final rank	5	4	3	**1**	2

The algorithms are run independently 30 times for the functions in Tables 1–3. The average best solution, the standard deviation (SD), the median of the best solution in the last iteration and the average search time of the algorithm (in seconds) are reported in Tables 5–7. The average search time is the required length of time to find the best solution by each algorithm. Moreover, an average rank and final rank are computed for each algorithm. The algorithms are ranked based on the average best solution in each table row, and the average rank of the algorithms is calculated in each table column. Boldfaces in these tables indicate the best results among the algorithms.

The results in Table 5 show that CAPSO performs the best and converges the fastest for the unimodal functions in Table 1, except for F_6. For that function, both LCAPSO and GSA return the global minimum; however, LCAPSO has a lower average search time. CAPSO used the lowest average search time for all the functions in the table. Thus, CAPSO is able to achieve

Table 16
Maximization results of the benchmark function in Table 4 for the binary search space (maximum iteration = 1000).

F_{24} (max-ones)	BCAPSO	BGSA	BPSO	LBCAPSO	LBPSO
n = 32					
Avg. best solution	**32**	**32**	31.93	**32**	31.6
SD	**0.00**	**0.00**	0.254	**0**	0.498
Median best solution	32	32	32	32	32
Avg. search time	**0.3**	8.8	0.5	0.8	1.6
n = 64					
Avg. best solution	**64**	**64**	59.57	**64**	57.97
SD	**0.00**	**0.00**	0.971	**0**	1.13
Median best solution	64	64	60	64	58
Avg. search time	**1.0**	31.29	1.6	3.4	4.4
n = 100					
Avg. best solution	**100**	**100**	86.9	**100**	83.6
SD	**0.00**	**0.00**	2.03	**0.00**	1.27
Median best solution	100	100	87	100	83.5
Avg. search time	**2.9**	52.0	1.7	8.4	7.5
n = 200					
Avg. best solution	**200**	199.97	154.1	**200**	148
SD	**0**	0.183	1.79	**0**	0.817
Median best solution	200	200	154	200	148
Avg. search time	12.98	94.4	5.3	39.4	19.8
Avg. rank	**1**	1.3	2.3	**1**	3.3
Final rank	**1**	2	3	**1**	4

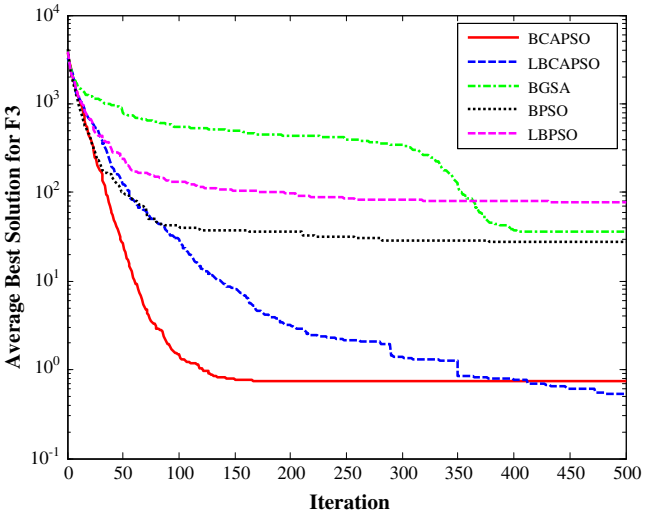


Fig. 7. Convergence performance of BCAPSO, LBCAPSO, BGSA, BPSO and LBPSO for *F3* with $n = 5$.

the best solution faster than the other algorithms. In addition, the CAPSO results show small standard deviations in the results for F_1 to F_5 and F_7. This indicates the consistency, stability and accuracy of CAPSO when solving unimodal functions. CAPSO ranks the best, and LPSO ranks the worst in the table. The superior convergence rate of CAPSO is shown Fig. 1. The results in this figure show that CAPSO tends to find the global optimum in F_3 faster than LCAPSO, GSA, PSO and LPSO and obtains the highest accuracy for the function from among all the algorithms.

Table 6 shows the average results for the multimodal high-dimensional functions in Table 2. The results show that PSO give better results for F_8; but GSA cannot tune itself and has fallen into a local minimum for this function. CAPSO yields the highest accuracy for function F_{10}, and LCAPSO performs better for F_{11} and F_{12}. The algorithms do not show good results for F_9. According to the theorem of "no free lunch" [57], one algorithm cannot offer better performance than the others in every aspect or in every type of problem. This is verified by our experimental results.

Fig. 2 shows the convergence characteristics of the algorithms solving F_{11}. The results show that LCAPSO performs the best.

