Novel Optimization Algorithm Inspired by Camel Traveling Behavior

Mohammed Khalid Ibrahim
, Department of Electrical Engineering
College of Engineering, University of Babylon, Iraq
mohammedkhalidibraheem@gmail.com

Ramzy Salim Ali
, Department of Electrical Engineering
College of Engineering, University of Basrah, Iraq
rsalwaily@ieee.org

Abstract: This article presents a novel optimization algorithm inspired by camel traveling behavior that called Camel algorithm (CA). Camel is one of the extraordinary animals with many distinguish characters that allow it to withstand the severer desert environment. The Camel algorithm used to find the optimal solution for several different benchmark test functions. The results of CA and the results of GA and PSO algorithms are experimentally compared. The results indicate that the promising search ability of camel algorithm is useful, produce good results and outperform the others for different test functions.

***Index Terms*- Evolutionary algorithms, Camel algorithm, Camel Traveling Behavior, Optimization algorithm.**

I. INTRODUCTION

In recently years, the emergence of many algorithms that mimicked or inspired by the different behavior of individual living creatures has been presented. It's mainly motivated by the need to present an efficient approach to deal with widespread optimization problems.

A genetic algorithm is the most popular evolutionary algorithms, which are used successfully in various engineering aspects [1]. Nowadays, most algorithms were developed based on swarm intelligence, biology, physical and chemical systems. In general, all the aforementioned algorithms may be called nature-inspired algorithms. The largest part of nature-inspired algorithms are known as a bio-inspired, and a subpart of the bio-inspired are known as a swarm intelligence based.

In literature, some of the algorithms based on swarm intelligence, like Particle swarm optimization (PSO) developed by Eberhart and Kennedy [2], inspired by the behavior of bird flocking and fish schooling. PSO has been developed to optimize a broad range of continuous functions successfully [3,4].

Ant colony optimization (ACO) [5] for routing in communication networks. Later, ACO algorithms have been developed and tested successfully in a different engineering fields.

Firefly algorithm, bat algorithm, and cuckoo search are also developed and tested in various engineering applications [6].

In the last years, several researchers deal with optimization algorithms inspired from animals behavior such as, Elephant search optimization that depends on habitual features of elephant herds [7]. Alireza Askarzadeh proposed a novel optimization search algorithm based on an intelligent behavior of crows [8].

The basic motivation to design the camel algorithm is inspired by the aim to present a simple structure algorithm that suitable to apply over a wide range of problem. This will lead to minimize the computation and processing consummation. In addition, the camel algorithm efficient performance was expected since it mimic one of the most successful behavior of survival and exploring in desert under extreme condition.

This paper presents a novel optimization algorithm inspired by the traveling behavior of

Camel in the desert in difficult environments. A Camel tends to move towards a region that contains food and water.

The rest of the paper is organized as follows: Section 2 presents theoretical concepts of the proposed Camel algorithm. Description of the CA and the benchmark test functions are given in Section 3. Section 4 illustrates the experimental settings and the experimental analysis of the proposed algorithm in comparison with GA and PSO algorithms. Finally, section 5 concludes the main achieved results.

II. CAMEL TRAVALING BEHAIVIOR

The CA is inspired by camel traveling movement and behavior. Under that consideration, several factors and operators are considered to outline CA algorithm procedure, including:

1. The temperature effect.
2. The supply (water and food).
3. The camel endurance.
4. Camel visibility (and /or hearing) range.
5. Random walk.
6. Group effect (multi-solution).
7. Termination condition (dying or moving back).
8. Land conditions (oasis, quick sand, storms, etc.).
9. Limitations (max speed, age and carrying weight).

Before discuss these factors and operators, and to avoid any confusion, in this work we will use $T, S,$ and E to refer to temperature, supply and endurance, respectively. The temperature effect is the primary random factor that will affect the journey of traveling camel and has an impact on camel's endurance. Furthermore, it can be vary from one camel to another since each camel is moving (searching) within the different sector in the desert (search space). For any j camel, the instantaneous temperature T_{now}^{j} can be expressed as:

$$T_{now}^{j} = (T_{Max} - T_{Min}) * Rand(0,1) + T_{Min} \quad (1)$$

where T_{Min} and T_{Max} are the minimum and maximum temperature, respectively. These values can be selected as desired such that $T_{Min} < T_{Max}$ (in this work, it has chosen that T_{Min}= 0 and T_{Max} = 100).

