Applied Soft Computing Journal 98 (2021) 106833

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

A carnivorous plant algorithm for solving global optimization problems

Ong Kok Meng, Ong Pauline *, Sia Chee Kiong

Faculty of Mechanical and Manufacturing Engineering, Universiti Tun Hussein Onn Malaysia (UTHM), 86400 Parit Raja, Batu Pahat, Johor, Malaysia

ARTICLE INFO

Article history:
Received 1 June 2020
Received in revised form 5 October 2020
Accepted 15 October 2020
Available online 6 November 2020

Keywords:
Carnivorous plant algorithm
Metaheuristic algorithm
Optimization
Population-based algorithm
Robotic arm

ABSTRACT

In this study, a novel metaheuristic algorithm, namely, carnivorous plant algorithm (CPA), inspired by how the carnivorous plants adapting to survive in the harsh environment, was proposed. The CPA was first evaluated on thirty well-known benchmark functions with different characteristics and seven CEC 2017 test functions. Its convergence characteristic and computational time were analysed and compared with seven widely used metaheuristic algorithms, with the superiority was validated using the Wilcoxon signed-rank test. The applicability of the CPA was further examined on mechanical engineering design problems and a real-world challenging application of controlling the orientation of a five degree-of-freedom robotic arm. Experimental simulations demonstrated the supremacy of the CPA in solving global optimization problems.

1. Introduction

Optimization is the process of searching the optimal parameters of a given objective function. For the single-objective optimization problem, the mathematical model can be written as:

$$\begin{aligned} &\text{Minimize/Maximize: } f\left(\vec{x}\right),\ \vec{x} = x_1, x_2, \ldots, x_{n-1}, x_n \\ &\text{Subject to: } g_j\left(\vec{x}\right) \geq 0, j = 1, 2, \ldots, J \\ &\qquad\qquad h_k\left(\vec{x}\right) = 0, k = 1, 2, \ldots, K \\ &\qquad\qquad Lb_i \leq x_i \leq Ub_i, i = 1, 2, \ldots, n \end{aligned} \tag{1}$$

where $f\left(\vec{x}\right)$ is the function of the single-objective optimization problem, $\vec{x} = x_1, x_2, \ldots, x_{n-1}, x_n$ is the inputs, n is the number of inputs, $g_j\left(\vec{x}\right)$ is the inequality constraint, J is the number of inequality constraints, $h_k\left(\vec{x}\right)$ is the equality constraint, K is the number of equality constraints, Lb_i is the lower boundary of the ith input, and Ub_i is the upper boundary of the jth input.

Before the emergence of the stochastic methods, the deterministic approaches, such as hill-climbing, Simplex method, Bundle method and Newton–Raphson, are commonly used to solve the optimization problem [1]. Despite these techniques continue to receive widespread attention in multitudinous domain, challenges still remain for local optima entrapment if given a poorly defined starting point [2]. Besides, the deterministic algorithm, especially the gradient-based approach, is ineffective in solving the problem that its derivative is unknown or computational expensive [3], and thus limited the applicability in solving complex real-world problems.

The rise of the stochastic method as a promising alternative way to the deterministic approach indeed lies with its inherent randomness and gradient-free calculation. The use of randomness can be found in different components of a stochastic optimization method, such as the crossover and mutation operators in the popular genetic algorithm (GA) and the hill-climbing method with random restart, allowing the stochastic method to escape from local optima. Moreover, moving the solutions towards the global optima is based on the evaluation of the objective function and a set of rules. This is in contrast with the mathematical optimization method that requires the calculation of gradient by derivative of the objective function. Given this superiority, rapid research in stochastic method spawns the development of a broad range of optimization solutions, falling into two categories: heuristic and metaheuristic [4].

The bio-inspired and population-based metaheuristic algorithms are gaining widespread interest today, often implemented in different domains. This is because the metaheuristic algorithm is simple to use, where it only needs the information of fitness function during optimization. Besides, the metaheuristic algorithm, which uses a set of solutions with probabilistic rules in finding the global optima in the search space, has also improved the success rate of optimization. In metaheuristic algorithm, two major elements, namely, exploration and exploitation, have played an important role during optimization. The exploration enables the algorithm to explore the promising areas in

* Corresponding author.
E-mail address: ongp@uthm.edu.my (Ong P.).

https://doi.org/10.1016/j.asoc.2020.106833
1568-4946/

the search space and also to escape from the local optima [5]. Meanwhile, exploitation enables the algorithm to obtain a highly accurate solution from the promising area [3]. In this regard, an algorithm with a good combination of these two elements will prevent itself from premature convergence in the early phase of the optimization process, and quickly converge towards the global optima at the end.

Most of the metaheuristic algorithms are inspired from nature, such as GA [6] - the first and most popular metaheuristic algorithm, imitates the biological evolution of mutation, recombination and selection of the biological systems. This has opened a new way of thinking for the researchers to link the nature to the mathematical computational skill to solve the challenging optimization problem. Since then, numerous metaheuristic algorithms have been developed and the list is still growing from time to time, which can be referred to [7].

These metaheuristic algorithms can be generally categorized based on their inspiration source as follows:

(a) Evolutionary techniques: GA [6] and differential evolution (DE) [8]. These algorithms are derived from biological evolution, such as mutation, crossover, selection and reproduction.
(b) Animal-based techniques: artificial bee colony (ABC) [9], particle swarm optimization (PSO) [10], cuckoo search algorithm (CSA) [11], bat algorithm (BA) [12], squirrel search algorithm (SSA) [13] and sailfish optimizer (SFO) [5]. Such algorithms are inspired from the behaviour of animals, for instance, bee, bird, bat, squirrel and fish.
(c) Plant-based techniques: invasive weed optimization (IWO) [14] and flower pollination algorithm (FPA) [15]. These algorithms imitate the plants' behaviour, where IWO mimics the process of weed invasion, while FPA simulates the pollination process of the flower.
(d) Human activity-based techniques: harmony search (HS) [16], teaching learning based optimization (TLBO) [17] and imperialist competitive algorithm (ICA) [18]. These kind of algorithms are derived from human activities, such as tuning guitar, teaching and learning method and colonization of an empire.
(e) Physic-based techniques: gravitational search algorithm (GSA) [19], water cycle algorithm (WCA) [20] and multiverse optimizer (MVO) [21]. These algorithms mimic the phenomena in earth and also universe, for example the law of gravity, the flow of rivers and streams towards the sea, as well as the concepts of white hole, black hole and wormhole.

Metaheuristic algorithms can also be classified into two groups, namely, single-based algorithm and population-based algorithm. In single-based algorithm, only one single solution is generated during the initialization. Hence, such algorithms require less number of function evaluation (NOFE) to solve a specific problem, but they experience premature convergence easily. For the population-based algorithm, multiple solutions are initialized and iteratively enhanced. Although these algorithms require larger NOFE, they can escape from local optima due to information sharing. Thus, population-based algorithms are preferable as they can explore the search space more effectively than a single-based algorithm, and move towards the promising regions of the search space based on the information exchange among the search agents. The examples of the single-based algorithm are Tabu search (TS) [22] and simulated annealing (SA) [23], while the examples of the population-based algorithms are GA, PSO, ABC, FPA and CSA.

Despite the consistently promising performance, a considerable amount of work has been continuously undertaken in the area of metaheuristic algorithm for performance enhancement, which can be divided into three main directions: improving the existing metaheuristic algorithm, hybridizing different metaheuristic algorithms, and proposing a new metaheuristic algorithm. Alteration of metaheuristic algorithm by incorporating different mathematical models into population initialization, local operator and global operator has been put forward to improve its optimization performance. The solution to this direction has seen numerous approaches, for instance, integration of metaheuristic algorithm with chaotic maps [24], adaptive strategy [25] and opposition-based learning strategy [26], are some of the proposed strategies. Hybridization of different metaheuristic algorithms while preserving the advantages of each algorithm and compensating the limitations with others, too, has been widely investigated in literature, exemplarily, the hybridization of grey wolf optimization (GWO) algorithm with CSA [27] and TLBO with neural network algorithm [28], to name a few.

The increasing complexity of real-world problem has prompted the development of more metaheuristic optimization approaches. One might question the need to have a new metaheuristic algorithm since there are many existing metaheuristic algorithms out there. A positive answer to this question is because of No Free Lunch (NFL) Theorem, proposed by Wolpert and Marcready [29]. According to the NFL theorem, if algorithm A outperformed algorithm B in the specific problem X, it is not necessary for algorithm A to outperform algorithm B in the specific problem Y. The performance of all algorithms is equally well in average. Concisely, there is no universal optimization procedure that works perfectly for all optimization problems and thus, the continuous flourish of the diversity of optimization algorithm is encouraged.

A newly proposed algorithm should be able to address the challenges concerning of high-dimensional design variables, existence of various constraints and search space with many local optima. It shall be able to search for the optimal solution within the shortest period of time despite a large number of design variables to be optimized. Besides, the increasing optimization difficulty due to the modelling constraints shall not impede the algorithm from reaching the optimal solutions with no violated constraints. Most importantly, a successful search mechanism shall not prone to stagnation due to the existence of local optima as in the multimodal function.

In this study, a new population-based metaheuristic algorithm, namely, carnivorous plant algorithm (CPA) is proposed. The CPA imitates how the carnivorous plants adapt to survive in the harsh environment, such as hunting insects for its food and pollinating for reproduction. To the best of our knowledge, the algorithm inspired by the survival skill of carnivorous plant has not yet been studied in the literature. It will be shown that the proposed CPA can successfully address the issues of high-dimensional design variables, existence of various constraints and search space with many local optima. The main contributions of this study are as follows:

(1) A novel bio-inspired CPA is proposed. The close mimicking of how the carnivorous plants adapt themselves to circumstances that are constantly changing is mathematically formulated.
(2) The performance of the proposed CPA is tested on 32 benchmark test functions. A thorough comparative analysis with other algorithms is performed using statistical analysis, convergence rate analysis and Wilcoxon's test.
(3) The practical application of the proposed algorithm is tested on eleven benchmark mechanical engineering design optimization problems and the results are compared with those available in literature.

Fig. 1. The carnivorous plant. (a) Pitcher Plant, (b) Sundew, (c) Bladderworts, (d) Venus Flytrap. *Source:* https://www.pinterest.com/pin/295056213082152740/, https://www.pulsk.com/640610/2/, https://www.pinterest.ca/pin/520095456952760120/, https://www.pinterest.com/pin/461830136786595642/.

(4) The effectiveness of the proposed algorithm is tested on a real-world application, which is to control the movement of 5-degree-of-freedom (DOF) robotic arm for gripping a target object accurately without colliding with any obstacle.

The rest of this paper is structured as follows: Section 2 presents the background and mathematical formulation of the CPA. The experimental analysis on benchmark test function and benchmark mechanical engineering design optimization problems are given in Section 3. Section 3 also provides the results of CPA in real-world application and lastly, the main findings of this study are concluded in Section 4.

2. Carnivorous plant algorithm (CPA)

2.1. Background

Most plants are direct food source for animals, but there are always exceptions. Carnivorous plants, also known as the insectivorous plants, are distinctly different from the common plant community. They usually grow in harsh habitats with scarce nutrients, for instances, swamp, marshes and muddy shores that are characterized as water abundant [30]. To gain additional nutrients like nitrogen and phosphorus for growth and reproduction, carnivorous plants attract, trap and consume animals such as flies, butterflies, lizard and mouse with their secreted enzymes [31].

A plant can only be considered as carnivorous if it possesses the abilities of attraction, trapping or killing, and digestion of the prey [32]. Some plants attract the prey for reproduction without killing them, and some immobilize or kill the attackers for defence purposes but do not digest the bodies. While some plants do absorb the nutrients from the dead animals, however, they do not have the ability to kill it. They merely consume the bodies found in the soil or on the leaf surface [31].

The pitcher plant, perhaps, is the most famous among the carnivorous plants (see Fig. 1(a)). It has an internal chamber with digestive enzymes exclusively used to break down the prey and release nutrients to be absorbed. To attract the prey, the outer surface of the chamber has vivid colours and alluring scent. The inner surface of the chamber is slippery, preventing the entrapped victim from climbing out. Drosera, also known as sundew, is the largest genera of carnivorous plants [33] (see Fig. 1(b)). Using its sweet and sticky mucilage, the Drosera attracts and traps the prey on the tentacles. Some special tentacles may even curl around the prey if prey movement is detected. Utricularia, also known as bladderworts, is an aquatic version of carnivorous plants [34] (see Fig. 1(c)). It has small and empty bladders growing from its stems, producing sugar for prey attraction. The inside of the bladder is partially vacuum. When the trigger hairs connected to the trapdoor are brushed by the moving prey, the trapdoor will open, causing water flows in to fill up the empty bladder. The prey will be trapped by this suction mechanism. Dionaea, also known as Venus flytrap, is another type of carnivorous plant uses different trapping mechanism to capture its prey (see Fig. 1(d)). The plant produces sweet-smelling nectar for prey attraction. When the prey touches the tiny hairs on the book-like leaves, the plant snaps shut the leaves around the prey within a second. Once closed, the plant starts digesting the prey with the leaves, acting as an external stomach.

2.2. Mathematical model of the carnivorous plant algorithm

The mathematical model used to simulate the attraction, trapping, digestion and reproduction strategies of the carnivorous plants are presented in this section. The CPA starts with initializing a set of solutions randomly. The solutions are then categorized as carnivorous plant and prey, and subsequently grouped for the growth and reproduction processes. Their fitness values are updated and all the solutions are combined. The process continues

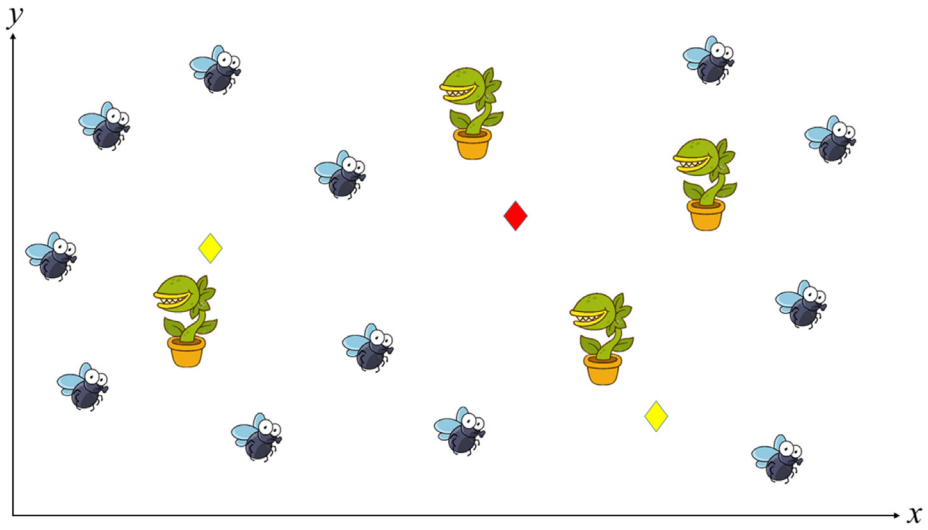


Fig. 2. A population of the carnivorous plants and the preys are staying in the wetland. The red diamond-shape indicates global optimum whereas the yellow diamond-shape denotes local optima.

until the termination condition is fulfilled. The detail of each process is explained as follows:

2.2.1. Initialization

The CPA is a population-based optimization algorithm and hence, it begins with initializing a population of potential solutions of the underlying problem. To start with, a population of n individuals, consisting of carnivorous plants and preys, are randomly initialized in a wetland. The number of carnivorous plants and preys are denoted as *nCPlant* and *nPrey*, respectively. The position of each individual is represented in a matrix as:

$$Pop = \begin{bmatrix} Individual_{1,1} & Individual_{1,2} & \cdots & \cdots & Individual_{1,d} \\ Individual_{2,1} & Individual_{2,2} & \cdots & \cdots & Individual_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ Individual_{n,1} & Individual_{n,2} & \cdots & \cdots & Individual_{n,d} \end{bmatrix} \tag{2}$$

where d is the number of dimensions, and n is the summation of *nCPlant* and *nPrey*. Each individual is initialized randomly using:

$$Individual_{i,j} = Lb_j + \left(Ub_j - Lb_j\right) \times rand \tag{3}$$

where Lb and Ub are the lower bound and upper bound of the search domain, respectively, with $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, d$. *rand* is a random number drawn from the range [0, 1].

From each ith individual, its fitness is evaluated by substituting each ith individual to the predefined fitness function. The obtaining fitness value is stored as:

$$Fit = \begin{bmatrix} f\left(Individual_{1,1}\right) & Individual_{1,2} & \cdots & \cdots & Individual_{1,d}\right) \\ f\left(Individual_{2,1}\right) & Individual_{2,2} & \cdots & \cdots & Individual_{2,d}\right) \\ & \vdots & & & \\ & \vdots & & & \\ f\left(Individual_{n,1}\right) & Individual_{n,2} & \cdots & \cdots & Individual_{n,d}\right) \end{bmatrix} \tag{4}$$

In Eq. (2), each ith individual indicates the solution vector of the optimization problem, while the fitness value in Eq. (4) depicts the quality of the particular solution vector. For minimization case, the lower the number of the fitness value, the higher the quality of that solution vector.

2.2.2. Classification and grouping

Next, each individual in Eq. (2) is sorted according to its fitness value in ascending order (consider the minimization problem). The top *nCPlant* solutions of the sorted population are considered as the carnivorous plants, *CP*, while the remaining solutions (*nPrey*) are the prey, *Prey*. The visualization of the carnivorous plants and the preys is presented in Fig. 2. The matrix of the sorted fitness and sorted population can be described as in Eqs. (5) and (6).

$$Sorted\_Fit = \begin{bmatrix} f_{CP(1)} \\ f_{CP(2)} \\ \vdots \\ f_{CP(nCPlant)} \\ f_{Prey(nCPlant+1)} \\ f_{Prey(nCPlant+2)} \\ \vdots \\ f_{Prey(nCPlant+nprey)} \end{bmatrix} \tag{5}$$

$$Sorted\_Pop = \begin{bmatrix} CP_{1,1} & CP_{1,2} & \cdots & \cdots & CP_{1,d} \\ CP_{2,1} & CP_{2,2} & \cdots & \cdots & CP_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ CP_{nCPlant,1} & CP_{nCPlant,2} & \cdots & \cdots & CP_{nCPlant,d} \\ Prey_{nCPlant+1,1} & Prey_{nCPlant+1,2} & \cdots & \cdots & Prey_{nCPlant+1,d} \\ Prey_{nCPlant+2,1} & Prey_{nCPlant+2,2} & \cdots & \cdots & Prey_{nCPlant+2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ Prey_{nCPlant+nPrey,1} & Prey_{nCPlant+nPrey,2} & \cdots & \cdots & Prey_{nCPlant+nPrey,d} \end{bmatrix} \tag{6}$$

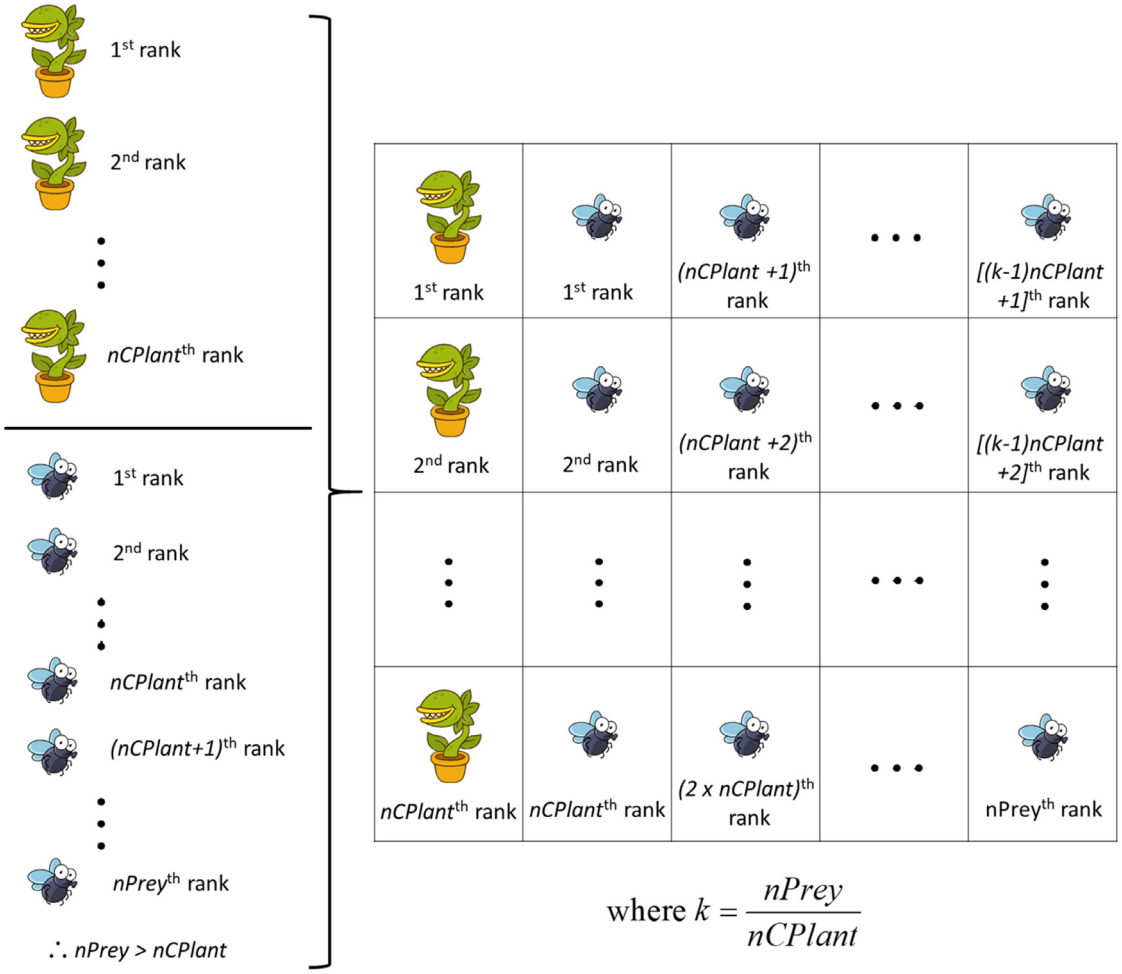


Fig. 3. The grouping process in CPA.

The process of grouping is required to simulate the environment of each carnivorous plant and its prey. During the grouping process, the prey with the best fitness value is allocated to the first-ranked carnivorous plant. Similarly, the second and third-ranked preys are assigned to the second and third-ranked carnivorous plants, respectively. The process is repeated until the *nCPlant*th-rank prey is distributed to the *nCPlant*th-rank carnivorous plant. Then, the *nCPlant+1*th-rank prey is assigned to the first-ranked carnivorous plant and so on. The grouping process in CPA is illustrated in Fig. 3. The grouping is essential to reduce the possibility of having many poor quality preys that will contribute to the growth of carnivorous plants, which is important to improve the survivability of the carnivorous plants.

2.2.3. Growth (exploration)

Due to the nutrient-poor soil, carnivorous plants attract, trap and digest the prey for their growth. The prey is lured to the plant by its sweet scent, but may successfully escape from the clutches of the carnivorous plants intermittently. Here, an attraction rate is introduced.

For each group, a prey is randomly chosen. If the attraction rate is higher than a random generated number, the prey is captured and digested by the carnivorous plant for the growth. The growth of the new carnivorous plant is modelled as:

$$NewCP_{i,j} = growth \times CP_{i,j} + (1 - growth) \times Prey_{v,j} \tag{7}$$

$$growth = growth\_rate \times rand_{i,j} \tag{8}$$

where $CP_{i,j}$ is the ith rank carnivorous plant, $Prey_{v,j}$ is the randomly chosen prey, the growth rate is the predefined value and *rand* is the random value chosen from the range [0, 1]. In CPA, there is only one carnivorous plant in each group, while the number of preys must be more than two. The attraction rate in CPA is assigned as 0.8 for most cases.

On the other hand, if the attraction rate is lower than the generated random value, the prey manages to escape from the trap and continues to grow, which is mathematically represented as:

$$NewPrey_{i,j} = growth \times Prey_{u,j} + (1 - growth) \times Prey_{v,j}, \quad u \neq v \tag{9}$$

$$growth = \begin{cases} growth\_rate \times rand_{i,j} & f(prey_v) > f(prey_u) \\ 1 - growth\_rate \times rand_{i,j} & f(prey_v) < f(prey_u) \end{cases} \tag{10}$$

where $Prey_{u,j}$ is another randomly selected prey in the ith rank group. The growth process of both carnivorous plant and prey is repeated until it reaches a predefined iteration, *group_iter*.

Eqs. (7) and (9) are used to direct the new solution towards the search space with good quality solution. To ensure this is also working for the growth of prey, Eq. (10) is introduced since the quality of the selected $Prey_u$ might be worse than $Prey_v$. As shown in Eqs. (7) and (9), the exploration of the algorithm is influenced by the growth rate. The higher the growth rate, the wider the exploration is and in turn, the higher possibility to miss the global optima. Hence, a suitable growth rate has to be selected.

Carnivorous Plant Algorithm

Define objective function $f(\vec{x}), \quad \vec{x}=(x_1,x_2,..,x_d)$
Define iteration number within a group, *group_iter*, attraction rate, *attraction_rate*, growth rate, *growth_rate*, reproduction rate, *reproduction_rate*, number of carnivorous plants, *nCPlant* and number of prey, *nPrey*
Initialize a population of *n* individuals with *d* dimension randomly
Evaluate the fitness of each individual
Sort the individuals based on the fitness value
Identify the best individual, *g** as the first rank carnivorous plant

Repeat until stopping condition is met
  Classify the top *nCPlant* individuals as carnivorous plants
  Classify the remaining *nPrey* individuals as prey
  Group the carnivorous plants and prey

  // Growth process of both carnivorous plants and prey
  for *i* = 1:*nCPlant*
    for *Group_cycle* = 1:*group_iter*
      if *attraction_rate* > a generated random number
        //The prey is trapped and digested
        Generate new carnivorous plant using Equation (7)
      else
        //The *prey* escapes from the trap
        Generate new prey using Equation (9)
    end for
  end for

  //Reproduction process of the first rank carnivorous plant
  for *i = 1:nCPlant*
    Generate new carnivorous plant based on the first rank carnivorous plant using Equation (11)
  end for

  Evaluate the fitness of each new carnivorous plant and new prey
  Combine the previous and newly generated carnivorous plants and preys
  Sort the individuals and select top *n*-ranked individuals to next generation
  Identify the current best individual, *g** as the first rank carnivorous plant
end while
Display the current best solution, *g**

Fig. 4. The pseudocode of CPA.

2.2.4. Reproduction (exploitation)

Carnivorous plant assimilates the nutrients from the prey and uses the nutrients for growth and reproduction. For the reproduction, only the first-ranked carnivorous plant, *i.e.*, the best solution in the population, is allowed to reproduce. This is to ensure that the exploitation of the CPA is only focusing on the best solution. Unnecessary exploitation on other solutions can be avoided and thus, leading to computational cost saving. The reproduction process for the first-ranked carnivorous plant is expressed as:

$$NewCP_{i,j} = CP_{1,j} + Reproduction\_rate \times rand_{i,j} \times mate_{i,j} \tag{11}$$

$$mate_{i,j} = \begin{cases} CP_{v,j} - CP_{i,j} & f(CP_i) > f(CP_v) \\ CP_{i,j} - CP_{v,j} & f(CP_i) < f(CP_v) \end{cases}, i \neq v \neq 1 \tag{12}$$

where $CP_{1,j}$ is the best solution, $CP_{v,j}$ is the randomly selected carnivorous plant and the reproduction rate is a predefined value for exploitation. This process is repeated for *nCPlant* times. During the reproduction process, *v* carnivorous plant is randomly reselected for every *j*th dimension. In the growth process, the prey is randomly reselected regardless of the *j*th dimension.