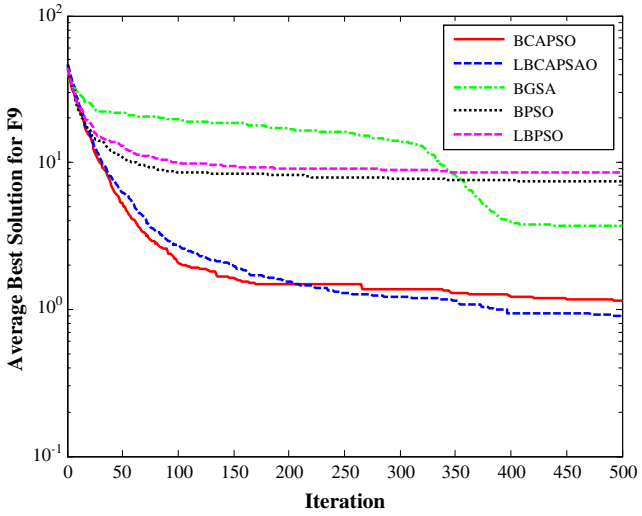


Fig. 8. Convergence performance of BCAPSO, LBCAPSO, BGSA, BPSO and LBPSO for F_9 with $n = 5$.

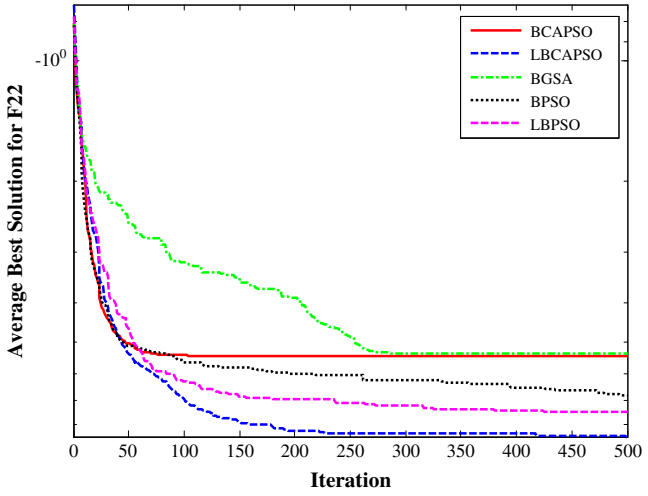


Fig. 9. Convergence performance of BCAPSO, LBCAPSO, BGSA, BPSO and LBPSO for F_{22} with $n = 4$.

In Table 7, the results of the algorithms for F_{14} to F_{23}, the multimodal fixed dimension functions, are presented. The results show that the algorithms return the global minimum or a good near-global optimum for several functions, such as F_{16},F_{17}, F_{18} and F_{19}. Additionally, LCAPSO performs better and converges faster for F_{15} and F_{21}; and GSA yields poor results for F_{14}, F_{20} and F_{21} compared with the other algorithms.

According to the final rank in Table 7, the best and the second best algorithms are LCAPSO and LPSO, respectively. Therefore, local topology algorithms exhibit better performance for these functions. The progress of the average best solution for 30 runs for function F_{15} with $n = 4$ is illustrated in Fig. 3. The results in this figure show that LCAPSO provides the best solution.

In the next subsections, CAPSO, LCAPSO, GSA, PSO and LPSO are evaluated using different dimensions for the benchmark functions in Tables 1 and 2. The results of the proposed methods for the functions in the tables are compared with several well-known PSO algorithms from the literature. In addition, the performance of the algorithms is tested for four selected functions from the tables for which the global optima are shifted to other points in various dimensions [53].

4.1.1. Comparison of algorithms with different dimensions

CAPSO, LCAPSO, GSA, PSO and LPSO are independently run 30 times for thirteen functions from Tables 1 and 2 with variable dimensions. The average best results and the SD of the best solution in the last iteration are reported in Tables 8 and 9.

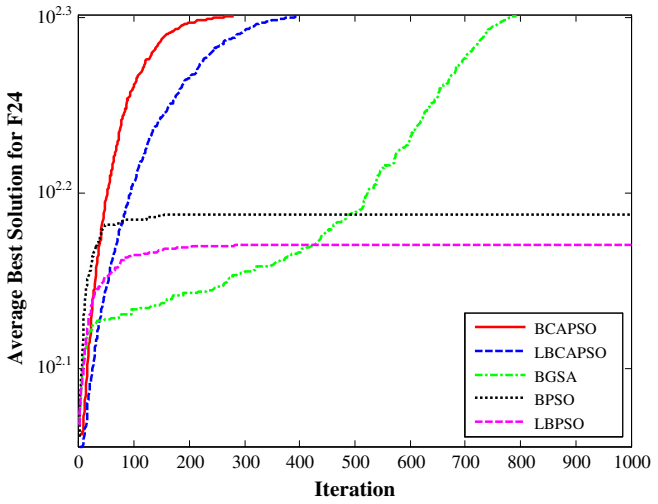


Fig. 10. Convergence performance of BCAPSO, LBCAPSO, BGSA, BPSO and LBPSO for F_{24} with $n = 200$.