The supply (water and food) effect is a critical factor that reversely depends on journey duration. The remaining supply S_{now}^{j} can expressed as decreasing function for any j camel as:

$$S_{now}^{j} = S_{Past}^{j} * \left(1 - \omega * \frac{Travaled\ steps}{Total\ journey\ steps}\right) \quad (2)$$

where ω is a burden factor $\in$ (0,1]. At first journey step, the S_{Past}^{j} is equal to $S_{initial}^{j}$ that indicate initial full supply (suitable positive value). Each camel has its own initial supply at the begin of the journey, which can set to be different or equal (in this work, for simplicity, it's chosen that $S_{initial}^{1} = S_{initial}^{2} = \cdots = 1$ to indicate initial full supply for 1st camel, 2nd camel,…).

After each journey step, the supply will recurrently updated:

$$S_{Past}^{j} = S_{now}^{j} \quad (3)$$

Both temperature and journey duration affect camel endurance. For any j camel, this effect can express as decreasing function as:

$$E_{now}^{j} = E_{Past}^{j} * \left(1 - \frac{T_{now}^{j}}{T_{Max}}\right) * \left(1 - \frac{Travaled\ steps}{Total\ journey\ steps}\right) \quad (4)$$

At first journey step, the E_{Past}^{j} is equal to $E_{initial}^{j}$ that indicate initial full endurance. (In this work, for simplicity, it's chosen that $E_{initial}^{j} = E_{initial}^{j} = \cdots = 1$ for 1st camel, 2nd camel,…).

After each journey step, the endurance will recurrently update as:

$$E_{Past}^{j} = E_{now}^{j} \quad (5)$$

CA algorithm uses camel group effect to improve search. The fundamental updating equation of new solution for any j camel can express as:

$$x_{new}^{j} = x_{old}^{j} + \delta^{j} * \left(1 - \frac{E_{now}^{j}}{E_{initial}^{j}}\right) * e^{\left(1 - \frac{S_{now}^{j}}{S_{initial}^{j}}\right)} * (x_{best}^{*} - x_{old}^{j}) \quad (6)$$

Where x_{new}^{j} is the updated value (i.e. new solution) of j Camel. x_{best}^{*} is the best value obtained so far from any Camel. δ^{j} is a random walk factor represents the direction of moving, $\delta^{j} \in$ [-1,1]. At first journey step, the old solution x_{old}^{j} is the same $x_{initial}^{j}$ that generate randomly within the allowed range specified by the problem.

To decide the acceptance of solution, there are several factors added for cases such as:

1. When a new camel replaces the old one that is dying due quicksand, storms, etc. with dying rate $\mu_d \in [0,1)$ (the low-quality solution disappear and replace by new one). This can be in form such as:

$$if\ Fitness\,{}_{now}^{j} < (\mu_d * Fitness\,{}_{old}^{j})$$

Generate x_{new}^{j} randomly (within the problem allowed rang of solution)

$end\ if$

Where $Fitness\,{}_{old}^{j}$ is the fitness of soluation found by camel in last past journey step. If it is not required to replace camel, this can be simply done by set $\mu_d = 0$.

2. Moving back or stop (neglect last updating movement and restore old solution). This practically useful to keep the camel in it is search region or location (i.e. the camel step back to its previous location when to leave the allowed search area).

$if\ x_{new}^{j}$ is not within the allowed rang of solution

$$x_{new}^{j} = x_{old}^{j}$$

$end\ if$

When camel reaches the promising area with better solutions of its journey or search (oasis), the helpful factors can be increased (supply and endurance increase or back to initial state). Also, it is proper to assume that temperature does not change by oasis since it is random and external. The oasis effect can be directly linked to view range α of the camel to provide more lifelike mimicking of camel. Thus, for any j camel, the oasis effect can state as:

$$if\ \ [Rand(0,1) > (1 - \alpha^{j})$$

$$\&\&\ (Fitness\,{}_{now}^{j} > Fitness\,{}_{old}^{j})]$$

$$S_{past}^{j} = S_{initial}^{j}$$

$$E_{past}^{j} = E_{initial}^{j}$$

$end\ if$

where $Fitness\,{}_{now}^{j}$ is the fitness of recent soluation measurered by objective or fitness function.