2.2.5. Fitness update and combination

The newly generated carnivorous plants and preys are combined with the previous population, resulting in a new group with $[n+nCPlant(group\_iter)+nCPlant]\times d$ dimension. To be more specific, the *n* individuals, $nCPlant(group\_iter)$ individuals and *nCPlant* individuals are the individuals from original population, growth process and reproduction process, respectively. Subsequently, this new group of individuals are sorted in ascending order according to the fitness value. The top *n*-rank individuals are then selected from this group as the new candidate solutions. Thus, this elitism selection strategy ensures that the fitter solutions are selected for reproduction in the next generation.

2.2.6. Stopping criterion

The processes of classification, grouping, growth and reproduction are repeated until the stopping condition is satisfied.

Fig. 4 presents the pseudocode of the CPA. The code of CPA has been made available at https://www.mathworks.com/matlabcentral/fileexchange/82703-carnivorous-plant-algorithm-cpa.

3. Simulation results and discussion

The performance of CPA was assessed on solving 30 classical benchmark test functions and further evaluated with seven bound-constrained benchmark test functions from CEC 2017. The classical benchmark test functions can be divided into four types:

De Jong Test Function, Dim = 15

Growth Rate \ Reproduction Rate	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
1.0	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1350824	106696	5980	5862	5388	5260	5218	5270	5250	5066	5614	5336
1.1	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1201340	255644	5902	5184	5050	4962	4834	4950	5006	5242	5106	5316
1.2	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	952080	305538	5802	4712	4862	4668	4710	4518	4644	4760	4964	5016
1.3	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1300722	653222	104916	5128	4726	4484	4394	4388	4344	4530	4522	4630	4550
1.4	1500000	1500000	1500000	1500000	1500000	1500000	1450168	1350412	851992	303622	54266	4376	4212	4144	4124	4128	4044	4082	4220	4414	4326
1.5	1500000	1500000	1500000	1500000	1500000	1400238	1250712	652280	153868	3942	3804	3652	3970	3884	3838	3896	3892	4032	3936	3982	4162
1.6	1500000	1500000	1500000	1500000	1500000	1001522	403116	203660	3752	3782	3818	3738	3776	3828	3808	3792	3932	3916	4052	4018	4096
1.7	1500000	1500000	1500000	1500000	1355978	542922	203632	3946	3936	3918	3862	3926	3980	3996	3960	4098	4122	4086	4214	4202	4148
1.8	1500000	1500000	1414106	907112	264632	112958	4756	4054	4136	4152	4154	4258	4328	4260	4340	4454	4370	4550	4542	4604	4486
1.9	1450310	1217878	815864	220660	5950	4762	4412	4268	4412	4530	4614	4732	4688	4876	4904	4930	4884	5000	4980	5036	5120
2.0	26776	12826	11000	6768	5242	4744	4530	4680	4770	4940	5066	5250	5338	5486	5488	5540	5692	5566	5618	5810	5728
2.1	9384	7800	6782	5970	5006	4776	4816	4856	5170	5394	5682	5848	5964	6144	6258	6436	6424	6586	6502	6528	6604
2.2	8268	7052	6108	5588	5258	4824	4984	5196	5600	5762	6200	6480	6696	7020	7178	7198	7454	7460	7496	7466	7730
2.3	7034	6752	6128	5298	5230	5182	5176	5596	5812	6424	6872	7278	7458	7784	8092	8288	8232	8698	8656	8772	8782
2.4	6972	6620	5934	5590	5238	5166	5292	5844	6194	6738	7300	7738	8364	8812	9200	9498	9860	10260	10078	10000	10266
2.5	6954	6404	5786	5722	5292	5200	5472	6046	6334	7178	7898	8700	9226	10072	10210	10848	11348	11452	11572	11698	12068
2.6	6960	6248	5892	5772	5386	5458	5670	6086	6658	7696	8354	9368	10022	10822	11946	12290	12530	13214	13734	13520	13682
2.7	6988	6370	6120	5780	5606	5486	5620	6290	7114	8022	8826	10216	10948	12084	13024	13778	14516	15092	15514	16074	16200
2.8	7138	6704	6234	5692	5722	5614	5896	6560	7168	8214	9418	10810	12092	13334	14642	15804	16584	17212	17592	17926	18334
2.9	7394	6908	6280	5968	5892	5896	5900	6428	7498	8500	10124	11050	13062	14426	16142	16784	18272	19324	20512	21794	21498
3.0	7510	6758	6356	6052	5940	6006	6192	6480	7358	8860	10362	11828	14266	15602	17754	19148	20356	22388	22252	24084	24756

Fig. 5. The number of function evaluation required to solve De Jong test function.

unimodal, multimodal, separable and non-separable, and were used to evaluate the exploration and exploitation capabilities of the CPA. To be specific, a unimodal function with only one global optimum in the search space is used to test the exploitation capability of an algorithm. Meanwhile, a multimodal function with many local optimums is used to examine the exploration capability of an algorithm. To reach the global optimum, the algorithm should have the capability to effectively explore the search space without getting trapped in local optima. A separable function, on the other hand, is easier to solve as compared to a non-separable function. This is because the separable function contains design variables that are independent of each other, while non-separable function has design variables that are interrelated to each other, and hence, increasing the difficulty in reaching the global optima. For the CEC test functions, seven shifted and rotated test functions were utilized to ensure the proposed CPA has no structural bias. The simulation results were compared to FPA, improved PSO with inertia function, DE, CSA, BAT, SSA, GA, Linear Population Size Reduction Success History based Adaptive Differential Evolution (LSHADE) and Effective Butterfly Optimizer with Covariance Matrix Adapted Retreat Phased (EBOwithCMAR). Besides, the complexity of the CPA in terms of Big O notation was developed. Furthermore, seven mechanical engineering design problems and four CEC2011 problems were optimized using the CPA and subsequently compared with other competitive algorithms available in literature. Lastly, the performance of CPA in solving a real-world problem was assessed, specifically, the control of a 5-DOF robotic arm. All simulations were performed on a computer with 3.50 GHz Intel (R) Xeon (R) CPU E3-1240 v5 and 32GB of RAM using MATLAB R2016a.

3.1. Test I – Parameter sensitivity analysis

The parameters of growth rate and reproduction rate of CPA have to be assigned judiciously, as they affect the optimization performance of CPA and may cause stagnation in local optima if improperly selected. Thus, the parameter sensitivity analysis was performed so that the appropriate values of these parameters can be identified. In this analysis, five test functions, namely, De Jong, Ackley, Schwefel 1.2, Sum of Different Power and Schwefel 2.21, were selected. By varying the reproduction rate and growth rate from 0.5 to 2.5 and 1.0 to 3.0, respectively, the NOFE required by the CPA to solve the test function was recorded. The CPA was simulated for 30 independent runs, and subsequently, the required NOFEs for all runs were averaged.

The convergence analysis of the CPA by varying the reproduction rate and growth rate and evaluated on several benchmark test functions is summarized in Figs. 5 to 9. In these figures, the red colour means that the CPA is getting stuck in local optima and thus, these values are not the appropriate selection. Meanwhile, the green colour means that these values of reproduction rate and growth rate are recommended for that particular test function. It can be seen that the CPA took a considerably lower number of NOFE to converge. By averaging the NOFE required for each test function, as illustrated in Fig. 10, the recommended value of reproduction rate is within 1.5 and 2.5, while the suggested range of growth rate is within 2.0 and 3.0.

3.2. Test II – Classical benchmark test functions

To examine the optimization performance of CPA, thirty frequently used benchmark test functions were chosen from literature [35]. The detailed information of each test function is tabulated in Table 1. The convergence performance of the CPA was compared against other well-known algorithms, specifically, FPA, Improved PSO, DE, CSA, BAT, SSA and GA. In this study, all algorithms were simulated for 30 independent runs. Each simulation was terminated when the obtained tolerance error was less than 10^{-5}, or when the NOFE reached the maximum predefined value. If the algorithm failed to reach the tolerance error within the predefined number of NOFE, the respective run was considered unsuccessful. The parameter setting of the proposed CPA was in accordance with the parameter sensitivity analysis as discussed in Section 3.1. The parameters of other competing algorithms

Ackley Test Function, Dim = 15

Growth rate \ Reproduction rate	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
1.0	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	661534	260356	110362	208966	109378	59364	9768	158874	109336	158912	59936
1.1	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1005002	211148	159854	60114	207928	158692	208376	108788	258006	208306	208340
1.2	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1400738	904828	260962	457542	207846	157872	108358	158258	158296	257576	108292	158326
1.3	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1450426	656918	59370	207790	157386	107364	157434	57742	157530	107994	157764	157936
1.4	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1450316	953138	256896	107700	107144	106864	107056	156712	256144	206774	107382	206786	256674
1.5	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1350674	803578	107214	6986	106314	156260	156258	56736	206054	56856	57022	106636	355572	206374
1.6	1500000	1500000	1500000	1500000	1500000	1450264	1201460	704274	106672	106320	205504	56396	56434	155986	205940	106396	156302	106490	156270	106396	56812
1.7	1500000	1500000	1500000	1500000	1500000	1400456	554678	160786	106374	56534	205790	106412	6840	6928	56654	56846	106674	106704	106766	57068	57076
1.8	1500000	1500000	1500000	1500000	1350958	804918	256396	255920	106604	106758	156510	57040	156770	107022	256282	57338	57546	57500	57522	107288	57732
1.9	1500000	1500000	1450656	1351290	708330	406314	156952	206462	107082	107364	57804	157326	8260	107820	8460	8554	8606	8710	8754	58510	8758
2.0	1453036	1450552	1154190	1202378	754564	654796	356130	256714	8272	58320	8992	207784	9284	9388	59210	59314	59400	9826	9824	9998	59668
2.1	1401076	1351530	1301738	953734	853996	654684	356382	257312	58664	59224	59462	59828	60194	10658	10858	11038	11102	11306	60900	11462	61190
2.2	1400964	1400680	1251838	1251770	903548	654754	257162	108496	158620	10322	60540	160258	11786	12198	12518	12646	12848	62516	13290	13266	13272
2.3	1500000	1400816	1301326	1251530	903510	853778	704912	208182	109642	60926	12016	62282	13332	13590	14170	14456	14778	15038	15302	15166	15392
2.4	1400770	1450362	1450300	1350910	1102392	456390	357198	158962	159776	61328	12900	13824	64432	15532	16200	16796	16940	17258	17784	17862	17960
2.5	1450500	1450336	1350986	1301420	1102562	1003250	655432	60242	209568	62232	14062	15108	16268	17386	18402	18682	19322	19946	20844	20786	70316
2.6	1450388	1351154	1400666	1400650	1102548	953512	556218	407834	309540	62830	113852	16448	17904	19304	20334	21778	21908	23294	23886	24208	24450
2.7	1450452	1400916	1251838	1350998	1201906	903894	556502	656198	161108	211996	164414	18336	69098	21682	72616	24440	25980	26380	75988	27906	28040
2.8	1500000	1351176	1400928	1500000	1152242	1053040	755208	259052	310254	163324	66096	19238	21224	23910	74794	27534	28598	29702	80232	32486	81714
2.9	1252078	1450374	1400742	1301350	1202152	1003378	804974	507454	409522	64512	17254	69476	23196	26034	28504	30834	32290	34498	36428	37728	37362
3.0	1351254	1301442	1400692	1450348	1251704	1102702	1102874	557606	360266	163726	67626	70504	73644	27578	31000	34116	36724	87762	88668	41816	43776

Fig. 6. The number of function evaluation required to solve Ackley test function.

Schwefel 1.2 Test Function, Dim = 15

Growth rate \ Reproduction rate	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
1.0	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	484434	38026	30222	28908	26102	25272	25230	25394	24672	26068	26336
1.1	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	957010	41396	31956	28912	26244	27188	25066	25074	25582	26076	26826
1.2	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1184874	40978	32352	27602	26524	25744	26222	24144	25660	25474	25032
1.3	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	933382	38662	29452	25954	26192	25836	25604	24844	24314	24390	25510
1.4	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	110356	33060	26842	26926	24768	23748	23948	23790	23406	23210	23156
1.5	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	944214	42106	27906	23802	22700	21686	21668	21422	21000	22208	21908	22114
1.6	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1473394	199998	25070	20604	21626	20790	20468	19434	20112	19870	20068	20558	20312
1.7	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	113006	25270	20322	18826	18128	17950	18524	18782	18938	19004	19788	19404	20440
1.8	1500000	1500000	1500000	1500000	1500000	1500000	1301108	51564	23842	18114	17394	17358	17640	17834	16972	18302	18912	18534	18738	19304	19958
1.9	1500000	1500000	1500000	1500000	1500000	306964	44742	22890	18038	17386	16804	16790	17286	17536	17710	18104	18324	18638	18904	19296	20222
2.0	1500000	1500000	1357860	392718	107032	38514	23624	18762	16668	16566	16530	17146	17586	18646	18806	19888	19504	19746	20668	19600	20566
2.1	480664	269458	112286	65576	37650	24278	18990	16728	16340	17128	17992	18600	18984	20682	20208	20458	21222	21354	22234	22788	22502
2.2	115004	80536	48982	34520	25768	20764	17156	16084	16978	17986	18930	19852	21290	21606	21854	23244	23812	24468	23748	25078	25032
2.3	70792	45866	36504	29764	22612	17708	17156	16362	17610	18420	21034	21604	23702	24216	25182	25968	26670	27056	27336	27496	27576
2.4	48486	37068	30816	25292	20712	17902	16852	16988	18456	19656	21914	23856	25314	26902	28426	30122	29962	30430	30888	31826	31640
2.5	43188	34934	27868	23568	19158	17012	17244	17466	19396	22218	24342	25570	29260	30774	32352	34210	33488	36322	35850	35326	37050
2.6	36710	30868	24854	21868	18950	17234	17428	18704	20916	22622	25828	29072	31328	33668	36908	38778	38570	40128	41918	42724	44366
2.7	32458	28610	23758	21058	19044	17956	17692	18506	20658	24426	28054	31212	34680	36556	40330	41898	45400	45200	47860	47808	50156
2.8	30312	25984	22652	19814	19502	17684	17554	19252	21692	25532	30650	32794	39454	40610	45276	47798	50910	51016	54636	56768	55140
2.9	30866	26132	22968	19826	18712	17226	17980	19662	22778	26818	32022	36770	42614	45954	51412	56128	56462	61886	62150	62826	65378
3.0	30838	25212	21732	19846	17996	18132	18516	20212	23574	27586	32328	40772	45112	51884	58120	60244	67730	71996	71032	74962	77198

Fig. 7. The number of function evaluation required to Solve Schwefel 1.2 test function.

were assigned to the fine-tuned values as recommended in the pioneering work [4,8,11,12,15]. The parameters of all algorithms were summarized as follows:

(1) CPA: group iteration = 2; Attraction rate = 0.8; Growth rate = 2; Reproduction rate = 1.8; nCPlant = 10; nPrey = 20
(2) FPA: population size = 25; Probability switch, p = 0.8
(3) Improved PSO: population size = 20; Inertial weight, $w = 0.9-0.5\left(\frac{iteration}{1000}\right)$; Acceleration factor, C1 = 2; Acceleration factor, C2 = 2
(4) DE: population size = 20; Scaling, F = 0.7, Crossover probability, Cr = 0.9;
(5) CSA: population size = 25; Discovery rate, pa = 0.25

Sum of Different Power Test Function, Dim = 15

Growth rate \ Reproduction rate	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
1.0	1500000	1500000	1500000	1500000	1350158	1450046	1250230	1150380	950832	551834	103326	3236	2510	2852	2938	2796	2912	3062	2588	3034	3012
1.1	1500000	1500000	1500000	1450046	1500000	1450064	1250232	1300220	900778	601262	252236	3460	3582	2778	2908	2934	2842	3288	3050	2598	3284
1.2	1500000	1450034	1500000	1450088	1350162	1400094	1350170	1250324	900910	801056	252458	53294	3192	3010	2810	2912	2940	2686	3048	2574	2940
1.3	1500000	1500000	1500000	1450056	1450042	1450048	1200354	950802	1001026	351802	252286	52936	2510	2744	2676	2488	2614	2678	2642	2752	2764
1.4	1500000	1500000	1450062	1450054	1300244	1350164	1200400	950758	751284	2628	52800	2562	2546	2270	2602	2472	2670	2492	2440	2584	2604
1.5	1500000	1500000	1500000	1350184	1350200	1150442	851126	751452	202416	52574	2310	2372	2484	2522	2310	2392	2596	2610	2280	2460	2452
1.6	1500000	1500000	1500000	1450128	1350244	1150740	651978	183034	2832	2802	2584	2360	2458	2330	2384	2482	2452	2398	2576	2458	2506
1.7	1500000	1500000	1400240	1450108	1150932	336654	103296	2788	2548	2480	2482	2572	2486	2434	2548	2572	2568	2598	2466	2586	2664
1.8	1500000	1500000	1350746	594722	234364	105540	3104	2868	2610	2560	2568	2534	2664	2628	2748	2724	2810	2680	2810	2754	2726
1.9	1294508	1167564	408176	162556	4934	3568	3028	2976	2808	2704	2840	2730	2834	2924	2852	2902	2976	3084	2860	3090	3072
2.0	13774	9312	6894	5326	4504	3306	2952	2954	2910	3046	3118	3048	3086	3142	3282	3212	3298	3356	3418	3310	3344
2.1	7020	6250	5628	4606	3768	3450	3128	3044	3188	3158	3304	3496	3482	3564	3682	3700	3624	3742	3812	3858	3828
2.2	6256	5204	4626	4024	3602	3484	3172	3288	3266	3456	3624	3734	3966	3996	4220	4134	4238	4318	4334	4304	4310
2.3	5520	5164	4484	3874	3674	3416	3212	3344	3424	3822	4080	4092	4376	4354	4740	4790	4768	4894	4884	5060	5046
2.4	5348	4944	4318	4024	3636	3430	3418	3568	3676	4032	4362	4606	4722	4884	5192	5442	5192	5566	5736	5788	5890
2.5	5052	4654	3980	4120	3688	3578	3568	3594	3850	4384	4784	5048	5502	5738	5854	6010	6218	6328	6458	6558	6600
2.6	4824	4376	4242	4080	4058	3634	3652	3774	4218	4600	5038	5586	6104	6408	6596	6826	7262	7496	7226	7496	8040
2.7	4840	4298	4184	3770	3910	3752	3686	3980	4502	4832	5408	6004	6528	6842	7588	7602	8252	8324	8732	9212	9054
2.8	4748	4416	4300	3938	3800	3816	3776	4118	4650	5098	5626	6392	7304	7808	8424	8698	9172	9682	9968	10224	10106
2.9	4696	4358	4182	3712	3654	3756	3802	4118	4598	5290	5770	6706	7828	8548	9344	9474	10352	11106	11212	11866	11868
3.0	4788	4420	4132	4066	3976	3816	4236	4240	4712	5488	6604	7442	8348	9214	10270	10996	11080	12444	12864	13844	13090

Fig. 8. The number of function evaluation required to solve sum of different power test function.

Schwefel 2.21 Test Function, Dim = 15

Growth rate \ Reproduction rate	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
1.0	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000
1.1	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000
1.2	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000
1.3	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000
1.4	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000
1.5	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1248946
1.6	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1420702	846876	577810	428350	337980	296028
1.7	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1360758	555902	286780	222398	167840	153776	139700	132754
1.8	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	771542	246680	143512	109302	82884	75870	63990	66592	61028	58572
1.9	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1181874	214468	84720	51890	42612	36606	38190	35908	35002	34614	34064	35434
2.0	1500000	1500000	1500000	1500000	1500000	1500000	1500000	1500000	404802	79696	43898	33310	28396	26920	27134	26566	27874	27746	26676	28494	26432
2.1	1500000	1500000	1500000	1500000	1500000	1500000	1500000	264112	72106	32020	26886	26732	24180	24870	24882	25292	25328	25908	25882	25342	26382
2.2	1500000	1500000	1500000	1500000	1500000	1497832	323714	73352	36808	27296	26300	24610	24760	25596	26302	25828	26008	26962	26604	27628	27196
2.3	1500000	1500000	1500000	1500000	1500000	435800	99382	43640	29124	27822	26450	26442	26846	28792	28484	28894	29454	29850	29680	29912	29990
2.4	1500000	1500000	1500000	1471642	499732	138970	58278	37770	29458	28052	28284	28682	29314	31116	31058	32288	32806	33714	34102	34614	34838
2.5	1500000	1499170	1192652	499670	208502	85408	45850	35724	31354	30126	30744	32162	32686	33996	35800	36428	38796	39098	39206	39738	40208
2.6	1173818	774236	449500	234376	124492	69838	44354	34488	32106	32018	34170	33702	36848	38126	39786	41298	41898	42714	44492	44210	46038
2.7	495346	375250	240318	154336	90818	60870	42556	35988	34164	35364	37266	38930	40782	42098	45176	47888	47928	50390	50972	53084	52200
2.8	281598	216872	161910	111176	73524	55556	43914	36108	36622	38634	42014	43774	44902	49512	52832	53880	57478	59982	60688	61570	62010
2.9	182144	142446	128598	93714	67964	54012	45262	40676	39876	39266	43282	45710	49920	55472	58402	60824	63612	67946	69224	72058	72120
3.0	130424	117816	102642	77620	66092	53036	45670	42756	40344	42620	47332	50440	54776	61348	65642	69780	73346	77798	80120	80526	85674

Fig. 9. The number of function evaluation required to solve Schwefel 2.21 test function.

(6) BAT: population size = 30; Frequency maximum, Qmax = 2; Frequency minimum, Qmin = 0; Pulse rate, r = 0.5; Loudness, A = 0.5
(7) SSA: population size = 30
(8) GA: population size = 25; Number of mutation sites, nsite = 5; Crossover probability, pc = 0.95; Mutation probability, pm = 0.05; String length, nsbit = 16

Table 2 summarizes the average NOFE required by each algorithm to reach the desired tolerance error for each test function. Also, each algorithm was ranked according to the attained average NOFE. Particularly, the lower the average NOFE required, the higher the rank is. If the algorithm did not reach the tolerance error within the predefined number of NOFE, the algorithm was assigned with the lowest rank. Hence, for a particular test function, some algorithms may attain the same rank. The success rate

Average

Growth rate \ Reproduction rate	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
1.0	7500000	7500000	7500000	7500000	7350158	7450046	7250230	7150380	6950832	6402658	2855990	1807598	1648956	1746114	1643678	1592650	1543180	1692580	1641662	1693628	1594620
1.1	7500000	7500000	7500000	7450046	7500000	7450064	7250232	7300220	6900778	6302602	3969892	1761906	1700576	1596854	1742042	1693648	1741234	1642156	1791880	1742086	1743766
1.2	7500000	7450034	7500000	7450088	7350162	7400094	7350170	7250324	6900910	6153874	4147698	1861036	1997798	1743320	1691874	1641724	1691938	1689770	1791044	1641304	1691314
1.3	7500000	7500000	7500000	7450056	7450042	7450048	7200354	6950802	6801748	5455450	3447502	1656096	1744478	1690568	1640626	1690146	1590304	1689582	1639472	1689536	1690760
1.4	7500000	7500000	7450062	7450054	7300244	7350164	7150568	6801170	6053592	4259388	1974318	1647698	1640744	1640204	1638550	1687060	1786806	1737138	1637448	1736994	1786760
1.5	7500000	7500000	7500000	7350184	7350200	7050680	6601838	5754406	4159862	2607944	1555206	1640244	1686516	1685364	1584570	1734010	1584766	1584664	1635060	1883922	1484048
1.6	7500000	7500000	7500000	7450128	7350244	6602526	5256554	4090968	3086650	1812902	1736976	1583098	1584294	1682934	1732600	1552806	1029674	710484	611316	471410	379754
1.7	7500000	7500000	7400240	7450108	7006910	5280032	3861606	3167520	1725864	1588202	1732456	1631736	1531434	1392066	637588	369078	354700	300232	287010	222960	217082
1.8	7500000	7500000	7264852	6001834	4849954	4023416	3065364	1814406	1637192	1631584	1680626	852732	428082	275256	389644	165702	159508	147254	150204	194978	143474
1.9	7244818	6885442	5674696	4734506	3719214	2221608	1709134	1736596	1632340	1313858	296530	266298	84958	175768	70532	72680	70698	70434	70112	119996	72606
2.0	4493586	4472690	4029944	3107190	2371342	2201360	1887236	1783110	437422	162568	77604	266538	63690	63582	113920	114520	115768	66240	66204	67212	115738
2.1	3398144	3135038	2926434	2529886	2400420	2187188	1883316	546052	155468	116924	113326	114504	112804	65918	65888	66924	67700	68896	119330	69978	120506
2.2	3030492	2993472	2811554	2795902	2438176	2181658	606188	206416	221272	64822	115594	214934	68498	70416	72072	73050	74360	125724	75472	77742	77540
2.3	3083346	2958598	2848442	2790466	2435026	1315884	829838	277124	165612	117414	70452	121698	75714	78736	80668	82396	83902	85536	85858	86406	86786
2.4	2961576	2998994	2991368	2857458	1631710	621858	441038	223132	217560	119806	74760	78706	132146	87246	90076	94146	94760	97228	98588	100090	100594
2.5	3005694	2995498	2581272	1834500	1339202	1114448	727566	123072	270502	126138	81830	86588	92942	97966	102618	106178	109172	113146	113930	114106	166242
2.6	2672700	2166882	1885154	1666746	1255434	1049676	627322	470886	373438	129766	187242	94176	102206	108328	115570	120970	122168	126846	131256	132158	136576
2.7	1990084	1815444	1526218	1535942	1321284	991958	626056	720962	227546	284640	243968	104698	162036	119262	178734	135606	142076	145386	199066	154084	155650
2.8	1823796	1605152	1596024	1640620	1254790	1135710	826348	325090	380386	240802	153804	113008	124976	135174	195968	153714	162742	167594	223116	178974	227304
2.9	1477178	1630218	1562770	1424570	1298374	1084268	877918	578338	484272	144386	108452	169712	136620	150434	163804	174044	180988	194760	199526	206272	208226
3.0	1524814	1455648	1535554	1557932	1345708	1183692	1177488	631294	436254	248280	164252	180986	196146	165626	182786	194284	209236	272388	274936	235232	244494

Fig. 10. The average number of function evaluation required to solve five test functions.

of each algorithm to meet the tolerance error within the allowable maximum NOFE through 30 independent runs is presented in Table 3. The success rate was calculated based on the number of successful runs over total independent runs.

For high-dimensional functions (f_{01} to f_{19}), Table 2 clearly shows that the proposed CPA outperforms others with the fastest convergence rate, except for f_{16}. Even though DE is ranked first for test function f_{16}, its successful convergence to the global optimum in all runs is not guaranteed, as seen in Table 3. Besides, the optimization performances of BAT and GA are less satisfactory. Among 19 test functions, BAT only can converge to the global optimum for test function f_{18}. The GA, on the other hand, could not locate the global optimum within the allowable iteration in 15 out of the 19 test functions. The performance of the recently proposed SSA is not promising as well, since zero success rate is achieved for 12 out of the 19 test functions. The experiment results show that the CPA does not encounter a curse of dimensionality problem, since it obtains promising results for these high-dimensional functions. For the low-dimensional functions (f_{19} to f_{30}), it can be seen that the CPA achieves better results than its competitors for all test functions, with the significantly lower NOFE is attained. Hence, it can be concluded that the CPA has better comparative performance on the unimodal, multimodal, separable and non-separable test functions.