Table 17
Overall rankings of the functions in Tables 5–7 for the real search space and those in Tables 13–16 for the binary search space.

Search space	Real					Binary				
Algorithm	CAPSO	GSA	PSO	**LCAPSO**	LPSO	BCAPSO	BGSA	BPSO	**LBCAPSO**	LBPSO
Overall Avg. rank	2.2	2.5	2.9	**2.1**	3.1	2.3	2.7	3.1	**1.1**	3.3
Overall rank	2	3	4	**1**	5	2	3	4	**1**	5

The dimension (n) is 100 and the maximum number of iterations is 2000 in Table 8; $n = 200$ and maximum iteration = 3000 in Table 9. The variable N is set to 50 in both tables. The results in Table 8 show that CAPSO and LCAPSO are ranked first and second, respectively, yielding the best results for all functions in the table except for F_8 and F_9. In these functions, PSO and GSA yield the better results. CAPSO performs better than LCAPSO for the unimodal functions F_1 to F_5 and F_7, and LCAPSO gives the global optimum for F_6. CAPSO yields the best results for F_{10} and F_{11} and LCAPSO for F_{12} and F_{13}. Moreover, the results of PSO and LPSO are far from those of CAPSO and LCAPSO. That is, PSO and LPSO cannot tune themselves and are trapped in local minima. As shown in Fig. 4, CAPSO performs well for F_4; in contrast, GSA, PSO and LPSO show poor results.

Table 9 shows the best results and the SD of the algorithms for the functions F_1 to F_{13} with $n = 200$. CAPSO and LCAPSO rank first and second among the algorithms, respectively. In this table, GSA, PSO and LPSO show poor results; and the proposed algorithms perform better for the functions F_1 to F_7 and F_{10} to F_{13}. For F_8 and F_9, LPSO and GSA show better results, respectively.

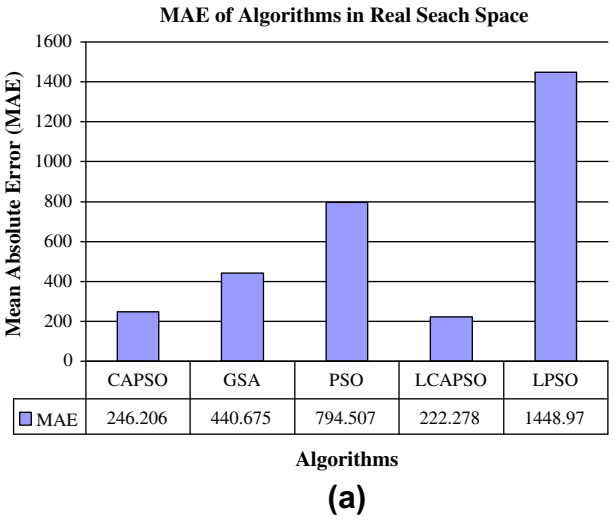
Fig. 5 shows the performance of the proposed methods compared with GSA, PSO and LPSO for F_7 with $n = 200$. The results show that CAPSO returns the best solution among the algorithms for this function.

Therefore, GSA, PSO and LPSO yield results far from the global optimum when the function dimensions are increased in Tables 8 and 9; whereas CAPSO and LCAPSO yield good near-global optimum results for the majority of the benchmark functions.

4.1.2. Comparison with other PSO algorithms

The eleven benchmark functions from reference [63] that are listed in Tables 1 and 2 are used to compare the performance of CAPSO and LCAPSO with seven existing PSO algorithms. The details of these algorithms are listed in Table 10. The algorithms GPSO [52], LPSO [28], HPSO-TVAC [47], VPSO [29], DMS-PSO [33], CLPSO [32] and APSO [63] are used for the comparison. GPSO with global star topology, LPSO with a ring neighborhood, and VPSO with a Von Neumann neighborhood are standard PSOs. The topology used in HPSO-TVAC and APSO is global star; DMS-PSO employs dynamic multi-swarm and CLPSO employs comprehensive learning.