III. CAMEL ALGORITHM

In contrast to the other existing algorithms, camel algorithm search for the best solution by set its parameters initially to proper values and apply its operators as explained before. The initial solutions is generated randomly within the range of x_i specified by each benchmark test function. The camel algorithm pseudocode is shown in Fig. 1.

```
% Camel Algorithm (CA)
Express problem information and objective or fitness function.
Express Camel Algorithm initial parameters values.
Create an initial Camel Caravan (multi-solutions).
Compute Camel Caravan individual fitness and find the current best one.
While (Counter < Total journey steps)
   For i =1: Camel Caravan
         Compute T_now^j, S_now^j, E_now^j in equations 1, 2 and 4 for each j camel in Caravan.
         Update camels' locations as expressed in equation (6).
         Decide the acceptance of new camels' locations (depend on solution quality measured by fitness function and range limitation).
         If (oasis condition occur)
                  Replenish Supply and Endurance
         End If
         Rank Camel Caravan individuals and find the current best one.
   End For i
End While
State the final results and the required plots.
```

Fig.1 Camel Algorithm pseudocode

A. Benchmark Test Functions

To evaluate the competence and suitability of new optimization algorithm, it is essential to test an algorithm using several well-known benchmark test functions with various characters [6]. Also, such test functions were used to perform a comparison between many exists optimization algorithm to verify that one algorithm surpassed others on a given set of problems.

In this work, several benchmark test functions considered to evaluate the performance of CA algorithm as follows:

1. Sphere function

Sphere function or De Jong's function is unimodal and convex well-known test function that given as [6]:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i{}^2 \qquad (7)$$

Where *n* is the dimension (*n*=1, 2, 3….) and the evaluation is usually done for a range of - 5.12 $\leq x_i \leq$ 5.12. Sphere function is known to have global minimum in 0 at *x* = (0, 0... 0). The two-dimension Sphere function is shown in Fig.2.

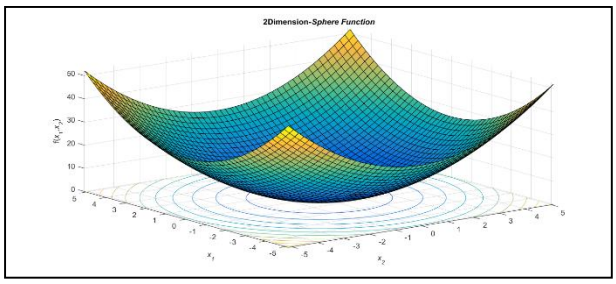


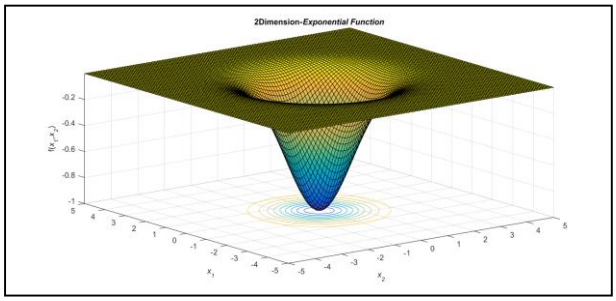
Fig.2 Two-dimension Sphere function

2. Exponential function

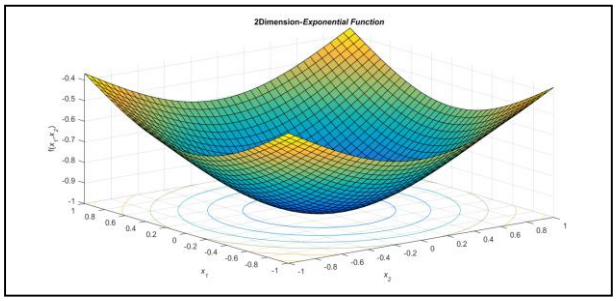
Exponential test function that was given as [6]:

$$f(x) = -e^{-0.5\left(\sum_{i=1}^{n} x_i{}^2\right)} \qquad (8)$$

Where *n* is the dimension (*n*=1, 2, 3) and the evaluation is usually done for a range of -1 $\leq x_i \leq$ 1. Exponential function has a global minimum in -1 at *x* = (0, 0... 0). The two-dimension Exponential function is shown in Fig.3 with different ranges.



(a)



(b)

Fig.3 Two-dimension Exponential function

3. Ackley function

Ackley function is well-known multimodal test function that given as [6]:

$$f(\boldsymbol{x}) = -20e^{-0.2\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i{}^2}\right)} - e^{-0.2\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)} + 20 + e^1 \quad (9)$$

Where *n* is the dimension (*n*=1, 2, 3….) and the evaluation is usually done for a range of $-32.768 \leq x_i \leq 32.768$. Ackley function is known to have global minimum f_{min} = 0 at *x* = (0, 0... 0). The two-dimension Ackley function is shown in Fig.4.

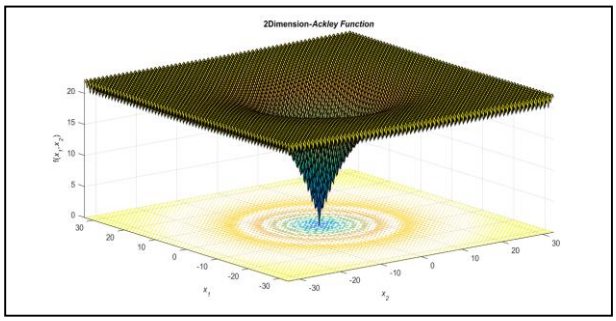


Fig.4 Two-dimension Ackley function

4. Rastrigin function

Rastrigin function is highly multimodal test function that is given as [6]:

$$f(\boldsymbol{x}) = 10n + \sum_{i=1}^{n}(x_i{}^2 - 10\cos(2\pi x_i)) \quad (10)$$

Where *n* is the dimension (*n*=1, 2, 3….) and the evaluation range $-5.12 \leq x_i \leq 5.12$. Rastrigin function's global minimum is f_{min} = 0 at *x* = (0, 0... 0). The two-dimension Rastrigin function is shown in Fig.5.

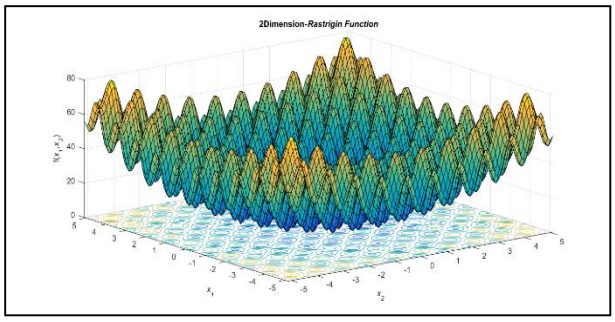


Fig.5 Two-dimension Rastrigin function

5. Griewank function

Griewank function is highly multimodal test function that is given as [6]:

$$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i{}^2 + \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (11)$$

Where *n* is the dimension (*n*=1, 2, 3….) and the evaluation is usually done for a range of $-600 \leq x_i \leq 600$. Griewank function is known to have global minimum f_{min} = 0 at *x* = (0, 0... 0). The two-dimension Griewank function is shown in Fig.6, with different ranges to demonstrate the local minimum points.