Furthermore, except for test function f_{16}, the percentage of improvement by comparing the obtained NOFE of CPA to the second rank algorithm is summarized in Table 4. The results show that the proposed CPA converges to the global optima with significant lower NOFE, where at least 80% performance improvement is achieved for test functions f_{03}, f_{05}, f_{11}, f_{21}, f_{22}, f_{24}, f_{25}, f_{26} and f_{29}. The proposed CPA achieves overwhelmingly better result on f_{25}, where the CPA only requires 770 NOFE to reach the global optima, while the Improved PSO uses 7680 NOFE to solve this test function.

Fig. 11 presents the convergence curve of all algorithms for the high dimensional test function f_{01}. It can be seen in this figure that SSA converges rapidly as compared to CPA at the initial stage. However, the search process of SSA goes stagnant in the later phase while CPA continues exploiting the global optima. Notably, CPA shows better performance than Improved PSO, DE, CSA, BAT and GA with respect to the convergence rate. The similar convergence trend can be observed for other high dimensional test functions of f_{02}, f_{03}, f_{04}, f_{06}, f_{09}, f_{10}, f_{11}, f_{13}, f_{14}, f_{15}, f_{16}, f_{18} and f_{19}. The convergence curves of these test functions were omitted here for brevity (see Appendices A.1–A.3, A.5, A.8–A.10, A.12–A.15, A.17 and A.18).

For the lower dimensional functions, CPA exhibits faster convergence behaviour than the competitive algorithms (see Appendix A.19 to Appendix A.27), with the exception for f_{26} and f_{29}, as shown in Fig. 12 and Fig. 13, respectively. The Improved PSO outperforms CPA on these functions, but it fails to converge to the optimal solution eventually, which is also evidenced by the obtained success rate of 57% and 17%, respectively, as seen in Table 3. These findings reveal that the proposed CPA is promising in optimizing both high-dimensional and low-dimensional functions with less NOFEs are required.

The performance improvement of the proposed CPA is probably due to three main reasons. Firstly, the grouping process partitions the population into several sub-groups (sub search spaces), each of which can explore new area in different search directions. The sub-groups are mixed after a predefined number of iterations, allowing the information sharing independently gained by each group. This process also enhances the survivability of good quality solutions since the grouping is based on the ranking of each individual. Secondly, during the reproduction process, each dimension of the first-ranked carnivorous plant is updated using various good quality solutions. Hence, by integrating different information from multiple good quality solutions, the first-ranked carnivorous plant can exploit the search space more effectively. For the third reason, it is due to the elitism selection strategy in the combination process. After combining the previous population and current population, the elitism selection strategy maintains the good solutions and abandons the not so good solutions. Hence, this process allows the algorithm

Table 1
Classical test functions used in Test II (M: multimodal, U: unimodal, S: separable, N: non-separable, D: dimension, Range: range of search space, optimum: global optimal value).

Test Function	Name	Type	D	Range	Optimum
$f_{01}(x) = \sum_{i=1}^{D} (x_i + 0.5)^2$	Step	US	100	$[-5.12, 5.12]$	0
$f_{02}(x) = \sum_{i=1}^{D} x_i^2$	De Jong	US	100	$[-5.12, 5.12]$	0
$f_{03}(x) = (x_1 - 1)^2 + \sum_{i=1}^{D} i\left(2x_i^2 - x_i - 1\right)^2$	Dixon–Price	UN	50	$[-10, 10]$	0
$f_{04}(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	Griewank	MN	50	$[-100, 100]$	0
$f_{05}(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + \exp(1)$	Ackley	MS	30	$[-32.768, 32.768]$	0
$f_{06}(x) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	Schwefel 2.22	UN	30	$[-10, 10]$	0
$f_{07}(x) = \sum_{j=1}^{D} \left(\sum_{i=1}^{j} x_i\right)^2$	Schwefel 1.2	UN	30	$[-100, 100]$	0
$f_{08}(x) = \max_i \{\lvert x_i\rvert, 1 \le i \le D\}$	Schwefel 2.21	US	30	$[-100, 100]$	0
$f_{09}(x) = \sum_{i=1}^{D} (x_i + 0.5)^2$	Step	US	30	$[-5.12, 5.12]$	0
$f_{10}(x) = \sum_{i=1}^{D} ix_i^2$	Sum Squares	US	30	$[-10, 10]$	0
$f_{11}(x) = (x_1 - 1)^2 + \sum_{i=1}^{D} i\left(2x_i^2 - x_i - 1\right)^2$	Dixon–Price	UN	30	$[-10, 10]$	0
$f_{12}(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^4$	Zakharov	UN	30	$[-5, 10]$	0
$f_{13}(x) = \sum_{i=1}^{D} x_i^2$	De Jong	US	15	$[-5.12, 5.12]$	0
$f_{14}(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	Griewank	MN	15	$[-100, 100]$	0
$f_{15}(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + \exp(1)$	Ackley	MS	12	$[-32.768, 32.768]$	0
$f_{16}(x) = \sum_{i=1}^{D/4} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4]$	Powell	UN	12	$[-4, 5]$	0
$f_{17}(x) = 10D + \sum_{i=1}^{D}[x_i^2 - 10\cos(2\pi x_i)]$	Rastrigin	MS	10	$[-5.12, 5.12]$	0
$f_{18}(x) = \sum_{i=1}^{D} \lvert x_i\rvert^{i+1}$	Sum of Different Power	US	10	$[-1, 1]$	0
$f_{19}(x) = \sin^2(\pi w_1) + \sum_{i=1}^{d-1}(w_i - 1)^2[1 + 10\sin^2(\pi w_i + 1)] + (w_d - 1)^2[1 + \sin^2(2\pi w_d)]$, where $w_i = 1 + \frac{x_i - 1}{4}$, for all $i = 1, \ldots, d$	Levy	MN	10	$[-10, 10]$	0
$f_{20}(x_1, x_2, x_3, x_4) = 100(x_1 - x_2^2)^2 + (1 - x_1)^2 + (1 - x_3)^2 + 90(x_4 - x_3^2)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1)$	Colville	MN	4	$[-10, 10]$	0
$f_{21}(x) = -\sum_{i=1}^{D} \sin(x_i)\sin^2 0(\frac{ix_i^2}{\pi})$	Michaelwicz	UN	2	$[0, 5]$	-1.8013
$f_{22}(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$	Easom	UN	2	$[-100, 100]$	-1

(*continued on next page*)

Table 1 (*continued*).

Test Function	Name	Type	D	Range	Optimum
$f_{23}(x_1,x_2)=(1.5-x_1+x_1x_2)^2+(2.25-x_1+x_1x_2^2)^2+(2.625-x_1+x_1x_2^3)^2$	Beale	MN	2	$[-4.5, 4.5]$	0
$f_{24}(x_1,x_2)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	Goldstein–Price	MN	2	$[-2, 2]$	3
$f_{25}(x_1,x_2)=(\sum_{i=1}^{5} i\cos((i+1)x_1+i))(\sum_{i=1}^{5} i\cos((i+1)x_2+i))$	Shubert	MN	2	$[-10, 10]$	−186.7309
$f_{26}(x_1,x_2)=-\left\lvert\sin(x_1)\cos(x_2)\exp(\left\lvert 1-\frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert)\right\rvert$	Holder Table	MN	2	$[-10, 10]$	−19.2085
$f_{27}(x_1,x_2)=\sin(x_1+x_2)+(x_1-x_2)^2-1.5x_1+2.5x_2+1$	McCormick	MN	2	$-1.5\le x_1\le 4$ $-3\le x_2\le 3$	−1.9133
$f_{28}(x_1,x_2)=(4-2.1x_1^2+\frac{x_1^4}{3})x_1^2+x_1x_2+(-4+4x_2^2)x_2^2$	Six-Hump Camel	MS	2	$-3\le x_1\le 3$ $-2\le x_2\le 2$	−1.0316
$f_{29}(x_1,x_2)=-0.0001(\left\lvert\sin(x_1)\sin(x_2)\exp(\left\lvert 100-\frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert)\right\rvert+1)^{0.1}$	Cross-In-Tray	MN	2	$[-10, 10]$	−2.0626
$f_{30}(x_1,x_2)=(x_1+2x_2-7)^2+(2x_1+x_2-5)^2$	Booth	UN	2	$[-10, 10]$	0

Table 2
The average number of function evaluation and rank obtained by CPA, FPA, Improved PSO, DE, CSA, BAT, FA and GA through 30 independent runs in solving classical test functions.

Test Function	CPA		FPA		Improved PSO		DE		CSA		BAT		SSA		GA	
	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank
f_{01}	89,364	1	720,601	4	1,500,000	5	170,331	2	177,608	3	1,500,000	5	1,500,000	5	1,755,173	5
f_{02}	88,435	1	718,529	4	1,500,000	5	171,991	2	175,495	3	1,500,000	5	1,500,000	5	1,755,033	5
f_{03}	32,060	1	460,696	3	1,500,000	5	156,545	2	674,688	4	1,500,000	5	1,500,000	5	1,754,989	5
f_{04}	72,150	1	1,011,051	4	1,500,000	6	881,697	3	344,172	2	1,500,000	6	1,450,930	5	1,748,200	6
f_{05}	34,704	1	1,500,000	4	1,500,000	4	1,355,281	3	267,965	2	1,500,000	4	1,500,000	4	1,754,856	4
f_{06}	13,786	1	216,925	4	600,000	5	47,800	2	106,500	3	600,000	5	600,000	5	584,982	5
f_{07}	93,761	1	492,650	4	500,000	5	123,340	2	331,150	3	600,000	5	600,000	5	585,007	5
f_{08}	455,078	1	1,500,000	2	1,500,000	2	1,500,000	2	1,500,000	2	1,500,000	2	1,500,000	2	1,755,026	2
f_{09}	9046	1	1,57,575	5	228,680	6	26,500	3	49,050	4	510,000	8	23,220	2	922,003	7
f_{10}	11,514	1	197,125	4	452,280	5	33,700	2	61,250	3	510,000	6	510,000	6	585,044	6
f_{11}	13,445	1	302,525	3	1,401,620	5	122,420	2	741,150	4	1,500,000	6	1,500,000	6	1,500,000	6
f_{12}	72,464	1	443,350	4	1,500,000	5	113,260	2	289,600	3	1,500,000	5	1,500,000	5	1,755,023	5
f_{13}	3439	1	67,359	6	14,920	3	10,660	2	25,100	5	90,000	8	22,080	4	73,009	7
f_{14}	33,760	1	576,100	3	561,060	5	258,707	4	113,250	2	600,000	8	580,830	7	562,104	6
f_{15}	6177	1	95,588	5	17,140	2	18,200	3	54,200	4	90,000	7	46,650	6	73,107	7
f_{16}	13,328	2	34,400	4	34,000	5	12,520	1	29,600	3	36,000	5	39,000	5	36,547	5
f_{17}	64,570	1	1,631,768	3	1,000,000	4	1,000,000	4	176,900	2	900,000	4	1,800,000	4	975,052	4
f_{18}	556	1	8275	7	7340	6	2060	3	5250	4	1320	2	13,470	8	5445	5
f_{19}	4908	1	48,125	5	12,820	3	8820	2	32,950	4	51,000	7	71,880	6	48,724	7
f_{20}	16,708	1	47,825	4	43,120	5	17,100	2	41,800	3	48,000	7	44,190	6	48,736	7
f_{21}	219	1	5780	5	4576	4	1880	2	3950	3	42,540	7	17,273	6	52,516	8
f_{22}	409	1	9700	3	6920	6	3720	2	11,200	4	8730	7	19,260	5	9722	7
f_{23}	476	1	6125	6	4260	4	1860	2	4750	5	2130	3	14,340	7	9421	8
f_{24}	370	1	7100	4	5320	8	2080	2	5800	3	6630	6	17,940	5	8835	7
f_{25}	770	1	88,125	6	7680	2	20,380	4	11,300	3	64,950	7	20,940	5	92,409	8
f_{26}	385	1	11,700	3	7280	7	5940	5	3550	2	3000	6	16,980	4	12,155	8
f_{27}	180	1	3125	5	2820	7	980	3	2750	4	360	2	13,020	6	3300	8
f_{28}	218	1	5050	6	4220	5	1660	3	2500	4	480	2	13,830	7	5308	8
f_{29}	205	1	2950	4	2740	7	1760	2	2050	3	600	6	11,520	5	2641	8
f_{30}	280	1	4275	5	4600	7	1320	3	3800	4	1230	2	16,440	6	4851	8

to converge more rapidly, thereby increasing the performance of the CPA.

3.3. Test III – Modern benchmark test functions

Some algorithms work excellently for classical test functions, however, the performance deteriorates when the global optimum of the classical function shifts to different values for each dimension [36]. This phenomenon occurs when the algorithm exhibits structural bias, *i.e.* centre-seeking bias (CSB) and initialization-region bias (IRB) [37]. CSB means the solutions of an algorithm will eventually move towards the origin, irrespective of the fitness evaluation [38]. Meanwhile, IRB refers to the solutions of an algorithm will not move far away from their initial points [38]. Due to the structural bias, the solutions are unable to cover the entire search space, and thus, the search process goes stagnant if the global optimum is not located at the origin or the initial population of the algorithm is not generalized near the global optimum.

To identify whether a structural bias arises in the proposed CPA, seven test functions from CEC2017 were selected [39] and the details were as in Table 5. By using the same parameter settings, the optimization results of the CPA and other competitors for the CEC2017 test functions are presented in Table 6. The LSHADE algorithm [40] and EBOwithCMAR [41] - winner of the CEC 2014 and CEC2017 competition, respectively, were used for

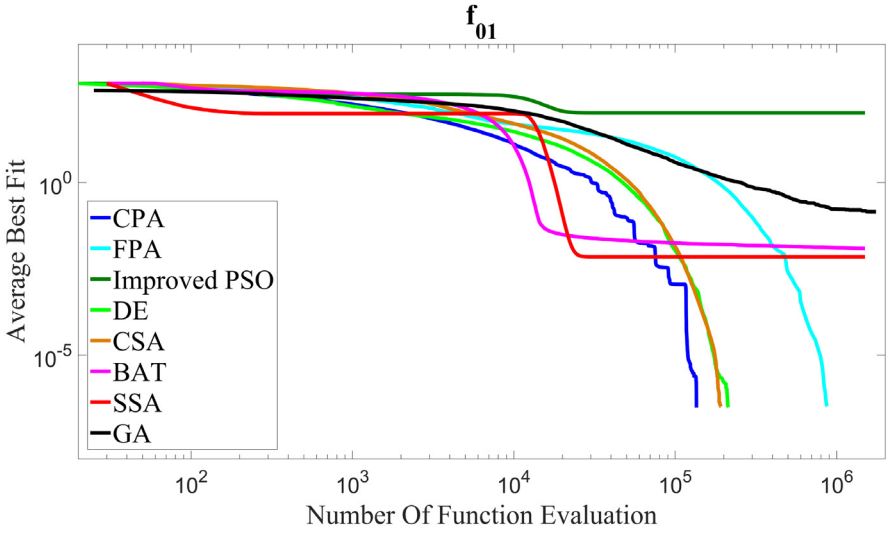


Fig. 11. Convergence curve of CPA and other competitive algorithms for Step test function (d = 100).

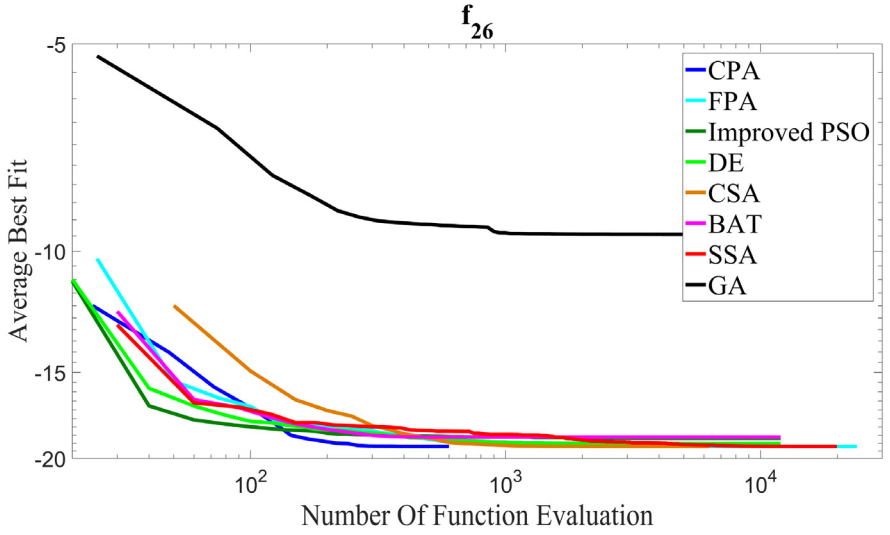


Fig. 12. Convergence curve of CPA and other competitive algorithms for Holder Table test function (d = 2).

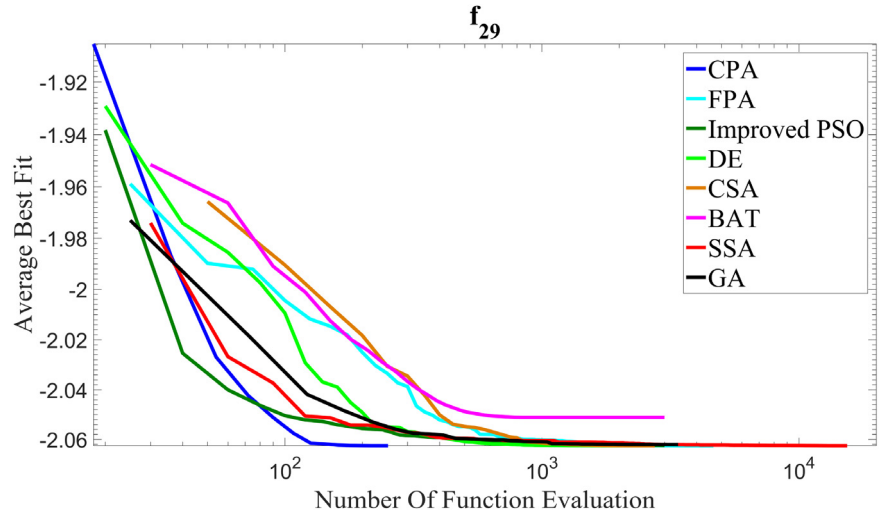


Fig. 13. Convergence curve of CPA and other competitive algorithms for Cross-In-Tray Test Function (d = 2).

Table 3
The success rate of CPA, FPA, Improved PSO, DE, CSA, BAT, FA and GA through 30 independent runs in solving classical test functions.

Test Function	Success rate, %							
	CPA	FPA	Improved PSO	DE	CSA	BAT	SSA	GA
f_{01}	100	100	0	100	100	0	0	0
f_{02}	100	100	0	100	100	0	0	0
f_{03}	100	100	0	100	100	0	0	0
f_{04}	100	43	0	43	83	0	3	0
f_{05}	100	0	0	10	90	0	0	0
f_{06}	100	100	0	100	100	0	0	0
f_{07}	100	100	0	100	100	0	0	0
f_{08}	100	0	0	0	0	0	0	0
f_{09}	100	100	57	100	100	0	100	100
f_{10}	100	100	10	100	100	0	0	0
f_{11}	100	100	7	100	100	0	0	0
f_{12}	100	100	0	100	100	0	0	0
f_{13}	100	100	100	100	100	0	100	3
f_{14}	100	100	7	67	100	0	3	7
f_{15}	100	100	100	100	100	0	63	0
f_{16}	100	100	0	100	100	0	0	0
f_{17}	100	100	0	0	100	0	0	0
f_{18}	100	100	100	100	100	100	100	100
f_{19}	100	100	100	100	100	0	27	0
f_{20}	100	100	33	100	100	0	17	0
f_{21}	100	100	100	100	100	57	100	50
f_{22}	100	100	73	100	100	0	100	0
f_{23}	100	100	100	100	100	90	100	3
f_{24}	100	100	0	100	100	57	100	27
f_{25}	100	100	100	100	100	43	100	13
f_{26}	100	100	57	83	100	80	100	0
f_{27}	100	100	67	100	100	100	100	10
f_{28}	100	100	100	100	100	100	100	20
f_{29}	100	100	17	100	100	93	100	40
f_{30}	100	100	43	100	100	100	100	0

Table 4
The percentage improvement of the average NOFE between first rank (CPA) and second rank optimizer.

Test Function	CPA	Improved PSO	DE	CSA	BAT	SSA	Percentage Improvement, %
	Average Number Of Function Evaluation	Average Number Of Function Evaluation	Average Number Of Function Evaluation	Average Number Of Function Evaluation	Average Number Of Function Evaluation	Average Number Of Function Evaluation	
f_{01}	89,364	–	170,331	–	–	–	48
f_{02}	88,435	–	171,991	–	–	–	49
f_{03}	32,060	–	156,545	–	–	–	80
f_{04}	72,150	–	–	344,172	–	–	79
f_{05}	34,704	–	–	267,965	–	–	87
f_{06}	13,786	–	47,800	–	–	–	71
f_{07}	93,761	–	123,340	–	–	–	24
f_{09}	9046	–	–	–	–	23,220	61
f_{10}	11,514	–	33,700	–	–	–	66
f_{11}	13,445	–	122,420	–	–	–	89
f_{12}	72,464	–	113,260	–	–	–	36
f_{13}	3439	–	10,660	–	–	–	68
f_{14}	33,760	–	–	113,250	–	–	70
f_{15}	6177	17,140	–	–	–	–	64
f_{17}	64,570	–	–	176,900	–	–	63
f_{18}	556	–	–	–	1320	–	58
f_{19}	4908	–	8820	–	–	–	44
f_{20}	16,708	–	17,100	–	–	–	2
f_{21}	219	–	1880	–	–	–	88
f_{22}	409	–	3720	–	–	–	89
f_{23}	476	–	1860	–	–	–	74
f_{24}	370	–	2080	–	–	–	82
f_{25}	770	7680	–	–	–	–	90
f_{26}	385	–	–	3550	–	–	89
f_{27}	180	–	–	–	360	–	50
f_{28}	218	–	–	–	480	–	55
f_{29}	205	–	1760	–	–	–	88
f_{30}	280	–	–	–	1230	–	77

performance comparison. The parameter settings of LSHADE are initially referred to [40]. However, simulation of LSHADE using these parameter setting portraits a not so good result as tabulated in Table 6. Hence, the initial population of LSHADE is tuned for

Table 5
CEC 2017 modern test functions used in Test III (M: multimodal, U: unimodal, D: dimension, Range: range of search space, optimum: global optimal value).

Test Function	Name	Type	D	Range	F*	Optimum
$CEC_{01}(x) = SR_{01}\left(M\left(\frac{0.5}{100}(x-o)\right)\right) + F^*_{01}$ where $SR_{01}(x) = \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1$	Shifted and Rotated Griewank	M	10	[−100, 100]	0	0
$CEC_{02}(x) = SR_{02}\left(M\left(\frac{5.12}{100}(x-o)\right)\right) + F^*_{02}$ where $SR_{02}(x) = \sin^2(\pi w_1) + \sum_{i=1}^{d-1}(w_i-1)^2[1+10\sin^2(\pi w_i+1)] + (w_d-1)^2[1+\sin^2(2\pi w_d)]$, where $w_i = 1 + \frac{x_i-1}{4}$, for all $i = 1, \ldots, d$	Shifted and Rotated Levy	M	10	[−100, 100]	900	900
$CEC_{03}(x) = SR_{03}\left(M\left(\frac{0.5}{100}(x-o)\right)\right) + F^*_{03}$ where $SR_{03}(x) = g(x_1,x_2) + g(x_2,x_3) + \ldots + g(x_{D-1},x_D) + g(x_D,x_1)$, where $g(x,y) = 0.5 + \frac{\left(\sin^2\left(\sqrt{x^2+y^2}\right) - 0.5\right)}{\left(1+0.001\left(x^2+y^2\right)\right)^2}$	Shifted and Rotated Schaffer's F6	M	10	[−100, 100]	600	600
$CEC_{04}(x) = SR_{04}\left(M\left(\frac{0.5}{100}(x-o)\right)\right) + F^*_{04}$ where $SR_{04}(x) = \left[\frac{1}{D-1}\sum_{i=1}^{D-1}\left(\sqrt{s_i}\left(\sin\left(50.0s_i^{0.2}\right)+1\right)\right)\right]^2$, $s_i = \sqrt{x_i^2+x_{i+1}^2}$	Shifted and Rotated Schaffer's F7	M	10	[−100, 100]	0	0
$CEC_{05}(x) = SR_{05}(M(x-o)) + F^*_{05}$ where $SR_{05}(x) = \sum_{i=1}^{D}\lvert x_i\rvert^{i+1}$	Shifted and Rotated Sum of Different Power	U	10	[−100, 100]	200	200
$CEC_{06}(x) = SR_{06}(M(x-o)) + F^*_{06}$ where $SR_{06}(x) = \sum_{i=1}^{D}x_i^2 + \left(\sum_{i=1}^{D}0.5x_i\right)^2 + \left(\sum_{i=1}^{D}0.5x_i\right)^4$	Shifted and Rotated Zakharov	U	10	[−100, 100]	300	300
$CEC_{07}(x) = SR_{07}\left(M\left(\frac{0.5}{100}(x-o)\right)+1\right) + F^*_{07}$ where $SR_{07}(x) = f_{02}(f_{01}(x_1,x_2)) + f_{02}(f_{01}(x_2,x_3)) + \ldots + f_{02}(f_{01}(x_{D-1},x_D)) + f_{02}(f_{01}(x_D,x_1))$ where $f_{01}(x) = \sum_{i=1}^{D-1}\left(100\left(x_i^2-x_{i+1}\right)^2 + (x_i-1)^2\right)$ $f_{02}(x) = \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1$	Shifted and Rotated Expanded Griewank's plus Rosenbrock	M	10	[−100, 100]	0	0

Table 6(a)
The average number of function evaluation and rank obtained by CPA, FPA, Improved PSO, DE, CSA, BAT, FA, GA and LSHADE through 30 independent runs in solving seven modern test functions.

Test Function	CPA		FPA		Improved PSO		DE		CSA		BAT	
	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank
CEC_{01}	1618	2	12,875	6	483,680	9	4000	4	10,200	5	84,684	8
CEC_{02}	6174	3	37,875	6	402,780	9	9420	4	42,050	7	500,010	10
CEC_{03}	1835	2	16,125	7	483,700	10	4900	4	12,600	5	73,620	9
CEC_{04}	24,443	1	881,150	6	1,450,900	9	1,500,000	9	934,310	5	1,023,960	7
CEC_{05}	7007	3	26,025	6	500,000	10	10,100	4	29,000	7	148,980	8
CEC_{06}	12,473	3	35,675	7	342,180	9	15,920	4	56,450	8	500,010	10
CEC_{07}	49,594	1	278,825	6	1,500,000	8	137,700	4	414,900	7	1,500,000	8

Table 6(b)
The average number of function evaluation and rank obtained by CPA, FPA, Improved PSO, DE, CSA, BAT, FA, GA and LSHADE through 30 independent runs in solving seven modern test functions.

Test Function	SSA		GA		LSHADE (With Different NP)		LSHADE		EBOwithCMAR (With Different NP)	
	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank	NOFE	Rank
CEC_{01}	16,410	7	584,958	9	1558	1	11,986	6	1713	3
CEC_{02}	372,300	8	584,954	10	5189	2	21,065	5	4413	1
CEC_{03}	18,150	8	585,007	11	1823	1	13,973	6	1879	3
CEC_{04}	1,401,409	8	1,754,992	9	252,867	4	178,725	3	75,268	2
CEC_{05}	308,640	9	584,954	10	6209	1	25,391	5	6440	2
CEC_{06}	25,835	5	585,055	10	11,931	2	31,775	6	10,758	1
CEC_{07}	1,500,000	8	1,755,025	8	269,240	5	72,190	3	51,814	2

Table 7
The success rate of CPA, FPA, Improved PSO, DE, CSA, BAT, FA and GA through 30 independent runs in solving seven modern test functions.