The computational results of the algorithms are taken directly from [63] and shown in Table 11. In the table, the maximum iteration is 2×10^5, n is 30, and N is 20. The parameter configurations are set according to the corresponding references. CAPSO and LCAPSO are run 30 times. The average best solution and the SD of the best solution in the last iteration are shown in the table. The results show that the global best solution for F_6 is obtained by all PSO algorithms. For F_1, F_2



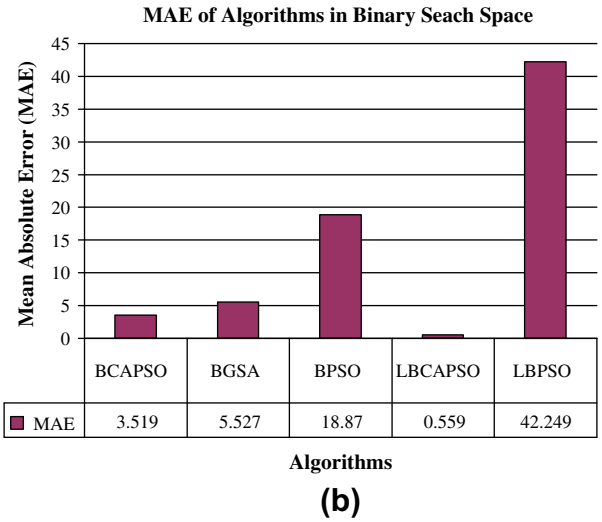


Fig. 11. Mean Absolute Error (MAE) of the best results obtained by the algorithms (a) for the functions in Tables 5–7 in the real search space and (b) for the functions in Tables 13–16 in the binary search space.

and F_3, the average best results of zero are returned by CAPSO and LCAPSO. LCAPSO finds the zero value in F_{11} and APSO gives an optimal solution in F_8. CAPSO and LCAPSO yield better results for F_7 and F_{10},respectively. For the functions F_5 and F_9, APSO yields a better solution; DMS-PSO performs well for F_{12}. Therefore, CAPSO and LCAPSO perform better for seven of the eleven benchmark functions in the table. As shown in Fig. 6, the average best results obtained by CAPSO and LCAPSO for the functions F_1, F_2 and F_3 are zero. That is, the proposed methods acquire the global optimum under the 30,000th iteration in these functions.

4.1.3. Shifted functions with different dimensions

In this section, the global optima for four functions from Tables 1 and 2 ($F4$, $F6$, $F10$ and $F11$) are shifted to other points to avoid the global optima positioning itself at the center of the search range. The performance of CAPSO, LCAPSO, GSA, PSO and LPSO in different dimensions; 30, 100 and 200, are evaluated for each of the functions. The maximum iteration is varied by 1000, 2000 and 3000 for dimensions $n = 30$, $n = 100$ and $n = 200$. The global optima are shifted for functions F_4 and F_6 to -450, for F_{10} to -140 and $F11$ to -180 [53]. The algorithms are run 30 times, and the average results are shown in Table 12.

The results in Table 12 show that the proposed methods, CAPSO and LCAPSO, rank first and second, respectively. Although GSA performs well in some cases, its results are far from the global optimum for F_6 and F_{11} with $n = 200$. PSO and LPSO show poor results in these functions for $n = 100$ and $n = 200$.

4.2. Analysis of BCAPSO and LBCAPSO in the binary search space

To test the performance of the proposed algorithms in the binary search space, BCAPSO, LBCAPSO, BGSA, BPSO and LBPSO are randomly initialised. The maximum iteration is 500 for the functions in Tables 1–3, and 1000 for those in Table 4.

The parameter G in BGSA is calculated using Eq. (19) and G_0 is 100. The dimension of the functions from Tables 1 and 2 is 5 ($n = 5$) and for the functions from Table 3 ($F_{14} - F_{23}$), it is fixed. In Table 4, n equals 32, 64, 100 and 200 to evaluate the performance of the algorithms with F_{24}. In this context, 15 bits are used to represent each continuous variable in Tables 1–3. Therefore, the dimension of each particle or agent is $dim = n \times 15$. Other parameters used in this section are the same as those used in Section 4.1. The results are averaged over 30 independent runs. The average best solution, the SD, the median of the best solution in the last iteration, the average search time, the average rank and the final rank of the algorithms are shown in Tables 13–16.

The results in Table 13 show that BCAPSO and LBCAPSO tend to find the global minimum in the unimodal functions of Table 1 and provide the same or better results than the other algorithms. It can be concluded that the best value obtained by both BCAPSO and LBCAPSO for F_3, and by LBCAPSO for F_5,differ significantly from the results of the other algorithms. Furthermore, both BCAPSO and LBCAPSO rank first and show the best SD for all the functions in the table. This all indicates that these algorithms are more powerful and more robust than the others for solving unimodal functions. The best performance of BCAPSO and LBCAPSO is shown in Fig. 7. This figure shows that the algorithms converge at a faster rate and provide good results for F_3; whereas BGSA, BPSO and LBPSO converge at a slower rate for the same function.

Table 14 shows the average results for the multimodal high-dimensional functions in Table 2. LBCAPSO yields the best solution and ranks first. LBPSO ranks last. Moreover, best results from BPSO for these functions are better than LBPSO and worse than BGSA. The results in Fig. 8 show that LBCAPSO has the best convergence rate and yields the best average solution for the function.