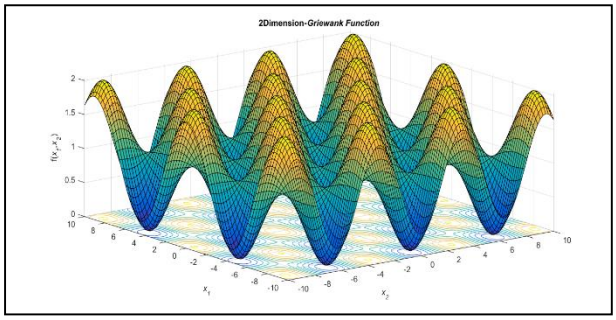


Fig.6 Two-dimension Griewank function

6. Schwefel function

Schwefel test function considered as one of most difficult test problems for optimization algorithms. Schwefel test function is multimodal with very deep sinusoidal indentations. Schwefel test function global minimum is located in a distant away from the next local minima, which is likely lead to convergence in the wrong direction. Schwefel test function was given as [6]:

$$f(x) = 418.9829n - \sum_{i=1}^{n} x_i \quad (12)$$

Where *n* is the dimension (*n*=1, 2, 3….) and the evaluation is usually done for a range of $-500 \leq x_i \leq 500$. Schwefel function is known to have global minimum f_{min} = 0 at *x* = (420.9687, 420.9687... 420.9687). The two-dimension Schwefel function shown in Fig.7.

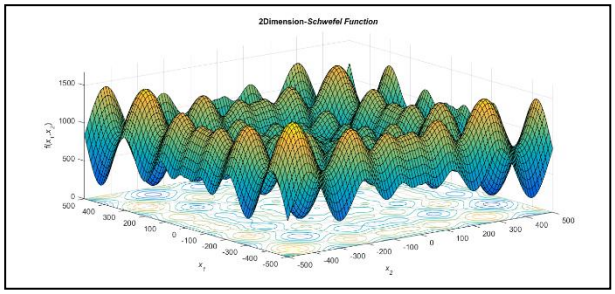


Fig.7 Two-dimension Schwefel function

B. Settings of Algorithms

The following values are used for Camel algorithm parameters:

1. Total journey steps = 100 or 1000
2. Minimum temperature = 0
3. Maximum temperature = 100
4. Initial supply = 1
5. Initial endurance = 1
6. Visibility = 0.5
7. Camel Caravan = 50
8. Dying rate = 0
9. benchmark test functions dimension selected = [2, 10, 20]

The experimental setting for GA is as follows:

1. Total generation = 100 or 1000
2. Crossover rate = 0.8 (one point crossover type)
3. Mutation rate = 0.1 (one point mutation type)
4. Individuals = 50
5. Roulette Wheel Selection method and elitism method
6. Benchmark test functions dimension selected = [2, 10, 20]

The experimental setting for PSO is as follows:

1. Total iterations = 100 or 1000
2. Inertia weight = 1
3. Inertia weight decreasing ration = 0.99
4. Local learning coefficient = 2
5. Global learning coefficient = 2
6. Maximum velocity = 10
7. Minimum velocity = -10
8. Number of particles=50
9. Benchmark test functions dimension selected=[2, 10, 20]

It can be seen from the setting parameters that there are some parameters like the total journey steps, the total generations, and the total iterations have principly similar meaning. Also the number of Camel caravan, the individuals, and the number of particles equal to 50 for all algorithms CA, GA, and PSO, respectively. We select three different dimensions: 2, 10 and 20 for all results. The other parameters have a different meaning for all three algorithms.

IV. EXPERIMENTAL RESULTS

The benchmark test functions indicated above programmed in a Matlab's m-files and used to evaluate the ability of the Camel algorithm to outperform the other algorithms. In the experiments, the number of iterations is chosen the same for all three algorithms to measure the performance and the speed of the convergence that obtains the best fitness of all benchmark test functions. Table I indicates the comparison results for the above algorithms with the best fitness, mean and standard deviation. It can be noticed that the CA reaches to the best fitness within the 100 iterations for all test functions except Schwefel function and gives better results than the other algorithms. The CA gives bad results for Schwefel function in 100 iterations, but the results have been improved for 1000 iterations. These difficulties were shown in all three algorithms, but the CA reaches to near global minimum for all test functions while GA and PSO are failed as seen in Table.I.