Test Function	Success rate, %										
	CPA	FPA	Improved PSO	DE	CSA	BAT	SSA	GA	LSHADE (With Different NP)	LSHADE	EBOwithCMAR (With Different NP)
CEC_{01}	100	100	0	100	100	100	100	0	100	100	100
CEC_{02}	100	100	20	100	100	0	27	0	100	100	100
CEC_{03}	100	100	3	100	100	100	100	0	100	100	100
CEC_{04}	100	63	0	0	80	33	7	0	93	100	100
CEC_{05}	100	100	0	100	100	100	40	0	100	100	100
CEC_{06}	100	100	33	100	100	0	100	0	100	100	100
CEC_{07}	100	100	0	100	100	0	0	0	100	100	100

Table 8
Percentage improvement of the average number of function evaluation between rank CPA and following rank optimizer.

Test Function	CPA	DE	EBOwithCMAR	Percentage Improvement, %
	Average Number Of Function Evaluation	Average Number Of Function Evaluation	Average Number Of Function Evaluation	
CEC_{01}	1618	–	1713	5.55
CEC_{02}	6174	9420	–	34.46
CEC_{03}	1835	–	1879	2.34
CEC_{04}	24,443	–	75,268	67.53
CEC_{05}	7007	10,100	–	30.62
CEC_{06}	12,473	15,920	–	21.65
CEC_{07}	49,594	–	51,814	4.28

each CEC test function. The number of initial population (NP) in LSHADE for each test function is assigned as: $NP = 20$ for CEC01 and CEC03, $NP = 40$ for CEC02, CEC04 and CEC05, $NP = 60$ for CEC06 and $NP = 100$ for CEC07. Similar to LSHADE, the number of NP in EBOwithCMAR for each test function is assigned as: $NP = 20$ for CEC01 and CEC03, $NP = 40$ for CEC02 and CEC05, $NP = 60$ for CEC04 and CEC06 and $NP = 18D$ for CEC07.

As shown in Table 6, LSHADE and EBOwithCMAR with the fine-tuned NP is the best optimizer in most of the CEC test functions and followed by the CPA. It is pertinent to note that LSHADE is a variant of DE that has been improved for many generations, but the CPA still outperforms LSHADE with fine-tuned NP in CEC04 and CEC07, as shown in Table 6. In addition, CPA surpasses EBOwithCMAR in CEC01, CEC03, CEC04 and CEC07 even though EBOwithCMAR is the improved version of an effective butterfly optimizer. Except LSHADE and EBOwithCMAR, the CPA outperforms other optimization algorithms in all the considered CEC test functions, with second or third least NOFE is required and 100% success in attaining the global optima in all runs (see Table 7). Thus, the CPA is still a competitive algorithm for these test functions. The results demonstrate that CPA does not exhibit structural bias in its search operator. It can locate the optimum although the global optimum has been shifted from the origin in the search space.

As shown in Table 7, only CPA, LSHADE and EBOwithCMAR can achieve 100% convergence success rate for all CEC test functions. The Improved PSO and GA are underachieving in this regard, possibly due to the inherent structural bias. It has been reported in [38] that the classical PSO suffers from such bias regardless of the population size, while the GA exhibits structural bias when the population size exceeds 20. Since the search operators of the Improved PSO and classical PSO remain the same, the Improved PSO still inherits the behaviour of structural bias as in the PSO.

Table 8 summarizes the percentage of improvement made by the CPA by comparing its NOFE to the one attained by the successive ranked algorithm. A remarkable improvement of 67.53% can be observed for the CEC_{04} test function. Although FPA, CSA, BAT and SSA are newly proposed optimizers as compared to the DE, they still unable to outperform DE in most of the test functions. Meanwhile, only the proposed CPA performs better than DE in all the test functions, as clearly shown in Table 8.

The good convergence rate of CPA, again, can be observed in the convergence curves for the CEC2017 test functions. For CEC_{02} and CEC_{03}, as shown in Figs. 14 and 15, although GA and Improved PSO converge more rapidly than CPA at initial, these algorithms experience stagnation subsequently, which in turn deteriorates their performances. This is why the success rate of the Improved PSO in solving these test functions is below 25%, while zero success rate is attained by GA.

The similar premature convergence can be observed for PSO, BAT SSA and GA in optimizing the CEC_{04} (see Appendix A.29). Fig. 15 also demonstrates the good convergence characteristic of CPA when optimizing the CEC_{03}. It can be seen that the CPA tends to converge to the global optimum faster than other competitive algorithms, except for LSHADE and EBOwithCMAR with the fine-tuned NP. The similar convergence trend can be observed for CEC_{01}, CEC_{05} and CEC_{07} (see Appendices A.28, A.30 and A.31).

Although the obtained results shed light on the optimization performance of the proposed CPA, the Wilcoxon signed-rank test is also employed to validate the statistical significance difference of CPA from other optimizers. In this statistical test, the null hypothesis states that the median of the results obtained from the CPA has no statistically significant difference from its competitor. To reject the null hypothesis, the average of the best results produced from 30 runs of each algorithm against each test function is tabulated in Table 9. Then, the "R+" and "R-" are calculated and shown in Table 10. The "R+" in Table 10 represents the sum of ranks for the test functions of which CPA surpassed the competitor, whereas the "R-" is the contrary. These values are evaluated to determine whether to accept or reject the null hypothesis [42]. The Wilcoxon signed-rank test shows that the null hypothesis is rejected, meaning the optimization

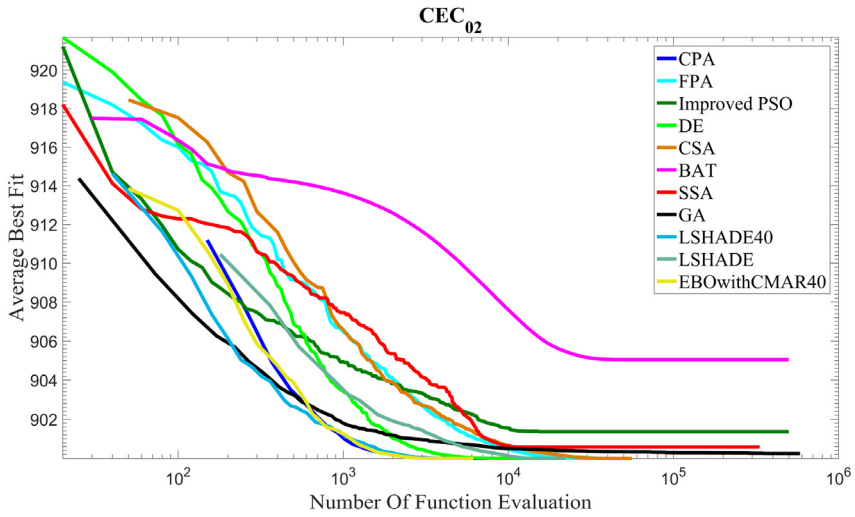


Fig. 14. Convergence curve of CPA and other competitive algorithms for Shifted and Rotated Levy test function.

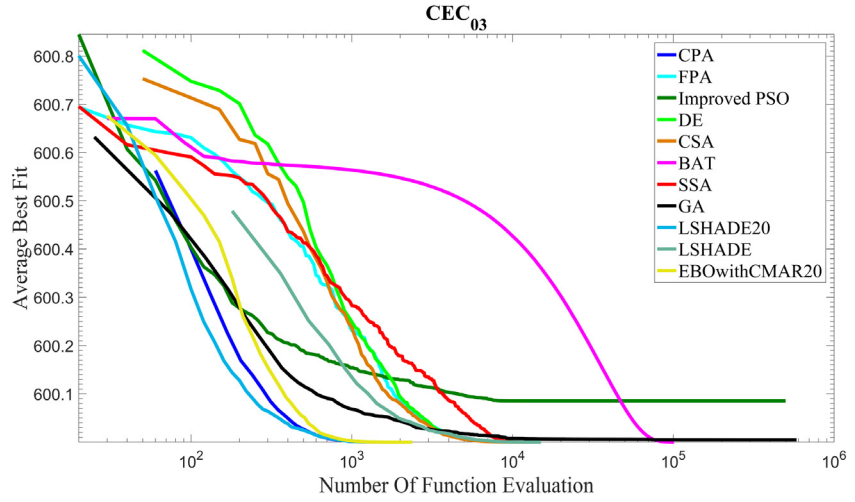


Fig. 15. Convergence curve of CPA and other competitive algorithms for Shifted and Rotated Schaffer F6 test function.

performance of the CPA is statistically significant better than its competitors.

3.4. Test IV - Time complexity of carnivorous plant algorithm

The bio-inspired metaheuristic algorithms are stochastic in nature, with random operations are executed during the exploration and exploitation. Therefore, the complexity of the bio-inspired metaheuristic algorithm is better to be described using the mathematical notation, called Big O notation, instead of from a deterministic point of view [43].

To determine the time complexity of CPA, the relationship between the increasing size of a problem and computational time is studied [44]. The performance of the CPA when optimizing De Jong test function with parameters: maximum number of function evaluation = 1000, population size = 30, group = 6, group iteration = 5, attraction rate = 0.8, growth rate = 2, reproduction rate = 1.8 and five values for problem size, dim = [10, 100, 1000, 10,000, 100,000], is assessed. The time complexity of the CPA, as shown in Fig. 14, is a linear time O(N), where N is the size of problem.

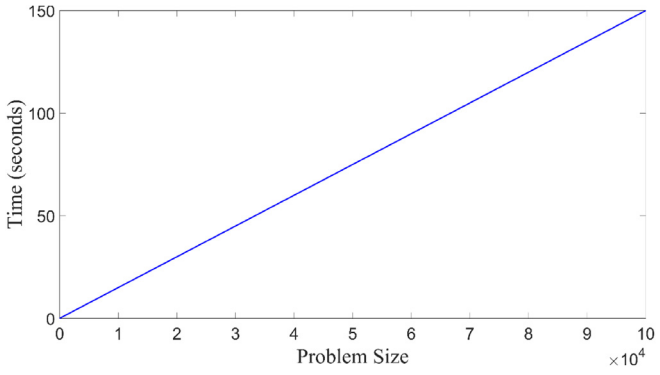


Fig. 16. The effect of increasing problem size on the CPA performance in terms of runtime.

3.5. Test V – Mechanical engineering design problems and CEC2011 problems

In addition to unconstrained optimization problems, the performance of CPA was evaluated through seven mechanical engineering design problems and four CEC2011 problems involving a

Table 9
Average error obtained in optimizing the benchmark test functions.

Test Function	CPA	FPA	PSO Improved	DE	CSA	BAT	SSA	GA
f_{01}	3.07E−07	3.33E−07	1.03E+02	3.11E−07	3.30E−07	1.22E−02	7.03E−03	1.42E−01
f_{02}	3.25E−07	3.33E−07	1.04E+02	3.31E−07	3.33E−07	1.20E−02	6.13E−03	4.18E−05
f_{03}	3.22E−07	3.32E−07	5.51E+05	3.30E−07	3.24E−07	6.24E+00	1.09E+01	3.22E−07
f_{04}	6.49E−07	1.48E−02	3.56E+00	1.82E−02	1.23E−03	7.73E−03	6.86E−03	1.14E−01
f_{05}	3.13E−07	2.26E+00	8.61E+00	1.85E+00	1.16E−01	1.84E+01	2.32E+00	1.49E+00
f_{06}	3.33E−07	3.20E−07	5.40E+01	3.31E−07	3.30E−07	1.64E+05	9.17E−01	1.52E−02
f_{07}	3.33E−07	3.33E−07	2.53E+04	3.23E−07	3.31E−07	2.01E−03	2.75E+02	2.87E+03
f_{08}	3.33E−07	1.30E+01	1.37E+01	1.16E+01	1.99E+00	3.00E+00	8.63E+00	8.91E−02
f_{09}	3.33E−07	3.31E−07	1.21E+01	3.09E−07	3.16E−07	9.33E−04	6.49E−07	3.16E−07
f_{10}	3.10E−07	3.23E−07	9.13E+02	3.33E−07	3.31E−07	1.52E−02	3.62E−01	2.76E−04
f_{11}	3.31E−07	3.33E−07	7.47E+04	3.26E−07	3.25E−07	5.63E−01	4.99E+00	1.34E−03
f_{12}	3.32E−07	3.32E−07	5.36E+02	3.30E−07	3.23E−07	1.36E−03	1.62E+00	4.77E+01
f_{13}	2.24E−07	9.37E−06	8.90E−06	8.99E−06	8.93E−06	2.01E−04	3.08E−07	3.17E−04
f_{14}	6.05E−07	9.16E−06	5.06E−02	4.51E−03	9.23E−06	2.27E+00	6.17E−02	9.80E−02
f_{15}	3.25E−07	9.38E−06	9.35E−06	9.30E−06	9.33E−06	1.77E+01	5.96E−01	1.86E+00
f_{16}	3.31E−07	8.65E−06	1.78E+02	8.37E−06	8.30E−06	4.70E−03	1.83E−02	7.45E+00
f_{17}	1.53E−07	8.90E−06	5.64E+00	9.08E+00	8.66E−06	4.55E+01	1.87E+01	3.18E+00
f_{18}	3.30E−07	6.92E−06	6.63E−06	6.25E−06	6.89E−06	8.47E−06	8.80E−07	2.95E−06
f_{19}	7.27E−07	8.21E−06	8.24E−06	8.21E−06	8.24E−06	9.49E+00	8.80E−01	4.70E−02
f_{20}	2.81E−07	6.47E−06	3.31E+00	7.28E−06	6.39E−06	2.31E+00	1.11E+00	3.13E+00
f_{21}	3.68E−06	4.10E−06	4.29E−06	4.26E−06	4.63E−06	2.90E−01	4.17E−06	7.93E−03
f_{22}	4.37E−06	5.45E−06	2.67E−01	4.84E−06	4.37E−06	9.67E−01	5.16E−06	3.90E−01
f_{23}	6.72E−08	4.54E−06	5.40E−06	4.46E−06	3.36E−06	7.62E−02	8.69E−08	7.10E−03
f_{24}	4.49E−06	5.15E−06	4.62E−06	4.86E−06	4.96E−06	7.79E+00	4.82E−06	3.57E−03
f_{25}	5.43E−06	4.85E−06	4.65E−06	4.42E−06	4.05E−06	4.29E+01	5.74E−06	4.93E−02
f_{26}	3.84E−06	4.01E−06	5.15E−01	1.70E−01	5.15E−06	5.88E−01	4.21E−06	9.75E+00
f_{27}	3.20E−06	3.93E−06	1.38E−05	4.86E−06	4.87E−06	3.80E−06	4.24E−06	1.25E−03
f_{28}	3.80E−06	5.58E−06	4.85E−06	4.81E−06	4.15E−06	6.11E−06	5.45E−06	3.47E−03
f_{29}	4.07E−06	4.90E−06	1.62E−04	4.57E−06	4.66E−06	1.15E−02	4.03E−06	4.51E−04
f_{30}	2.77E−07	4.85E−06	4.05E−05	4.16E−06	5.28E−06	5.91E−06	2.18E−08	2.85E−01
CEC_{01}	7.41E−07	3.31E−07	7.46E−03	2.29E−07	3.06E−07	3.33E−07	2.59E−07	3.60E−04
CEC_{02}	7.45E−06	8.64E−06	1.36E+00	8.01E−06	9.72E−06	5.04E+00	5.78E−01	2.46E−01
CEC_{03}	8.29E−06	7.85E−06	8.53E−02	8.74E−06	6.91E−06	9.47E−06	9.47E−06	4.83E−03
CEC_{04}	3.41E−07	1.56E−05	9.80E−03	5.93E−03	3.29E−06	4.16E−02	3.19E−03	3.43E−03
CEC_{05}	8.25E−06	7.24E−06	9.95E+12	6.07E−06	7.48E−06	8.36E−06	1.53E+03	2.01E+09
CEC_{06}	8.43E−06	8.20E−06	1.22E+04	8.27E−06	8.57E−06	1.02E−04	8.15E−06	1.46E+03
CEC_{07}	3.32E−07	9.86E−04	8.42E−02	3.24E−07	3.23E−07	8.79E−01	4.74E−02	5.65E−02

set of constraints. To handle the design constraints, the commonly used penalty method was utilized as the constraint handling method. The fitness value was penalized by adding a large value to any of the constraint violation. The penalty value of 10^{30} was assigned in this study. The optimized results of CPA, specifically, the best, mean and worst obtained fitness value as well as the standard deviation (SD), were then compared to other optimizers obtained from the literature.

3.5.1. Welded beam design problem

The main purpose of the welded beam design problem was to minimize the total cost of the welded beam, which could withstand the maximum bending stress σ_d of 30,000 psi, maximum end deflection δ_d of 0.25 in, shear stress τ_d of 13,600 psi and loading condition P of 6000 lb. There were four design variables in this problem: the width h (x_1) and length l (x_2) of the welded area, the depth t (x_3) and thickness b (x_4). For more details regarding the mathematical model and its constraints, please see Appendix B.1.

From the statistical findings in Table 11, it can be observed that CPA shows better performance than its competitive counterpart, specifically, differential search algorithm (DSA) [45], modified grey wolf optimization (MGWO)-III [46], MVO [46], ABC [47], sine cosine algorithm (SCA) [46] and GWO [46], with the lowest total cost of 1.72485231. To achieve this best result, CPA requires only 4494 NOFE and the computational time of 0.1983832 s.

3.5.2. Multiple disk clutch brake design problem

The aim of multiple disk clutch brake design problem was to minimize its overall mass. There were five design variables to be optimized, namely, number of friction surface Z, actuating force F, the thickness of the disk t, outer radius R_o and inner radius R_i, which must be discrete value. The mathematical model of this design problem can be referred to Appendix B.2.

Table 10
Wilcoxon signed rank test results.

Comparison	R^+	R^-	Null Hypothesis	p-value
CPA versus FPA	619	84	Reject	0.00005
CPA versus PSO Improved	700	3	Reject	0.00001
CPA versus DE	603	100	Reject	0.00015
CPA versus CSA	598	105	Reject	0.00020
CPA versus BAT	701	2	Reject	0.00001
CPA versus SSA	700	3	Reject	0.00001
CPA versus GA	682	21	Reject	0.00001

This design problem was recently optimized by using IAPSO [48], APSO [4], WCA [20], ABC [47] and TLBO [17]. The comparative statistical results are tabulated in Table 12. It can be seen that the best solution of 0.282541 obtained by the proposed CPA is the lowest among all, which has improved 10% of the solution accuracy obtained by others. The best fitness value obtained by APSO is the highest. Moreover, the proposed CPA requires only 360 NOFE and 0.28254094 s to solve this problem. The NOFE required by the APSO in solving this problem is the highest, while CPA uses 82% less NOFE than the APSO.

3.5.3. Rolling bearing design problem

The bearing has been invented to reduce friction for rotational or linear movement in a mechanical system. The goal of this design problem was to maximize its dynamic load carrying capacity,

Table 11
Comparison of best result and statistical results attained by different optimizers for welded beam design problem.

	Algorithm						
	DSA	GWO	ABC	MVO	SCA	MGWO-III	CPA
x_1	0.2054068	0.20567800	0.21267803	0.20561100	0.20469500	0.20566700	0.20572964
x_2	3.4825428	3.47140300	3.39734605	3.47210300	3.53629100	3.47189900	3.47048868
x_3	9.0504378	9.03696400	8.88418618	9.04093100	9.00429000	9.03667900	9.03662390
x_4	0.2056651	0.20572900	0.21420809	0.20570900	0.21002500	0.20573300	0.20572964
Best	1.7278830	1.72499500	1.76259824	1.72547200	1.75917300	1.72498400	1.72485231
Mean	1.9308745	1.72522800	1.93319894	1.72968000	1.81765700	1.72515600	1.83765549
Worst	2.2516695	1.72566400	2.23355071	1.74165100	1.87340800	1.72542000	2.19638593
SD	0.2054068	0.000187000	0.118909566	0.004866000	0.027543000	0.000137000	0.16481285

Table 12
Comparison of best result and statistical results attained by different optimizers for multiple disk clutch brake design problem.

	Algorithm					
	ABC	TLBO	WCA	APSO	IAPSO	CPA
R_i	–	70	70	76	70	73
R_o	–	90	90	96	90	93
t	–	1	1	1	1	1
F	–	810	910	840	900	990
Z	–	3	3	3	3	2
Best	0.313657	0.313657	0.313657	0.337181	0.313657	0.282541
Mean	0.324751	0.327166	0.313656	0.506829	0.313656	0.342761
Worst	0.352864	0.392071	0.313656	0.716313	0.313656	0.529770
SD	–	–	1.69×10^{-16}	0.09767000	1.13×10^{-16}	0.05243258
NOFE	>900	>900	500	2000	400	360

***Bolded font* = *Infeasible solution*.**

C_d, with ten design variables of pitch diameter of the bearing D_m, ball diameter D_b, number of the balls Z, inner raceway curvature f_i, outer raceway curvature coefficients f_o, K_{Dmin}, K_{Dmax}, ϵ, e and ζ. The first five design variables gave impact on the internal geometries of the bearing while the last five design variables were part of the constraints. The mathematical model and the set of constraints are given in Appendix B.3.

The comparative best performances of CPA and the proposed approaches available in literature, namely, TLBO [17], WCA [20], MVO [46], SCA [46] and MGWO-III [46] are summarized in Table 13. Although the TLBO achieves the highest dynamic load carrying capacity of 81979.575272, the violation of constraints g_4, g_7, g_8 and g_9 could be observed. Hence, the produced optimal solutions of the TLBO is infeasible in this regard. The CPA is thus a better method, with the second highest fitness of 81849.21039788 is attained and without triggering any constraint violation. The proposed CPA uses 6294 NOFE and 0.3353442 s to achieve the best solution for this design problem.

3.5.4. Heat exchange design problem

The proper design of a heat exchanger was challenging due to all constraints were interrelated. The details regarding the mathematical model and its constraints are given in Appendix B.4.

Different improved versions of harmony search, such as improving proposed harmony search (IPHS) [49], proposed harmony search (PHS) [49] and new harmony search (NHS) [50], have been utilized to optimize this case study. Table 14 summarizes the comparative cost values with CPA. It can be concluded that the CPA completes the search with the lowest fitness value, which is 7049.96403564, as compared to the harmony search variants. In addition, the produced best fitness value by CPA is the closest to the true fitness value, with the difference of merely 0.008% is obtained.

3.5.5. Speed reducer design problem

A speed reducer was used to raise the torque and lower the speed of the gearbox output. The purpose of this design problem is to minimize the total mass of the speed reducer while satisfying all the constraints, such as bending stress of the gear teeth, surface stress, transverse deflections of the shafts due to transmitted force and stresses in the shafts. Seven decision parameters were involved, given as diameter of shaft 1 (d_1), diameter of shaft 2 (d_2), length of shaft 1 between bearings (l_1), length of shaft 2 between bearings (l_2), number of teeth on pinion (z), module of teeth (m) and face width (b). More details regarding the mathematical model and the involved constraints please refer to Appendix B.5.

As shown in Table 15, the performance comparison reveals that the proposed CPA outperforms its competitors, such as MGWO variants [46], GWO [51], SCA [52], MVO [21], and CSA [53], with the best lowest fitness value of 2996.21851300. Moreover, not only using fewer NOFE (4494) and shorter computational time (0.2145254 s) to complete the search process, the CPA demonstrates consistent performance, with the standard deviation of approximately 0 is observed. Upon closer inspection, it can be seen that the optimization performance of others is less stable, exhibiting the standard deviation within the range of 0.9788 to 92.5726.

3.5.6. Pressure vessel design problem

The goal of this design problem was to design a compressed air tank at minimum overall cost, including the forming cost, welding cost and material cost. The designed air tank should also can withstand a working pressure of 3000 psi and have a minimum volume of 750 ft^3. Besides, both ends of the cylindrical vessel were capped with hemispherical heads. There were four design variables involved, namely, the thickness x_1, the thickness of the head x_2, the inner radius x_3 and the length of the cylindrical section of the vessel x_4. Moreover, the thickness x_1 and thickness of the head x_2 should be discrete values, which were integer multiples of 0.0625 inch [54]. Please refer to Appendix B.6 for the mathematical model and the constraints of this design problem.

The comparative result obtained for the pressure vessel design problem (see Table 16) shows that the CPA and CSA have better

Table 13
Comparison of best result and statistical results attained by different optimizers for rolling element bearing design problem.

	TLBO	WCA	MVO	SCA	MGWO-III	CPA
D_m	125.719100	125.721167	125.640200	125.000000	125.705100	125.722718
D_b	21.425590	21.423300	21.422500	21.148340	21.422940	21.423301
Z	11.000000	11.001030	10.993380	10.969280	10.99905	11.001159
f_i	0.514945	0.515000	0.515019	0.521696	0.515009	0.515000
f_o	0.514945	0.515000	0.515019	0.521696	0.515009	0.515000
K_{Dmin}	0.424266	0.401514	0.499800	0.500000	0.487200	0.473508
K_{Dmax}	0.633948	0.659047	0.687820	0.700000	0.638340	0.617554
ε	0.300000	0.300032	0.301348	0.300000	0.300000	0.300000
e	0.068858	0.040045	0.046170	0.027780	0.076050	0.086504
ζ	0.799498	0.600000	0.600610	0.629120	0.673280	0.680706
g_1	0.000004	0.000005	0.001453	0.078566	0.000830	0
g_2	13.152560	14.740620	7.859000	7.296680	8.741880	9.701030
g_3	1.525180	3.286690	5.302400	6.703320	1.837920	0.382238
g_4	**2.559350**	−3.423300	−3.404200	−2.274740	−1.224540	−1.002116
g_5	0.719100	0.721167	0.640200	0	0.705100	0.722718
g_6	16.495400	9.290083	10.902300	6.945000	18.307400	20.903397
g_7	**−0.000022**	0.000091	0.013022	0.581328	0.009098	0
g_8	**−0.000055**	0	0.000019	0.006696	0.000009	0
g_9	**−0.000055**	0	0.000019	0.006696	0.000009	0
C_d	81979.575272	81848.559216	81765.800054	68945.201888	81817.916270	81849.21039788
Mean	–	–	–	–	–	81849.20254772
Worst	–	–	–	–	–	81849.01507547
SD	–	–	–	–	–	0.035667

Bolded font* = *Infeasible solution.

Table 14
Comparison of best result and statistical results attained by different optimizers for heat exchanger design problem.

	NHS	PHS	IPHS	CPA	Optimal Solution
x_1	500.0038	565.6476	598.1956	545.90884864	579.3167
x_2	1359.3110	1000	1285.3993	1384.02691346	1359.943
x_3	5197.9595	5543.5424	5167.7061	5120.02827352	5110.071
x_4	174.7263	180.8650	183.5732	179.16080043	182.0174
x_5	292.0817	278.2583	293.2945	295.19886906	295.5985
x_6	224.7054	219.1347	216.4256	220.83919957	217.9799
x_7	282.6446	302.6066	290.2749	283.96193137	286.4162
x_8	392.0817	378.2583	393.2936	395.19886906	395.5979
Best	7057.274414	7109.1901	7051.3012	7049.96403564	7049.330923
Mean	–	–	–	7468.39178713	–
Worst	–	–	–	9147.50804836	–
SD	–	–	–	435.9452475	–

Table 15
Comparison of best result and statistical results attained by different optimizers for speed reducer design problem.