Table 15 also shows the results of applying the algorithms to the multimodal low-dimensional functions of Table 3 in the binary search space. The results show that LBCAPSO performs the same or better than the others. It ranks first overall and provides an optimal solution or good near-global optimum for the majority of the functions. The algorithms with local topology yield better solutions than the global algorithms. LBPSO performs well for functions F_{13} to F_{23}; however, it does not provide good results for the unimodal and the multimodal high-dimensional functions in the binary search space.

Fig. 9 shows the average best solution of the algorithms for 30 independent runs for function F_{22} with $n = 4$. The algorithms with local topology converge faster than the global ones for this function and LBCAPSO performs the best.

As shown in Tables 13–15, GSA uses the most average search time. BCAPSO, for both *gbest* and *lbest* models, uses less average search time than BPSO and LBPSO; the proposed algorithms require more time than BPSO and LBPSO, while they return better results. BCAPSOL has the best SD for the majority of functions in Tables 13–15. That is, BCAPSOL is more stable, robust and accurate than the others in solving high-dimensional functions.

To evaluate the ability of BCAPSO and LBCAPSO in the binary search space, a maximization binary function detailed in Table 4 is used. Several values of n (32, 64, 100 and 200) are run for the function, as shown in Table 16. The *dim* of particles or agents is equal to n. All algorithms are run 30 times for the function and their results are given in Table 16. The results show that only BCAPSO and LBCAPSO provide the optimal solution for all cases and rank first on average. Though BGSA returns good results, it requires a long time to obtain them. The results show that BPSO and LBPSO return the worst results as n increases and cannot tune themselves to obtain good results. The results in Fig. 10 show the faster convergence rate of BCAPSO and LBCAPSO for F_{24} with $n = 200$; in contrast, BPSO and LBPSO cannot obtain good results for this function.

4.3. Overall performance comparison of algorithms

In this section, the overall rank and mean absolute error (MAE) of the algorithms are calculated to compare the performance of the proposed methods with the other algorithms in the real and the binary search spaces. The comparisons are performed based on the results of 23 functions in Tables 5–7 for the real search space and those of 24 functions in Tables 13–16 for the binary search space. For Table 16, the results of $n = 200$ are used.

Table 17 summarises the overall ranking of the algorithms. As shown in this table, LCAPSO in the real search space and LBCAPSO in the binary search space rank first and the CAPSO and BCAPSO rank second in their corresponding search spaces. PSO with local topology ranks lowest in both the real and the binary search spaces.

The MAE is a statistical measure that shows how far estimates or forecasts are from the actual values. It is computed as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|t_i - y_i|, \tag{39}$$

where n is the number of functions, t_i is the global optimum of the ith function and y_i is the average best result of the ith function obtained by the algorithms.

The mean absolute error (MAE) in the real and the binary search spaces is shown in Fig. 11. The LCAPSO results had the smallest MAE in real search space and BLCAPSO the smallest in the binary search space. The MAE from the BLCAPSO runs (0.559) is the minimum error from among all the algorithms. This indicates that the binary mode of the proposed algorithm performs well in the search space and provides the global optimum or a good near-global optimum for each function.

5. Conclusions

In this study, a new optimization algorithm, centripetal accelerated particle swarm optimization (CAPSO) is introduced for both real and binary search spaces. CAPSO is inspired by Newton's laws of motion and the PSO algorithm. It is a global search algorithm with several advantages. Because of these advantages, the algorithm is convenient to use. The advantages include the following: CAPSO is easy to implement; it is not sensitive to the scale of the design variables; it is not necessary to tune CAPSO to any algorithmic parameter; and it is easy to parallelise it to perform concurrent processing. The algorithm applies *gbest* and *lbest* models, similar to PSO, to find the best solution. A set of standard benchmarks, including unimodal and multimodal functions, are used to evaluate the algorithm in the search spaces and the average results are compared with the GSA and PSO algorithms.

In the real search space, CAPSO shows significantly superior performance and uses the least average search time for the unimodal functions. For the multimodal functions, LCAPSO provides good results. GSA, PSO and LPSO cannot tune themselves with increasing functions dimension of 100 and 200 in both the unimodal and the multimodal high-dimensional functions, thus delivering poor results. The comparisons of CAPSO and LCAPSO with well-known PSO algorithms in the literature indicate that the proposed algorithms yield better results.

In the binary search space, the results from BCAPSO and LBCAPSO are better than those from BGSA, BPSO and LBPSO for the unimodal functions. For the multimodal functions, LBCAPSO performs well. The results for the max-ones function as a binary function show that both BCAPSO and LBCAPSO obtain the best solution and converge faster in all cases.

PSO and BPSO perform better than LPSO and LBPSO for the unimodal and multimodal high-dimensional functions but the results are opposite for the multimodal low-dimensional functions. GSA takes the longest average search time.