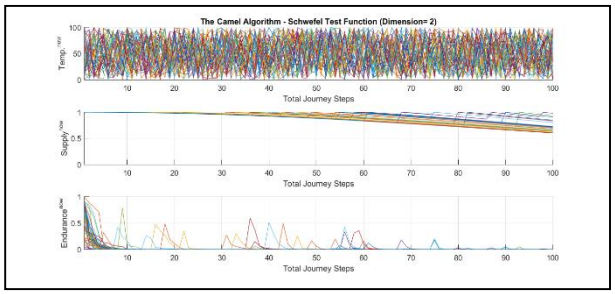
The speed of convergence to the global minimum is an important measure for evaluating the algorithms, Fig. 8 shows the Schwefel function results for CA. Fig.8-a indicates the changes in temperature, supply that represented by food and water, and endurance with total journey steps (iterations).

TABLE I
BEST FITNESS, MEAN and STANDERAD DEVIATION of SIX TEST FUNCTIONS

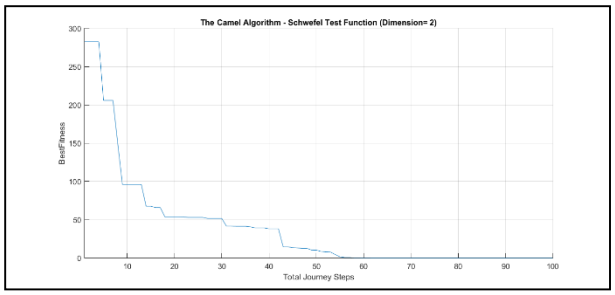
		GA		PSO		CA	
Fun	**Dim**	**B.F.**	**Mean±Stdv**	**B.F.**	**Mean±Stdv**	**B.F.**	**Mean±Stdv**
Sph.	2	1.9043	0.5147 ±1.1907	2.1898e-12	-5.4627e-08 ±1.4778e-06	0	0±0
	10	3.4521e+03	-1.8986 ±34.6458	0.0518	0.0057 ±0.0756	0	0±0
	20	1.1845e+04	2.9004 ±36.3079	11.5221	0.2649 ±0.7298	0	0±0
Exp.	2	-0.9999	0.0328 ±0.0559	-1.0000	-1.1288e-08 ±3.6780e-08	-1	0±0
	10	-0.7634	-0.0973 ±0.4413	-0.4885	-0.3000 ±0.6749	-1	-3.1077e-10 ±9.7721e-10
	20	-0.3956	0.1356 ±0.4866	-0.5793	0.0478 ±0.2347	-1	3.6227e-10 ±1.6201e-09
Ack.	2	1.7841	-0.1647 ±0.0779	6.9359e-06	-9.0105e-07 ±2.0951e-06	8.8818e-16	0±0
	10	15.9419	-0.4162 ±12.2627	0.3422	-0.0012 ±0.0544	8.8818e-16	0±0
	20	18.8913	1.4832 ±13.5131	4.1008	-0.2590 ±0.6588	8.8818e-16	0±0
Ras.	2	0.1440	0.0283 ±0.0583	1.3075e-09	-1.2570e-06 ±1.8521e-06	0	0±0
	10	45.6585	-0.4945 ±2.2429	73.4155	-0.0918 ±2.7353	0	-3.0611e-13 ±9.6801e-13
	20	168.2051	-0.0553 ±1.9153	264.8060	-0.0530 ±3.7258	0	0±0
Gri.	2	0.1061	-9.7361 ±8.4242	0.0099	-3.1400 ±4.4407	0	0±0
	10	36.6906	-41.8454 ±195.520	0.2287	0.5029 ±9.7193	0	-5.4580e-11 ±1.7260e-10
	20	107.6069	43.257 ±209.2067	0.9852	-0.2333 ±2.3385	0	0±0
Sch.	2	5.5065	64.0740 ±497.621	2.5455e-05	420.9688 ±3.9228e-06	2.5455e-05	420.9688 ±8.8482e-06
	10	1.8087e+03	57.4763 ±327.642	1.9941e+03	111.0072 ±255.0457	18.8830	420.1279 ±3.9868
	20 (100 step)	4.7606e+03	22.2621 ±283.554	4.7698e+03	-18.6564 ±219.2892	2.030e+03	259.3556 ±284.6770
	20 (1000 step)	4.0038e+03	-3.3902 ±313.5934	4.7584e+03	-18.2390 ±218.9861	0.0160	420.9799 ±0.0801