	Algorithm					
	MVO	SCA	GWO	MGWO-II	MGWO-III	CPA
b	3.508502	3.508755	3.50069	3.500569	3.5001	3.50000005
m	0.7	0.7	0.7	0.7	0.7	0.70000000
z	17	17	17	17	17	17.00000013
l_1	7.392843	7.300000	7.310933	7.319115	7.300568	7.30000094
l_2	7.816034	7.800000	7.814726	7.804140	7.806687	7.80000020
d_1	3.358073	3.461020	3.351047	3.350477	3.350680	3.35021478
d_2	5.286777	5.289213	5.286741	5.286695	5.286903	5.28668325
Best	3002.928000	3030.563000	2997.288000	2996.905000	2996.798000	2996.21851300
Mean	3028.841000	3065.917000	2999.640000	2998.517000	2997.978000	2996.21937612
Worst	3060.958000	3104.779000	3003.889000	3000.739000	2999.825000	2996.22293629
SD	13.018600	18.074200	1.931930	1.102990	0.978800	8.770273×10^{-4}

performance than other approaches, *i.e.*, WOA [55], GA [56], co-evolutionary particle swarm optimization (CPSO) [57], CSA [53], MBA [58] and MGWO-III [46], with the attained lowest manufacturing cost of 6059.7143. However, the CPA is superior to CSA in terms of stability, as observed from the lower standard deviation value. Although the performance of CPA is less consistent than CPSO and GA as the yielded standard deviation is higher, CPA has faster convergence rate as it requires less NOFE to find the optimal solution. In addition, the MGWO-III and MBA consider the thickness x_1 and thickness of the head x_2 as continuous value, which should be discrete value.

3.5.7. Car side impact design problem

The mathematical model of a car side impact was developed using derivation response surface method in accordance with [59]. Eleven design variables were considered in this study, namely, thicknesses of B-Pillar inner x_1, B-Pillar reinforcement x_2, floor side inner x_3, cross members x_4, door beam x_5, door beltline reinforcement x_6, roof rail x_7, materials of B-Pillar inner x_8, materials of floor side inner x_9, barrier height x_{10}, and hitting position x_{11}. The purpose of this study was to minimize the weight of the car, with nine constraints were taken into

Table 16
Comparison of best result and statistical results attained by different optimizers for pressure vessel design problem.

	Algorithm						
	WOA	GA	CPSO	CSA	MBA	MGWO-III	CPA
x_1	0.8125000	0.8125000	0.8125000	0.8125000	**0.7802000**	**0.7782000**	0.8125000
x_2	0.4375000	0.4375000	0.4375000	0.4375000	**0.3856000**	**0.3847000**	0.4375000
x_3	42.0982699	42.0974000	42.0913000	42.0984000	40.4292000	40.3155000	42.0984456
x_4	176.638998	176.654000	176.746500	176.636600	198.496400	199.959300	176.636596
Best	6059.7410	6288.7445	6061.0777	6059.7143	5889.3216	5884.0616	6059.7143
Mean	6068.0500	6293.8432	6147.1332	6447.7360	6200.6477	5884.7820	6493.2009
Worst	–	6308.4970	6363.8041	6495.3470	6392.5062	5886.3961	7332.8415
SD	65.6519	7.4133	86.4500	502.6930	160.3400	0.9770	417.7136
NOFE	6300	900,000	240,000	15,000	70,650	–	12,020

***Bolded font* = *Infeasible solution*.**

Table 17
Comparison of best result and statistical results attained by different optimizers for car side impact design problem.

	PSO	DE	FA	CS	TLBO	TLCS	CPA
x_1	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000000
x_2	1.11670	1.11670	1.36000	1.11643	1.11350	1.11630	1.11575860
x_3	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000000
x_4	1.30208	1.30208	1.20200	1.30208	1.30700	1.30230	1.30321196
x_5	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000000
x_6	1.50000	1.50000	1.12000	1.50000	1.50000	1.50000	1.50000000
x_7	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000000
x_8	0.34500	0.34500	0.34500	0.34500	0.34500	0.34500	0.34500000
x_9	0.19200	0.19200	0.19200	0.19200	0.19200	0.19200	0.27247957
x_{10}	−19.54935	−19.54935	8.87307	−19.54935	−20.06550	−19.57210	−19.67009727
x_{11}	−0.00431	−0.00431	−18.99808	−0.00431	0.11390	0.01570	0.00000206
g_1	−0.61751	−0.61751	−0.47297	−0.61742	−0.62362	−0.61770	−0.63834
g_2	−0.09271	−0.09271	−0.10214	−0.09270	−0.09286	−0.09271	−0.08303
g_3	−0.06878	−0.06878	−0.09297	−0.06879	−0.06864	−0.06878	−0.06673
g_4	0.57886	0.57886	0.71831	0.57874	0.57712	0.57867	0.57168
g_5	−4.27833	−4.27833	−3.23254	−4.27776	−4.30866	−4.27912	−3.99122
g_6	−7.28381	−7.28381	−4.03456	−7.28210	−7.35052	−7.28511	−6.68735
g_7	−0.00264	−0.00264	−0.83813	**0.00004**	**0.00047**	**0.00006**	0.00000
g_8	−0.00002	−0.00002	**0.02129**	0.00000	**0.00001**	0.00000	0.00000
g_9	−0.96543	−0.96543	−0.53505	−0.96515	−0.97391	−0.96554	−0.96723
g_{10}	−0.16660	−0.16660	**0.14839**	−0.16660	−0.17216	−0.16701	−0.23604
Best	22.84474	22.84474	22.84298	22.84294	22.8436	22.8430	22.84298982
Mean	22.89429	23.22828	22.89376	22.85858	22.9635	22.8443	23.12998013
Worst	23.21354	24.12206	24.06623	23.25998	23.4058	22.8501	23.38526236
SD	0.150700	0.344510	0.166670	0.076120	0.14610	0.0016	0.17871462

***Bolded font* = *Infeasible solution*.**

Table 18
Comparison of best result and statistical results attained by different optimizers for parameter estimation for frequency-modulated sound waves.

	Algorithm						
	CDASA	EA-DE-MA	SAMODE	DE-RHC	Adap. DE171	GA-MPC	CPA
Best	0	0	0	0.1060	0	0	0
Mean	13.9950	7.6022	3.0769	12.6000	4.8526	0	0
Worst	22.0810	15.4409	12.5473	20.4000	19.9208	0	0
SD	6.4643	6.0206	5.0495	6.0900	6.6900	0	0
NOFE	50,000	50,000	50,000	50,000	50,000	50,000	50,000

consideration. The details of mathematical model for this problem can refer to Appendix B.7.

From the best cost value obtained in Table 17, it can be concluded that the CPA and the competitive algorithms, namely, PSO [60], DE [60], FA [60], CSA [61], TLBO [62] and teaching leaning based cuckoo search (TLCS) [61], have similar performance, in which a similar fitness value of 22.84 is obtained. However, the optimal solutions produced by GA, FA, CSA, TLBO and TLCS could not be accepted due to some constraints are violated. Thus, the CPA is a better method for this problem by obtaining the minimum fitness value while satisfying all the constraints. Besides, the CPA only requires 16806 NOFE and 1.2545834 s of processing time to converge to the optimal solution.

3.5.8. Parameter estimation for frequency-modulated (FM) sound waves (CEC2011 problem)

In the current music system, frequency-modulated (FM) sound wave synthesis plays a vital function to produce a sound that is close to the target sound [63]. It is also one of the real-world numerical optimization problems in the CEC 2011 competition. There are six decision parameters of FM synthesizer in this problem that have to be optimized, which are $X = \{a_1, \omega_1, a_2, \omega_2, a_3, \omega_3\}$. For more details of the involved mathematical model and constraints, kindly refer to Appendix B.8.

In this study, the optimization results of the CPA are compared with the algorithms participated in the CEC 2011 competition, namely, continuous differential ant-stigmergy algorithm

Table 19
Comparison of best result and statistical results attained by different optimizers for the bi-functional catalyst blend optimal control problem.

	Algorithm						
	CDASA	EA-DE-MA	SAMODE	DE-RHC	Adap. DE171	GA-MPC	CPA
Best	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05
Mean	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05
Worst	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05	1.1515E−05
SD	0	0	0	0	0	0	0
NOFE	50,000	50,000	50,000	50,000	50,000	50,000	50,000

Table 20
Comparison of best result and statistical results attained by different optimizers for optimal control of a non-linear stirred tank reactor.

	Algorithm						
	CDASA	EA-DE-MA	SAMODE	DE-RHC	Adap. DE171	GA-MPC	CPA
Best	13.7710	13.8542	13.7708	20.3000	14.3291	13.7708	13.7708
Mean	14.1730	15.4119	14.3396	21.1000	18.3122	14.3692	14.0357
Worst	14.3290	20.8780	20.9574	21.8000	21.5396	19.5308	14.3291
SD	0.2559	1.9663	1.4047	0.3250	3.2548	1.1816	0.2795
NOFE	50,000	50,000	50,000	50,000	50,000	50,000	50,000

Table 21
Comparison of best result and statistical results attained by different optimizers for spread spectrum radar polyphase code design.

	Algorithm						
	CDASA	EA-DE-MA	SAMODE	DE-RHC	Adap. DE171	GA-MPC	CPA
Best	0.6760	0.5	0.8206	1.0800	0.5000	0.7752	0.6248
Mean	1.0262	0.5278	1.2862	1.3100	−0.5000	1.6151	1.0292
Worst	1.4663	0.6206	1.7012	1.5300	−0.5000	1.9160	1.3672
SD	0.1593	0.0355	0.1932	1.1700	0	0.3237	0.1914
NOFE	50,000	50,000	50,000	50,000	50,000	50,000	50,000

(CDASA) [64], evolutionary algorithm differential evolution memetic algorithm (EA-DE-MA) [65], self-adaptive multi-operator differential evolution (SAMODE) [66], hybrid differential evolution–random hill climber (DE-RHC) [67], adaptive differential evolution (Adap. DE171) [68] and genetic algorithm with a multi-parent crossover (GA-MPC) [69], as shown in Table 18. From Table 18, it can be seen that the CPA has the same performance as GA-MPC, which is the winner of the CEC 2011 competition. Furthermore, both CPA and GA-MPC outperformed other competitive algorithms in these problems.

3.5.9. The bi-functional catalyst blend optimal control problem (CEC2011 problem)

The bi-functional catalyst blend optimal control problem is a hard multimodal function with 300 local optima [70]. It is a chemical process which converts methylcyclopentane to benzene in a tubular reactor. The objective is to identify the optimal catalyst blend u along the length of the reactor at time interval $0 \leq t \leq 2000$ such that the performance index I is maximized. Details of the mathematical model and constraints involved can be referred in Appendix B.9.

As shown in Table 19, it can be observed that the CPA and the competitors, including CDASA [64], EA-DE-MA [65], SAMODE [66], DE-RHC [67], Adap. DE171 [68] and GA-MPC [69], behave equally, in which a similar fitness value of 1.1515E−05 has been achieved.

3.5.10. Optimal control of a non-linear stirred tank reactor (CEC2011 problem)

A first-order irreversible chemical reaction in a continuous stirred tank reactor has been formulated as a complex multimodal optimal control problem [71]. The objective of this problem was to identify the optimal value of u so that the performance index I was minimized when the initial condition is $x_0 = [0.09 \quad 0.09]$. The objective function of this control problem and two nonlinear differential equations that model the chemical process are given in Appendix B.10.

Table 20 compares the optimization results of the CPA with that of CDASA [64], EA-DE-MA [65], SAMODE [66], DE-RHC [67], Adap. DE171 [68] and GA-MPC [69]. As shown in Table 20, CPA, GA-MPC and SAMODE obtained the lowest best fitness. However, CPA outshines the GA-MPC and SAMODE as the mean fitness, worst fitness and SD obtained by CPA are lower. Besides, the Adap. DE171 has the worst stability due to its obtained SD is the highest among all, while DE-RHC has the worst performance as the obtained mean fitness is the highest.

3.5.11. Spread spectrum radar polly phase code design (CEC2011 problem)

When developing a radar device that uses pulse compression, the option of the correct waveform needs to be given a significant attention. There are other methods known for modulating the radar pulse, making the pulse compression possible. Polyphase codes are desirable because they reduce side lobes in the compressed signal and make the digital processing techniques easier for implementation. A new approach for the synthesis of polyphase pulse compression code was proposed by Dukic and Do-brosavljevic [72]. Based on their study, this problem was modelled as a min–max nonlinear non-convex optimization problem. The objective of this problem was to minimize the largest module among the samples. The details of mathematical model of this problem can be referred to Appendix B.11.

Table 21 shows the results of the CPA and the competitor algorithms, namely, CDASA [64], EA-DE-MA [65], SAMODE [66], DE-RHC [67], Adap. DE171 [68] and GA-MPC [69]. The EA-DE-MA and Adap. DE171 obtained the lowest best fitness, which is 0.5, as shown in Table 21. Although CPA is inferior as compared to these

approaches, CPA outperformed GA-MPC, which is the winner of the CEC2011 competition, in this study. In addition, the DE-RHC has the worst stability as its obtained SD is the highest among all, while GA-MPC has the worst performance due to the highest obtained mean fitness.

3.6. Test VI – Obstacle avoidance of 5-DOF robotic arm using carnivorous plant algorithm

Many tasks in industries are risky, dirty, dull, complicated and repetitive. Hence, the robotic arm with stable, fast and accurate positioning has received great attention over the last few decades. A redundant robotic arm, also known as the redundant manipulator, is a robot with more degrees of freedom (DOF) than required. These extra DOF enhance the flexibility of the robot so that it can reach any position without colliding with any obstacle. The DOF also produce multiple joint configurations for the robotic arm to reach its desired position. Such redundancy solutions, however, increase the complexity of the kinematics problem because the singularities configurations and non-linearity may exist in the solutions [73].

The kinematics problem of the robotic can be classified into forward kinematics (FK) and inverse kinematics (IK). The FK describes the relationship between the joint coordinates (input) and the position of the end-effector of the robotic arm (output). The reverse of FK is the IK. The FK problem is simple and straightforward because the equations are not complex to derive. Hence, every robotic arm has its forward kinematics solution. The general formula of FK is given as:

$$f_{FK}(\theta) = (X; Y; Z) \tag{13}$$

where θ is the orientation vector of the n joints and $(X; Y; Z)$ is the Cartesian coordinate of the end-effector. The forward kinematics model begins with

$$\begin{aligned} {}^{0}T_n &= {}^{0}T_1(\theta_1)\,{}^{1}T_2(\theta_2)\,{}^{2}T_3(\theta_3)\ldots{}^{n-1}T_n(\theta_n) \\ &= \prod_{i=1}^{n} {}^{i-1}T_i(\theta_i) \end{aligned} \tag{14}$$

where n is n joints, θ is the orientation vector and ${}^{0}T_n$ denotes a homogeneous matrix. The ${}^{0}T_n$ defines the position and orientation of the end effector posture by the product of the matrices represented by each link, from the beginning until the end. Each link homogeneous matrix ${}^{i-1}T_i(\theta_i)$ can be expressed as follows:

$${}^{i-1}T_i(\theta_i) = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{15}$$

where the parameters a_i, d_i and α_i represent link length, link offset and link twist, respectively. The c and s denote *sin* and *cos* operations, respectively. Thus, the homogeneous matrix ${}^{0}T_n$ can be written as:

$${}^{0}T_n = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n & o & a & p \\ & 0 & & 1 \end{bmatrix} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \tag{16}$$

where R is a rotation matrix, while vector p represents the (x, y, z) coordinate of the end-effector. Thus, the coordinate of the end-effector can be determined easily with the developed FK model when the orientation of each joint is known.

Table 22
Denavit–Hartenberg Parameters.

Link	θ_iRange	θ_i	d_i	a_i	α_i
1	$0 \le \theta_1 \le 180$	θ_1+90	120	20	90
2	$0 \le \theta_2 \le 180$	θ_2	0	160	0
3	$0 \le \theta_3 \le 180$	θ_3+90	0	0	90
4	$0 \le \theta_4 \le 180$	θ_4	120	0	−90
5	$0 \le \theta_5 \le 180$	θ_5-90	0	100	0

However, in most of the robotic arm application, the end-effector is required to follow a continuous path to fulfil a specific task. Thus, finding the optimal joint coordinates by given the coordinate of the end-effector is the main goal in practical application. This can be solved through IK. The general formula of the IK is presented as follows:

$$f_{IK}(X; Y; Z) = \theta \tag{17}$$

With IK, the orientation vector of the n joints can be computed by given the coordinate of end-effector, and hence, the posture of the robot can be controlled. However, IK is a much harder problem as compared to FK. This is because the computational power of solving IK problem is high when the singularities and non-linearity exist in the solutions [74]. Consequently, the control of the robotic arm becomes unrealistic in real-time. Besides, some structures of the robotic do not even have the exact solution of IK [75].

Several approaches have been developed to solve the IK problem. These techniques can be divided into two major categories, which are conventional approaches and artificial intelligence approaches [76]. There are three techniques, which are algebraic technique, geometric technique and numerical technique, in conventional approaches. They, however, are facing the difficulties and slow in solving the IK problem when the DOF of the robotic arm is large. Moreover, the algebraic technique does not always guarantee the exact solutions, while the geometric technique requires the closed-form solutions that must geometrically exist for the first three joints of the robotic arm [77]. The numerical technique, also known as iterative technique, is sensitive to the selection of starting point, and hence, may fail to converge to exact solutions [78].

Due to the ineffectiveness of conventional approaches, artificial intelligence approaches have gained popularity in robotic control recently. The neural network, which is one of the artificial intelligence approaches, is utilized to solve the nonlinear IK problem. Zhang and Wang have introduced dual neural network for solving obstacle avoidance of the redundant robotic arm [79]. In addition, Li et al. have developed recurrent neural networks for kinematic control of multiple robotic arms that collaborate with each other without collision [80]. Neural networks with Q-learning reinforcement method, which is proposed by Duguleana et al. has been implemented in redundant manipulators for solving the IK and obstacle avoidance problem [81]. Furthermore, Almusawi et al. have proposed a new design of an artificial neural network to solve the IK problem of the robotic arm [82]. Although many researchers have used this technique to solve the IK problem, it has a disadvantage where the neural network must be trained for a long time before the promising solution of IK can be found [78].

Another way to solve the IK problem is by converting the FK problems into the problem of optimization. Then, the metaheuristic algorithm is utilized to minimize the distance between

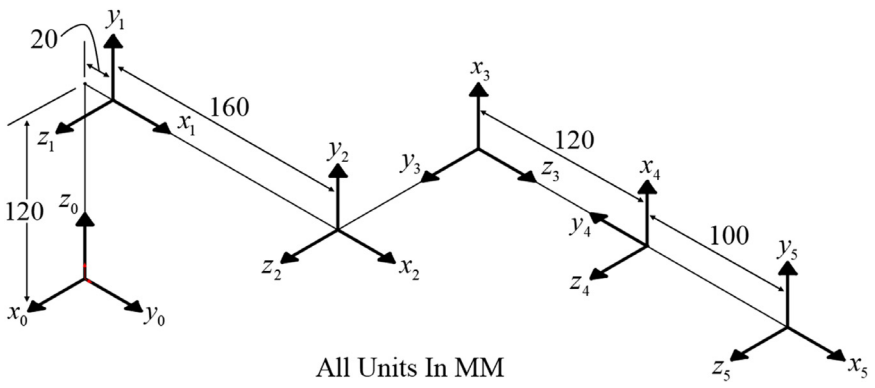


Fig. 17. Arm kinematic representation showing frame assign.

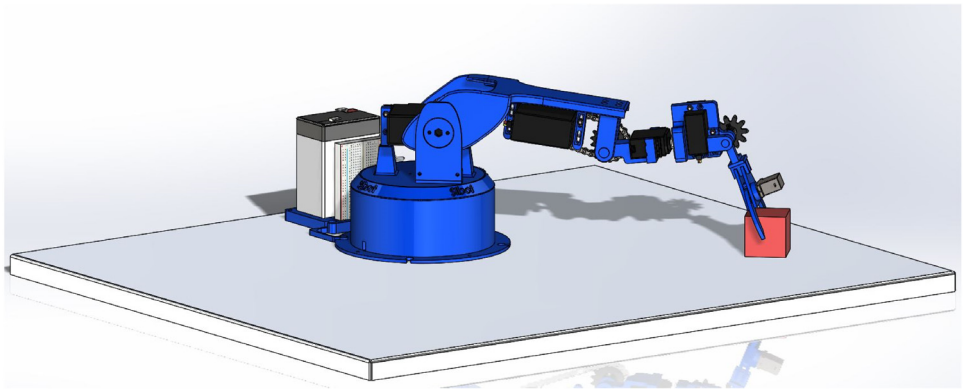


Fig. 18. Case 1: Optimal orientations obtained from CPA, with the position of the targeted object (red) at (−250, 225, 20).

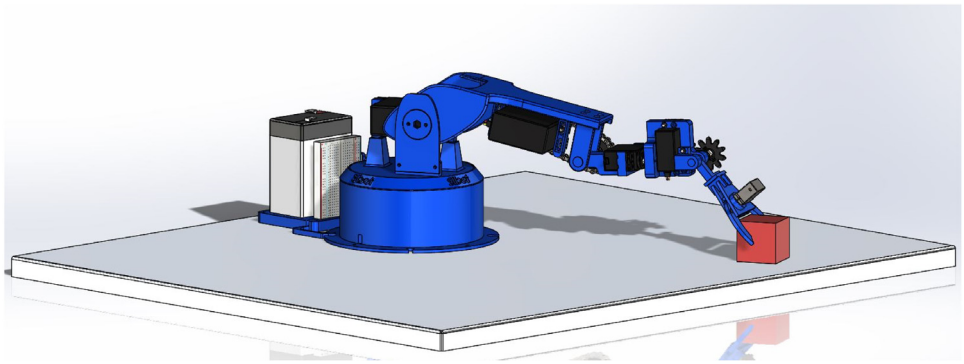


Fig. 19. Case 2: Optimal orientations obtained from CPA, with the position of the targeted object (red) at (−160, 325, 20).

the coordinate of the end effector and the targeted position [83–85]. In this study, the posture control of the 5-DOF robotic arm with obstacle avoidance is determined using CPA. To model the motion control of the 5-DOF robotic arm, the FK of the 5-DOF robotic arm has to be derived. The simplified kinematic representation of the 5-DOF robotic arm are presented in Fig. 17. The Denavit–Hartenberg (DH) parameters are obtained through the simplified kinematic representation of the robotic arm, as shown in Table 22. Then, the coordinate transformation matrix ${}^{0}T_{1}$, ${}^{1}T_{2}$, ${}^{2}T_{3}$, ${}^{3}T_{4}$ and ${}^{4}T_{5}$ for each link is calculated using Eq. (18) from the DH parameters. After that, the coordinate transformation matrix ${}^{0}T_{5}$ between the world coordinate system (x_0, y_0, z_0) and the end-effector coordinate system (x_5, y_5, z_5) are calculated using Eq. (19). Eq. (20) is known as FK model of this 5-DOF robotic arm. Finally, Eqs. (21) to Eq. (23) are used to determine the coordinate (x, y, z) of the end-effector, given the inputs $\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5.

$$
{}^{i-1}T_i(\theta_i) = \begin{bmatrix} \cos(\theta_i) & -\cos(\alpha_i)\cdot\sin(\theta_i) & \sin(\alpha_i)\cdot\sin(\theta_i) & a_i\cdot\cos(\theta_i) \\ \sin(\theta_i) & \cos(\alpha_i)\cdot\cos(\theta_i) & -\sin(\alpha_i)\cdot\cos(\theta_i) & a_i\cdot\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix},
$$

$$
i = 1, 2, \ldots, 5 \tag{18}
$$

$$
{}^{0}T_5 = {}^{0}T_1(\theta_1)\,{}^{1}T_2(\theta_2)\,{}^{2}T_3(\theta_3)\,{}^{3}T_4(\theta_4)\,{}^{4}T_5(\theta_5) \tag{19}
$$

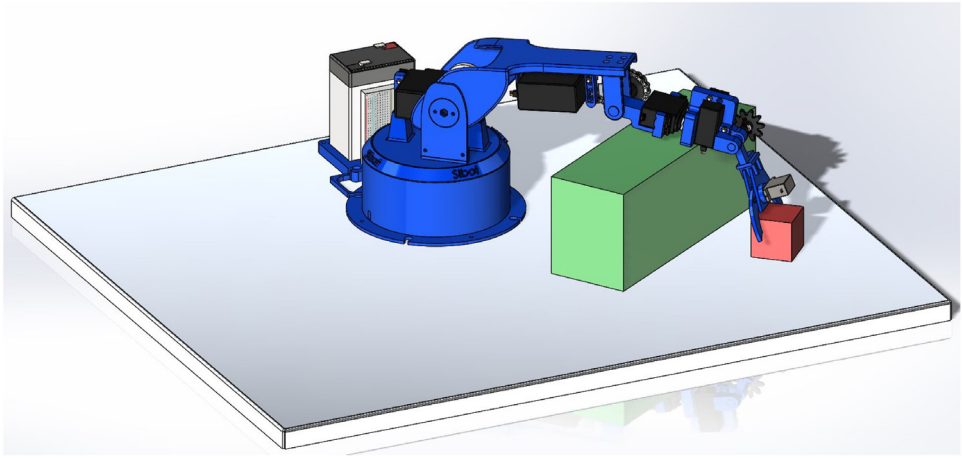


Fig. 20. Case 3: Optimal orientations obtained from CPA, with the position of the targeted object (red) at (−250, 225, 20).

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = {}^{0}T_{5} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{20}$$

$$\begin{aligned} x = {} & [\cos(\theta_1)\cdot\cos(\theta_2)\cdot\cos(\theta_3)\cdot\cos(\theta_4) \\ & -\cos(\theta_1)\cdot\sin(\theta_2)\cdot\sin(\theta_3)\cdot\cos(\theta_4) \\ & +\sin(\theta_1)\cdot\sin(\theta_4)]\cdot(100\cdot\cos(\theta_5)) \\ & -[\cos(\theta_1)\cdot\cos(\theta_2)\cdot\sin(\theta_3) \\ & +\cos(\theta_1)\cdot\sin(\theta_2)\cdot\cos(\theta_3)]\cdot(100\cdot\sin(\theta_5)) \\ & +120\cdot\cos(\theta_1)\cdot\cos(\theta_2)\cdot\sin(\theta_3) \\ & +120\cdot\cos(\theta_1)\cdot\sin(\theta_2)\cdot\cos(\theta_3) \\ & +160\cdot\cos(\theta_1)\cdot\cos(\theta_2)+20\cdot\cos(\theta_1) \end{aligned} \tag{21}$$

$$\begin{aligned} y = {} & [\sin(\theta_1)\cdot\cos(\theta_2)\cdot\cos(\theta_3)\cdot\cos(\theta_4) \\ & -\sin(\theta_1)\cdot\sin(\theta_2)\cdot\sin(\theta_3)\cdot\cos(\theta_4) \\ & -\cos(\theta_1)\cdot\sin(\theta_4)]\cdot(100\cdot\cos(\theta_5)) \\ & -[\sin(\theta_1)\cdot\cos(\theta_2)\cdot\sin(\theta_3) \\ & +\sin(\theta_1)\cdot\sin(\theta_2)\cdot\cos(\theta_3)]\cdot(100\cdot\sin(\theta_5)) \\ & +120\cdot\sin(\theta_1)\cdot\cos(\theta_2)\cdot\sin(\theta_3) \\ & +120\cdot\sin(\theta_1)\cdot\sin(\theta_2)\cdot\cos(\theta_3) \\ & +160\cdot\sin(\theta_1)\cdot\cos(\theta_2)+20\cdot\sin(\theta_1) \end{aligned} \tag{22}$$

$$\begin{aligned} z = {} & [\sin(\theta_2)\cdot\cos(\theta_3)\cdot\cos(\theta_4)+\cos(\theta_2)\cdot\sin(\theta_3)\cdot\cos(\theta_4)] \\ & \cdot(100\cdot\cos(\theta_5)) \\ & -[\sin(\theta_2)\cdot\sin(\theta_3)-\cos(\theta_2)\cdot\cos(\theta_3)]\cdot(100\cdot\sin(\theta_5)) \\ & +120\cdot\sin(\theta_2)\cdot\sin(\theta_3)-120\cdot\cos(\theta_2)\cdot\cos(\theta_3) \\ & +160\cdot\sin(\theta_2)+120 \end{aligned} \tag{23}$$

In Table 22, θ represents the angle rotated from the previous x-axis to current x-axis about the previous z-axis, d represents the distance moved from the previous z-axis to current y-axis, a represents the distance moved from the previous x-axis to current x-axis and α represents the angle rotated from the previous z-axis to current z-axis about the previous x-axis.