Acknowledgements

This research is supported by the Ministry of Higher Education (MOHE – LRGS/TD/2011/UTM/ICT/03-4L805). The authors thank the INS Editorial Board and the anonymous reviewers who contributed enormously to this work. They also wish to extend their grateful appreciation to Dr. Esmat Rashedi for providing her GSA and BGSA programs. They would like to thank *Soft Computing Research Group (SCRG)*, Universiti Teknologi Malaysia (UTM), Johor Bahru Malaysia, for supporting this study.

Appendix A

In this section, the details of functions of Table 3 are shown in Tables A.1–A.8.

Table A.1
Optimum values of functions in Table 3.

Test function	X_{opt}	F_{opt}
F_{14}	$(-32,32)$	1
F_{15}	$(0.1928,0.1908,0.1231,0.1358)$	0.00030
F_{16}	$(0.089,-0.712)$, $(-0.089,0.712)$	−1.0306
F_{17}	$(-3.14,12.27)$, $(3.14,2.275)$, $(9.42,2.42)$	0.398
F_{18}	$(0,-1)$	3
F_{19}	$(0.114,0.556,0.852)$	−3.86
F_{20}	$(0.201,0.15,0.477,0.275,0.311,0.657)$	−3.32
F_{21}	5 Local minima in a_{ij}, $i = 1,\ldots,5$	−10.1532
F_{22}	7 Local minima in a_{ij}, $i = 1,\ldots,7$	−10.4028
F_{23}	10 Local minima in a_{ij}, $i = 1,\ldots,10$	−10.5363

Table A.2
$a_{i\ j}$ in F_{14}.

A_{ij}
$\begin{pmatrix} -32,-16,0,16,32,-32,\ldots,0,16,32 \\ -32,-32,-32,-32,-16,\ldots,32,32,32 \end{pmatrix}$

Table A.3
a_i and b_i in F_{15}.

i	1	2	3	4	5	6	7	8	9	10	11
a_i	0.1957	0.1947	0.1735	0.1600	0.0844	0.0627	0.0456	0.0342	0.0323	0.0235	0.0246
b_i^{-1}	0.25	0.5	1	2	4	6	8	10	12	14	16

Table A.4
a_{ij} and c_i in F_{19}.

i	a_{ij}, $j = 1, 2, 3$			c_i
1	3	10	30	1
2	0.1	10	35	1.2
3	3	10	30	3
4	0.1	10	30	3.2

Table A.5
p_{ij} in F_{19}.

i	p_{ij}, $j = 1, 2, 3$		
1	0.3689	0.1170	0.2673
2	0.4699	0.4387	0.7470
3	0.1091	0.8732	0.5547
4	0.03815	0.5743	0.8828

Table A.6
a_{ij} and c_i in F_{20}.

i	a_{ij}, $j = 1, 2, 3, 4, 5, 6$						c_i
1	10	3	17	3.5	1.7	8	1
2	0.05	10	17	0.1	8	14	1.2
3	3	3.5	1.7	10	17	8	3
4	17	8	0.05	10	0.1	14	3.2

Table A.7
p_{ij} in F_{20}.

i	p_{ij}, $j = 1, 2, 3, 4, 5, 6$					
1	0.131	0.169	0.556	0.012	0.828	0.588
2	0.232	0.413	0.830	0.373	0.100	0.999
3	0.234	0.141	0.352	0.288	0.304	0.665
4	0.404	0.882	0.873	0.574	0.109	0.038

Table A.8
a_{ij} and c_i in F_{21}, F_{22} and F_{23}.

i	a_{ij}, $j = 1, 2, 3, 4$				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

References

[1] P. Avishek, J. Maiti, Development of a hybrid methodology for dimensionality reduction in mahalanobis-taguchi system using mahalanobis distance and binary particle swarm optimization, Expert System Application 37 (2) (2010) 1286–1293.
[2] Z. Beheshti, S.M. Shamsuddin, A review of population-based meta-heuristic algorithms, International Journal of Advances in Soft Computing and its Applications 5 (1) (2013) 1–35.
[3] Z. Beheshti, S.M. Shamsuddin, S. Hasan, MPSO: median-oriented particle swarm optimization, Applied Mathematics and Computation 219 (11) (2013) 5817–5836.
[4] Z. Beheshti, S.M. Shamsuddin, S.S. Yuhaniz, Binary accelerated particle swarm algorithm (BAPSA) for discrete optimization problems, Journal of Global optimization (2012) 1–23, http://dx.doi.org/10.1007/s10898-012-0006-1 (in press).
[5] F.V.D. Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, Information Sciences 176 (8) (2006) 937–971.
[6] J.M. Bishop, Stochastic searching networks, in: Proceedings 1st IEE Conference on Artificial Neural Networks, 1989, pp. 329–331.
[7] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, 1999.