Fig.8-b shows the convergence to best fitness during 100 iterations. It can be seen from Fig.8 that, CA reaches near global optimum=2.5455e-05 in only 55 iterations. Fig. 9 shows the best fitness convergence for Schwefel test function using GA. Fig.10 shows best fitness convergence for the same test function using PSO. From figures (8, 9 and 10), it can be seen that the PSO gives fast convergence compared with GA and CA. It seems that only this case that PSO algorithm has the best result compared with the other two. All other cases for all test functions, the CA algorithm has better ability to solve the optimization problems.

Figures (11-19) show the fitness convergence for the Schwefel test function with dimensions: 10 and 20 with 100 iterations and dimensions 20 with 1000 iterations for CA, GA and PSO algorithms. It is clear that the CA reaches near the global optimum 0.0160 while the others failed.



(a)



(b)

Fig. 8 Convergence of CA for Schwefel function with two dimensions with total journey steps (iterations), (a) the variation of temperature, supply and endurance (b) best fitness.

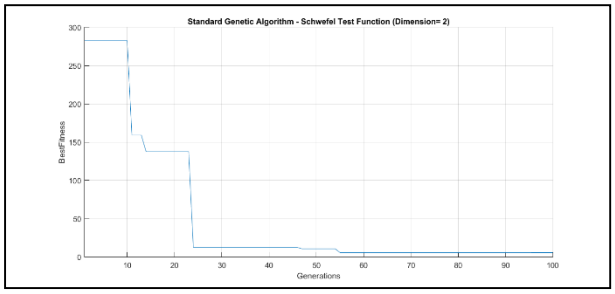


Fig. 9 Best fitness convergence of GA for Schwefel function with two dimensions

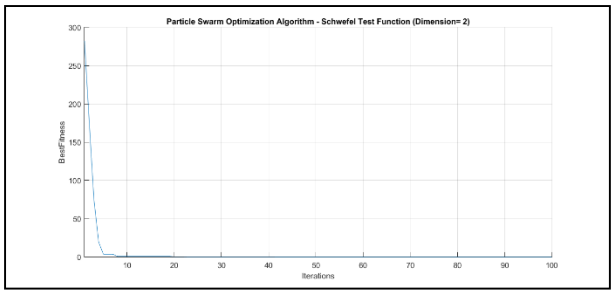
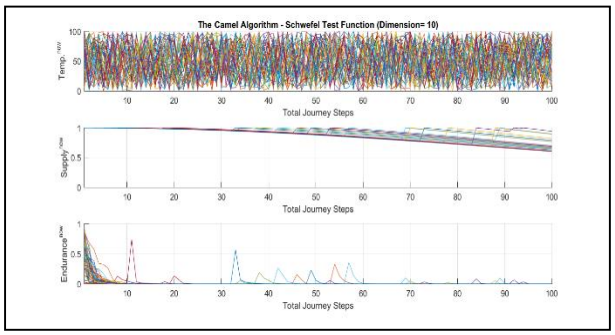
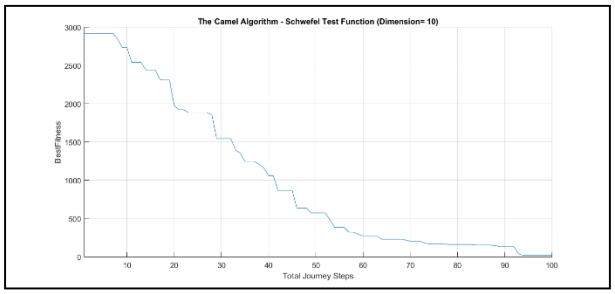


Fig. 10 Best fitness convergence of PSO for Schwefel function with two dimensions



(a)



(b)

Fig. 11 Convergence of CA for Schwefel function with ten dimensions with total journey steps (iterations), (a) the variation of temperature, supply and endurance (b) best fitness.