Since the (x, y, z) coordinates of the target object are known, the optimal values of $\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5 can be determined by the CPA in order for the robotic arm to grip the target object accurately without collision with obstacle. The objective function of this case study can be mathematically formulated as:

$$\text{Minimize } f\left(\vec{\theta}\right) = Position\_Error + collision \tag{24}$$

where

$$Position\_Error = \sqrt{(x_e - x_t)^2 + (y_e - y_t)^2 + (z_e - z_t)^2} \tag{25}$$

$$collision = \begin{cases} 0, & \text{safe} \\ 10^{15}, & \text{collide} \end{cases} \tag{26}$$

with the position of end effector was denoted as (x_e, y_e, z_e), while the position of the targeted object was defined as (x_t, y_t, z_t). The design variables were the five orientations, $\vec{\theta}$ of the 5-DOF robotic arm.

In this application, the 5-DOF robotic arm was set up in the environment of obstacle-free and with obstacle. In the obstacle-free environment, two case studies were formed with two different targeted positions at (−250, 225, 20) and (−160, 325, 20), as shown in Fig. 16 and Fig. 17. Meanwhile, three case studies were developed in the environment with an obstacle, as illustrated in Figs. 18 to 20. In these figures, the targeted object was marked as red colour while the obstacle was represented as green colour.

From previous section, it can be seen that both DE and CSA obtained good results in optimizing the benchmark test functions (see Tables 2 and 6) and hence, these algorithms were chosen as the competitive algorithm in this case study. Besides, FPA was chosen as another competitor as both FPA and CPA were plant-based optimizer. The SSA was chosen for performance comparison as well since it was one of the recently developed metaheuristic algorithms. The optimization process was repeated for 30 runs, with the parameter settings of all considered algorithms as given in Sections 3.1 and 3.2. The termination condition was fulfilled when the obtained tolerance error was below 10^{-5}, or when the NOFE achieved 10,000. If the simulation was terminated based on the latter condition, the algorithm was considered

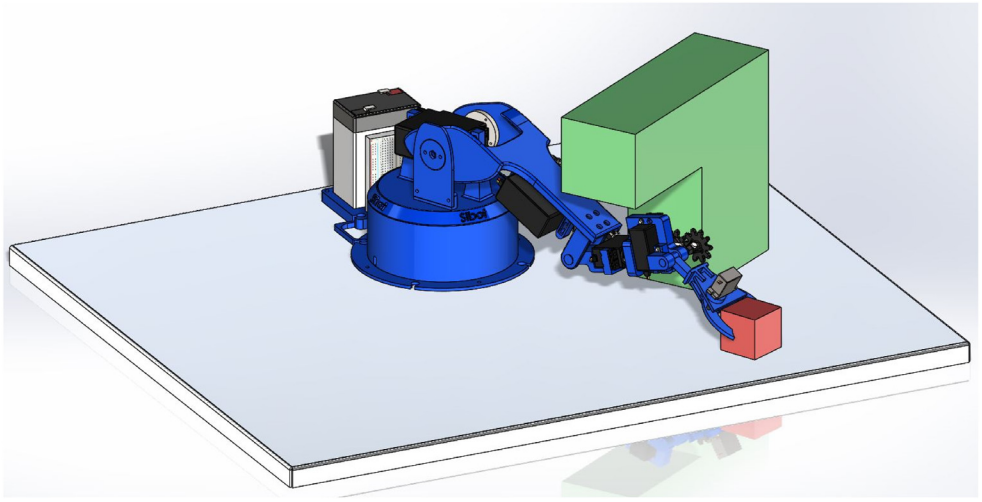


Fig. 21. Case 4: Optimal orientations obtained from CPA, with the position of the targeted object (red) at (−160, 325, 20).

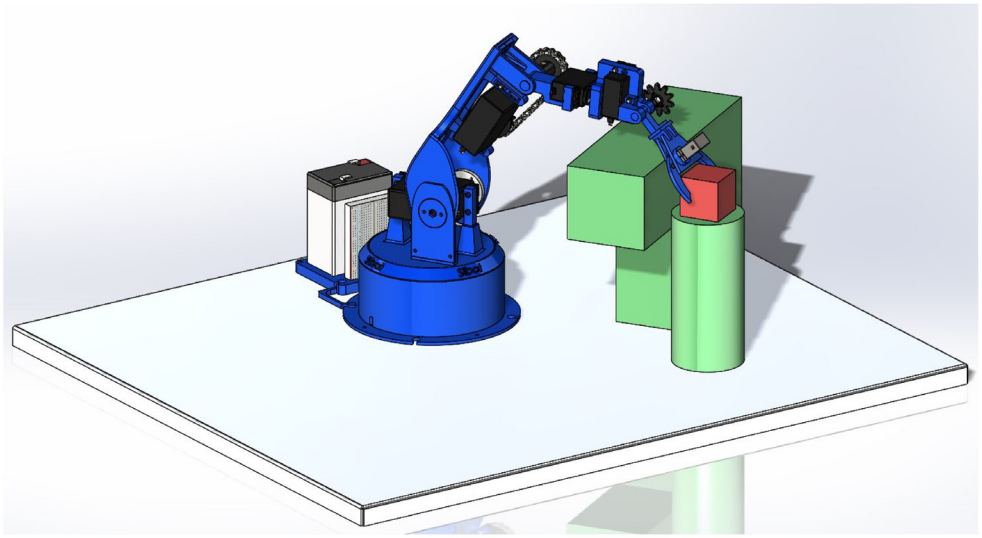


Fig. 22. Case 5: Optimal orientations obtained from CPA, with the position of the targeted object (red) at (−150, 250, 170).

as failed to solve the case study. Tables 23 to 27 present the comparative performance of each case study, while the convergence curves of these five optimizers are depicted in Figs. 23 to 27.

For the obstacle-free environment (see Tables 23 and 24), it can be seen that SSA and CPA obtain higher accuracy than others, but the CPA has a much lower standard deviation value, indicating its higher stability than SSA. Besides, the results reveal the superior performance of CPA in comparison with others in terms of NOFE and processing time. The competitive algorithms require almost two times longer of processing time than that of CPA, yet not getting to converge to global optimal solution, especially for FPA, DE, and CSA. The better convergence characteristic of CPA can be seen in Figs. 23 and 24.

In the environment with obstacle, both SSA and CPA are performing equally in Case 3, where the obtained position accuracy and stability are similar (see Table 25). The CPA is still can be deemed as a better method than SSA in this case, since the NOFE and processing time needed by CPA in solving this problem is the least. For Case 4 and Case 5, again, in comparison to others, the CPA can reach the targeted position with the minimum error of approximately zero and with the highest stability. Moreover, faster convergence of CPA can be observed in Figs. 25 to 27, where it improves over the SSA in terms of the best fitness value by 67.47%, 74.86% and 67.20% in Case 3, Case 4 and Case 5, respectively.

Lastly, the optimal orientations obtained by the CPA for each case study are illustrated in Figs. 18 to 22. Although the position of the targeted object is the same for Case 1 and Case 3, the optimal orientations obtained by the CPA for these two cases are different due to the existence of the obstacle. The middle arm of the robotic is higher in Case 3 than that in Case 1 in order to avoid collision with the obstacle. Besides, both targeted positions for Case 2 (without obstacle) and Case 4 (with obstacle) are the same. However, the CPA is still able to lower the middle arm of the robotic in Case 4 so that the collision with the obstacle can be avoided (see Fig. 21). It thus can be concluded that optimizing the orientation angles of a robotic arm by using CPA gives promising result.

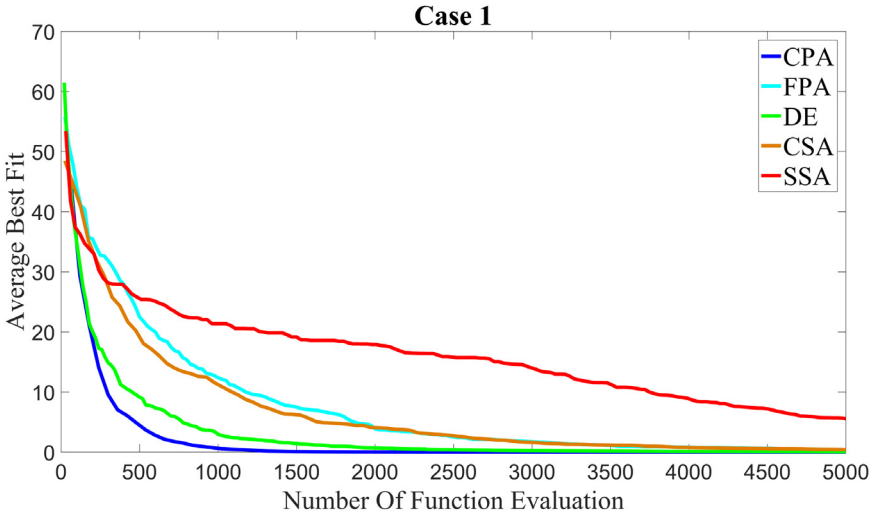


Fig. 23. Convergence curve of CPA and other competing algorithm for Case 1.

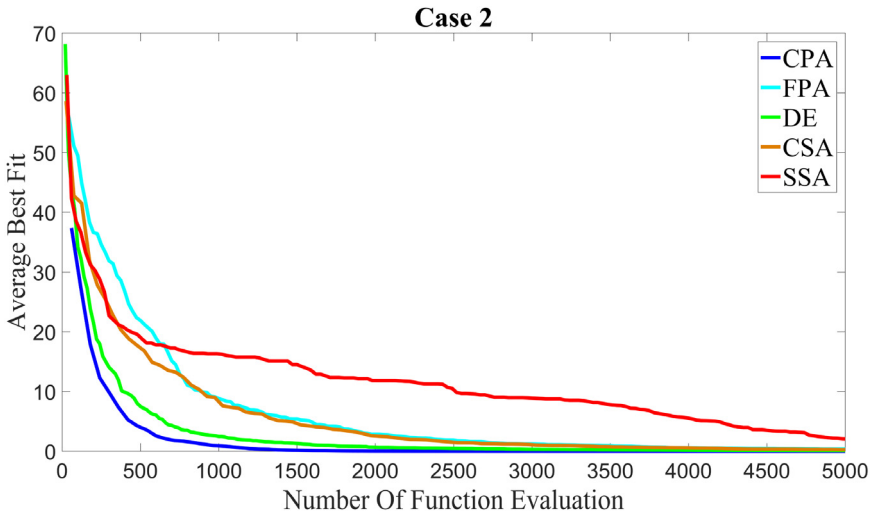


Fig. 24. Convergence curve of CPA and other competing algorithm for Case 2.

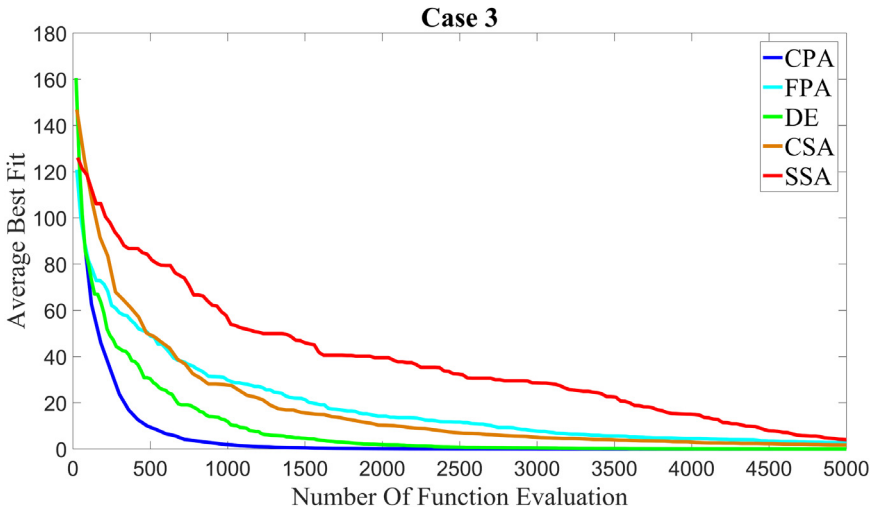


Fig. 25. Convergence curve of CPA and other competing algorithm for Case 3.

Table 23
Comparison of best result and statistical results attained by different optimizers for Case 1.

	Algorithm				
	FPA	DE	CSA	SSA	CPA
θ_1	41.967301	41.989793	41.978249	42.025052	42.016650
θ_2	41.584633	25.549602	45.102687	21.367426	36.637998
θ_3	138.109435	172.041637	129.869317	179.608836	149.246712
θ_4	90.072887	89.990876	90.038429	89.869141	89.896930
θ_5	158.928524	167.658040	154.105024	166.554926	163.866726
Best	0.013227	0.000978	0.007865	0.00000807	0.00000290
Mean	0.048125	0.009200	0.045736	3.342388	0.00000776
Worst	0.134062	0.025957	0.097267	20.054206	0.00000999
SD	0.027453	0.006744	0.024618	7.601518	1.847842×10^{-6}
Processing Time (s)	1.463018	1.598340	1.450890	1.460653	0.763963
NOFE	10,000	10,000	9975	9900	4440

Table 24
Comparison of best result and statistical results attained by different optimizers for Case 2.

	Algorithm				
	FPA	DE	CSA	SSA	CPA
θ_1	63.783152	63.783695	63.755813	63.781372	63.769508
θ_2	49.148511	51.510206	37.282510	51.609278	32.768944
θ_3	109.370232	98.964266	140.561673	98.256731	150.119411
θ_4	90.022231	90.106371	90.152337	90.182169	90.087319
θ_5	114.240068	99.506760	139.084681	98.288591	142.461065
Best	0.020909	0.000723	0.011021	0.00000706	0.00000361
Mean	0.088504	0.018958	0.051684	0.170058	0.00000707
Worst	0.204621	0.111774	0.158131	2.557776	0.00000987
SD	0.045291	0.023138	0.034744	0.505144	1.906424×10^{-6}
Processing Time (s)	1.774884	1.667708	1.450361	1.399536	0.884384
NOFE	10,000	10,000	9975	9840	5160

Table 25
Comparison of best result and statistical results attained by different optimizers for Case 3.

	Algorithm				
	FPA	DE	CSA	SSA	CPA
θ_1	41.949023	41.937236	42.047491	41.971171	41.938620
θ_2	48.366810	48.198418	49.503339	46.952457	47.841992
θ_3	121.934838	122.347954	119.087698	125.409692	123.228942
θ_4	90.145319	90.196517	89.752439	90.061653	90.189732
θ_5	148.495553	148.809015	146.246679	151.069007	149.474113
Best	0.011538	0.0000477	0.009134	0.00000661	0.00000215
Mean	0.097907	0.0005193	0.050126	0.00002681	0.00000725
Worst	0.848937	0.0019668	0.196352	0.00006044	0.00000999
SD	0.161021	0.0004655	0.049152	1.317181×10^{-5}	1.824249×10^{-6}
Processing Time (s)	1.482664	1.532912	1.661112	1.397121	0.830824
NOFE	10,000	10,000	9975	9960	4680

Table 26
Comparison of best result and statistical results attained by different optimizers for Case 4.

	Algorithm				
	FPA	DE	CSA	SSA	CPA
θ_1	63.793961	63.788544	63.831701	63.788604	63.801316
θ_2	4.267207	4.254785	22.143285	4.170738	15.313545
θ_3	177.948074	177.825088	169.041900	176.141634	177.944301
θ_4	89.818020	90.004039	89.796983	90.011863	89.934690
θ_5	94.620417	94.312936	141.794057	90.322606	134.759517
Best	0.012660	0.000117	0.017916	0.0000148	0.00000372
Mean	0.065332	0.004126	0.084048	0.0359885	0.00000713
Worst	0.182419	0.026795	0.309981	0.9070697	0.00000943
SD	0.037753	0.005233	0.068192	0.165440	1.746460×10^{-6}
Processing Time (s)	1.496524	1.543446	1.493558	1.916551	0.861789
NOFE	10,000	10,000	9975	10,020	4860

Table 27
Comparison of best result and statistical results attained by different optimizers for Case 5.

	Algorithm				
	FPA	DE	CSA	SSA	CPA
θ_1	59.107228	59.050613	59.002187	59.005387	59.052499
θ_2	99.843258	105.644210	83.740118	25.234230	98.849312
θ_3	78.239089	60.303878	116.037757	155.223803	80.816269
θ_4	89.752172	89.897444	90.087371	89.910010	89.941719
θ_5	141.172332	115.353674	179.660981	1.341071	144.406894
Best	0.025818	0.004578	0.022241	0.00000558	0.00000183
Mean	0.163438	0.023832	0.116158	1.52174480	0.00000695
Worst	0.637597	0.084975	0.272477	21.47633116	0.00000983
SD	0.132328	0.018485	0.067008	5.424815	2.087758×10^{-6}
Processing Time (s)	1.518849	1.517456	1.493249	1.407713	0.722806
NOFE	10,000	10,000	9975	9960	4080

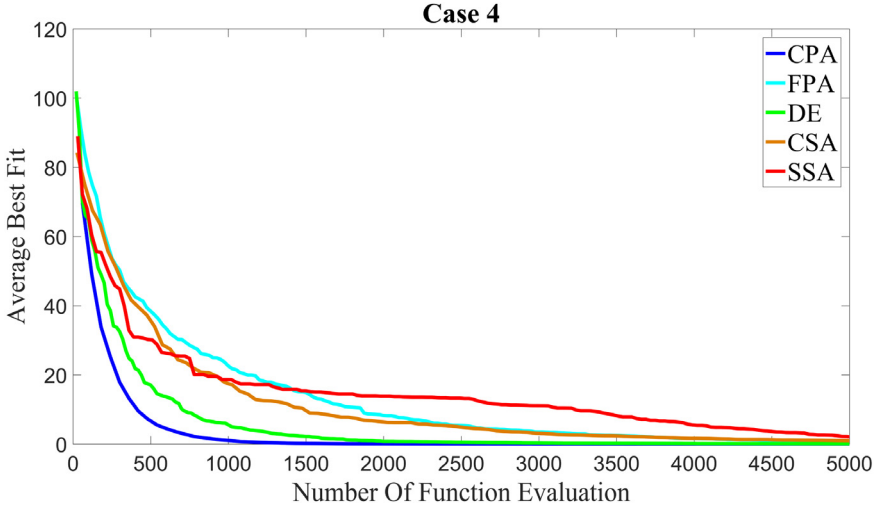


Fig. 26. Convergence curve of CPA and other competing algorithm for Case 4.

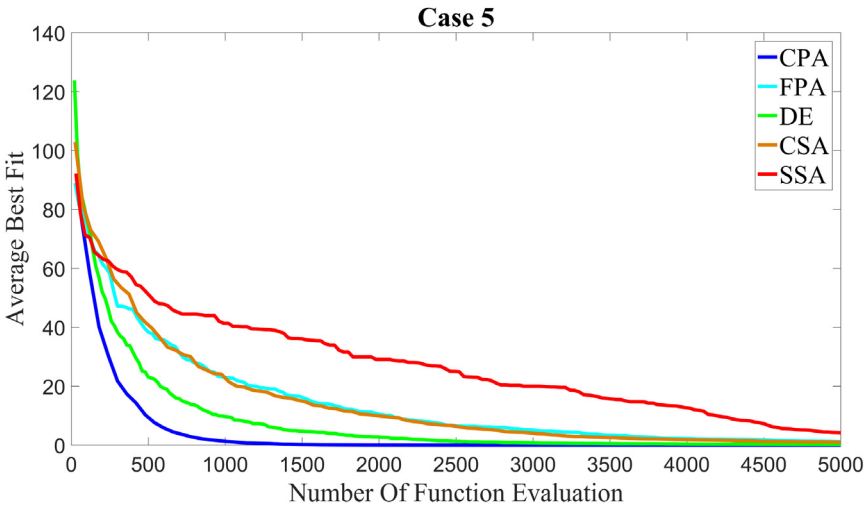


Fig. 27. Convergence curve of CPA and other competing algorithm for Case 5.

4. Conclusion

A novel plant-inspired population-based metaheuristic algorithm, namely, CPA, is proposed in this study for global optimization problems. The exploration and exploitation of CPA are inspired by the growth process and reproduction process of the carnivorous plant. To validate the robustness of CPA, several experiments are conducted. Firstly, thirty classical benchmark test functions are employed to test the exploration and exploitation of the CPA. Secondly, seven CEC2017 test functions are selected to evaluate whether there is a structural bias in the proposed CPA. Both first and second experiments are also used to assess the convergence rate of the CPA. Thirdly, the convergence rate of the CPA is observed and the superiority of CPA as compared with other competitors is assessed through Wilcoxon signed-rank test. In addition, the time complexity of the CPA is established using Big O notation. Seven mechanical engineering design problems and four CEC2011 problems are utilized to test the capability of CPA in application, where a set of constraints are involved, prior to the implementation of CPA to solve a real-world application, *i.e.,* controlling the orientation of a 5-DOF robotic arm so that the end-effector can reach the targeted position precisely without colliding with its surrounding obstacle.

In classical benchmark test functions, the unimodal and multimodal functions are utilized to test the exploitation and exploration of the CPA. Excessive exploration of an algorithm will consume high computational force, and hence, high NOFE is required to solve a function. Meanwhile, too much exploitation will cause the algorithm to become stagnant in the later phase as the algorithm cannot escape from the local minima. From the results, the CPA has outperformed almost all of its competitors by converging to the global optimum with less NOFE required. Some of its competitors, on the other hand, have gone stagnant after a few iterations. Thus, this has proven that the exploration and exploitation of CPA are well-balanced. For the simulation on CEC2017 test functions, it can be concluded that there is no structural bias exists in the CPA, since it still can converge to the global optimum although the global optimum of each dimension is shifted to a different value. The superior performance of the CPA in comparison to the competitive algorithms has been validated by the Wilcoxon signed-rank test. In the fourth experiment, it has been shown that the computational complexity of the CPA is linearly proportional to the problem size. The results in the fifth experiment showed that the CPA is also capable of dealing with constrained problems in addition to the unconstrained problems. Lastly, the last experiment confirmed the effectiveness of the CPA in solving the real-world challenging problem. The results have shown that the CPA can optimize the orientation of the 5-DOF robotic arm with minimum NOFE and processing time. The results also showed that the accuracy and stability of the CPA are higher than its competitors, in which the position error and SD obtained from CPA is below 10^{-5}. Hence, as a new optimization algorithm, the supremacy of the CPA to address the challenges concerning of high-dimensional design variables, existence of various constraints and search space with many local optima, was shown.

To date, theoretical convergence analysis of metaheuristic algorithms is still at early stage. Their convergence is primarily investigated experimentally in the literature and thus, mathematic analysis concerning the convergence of the CPA will be an interesting topic to pursue in future study.

CRediT authorship contribution statement

Ong Kok Meng: Conceptualization, Investigation, Methodology, Software, Draft preparation. **Ong Pauline:** Supervision, Writing - review & editing. **Sia Chee Kiong:** Supervision.

Declaration of competing interest

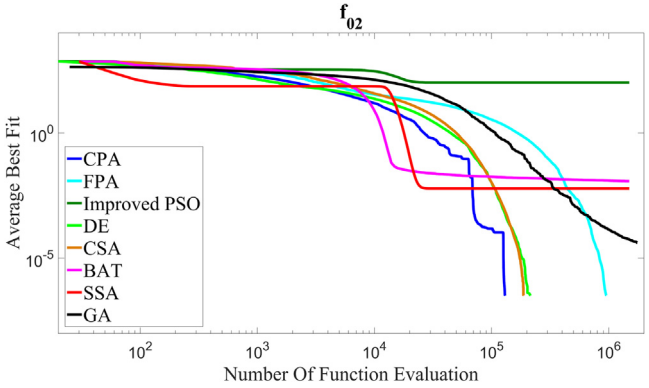
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

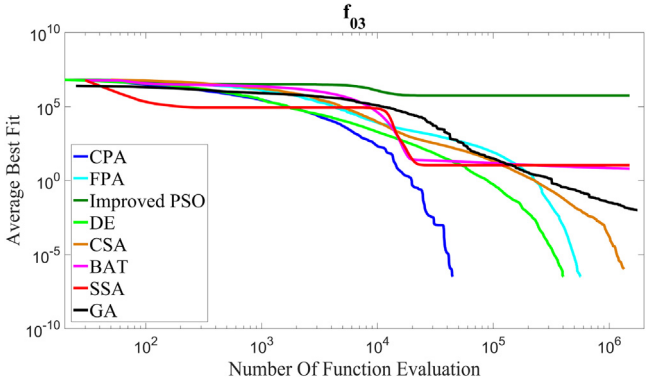
The authors would like to express the deepest appreciation to the Ministry of Higher Education Malaysia (MOHE), for funding this project through the Fundamental Research Grant Scheme (FRGS/1/2018/ICT02/UTHM/02/2 Vot K070). Additional support from Universiti Tun Hussein Onn Malaysia (UTHM) in the form of GPPS Vot U806 is also gratefully acknowledged.