[8] D. Bratton, J. Kennedy, Defining a standard for particle swarm optimization, in: Proceedings of the 2007 IEEE Swarm Intelligence Symposium, 2007, pp. 120–127.
[9] K. Cao, X. Yang, X. Chen, Y. Zang, J. Liang, J. Tian, A novel ant colony optimization algorithm for large-distorted fingerprint matching, Pattern Recognition 45 (1) (2012) 151–161.
[10] P. Chakraborty, S. Das, G.G. Roy, A. Abraham, On convergence of the multi-objective particle swarm optimizers, Information Sciences 181 (8) (2011) 1411–1425.
[11] Y. Chakrapani, K.S. Rajan, Genetic algorithm applied to fractal image compression, ARPN Journal of Engineering and applied Sciences 4 (1) (2009) 53–58.
[12] L.Y. Chuang, H.W. Chang, C.J. Tu, C.H. Yang, Improved binary PSO for feature selection using gene expression data, Computational Biology and Chemistry 32 (2008) 29–38.
[13] J.-F. Connolly, E. Granger, R. Sabourin, An adaptive classification system for video-based face recognition, Information Sciences 192 (2012) 50–70.
[14] E. Cuevas, D. Zaldívar, M. Pérez-Cisneros, H. Sossa, V. Osuna, Block matching algorithm for motion estimation based on artificial bee colony (ABC), Applied Soft Computing 13 (6) (2013) 3047–3059.
[15] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man and Cybernetics – Part B 26 (1) (1996) 29–41.
[16] I. Ellabib, P. Calamai, O. Basir, Exchange strategies for multiple ant colony system, Information Sciences 177 (5) (2007) 1248–1264.
[17] J.D. Farmer, N.H. Packard, A.S. Perelson, The immune system, adaptation and machine learning, Physica D 2 (1986) 187–204.
[18] A. Forestiero, C. Pizzuti, G. Spezzano, A single pass algorithm for clustering evolving data streams based on swarm intelligence, Data Mining and Knowledge Discovery 26 (1) (2012) 1–26.
[19] L. Gao, Y. Wang, Z. Tang, X. Lin, Newspaper article reconstruction using ant colony optimization and bipartite graph, Applied Soft Computing 13 (6) (2013) 3033–3046.
[20] W.F. Gao, S. Y Liu, L.L. Huang, Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique, Communications in Nonlinear Science and Numerical Simulation 17 (11) (2012) 4316–4327.
[21] D. Holliday, R. Resnick, J. Walker, Fundamentals of Physics, John Wiley and Sons, 1993.
[22] Y. Jiang, T. Hu, C.C. Huang, X. Wu, An improved particle swarm optimization algorithm, Applied Mathematics and Computation 193 (1) (2007) 231–239.
[23] Y.-T. Juang, S.-L. Tung, H.-C. Chiu, Adaptive fuzzy particle swarm optimization for global optimization of multimodal functions, Information Sciences 181 (2011) 4539–4549.
[24] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report-TR06-Erciyes University, 2005.
[25] A. Kaveh, S. Talatahari, A novel heuristic optimization method: charged system search, Acta Mechanica 213 (3–4) (2010) 267–289.
[26] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948.
[27] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of IEEE International Conference on Computational Cybernetics and Simulation, vol. 5, 1997, pp. 4104–4108.
[28] J. Kennedy, R. Mendes, Population structure and particle swarm performance, in: Proceedings of IEEE Congress on Evolutionary Computation, vol. 2, 2002, pp. 1671–1676.
[29] J. Kennedy, R. Mendes, Neighborhood topologies in fully informed and best-of-neighborhood particle swarms, IEEE Transactions on Systems, Man, and Cybernetics Part C 36 (4) (2006) 515–519.
[30] P. Korošec, J. Šilc, B. Filipič, The differential ant-stigmergy algorithm, Information Sciences 192 (2012) 82–97.
[31] X. Li, M.F. Baki, Y.P. Aneja, An ant colony optimization metaheuristic for machine-part cell formation problems, Computers and Operations Research 37 (12) (2010) 2071–2081.
[32] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Transactions on Evolutionary Computation 10 (3) (2006) 281–295.
[33] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer, in: Proceedings of IEEE Swarm Intelligence Symposium, 2005, pp. 124–129.
[34] M.B. Lin, J.F. Lee, G.E. Jan, A lossless data compression and decompression algorithm and its hardware architecture, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 14 (9) (2006) 925–936.
[35] Q. Liu, L. Wang, X. Liao, Stability analysis of swarms with interaction time delays, Information Sciences 192 (2012) 244–254.
[36] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler maybe better, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 204–210.
[37] A. Muruganandham, R.S.D. Wahida-banu, Effective MSE optimization in fractal image compression, International Journal of Computer Science and Information Security 8 (2) (2010) 238–243.
[38] R. Neapolitan, K. Naimipour, Foundations of algorithms using C++ pseudo code, third ed., Jones and Bartlett, 2004.
[39] F. Neri, E. Mininno, G. Iacca, Compact particle swarm optimization, Information Sciences 239 (2013) 96–121.
[40] M. Pant, T. Radha, V.P. Singh, A new particle swarm optimization with quadratic interpolation, in: Proceedings of IEEE International Conference on Computational Intelligence and Multimedia Applications, 2007, pp. 55–60.
[41] K.E. Parsopoulos, M.N. Vrahatis, UPSO: a unified particle swarm scheme, in: Lecture series on Computer and Computational Sciences, vol. 1, 2004, pp. 868–873.
[42] S.N. Qasem, S.M. Shamsuddin, Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis, Applied Soft Computing 11 (2011) 1427–1438.
[43] S.N. Qasem, S.M. Shamsuddin, S.Z.M. Hashim, M. Darus, E. Al-Shammari, Memetic multiobjective particle swarm optimization-based radial basis function network for classification problems, Information Sciences 239 (2013) 165–190.
[44] P. Rabanal, I. Rodríguez, F. Rubio, Using river formation dynamics to design heuristic algorithms, Lecture Notes in Computer Science 4618/2007 (2007) 163–177.
[45] E. Rashedi, S. Nezamabadi, S. Saryazdi, GSA: a gravitational search algorithm, Information Sciences 179 (13) (2009) 2232–2248.
[46] E. Rashedi, S. Nezamabadi, S. Saryazdi, BGSA: binary gravitational search algorithm, Natural Computing 9 (3) (2010) 727–745.
[47] A. Ratnaweera, S. Halgamuge, H. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 240–255.
[48] F.J. Rodriguez, M. Lozano, C. García-Martínez, J.D. González-Barrera, An artificial bee colony algorithm for the maximally diverse grouping problem, Information Sciences 230 (2013) 183–196.
[49] R.K. Santhi, S. Subramanian, Adaptive binary PSO based unit commitment, International Journal of Computer Applications 15 (4) (2011) 1–6.
[50] B. Schutz, Gravity From the Ground Up, Cambridge University Press, 2003.
[51] H. Shah-Hosseini, Problem solving by intelligent water drops, in: Proceedings of IEEE Congress on Evolutionary Computation, 2007, pp. 3226–3231.
[52] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: Proceedings of IEEE International Conference on Evolutionary Computation, 1998, pp. 69–73.
[53] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, in: Proceedings of IEEE Congress on Evolutionary Computation, 2005, pp. 1–50.
[54] M. Taherdangkoo, M.H. Shirzadi, M. Yazdi, M.H. Bagheri, A robust clustering method based on blind, naked mole-rats (BNMR) algorithm, Swarm and Evolutionary Computation 10 (2013) 1–11.