Appendix A

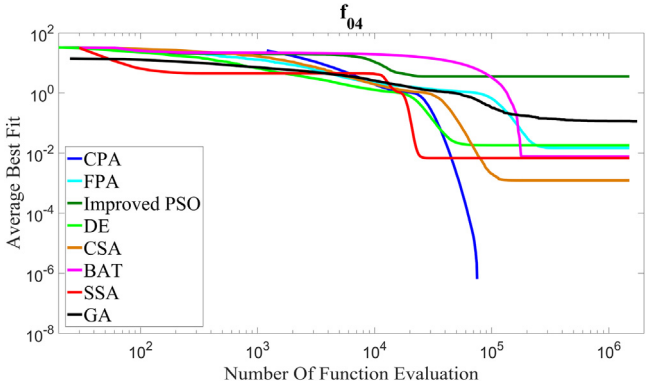
A.1. Convergence curve of CPA and other competitive algorithms for De Jong test function ($d = 100$)



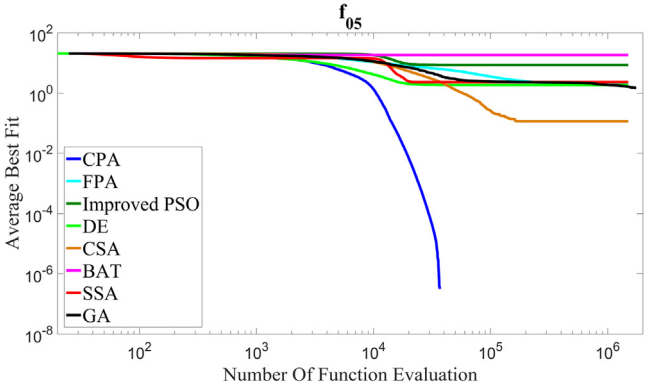
A.2. Convergence curve of CPA and other competitive algorithms for Dixon–Price test function ($d = 50$)



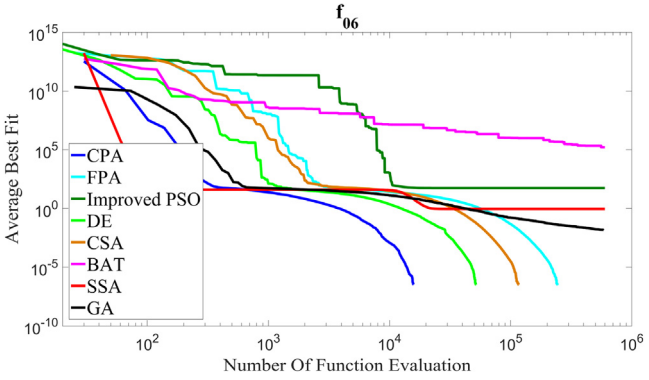
A.3. Convergence curve of CPA and other competitive algorithms for Griewank test function ($d = 50$)



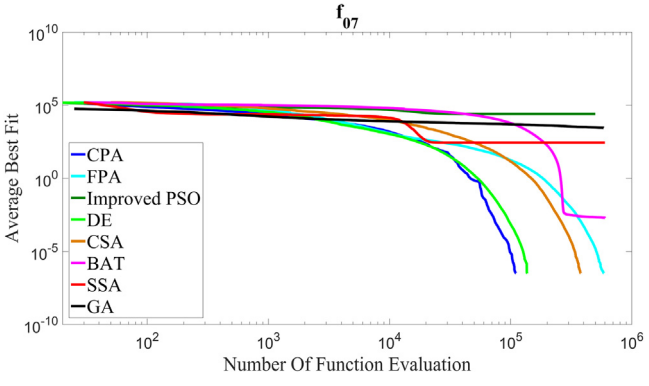
A.4. Convergence curve of CPA and other competitive algorithms for Ackley test function ($d = 30$)



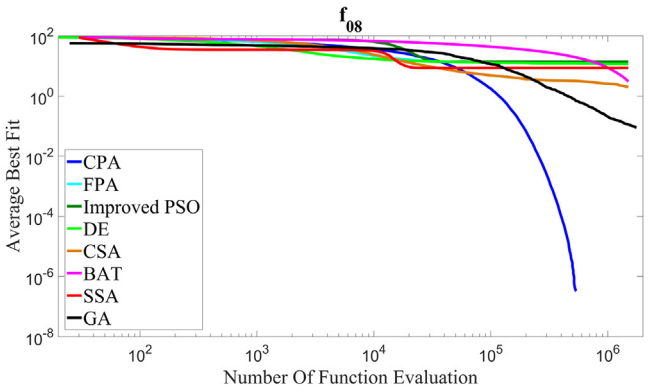
A.5. Convergence curve of CPA and other competitive algorithms for Schwefel 2.22 test function ($d = 30$)



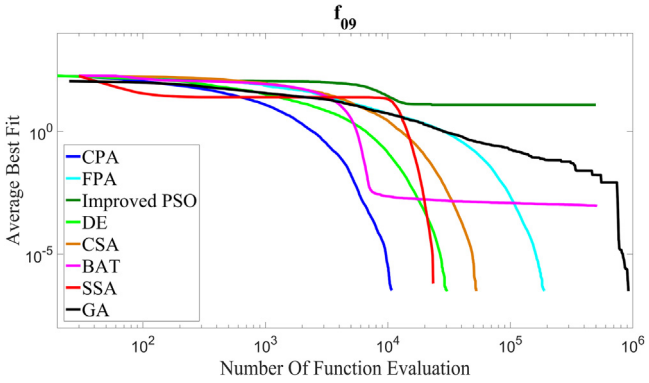
A.6. Convergence curve of CPA and other competitive algorithms for Schwefel 1.2 test function ($d = 30$)



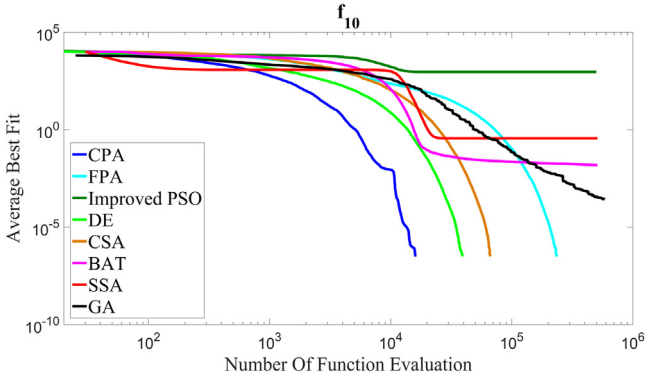
A.7. Convergence curve of CPA and other competitive algorithms for Schwefel 2.21 test function (d = 30)



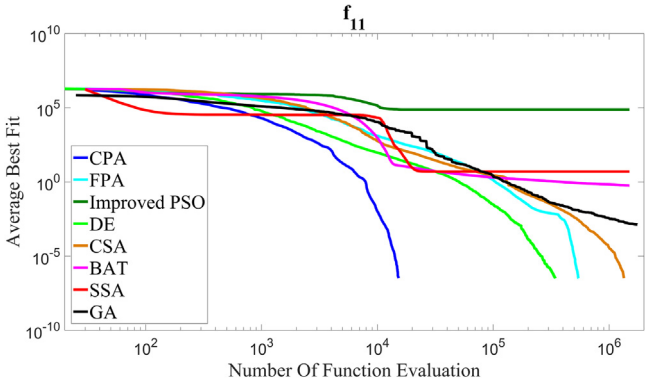
A.8. Convergence curve of CPA and other competitive algorithms for Step test function (d = 30)



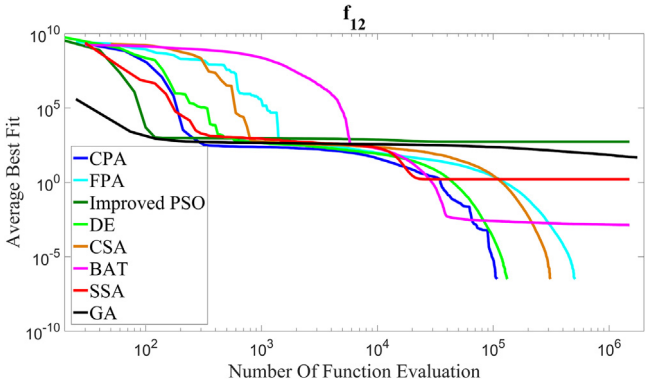
A.9. Convergence curve of CPA and other competitive algorithms for Sum Squares test function (d = 30)



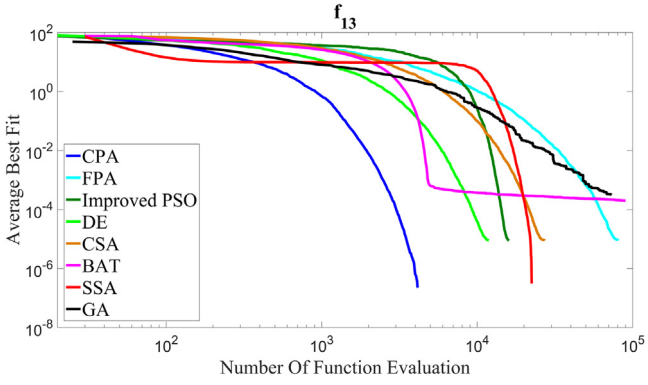
A.10. Convergence curve of CPA and other competitive algorithms for Dixon–Price test function (d = 30)



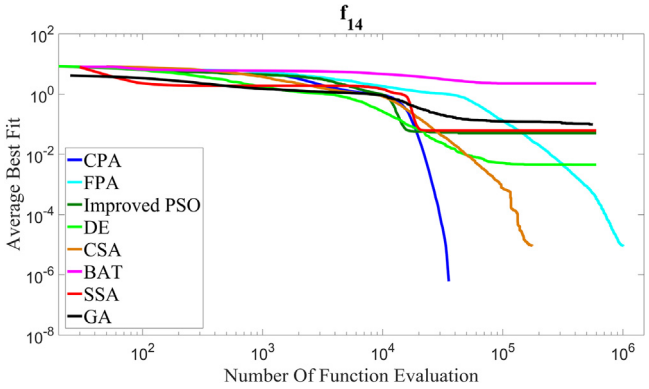
A.11. Convergence curve of CPA and other competitive algorithms for Zakharov test function (d = 30)



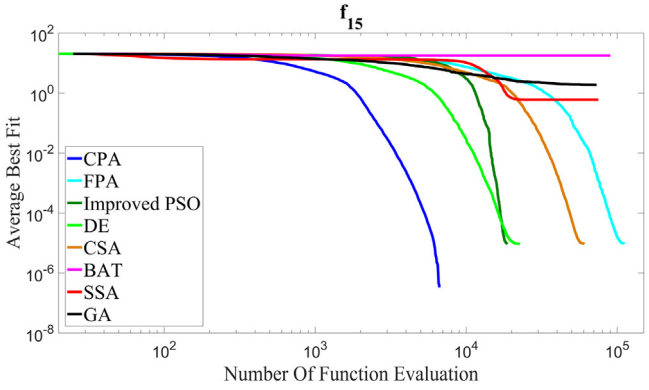
A.12. Convergence curve of CPA and other competitive algorithms for De Jong test function (d = 15)



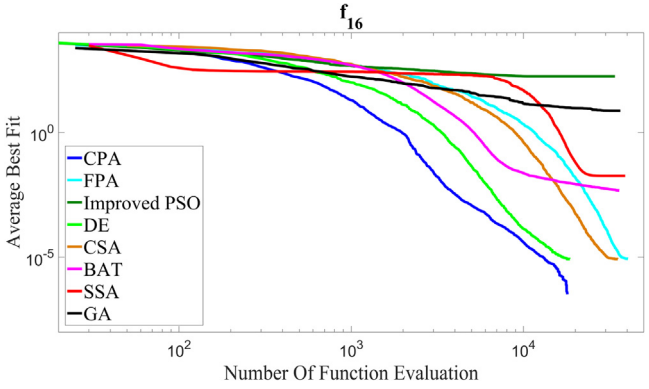
A.13. Convergence curve of CPA and other competitive algorithms for Griewank test function (d = 15)



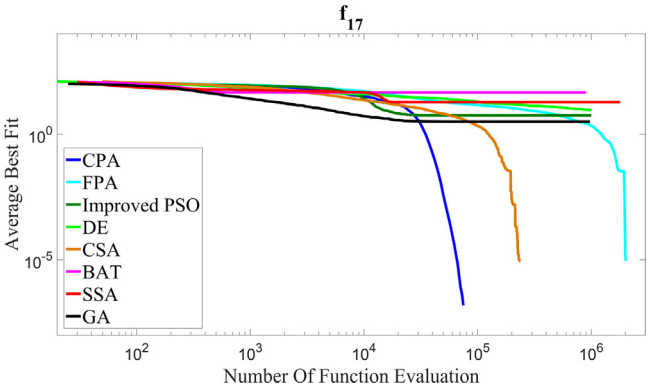
A.14. Convergence curve of CPA and other competitive algorithms for Ackley test function (d = 12)



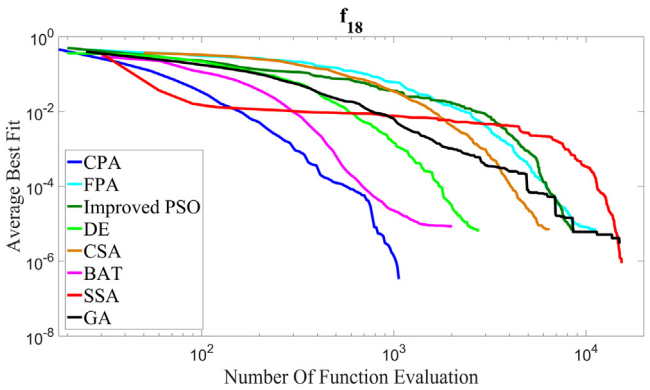
A.15. Convergence curve of CPA and other competitive algorithms for Powell test function (d = 12)



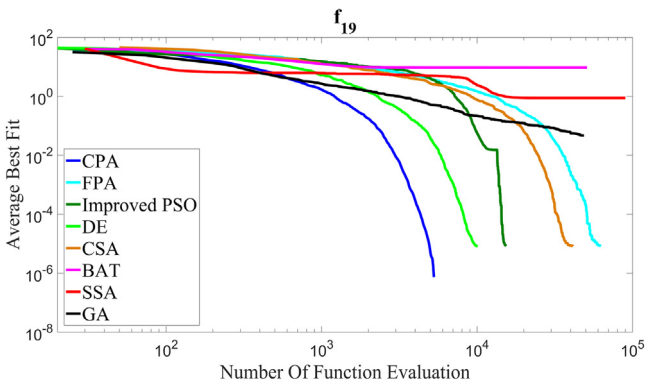
A.16. Convergence curve of CPA and other competitive algorithms for Rastrigin test function (d = 10)



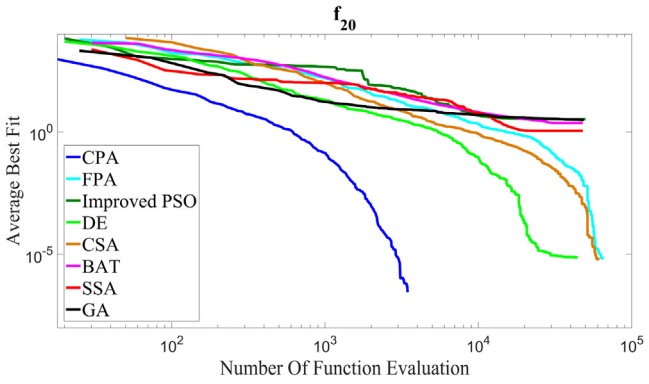
A.17. Convergence curve of CPA and other competitive algorithms for Sum of Different Power test function (d = 10)



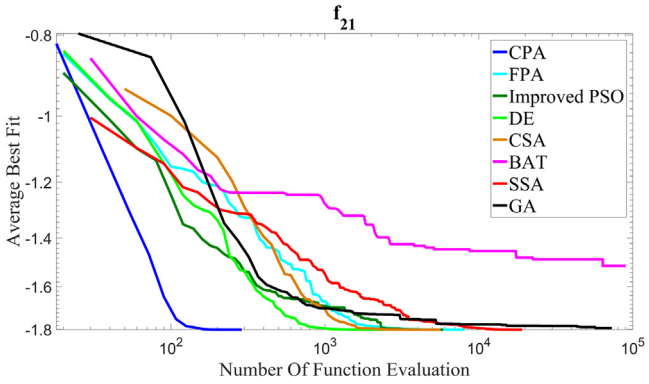
A.18. Convergence curve of CPA and other competitive algorithms for Levy test function (d = 10)



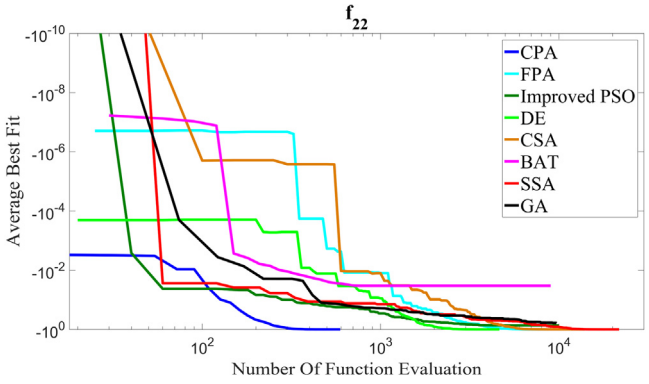
A.19. Convergence curve of CPA and other competitive algorithms for Colville test function (d = 4)



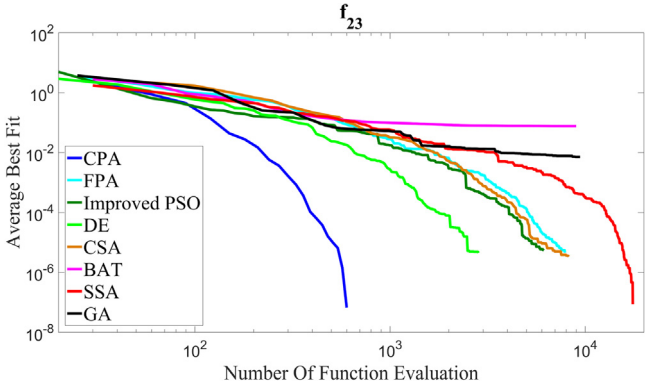
A.20. Convergence curve of CPA and other competitive algorithms for Michaelwicz test function (d = 2)



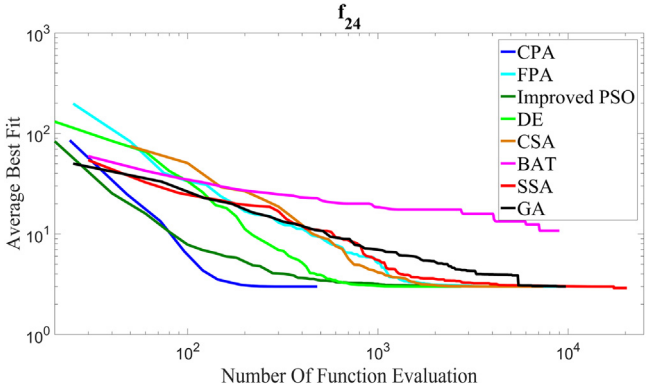
A.21. Convergence curve of CPA and other competitive algorithms for Easom test function (d = 2)



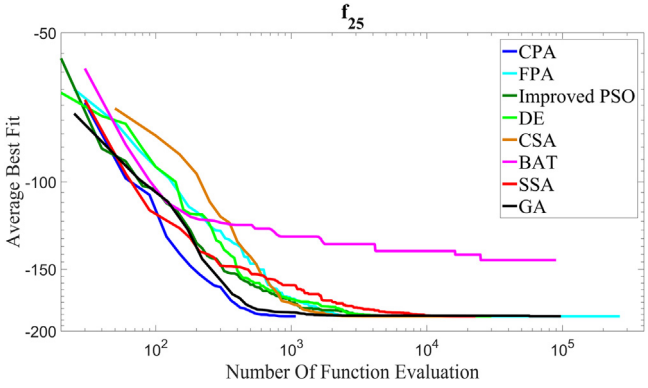
A.22. Convergence curve of CPA and other competitive algorithms for Beale test function (d = 2)



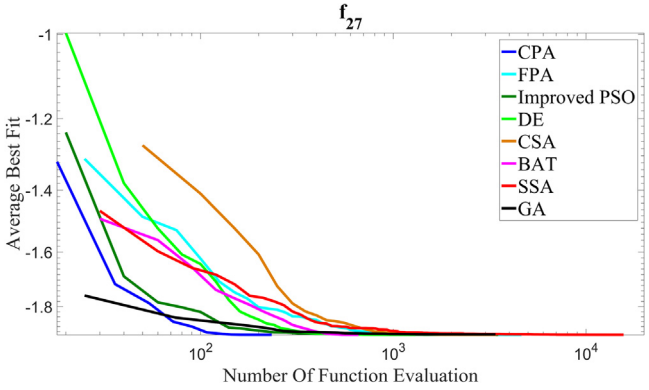
A.23. Convergence curve of CPA and other competitive algorithms for Goldstein–Price test function (d = 2)



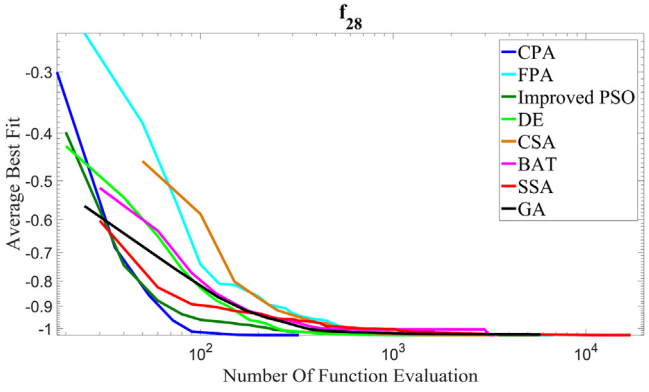
A.24. Convergence curve of CPA and other competitive algorithms for Shubert test function (d = 2)



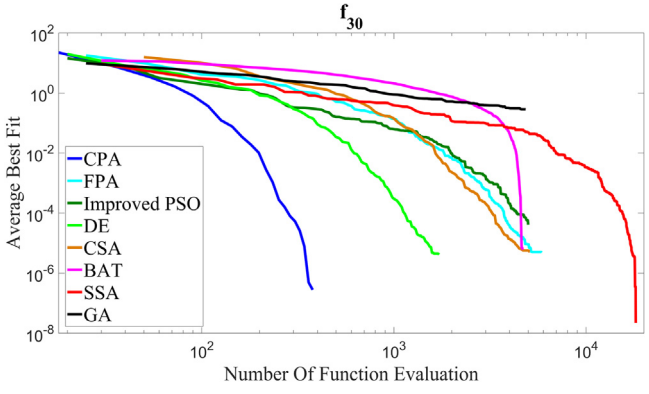
A.25. Convergence curve of CPA and other competitive algorithms for McCormick test function ($d = 2$)



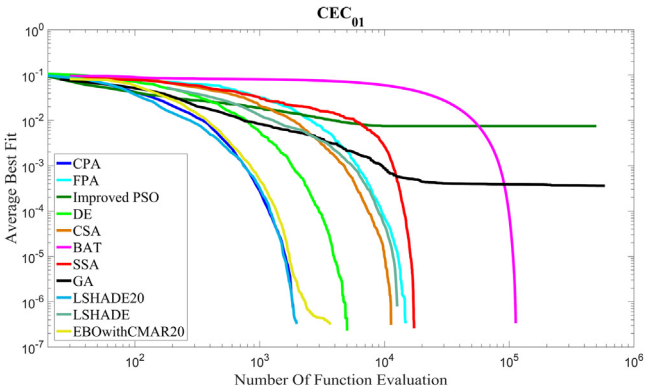
A.26. Convergence curve of CPA and other competitive algorithms for Six-Hump Camel test function ($d = 2$)



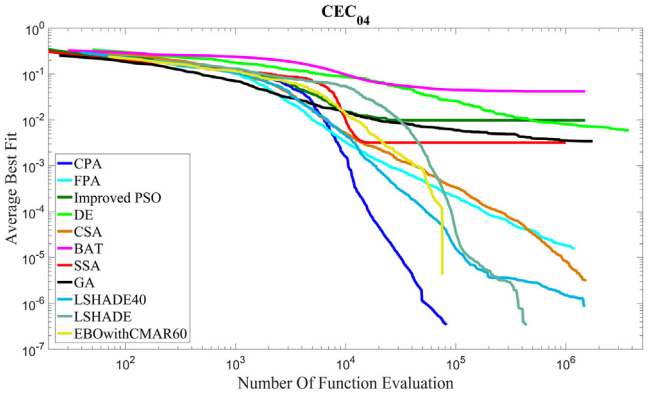
A.27. Convergence curve of CPA and other competitive algorithms for Booth test function ($d = 2$)



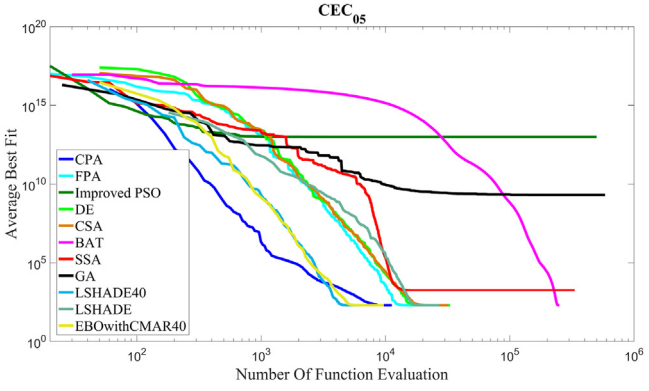
A.28. Convergence curve of CPA and other competitive algorithms for Shifted and Rotated Griewank test function



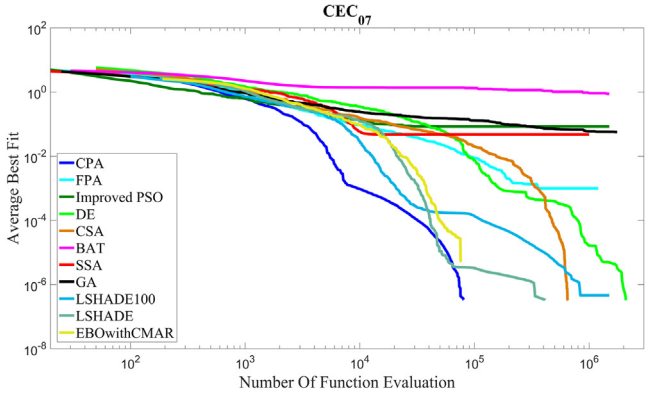
A.29. Convergence curve of CPA and other competitive algorithms for Shifted and Rotated Schaffer F7 test function



A.30. Convergence curve of CPA and other competitive algorithms for Shifted and Rotated Sum of Different Power test function



A.31. Convergence curve of CPA and other competitive algorithms for Shifted and Rotated Expanded Griewank's plus Rosenbrock test function



Appendix B

B.1. Welded beam design problem

$$\text{Minimize} f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2)$$

Subject to:

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0$$
$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0$$
$$g_3(\vec{x}) = x_1 - x_4 \le 0$$
$$g_4(\vec{x}) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0$$
$$g_5(\vec{x}) = 0.125 - x_1 \le 0$$
$$g_6(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0$$
$$g_7(\vec{x}) = P - P_c(\vec{x})$$

where

$$\tau(\vec{x}) = \sqrt{t_1^2 + \frac{t_1 t_2 x_2}{R} + t_2^2}$$
$$t_1 = \frac{P}{\sqrt{2}x_1x_2}, t_2 = \frac{MR}{J}$$
$$M = P\left(L + \frac{x_2}{2}\right), R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$
$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$
$$\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}$$
$$\delta(\vec{x}) = \frac{4PL^3}{Ex_4x_3^3}$$
$$P_c(x) = \frac{4.013E}{L^2}\left(\sqrt{\frac{x_3^2x_4^6}{36}}\right)\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$P = 6000lb, L = 14in, E = 30 \times 10^6 psi, G = 12 \times 10^6 psi$$

$$\tau_{\max} = 13600psi, \sigma_{\max} = 30000psi, \delta_{\max} = 0.25in$$

The range of design variables were $0.1 \le x_1 \le 2$, $0.1 \le x_2 \le 10$, $0.1 \le x_3 \le 10$ and $0.1 \le x_4 \le 2$.

B.2. Multiple disk clutch brake design problem

$$\text{Minimize} f(R_i, R_o, t, F, Z) = \pi(R_o^2 - R_i^2)(t)(Z + 1)(0.0000078)$$

Subject to:

$$g_1 = R_o - R_i - \Delta r \ge 0$$
$$g_2 = l_{\max} - (Z + 1)(t + 0.5) \ge 0$$
$$g_3 = P_{\max} - P_{rz} \ge 0$$
$$g_4 = P_{\max}(v_{sr\,\max}) - P_{rz}(v_{rz}) \ge 0$$
$$g_5 = v_{sr\,\max} - v_{rz} \ge 0$$
$$g_6 = T_{\max} - T \ge 0$$
$$g_7 = M_h - s(M_s) \ge 0$$
$$g_8 = T \ge 0$$

where

$$M_h = \frac{2}{3000}(\mu FZ)\left(\frac{R_o^3 - R_i^3}{R_o^2 - R_i^2}\right)$$
$$P_{rz} = \frac{F}{\pi\left(R_o^2 - R_i^2\right)}$$
$$v_{rz} = \frac{2\pi n(R_o^3 - R_i^3)}{90000(R_o^2 - R_i^2)}$$
$$T = \frac{I_z\pi n}{30(M_h - M_f)}$$

$\Delta r = 20$ mm, $I_z = 55$ kgmm^2, $P_{\max} = 1$ MPa, $T_{\max} = 15$ s,

$\mu = 0.5$, $s = 1.5$, $M_s = 40$ Nm, $M_f = 3$ Nm, $n = 250$ rpm,

$$v_{sr\,\max} = 10 \text{ m/s}, l_{\max} = 30 \text{ mm}$$

The range of design variables are $60 \le R_i \le 80$, $90 \le R_o \le 110$, $1 \le t \le 3$, $600 \le F \le 1000$ and $2 \le Z \le 9$.

B.3. Rolling bearing design problem

$$\text{Maximize } C_d = \begin{cases} f_cZ^{2/3}D_b^{1.8} & D_b \le 25.4 \text{ mm} \\ 3.647f_cZ^{2/3}D_b & D_b \ge 25.4 \text{ mm} \end{cases}$$

Subject to:

$$g_1 = \frac{\phi_o}{2\sin^{-1}(D_b/D_m)} - Z + 1 \ge 0$$
$$g_2 = 2D_b - K_{D\min}(D - d) \ge 0$$
$$g_3 = K_{D\max}(D - d) - 2D_b \ge 0$$
$$g_4 = \xi B_w - D_b \le 0$$
$$g_5 = D_m - 0.5(D + d) \ge 0$$
$$g_6 = (0.5 + e)(D + d) - D_m \ge 0$$
$$g_7 = 0.5(D - D_m - D_b) - \varepsilon D_b \ge 0$$
$$g_8 = f_i \ge 0.515$$
$$g_9 = f_o \ge 0.515$$

where

$$f_c = 37.91\left[1 + \left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i(2f_o - 1)}{f_o(2f_i - 1)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3} \times \left[\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{(1+\gamma)^{1/3}}\right]\left[\frac{2f_i}{2f_i - 1}\right]^{0.41}$$

$$\phi_o = 2\pi - 2\cos^{-1}\left(\frac{\left(\frac{D-d}{2} - \frac{3T}{4}\right)^2 + \left(\frac{D}{2} - \frac{T}{4} - D_b\right)^2 - \left(\frac{d}{2} + \frac{T}{4}\right)^2}{2\left(\frac{D-d}{2} - \frac{3T}{4}\right)\left(\frac{D}{2} - \frac{T}{4} - D_b\right)}\right)$$

$$T = D - d - 2D_b$$

$$f_i = \frac{r_i}{D_b}, f_o = \frac{r_o}{D_b}, \gamma = \frac{D_b}{D_m}\cos(\alpha)$$

$$D = 160, d = 90, B_w = 30, r_i = r_o = 11.033, \alpha = 0$$

The range of design variables rare $0.5(D+d) \le D_m \le 0.6(D+d)$, $0.15(D-d) \le D_b \le 0.45(D-d)$, $4 \le Z \le 50$, $0.515 \le f_i, f_o \le 0.6$, $0.4 \le K_{D\min} \le 0.5$, $0.6 \le K_{D\max} \le 0.7$, $0.3 \le \varepsilon \le 0.4$, $0.02 \le e \le 0.10$ and $0.6 \le \zeta \le 0.85$.

B.4. Heat exchange design problem

$$\text{Minimize} f\left(\overrightarrow{x}\right) = x_1 + x_2 + x_3$$

Subject to:

$$g_1 = 0.0025\,(x_4 + x_6) - 1 \le 0$$
$$g_2 = 0.0025\,(x_5 + x_7 - x_4) - 1 \le 0$$
$$g_3 = 1 - 0.01\,(x_8 - x_5) \ge 0$$
$$g_4 = x_1x_6 - 833.33252x_4 - 100x_1 + 83333.333 \ge 0$$
$$g_5 = x_2x_7 - 1250x_5 - x_2x_4 + 1250x_4 \ge 0$$
$$g_6 = x_3x_8 - x_3x_5 + 2500x_5 - 1250000 \ge 0$$

where $100 \le x_1 \le 10000$, $1000 \le x_2, x_3 \le 10000$ and $10 \le x_i \le 1000, i = 4, 5, 6, \ldots, 8$.

B.5. Speed reducer design problem

$$\text{Minimize f(b, m, z, } l_1, l_2, d_1, d_2)$$
$$= 0.7854bm^2(3.3333z^2 + 14.9334z - 43.0934)$$
$$-1.508b(d_1^2 + d_2^2) + 7.477(d_1^3 + d_2^3)$$
$$+0.7854(l_1d_1^2 + l_2d_2^2)$$

Subject to:

$$g_1 = \frac{27}{bm^2z} - 1 \le 0$$
$$g_2 = \frac{397.5}{bm^2z^2} - 1 \le 0$$
$$g_3 = \frac{1.93l_1^3}{mzd_1^4} - 1 \le 0$$
$$g_4 = \frac{1.93l_2^3}{mzd_2^4} - 1 \le 0$$
$$g_5 = \frac{1}{110d_1^3}\sqrt{\left(\frac{745l_1}{mz}\right)^2 + 16.9 \times 10^6} - 1 \le 0$$
$$g_6 = \frac{1}{85d_2^3}\sqrt{\left(\frac{745l_2}{mz}\right)^2 + 157.5 \times 10^6} - 1 \le 0$$
$$g_7 = \frac{mz}{40} - 1 \le 0$$
$$g_8 = \frac{5m}{b} - 1 \le 0$$
$$g_9 = \frac{b}{12m} - 1 \le 0$$
$$g_{10} = \frac{1.5d_1 + 1.9}{l_1} - 1 \le 0$$
$$g_{11} = \frac{1.1d_2 + 1.9}{l_2} - 1 \le 0$$

Table B.1 lists down the range of each design variable of this design problem.

Table B.1
The range of each design variable of speed reducer design problem.

Parameter	Range
b	2.6-3.6
m	0.7-0.8
z	17-28
l_1	7.3-8.3
l_2	7.8-8.3
d_1	2.9-3.9
d_2	5.0-5.5

B.6. Pressure vessel design problem

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

Subject to:

$$g_1(\overrightarrow{x}) = -x_1 + 0.0193x_3 \le 0$$
$$g_2(\overrightarrow{x}) = -x_2 + 0.00954x_3 \le 0$$
$$g_3(\overrightarrow{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0$$
$$g_4(\overrightarrow{x}) = x_4 - 240 \le 0$$

The range of each design variable were $1 \times 0.0625 \le x_1, x_2 \le 99 \times 0.0625$ and $10.0 \le x_3, x_4 \le 200.0$.

B.7. Car side impact design

$$\text{Minimize} f\left(\overrightarrow{x}\right) = 1.98 + 4.90x_1 + 6.67x_2 + 6.98x_3$$
$$+ 4.01x_4 + 1.78x_5 + 2.73x_7$$

Subject to:

$$g_1\left(\overrightarrow{x}\right) = 1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10}$$
$$- 0.484x_3x_9 + 0.01343x_6x_{10} - 1 \le 0$$

$$g_2\left(\overrightarrow{x}\right) = 0.261 - 0.0159x_1x_2 - 0.188x_1x_8$$
$$-0.019x_2x_7 + 0.0144x_3x_5$$
$$+0.0008757x_5x_{10} + 0.08045x_6x_9 + 0.00139x_8x_{11}$$
$$+0.00001575x_{10}x_{11}$$
$$-0.32 \le 0$$

$$g_3\left(\overrightarrow{x}\right) = 0.214 + 0.00817x_5 - 0.131x_1x_8$$
$$-0.0704x_1x_9 + 0.03099x_2x_6$$
$$-0.018x_2x_7 + 0.0208x_3x_8 + 0.121x_3x_9 - 0.00364x_5x_6$$
$$+0.0007715x_5x_{10} - 0.0005354x_6x_{10}$$
$$+0.00121x_8x_{11} - 0.32 \le 0$$

$$g_4\left(\overrightarrow{x}\right) = 0.74 - 0.061x_2 - 0.163x_3x_8$$
$$+0.001232x_3x_{10} - 0.166x_7x_9$$
$$+0.227x_2^2 - 0.32 \ge 0$$

Table B.2
Value of c.

2.918487e−003	−8.045787e−003	6.749947e−003	−1.416647e−003
9.509977e+000	−3.500994e+001	4.283329e+001	−1.733333e+001
2.682093e+001	−9.556079e+001	1.130398e+002	−4.429997e+001
2.087241e+002	−7.198052e+002	8.277466e+002	−3.166655e+002
1.350005e+000	−6.850027e+000	1.216671e+001	−6.666689e+000
1.921995e−002	−7.945320e−002	1.105660e−001	−5.033333e−002
1.323596e−001	−4.692550e−001	5.539323e−001	−2.166664e−001
7.339981e+000	−2.527328e+001	2.993329e+001	−1.199999e+001
−3.950534e−001	1.679353e+000	−1.777829e+000	4.974987e−001
−2.504665e−005	1.005854e−002	−1.986696e−002	9.833470e−003

$$g_5\left(\overrightarrow{x}\right) = 28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 - 7.7x_7x_8 + 0.32x_9x_{10} - 32 \le 0$$

$$g_6\left(\overrightarrow{x}\right) = 33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11x_2x_8 - 0.0215x_5x_{10} - 9.98x_7x_8 + 22x_8x_9 - 32 \le 0$$

$$g_7\left(\overrightarrow{x}\right) = 46.36 - 9.9x_2 - 12.9x_1x_8 + 0.1107x_3x_{10} - 32 \le 0$$

$$g_8\left(\overrightarrow{x}\right) = 4.72 - 0.5x_4 - 0.19x_2x_3 - 0.0122x_4x_{10} + 0.009325x_6x_{10} + 0.000191x_{11}^2 - 4 \le 0$$

where the range of design variables are $0.5 \le x_1 - x_7 \le 1.5$, $0.192 \le x_8, x_9 \le 0.345$ and $-30 \le x_{10}, x_{11} \le 30$.

B.8. Parameter estimation for frequency-modulated (FM) sound waves (CEC2011 problem)

$$\text{Minimize} f(X) = \sum_{t=0}^{100} \left(y_{est}(X,t) - y_{target}(X,t)\right)^2$$

with

$$y_{est}(X,t) = a_1 \sin(\omega_1 \cdot t \cdot \theta + a_2 \sin(\omega_2 \cdot t \cdot \theta + a_3 \sin(\omega_3 \cdot t \cdot \theta)))$$

$$y_{target}(X,t) = 1.0 \cdot \sin(5.0 \cdot t \cdot \theta + 1.5 \cdot \sin(4.8 \cdot t \cdot \theta + 2.0 \cdot \sin(4.9 \cdot t \cdot \theta)))$$

where $\theta = 2\pi/100$ and the range of decision parameters are $[-6.4\ 6.35]$.

B.9. The bi-functional catalyst blend optimal control problem (CEC2011 problem)

$$\text{Maximize} I = x_7(2000) \times 10^3$$

with a series of seven differential equations are represented as follows:

$$\dot{x}_1 = -k_1x_1$$
$$\dot{x}_2 = k_1x_1 - (k_2 + k_3)x_2 + k_4x_5$$
$$\dot{x}_3 = k_2x_2$$
$$\dot{x}_4 = -k_6x_4 + k_5x_5$$
$$\dot{x}_5 = -k_3x_2 + k_6x_4 - (k_4 + k_5 + k_8 + k_9)x_5 + k_7x_6 + k_{10}x_7$$
$$\dot{x}_6 = k_8x_5 - k_7x_6$$
$$\dot{x}_7 = k_9x_5 - k_{10}x_7$$

where the mole fractions of the chemical species are $x_i, i = 1, 2, \ldots, 7$, and the rate constants k_i are cubic functions of the catalyst blend u:

$$k_i = c_{i1} + c_{i2}u + c_{i3}u^2 + c_{i4}u^3, i = 1, 2, \ldots, 10$$

The coefficients c_{ij} were obtained through experimental measurements and tabulated in Table B.2 [70]. The initial state vector is $x_0 = [1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]$ and the range of catalyst blend u is $0.60 \le u \le 0.90$.

B.10. Optimal control of a non-linear stirred tank reactor (CEC2011 problem)

$$\text{minimize} I = \int_0^{t_f=0.72} \left(x_1^2 + x_2^2 + 0.1 \cdot u^2\right) dt$$

where the range of u is within [0 5]. Two nonlinear differential equations that model the chemical process are shown as follows:

$$\dot{x}_1 = -(2+u)(x_1 + 0.25) + (x_2 + 0.5)\exp\left(\frac{25x_1}{x_1 + 2}\right)$$

$$\dot{x}_2 = 0.5 - x_2 - (x_2 + 0.5)\exp\left(\frac{25x_1}{x_1 + 2}\right)$$

where x_1 is the deviation from dimensionless steady state temperature and x_2 is the deviation from dimensionless steady state concentration.

B.11. Spread spectrum radar polly phase code design (CEC2011 problem)

$$\text{global minimize}_{x \in X} f(x) = \max\{\phi_1(x), \ldots, \phi_{2m}(x)\},$$
$$X = \left\{(x_1, \ldots, x_n) \in R^n | 0 \le x_j \le 2\pi, j = 1, \ldots, n\right\},$$

where m = 2n−1 and

$$\phi_{2i-1}(x) = \sum_{j=1}^{n} \cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k\right), i = 1, \ldots, n$$

$$\phi_{2i}(x) = 0.5 + \sum_{j=i+1}^{n} \cos\left(\sum_{k=|2i-j|+1}^{j} x_k\right), i = 1, \ldots, n-1$$

$$\phi_{m+i}(x) = -\phi_i(x), i = 1, \ldots, m$$

References

[1] S. Mirjalili, Moth-flame optimization algorithm : A novel nature-inspired heuristic paradigm, Knowl.-Based Syst. 89 (2015) 228–249.
[2] S. Mirjalili, The ant lion optimizer, Adv. Eng. Softw. 83 (2015) 80–98.
[3] S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: Theory and application, Adv. Eng. Softw. 105 (2017) 30–47.
[4] X.S. Yang, Engineering Optimization: An Introduction with Metaheuristic Applications, John Wiley & Sons, Inc., Hoboken, New Jersey, 2010.
[5] S. Shadravan, H.R. Naji, V.K. Bardsiri, The Sailfish Optimizer : A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems, Eng. Appl. Artif. Intell. 80 (2019) 20–34.
[6] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1992.
[7] A. Cheraghalipour, M. Hajiaghaei-Keshteli, M.M. Paydar, Tree Growth Algorithm (TGA): A novel approach for solving optimization problems, Eng. Appl. Artif. Intell. 72 (April) (2018) 393–414.
[8] R. Storn, K. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[9] D. Karaboga, B. Basturk, An artificial bee colony (abc) algorithm for numeric function optimization, in: IEEE Swarm Intelligence Symposium 2006, 2006.
[10] J. Kennedy, R. Eberhart, Particle Swarm Optimization, IEEE Service Center, Piscataway, New Jersey.
[11] X.S. Yang, S. Deb, Cuckoo search via Lévy flights, in: 2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings, 2009, pp. 210–214.
[12] X.S. Yang, A new metaheuristic bat-inspired algorithm, Stud. Comput. Intell. 284 (2010) 65–74.
[13] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm Evol. Comput. (November) (2018) 1–28.
[14] A.R. Mehrabian, C. Lucas, A novel numerical optimization algorithm inspired from weed colonization, Ecol. Inform. 1 (2006) 355–366.
[15] X.-S. Yang, Flower pollination algorithm for global optimization, 2012, arXiv.
[16] Z. Geem, J. Kim, G.V. Loganathan, A new heuristic optimization algorithm: Harmony search, Simulation 76 (2) (2001) 60–68.
[17] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching-learning-based optimization : A novel method for constrained mechanical design optimization problems, Comput. Aided Des. 43 (3) (2011) 303–315.
[18] E. Atashpaz-Gargari, C. Lucas, Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, in: 2007 IEEE Congress on Evolutionary Computation, 2007, pp. 4661–4667.
[19] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA : A gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248.
[20] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm - A novel metaheuristic optimization method for solving constrained engineering optimization problems, Comput. Struct. 110—111 (2012) 151–166.
[21] S. Mirjalili, S.M. Mirjalili, A. Hatamlou, Multi-Verse Optimizer : a nature-inspired algorithm for global optimization, Neural Comput. Appl. 27 (2016) 495–513.
[22] F. Glover, Tabu search — Part I, ORSA J. Comput. 1 (3) (1989) 190–206.
[23] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4589) (1983) 671–680.
[24] K.M. Ong, P. Ong, C.K. Sia, E.S. Low, Effective moving object tracking using modified flower pollination algorithm for visible image sequences under complicated background, Appl. Soft Comput. 83 (2019) 105625.
[25] H. Liu, X.-W. Zhang, L.-P. Tu, A modified particle swarm optimization using adaptive strategy, Expert Syst. Appl. 152 (2020) 113353.
[26] S. Dhargupta, M. Ghosh, S. Mirjalili, R. Sarkar, Selective opposition based grey wolf optimization, Expert Syst. Appl. 151 (2020) 113389.
[27] W. Long, S. Cai, J. Jiao, M. Xu, T. Wu, A new hybrid algorithm based on grey wolf optimizer and cuckoo search for parameter extraction of solar photovoltaic models, Energy Convers. Manage. 203 (2020) 112243.
[28] Y. Zhang, Z. Jin, Y. Chen, Hybrid teaching–learning-based optimization and neural network algorithm for engineering design optimization problems, Knowl.-Based Syst. 187 (2020) 104836.
[29] D.H. Wolpert, W.G. Macready, No free lunch theorems for search, Most (1997) 1–38.
[30] J. Andreas, A. Sciligo, T. Witt, A.M. El-sayed, D.M. Suckling, Pollinator-prey conflict in carnivorous plants, Biol. Rev. 87 (3) (2011) 602–615.
[31] A. Pavlovic, M. Saganova, A novel insight into the cost-benefit model for the evolution of botanical carnivory, Ann. Bot. 115 (7) (2015) 1075–1092.
[32] A. Mithöfer, Carnivorous pitcher plants : Insights in an old topic, Phytochemistry 72 (2011) 1678–1682.
[33] S. McPherson, Carnivorous plants and their habitats, Redfern Nat. Hist. Prod. (2010) 1442.
[34] Salmon, Bruce, Carnivorous Plants of New Zealand, Ecosphere Publications, 2001.
[35] M. Jamil, X. s, Y. Blekinge, A literature survey of benchmark functions for global optimization problems, 2013, pp. 1–47.
[36] J.J. Liang, P.N. Suganthan, K. Deb, Novel composition test functions for numerical global optimization, in: Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005, 2005, pp. 68–75.
[37] F. Caraffini, A.V. Kononova, D. Corne, Infeasibility and structural bias in differential evolution, Inform. Sci. 496 (2019) 161–179.
[38] A.V. Kononova, D.W. Corne, P. De Wilde, V. Shneer, F. Caraffini, Structural bias in population-based algorithms, Inform. Sci. 298 (2015) 468–490.
[39] G. Wu, R. Mallipeddi, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization, 2017.
[40] R. Tanabe, A. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: 2014 IEEE Congress on Evolutionary Computation, CEC, 2014, pp. 1658–1665.
[41] A. Kumar, R.K. Misra, D. Singh, Improving the local search capability of Effective Butterfly Optimizer using Covariance Matrix Adapted Retreat Phase, in: 2017 IEEE Congress on Evolutionary Computation, CEC, 2017, pp. 1835–1842.
[42] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.
[43] S. Bansal, Performance comparison of five metaheuristic nature-inspired algorithms to find near-OGRs for WDM systems, Artif. Intell. Rev. (2020).
[44] H.N. Ghafil, K. Jármai, Dynamic differential annealed optimization: New metaheuristic optimization algorithm for engineering applications, Appl. Soft Comput. 93 (2020) 106392.
[45] P. Civicioglu, Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm, Comput. Geosci. 46 (2012) 229–247.
[46] V. Kumar, D. Kumar, An astrophysics-inspired grey wolf algorithm for numerical optimization and its application to engineering design problems, Adv. Eng. Softw. 112 (2017) 231–254.
[47] D. Karaboga, B. Basturk, Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems, 2015.
[48] N.B. Guedria, Improved accelerated PSO algorithm for mechanical engineering optimization problems, Appl. Soft Comput. J. 40 (2016) 455–467.
[49] M. Jaberipour, E. Khorram, Two improved harmony search algorithms for solving engineering optimization problems, Commun. Nonlinear Sci. Numer. Simul. 15 (11) (2010) 3316–3331.
[50] K.S. Lee, A new meta-heuristic algorithm for continuous engineering optimization : harmony search theory and practice, Comput. Methods Appl. Mech. Engrg. 194 (2005) 3902–3933.
[51] S. Mirjalili, S. Mohammad, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.
[52] S. Mirjalili, SCA : A Sine Cosine Algorithm for solving optimization problems, Knowl.-Based Syst. 96 (2016) 120–133.
[53] A.H. Gandomi, X.S. Yang, A.H. Alavi, Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems, Eng. Comput. 29 (1) (2013) 17–35, http://dx.doi.org/10.1007/s00366-011-0241-y.
[54] E. Sandgren, Nonlinear integer and discrete programming in mechanical design optimization, J. Mech. Des. 112 (2) (1990) 223–229.
[55] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67.
[56] C.A. Coello, Use of a self-adaptive penalty approach for engineering optimization problems, Comput. Ind. 41 (2) (2000) 113–127.
[57] Q. He, L. Wang, An effective co-evolutionary particle swarm optimization for constrained engineering design problems, Eng. Appl. Artif. Intell. 20 (1) (2007) 89–99.
[58] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Appl. Soft Comput. J. 13 (5) (2013) 2592–2612.
[59] B.D. Youn, K.K. Choi, R.J. Yang, L. Gu, Reliability-based design optimization for crashworthiness of vehicle side impact, Struct. Multidiscip. Optim. 26 (3–4) (2004) 272–283.
[60] A.H. Gandomi, X.-S. Yang, Benchmark problems in structural optimisation, in: S. Koziel, X.-S. Yang (Eds.), Computational Optimization, Methods and Algorithms, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 259–281.

[61] J. Huang, L. Gao, X. Li, An effective teaching-learning-based cuckoo search algorithm for parameter optimization problems in structure designing and machining processes, Appl. Soft Comput. J. 36 (2015) 349–356.
[62] P.J. Pawar, R.V. Rao, Parameter optimization of machining processes using teaching–learning-based optimization algorithm, Int. J. Adv. Manuf. Technol. 67 (5–8) (2013) 995–1006.
[63] F. Herrera, M. Lozano, Gradual distributed real-coded genetic algorithms, IEEE Trans. Evol. Comput. 4 (1) (2000) 43–62.
[64] P. Korošec, J. Šilc, The continuous differential Ant-Stigmergy Algorithm applied to dynamic optimization problems, in: 2012 IEEE Congress on Evolutionary Computation, 2012, pp. 1–8.
[65] H.K. Singh, T. Ray, Performance of a hybrid EA-DE-memetic algorithm on CEC 2011 real world optimization problems, in: 2011 IEEE Congress of Evolutionary Computation, CEC, 2011, pp. 1322–1326.
[66] S.M. Elsayed, R.A. Sarker, D.L. Essam, Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems, in: 2011 IEEE Congress of Evolutionary Computation, CEC, 2011, pp. 1041–1048.
[67] A. LaTorre, S. Muelas, J. Peña, Benchmarking a hybrid DE-RHC algorithm on real world problems, in: 2011 IEEE Congress of Evolutionary Computation, CEC, 2011, pp. 1027–1033.
[68] M. Asafuddoula, T. Ray, R. Sarker, An adaptive differential evolution algorithm and its performance on real world optimization problems, in: 2011 IEEE Congress of Evolutionary Computation, CEC, 2011, pp. 1057–1062.
[69] S.M. Elsayed, R.A. Sarker, D.L. Essam, GA with a new multi-parent crossover for solving IEEE-CEC2011 competition problems, in: 2011 IEEE Congress of Evolutionary Computation, CEC, 2011, pp. 1034–1040.
[70] R. Luus, B. Bojkov, Global optimization of the bifunctional catalyst problem, Can. J. Chem. Eng. 72 (1) (1994) 160–163.
[71] R. Aris, N.R. Amundson, An analysis of chemical reactor stability and control, Chem. Eng. Sci. 7 (1958) 121–147.
[72] M.L. Dukic, Z.S. Dobrosavljevic, A method of a spread-spectrum radar polyphase code design, IEEE J. Sel. Areas Commun. 8 (5) (1990) 743–749.
[73] C. Lopez-Franco, J. Hernandez-Barragan, A.Y. Alanis, N. Arana-Daniel, A soft computing approach for inverse kinematics of robot manipulators, Eng. Appl. Artif. Intell. 74 (2018) 104–120.
[74] H.-C. Huang, C.-P. Chen, P.-R. Wang, Particle swarm optimization for solving the inverse kinematics of 7-DOF robotic manipulators, in: 2012 IEEE International Conference on Systems, Man, and Cybernetics, SMC, 2012, pp. 3105–3110.
[75] A. Li, P. Stief, J.-y. Dantan, A. Etienne, A. Siadat, Kinematics analysis and trajectory planning of collaborative welding robot with multiple manipulators, Proc. CIRP 81 (2019) 1034–1039.
[76] A. El-Sherbiny, M.A. Elhosseini, A.Y. Haikal, A comparative study of soft computing methods to solve inverse kinematics problem, Ain Shams Eng. J. 9 (4) (2018) 2535–2548.
[77] R. Köker, A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization, Inform. Sci. 222 (2013) 528–543.
[78] X.-S. Wang, M.-L. Hao, Y.-H. Cheng, On the use of differential evolution for forward kinematics of parallel manipulators, Appl. Math. Comput. 205 (2) (2008) 760–769.
[79] Y. Zhang, J. Wang, Obstacle avoidance for kinematically redundant manipulators using a dual neural network, IEEE Trans. Syst. Man Cybern., Part B (Cybernetics) 34 (1) (2004) 752–759.
[80] S. Li, S. Chen, B. Liu, Y. Li, Y. Liang, Decentralized kinematic control of a class of collaborative redundant manipulators via recurrent neural networks, Neurocomputing 91 (2012) 1–10.
[81] M. Duguleana, F. Grigore, A. Teirelbar, G. Mogan, Obstacle avoidance of redundant manipulators using neural networks based reinforcement learning, Robot. Comput. Integr. Manuf. 28 (2) (2012) 132–146.
[82] A.R.J. Almusawi, L.C. Dulger, S. Kapucu, A new artificial neural network approach in solving inverse kinematics of robotic arm (Denso VP6242), Comput. Intell. Neurosci. 2016 (2016) 5720163.
[83] S. Dereli, R. Köker, Calculation of the inverse kinematics solution of the 7-DOF redundant robot manipulator by the firefly algorithm and statistical analysis of the results in terms of speed and accuracy, Inverse Probl. Sci. Eng. 28 (5) (2020) 601–613.
[84] S.V. Reyes, S.P. Gardini, Inverse kinematics of Manipulator Robot using a PSO Metaheuristic with Adaptively Exploration, in: 2019 IEEE XXVI International Conference on Electronics, Electrical Engineering and Computing, INTERCON, 2019, pp. 1–4.
[85] C.-J. Lin, T.-H.S. Li, P.-H. Kuo, Y.-H. Wang, Integrated particle swarm optimization algorithm based obstacle avoidance control design for home service robot, Comput. Electr. Eng. (2015).