[55] C. Tharini, P. VanajaRanjan, Design of modified adaptive Huffman data compression algorithm for wireless sensor network, Journal of Computer Science 5 (6) (2009) 466–470.
[56] P.K. Tripathi, S. Bandyopadhyay, S.K. Pal, Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients, Information Sciences 177 (22) (2007) 5033–5049.
[57] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1997) 67–82.
[58] T.H. Wu, C.C. Chang, S.H. Chung, A simulated annealing algorithm for manufacturing cell formation problems, Expert System Application 34 (3) (2008) 1609–1617.
[59] D. Yang, L. Jiao, M. Gong, F. Liu, Artificial immune multi-objective SAR image segmentation with fused complementary features, Information Sciences 181 (13) (2011) 2797–2812.
[60] X. Yang, J. Yuan, J. Yuan, H. Mao, A modified particle swarm optimizer with dynamic adaptation, Applied Mathematics and Computation 189 (2) (2007) 1205–1213.
[61] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation 3 (1999) 82–102.
[62] X. Yuan, H. Nie, A. Su, L. Wang, Y. Yuan, An improved binary particle swarm optimization for unit commitment problem, Expert Systems with Applications 36 (4) (2009) 8049–8055.
[63] Z.H. Zhan, J. Zhang, Y. Li, H.S. Chung, Adaptive particle swarm optimization, IEEE Transactions on Systems, Man and Cybernetics – Part B 39 (6) (2009) 1362–1381.
[64] Z.-H. Zhan, J. Zhang, Y. Li, Y.-H. Shi, Orthogonal learning particle swarm optimization, IEEE Transactions on Evolutionary Computation 15 (6) (2011) 832–847.
[65] W. Zhao, C.E. Davis, A modified artificial immune system based pattern recognition approach – an application to clinical diagnostics, Artificial Intelligence in Medicine 52 (1) (2011) 1–9.