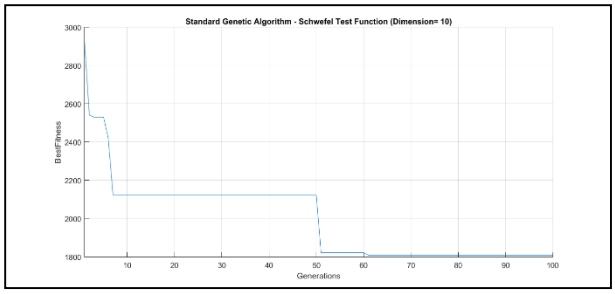


Fig. 12 Best fitness convergence of GA for Schwefel function with ten dimensions

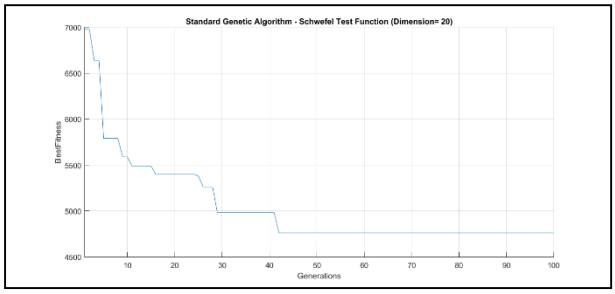


Fig. 15 Best fitness convergence of GA for Schwefel function with twenty dimensions and 100 generations

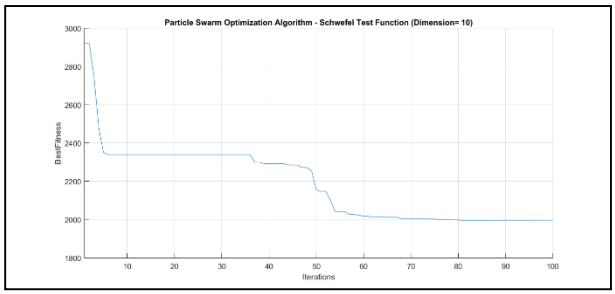


Fig. 13 Best fitness convergence of PSO for Schwefel function with ten dimensions

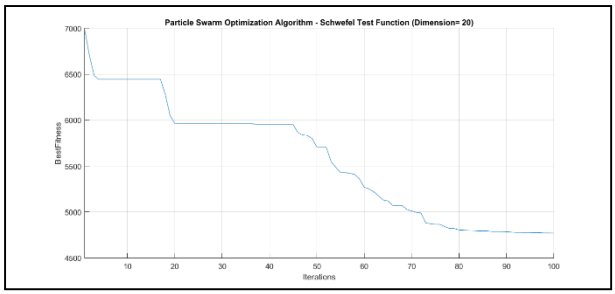
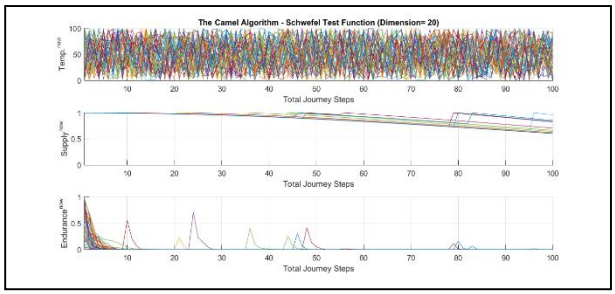
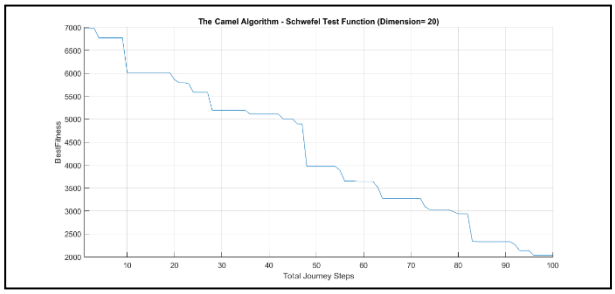


Fig. 16 Best fitness convergence of PSO for Schwefel function with twenty dimensions and 100 iterations

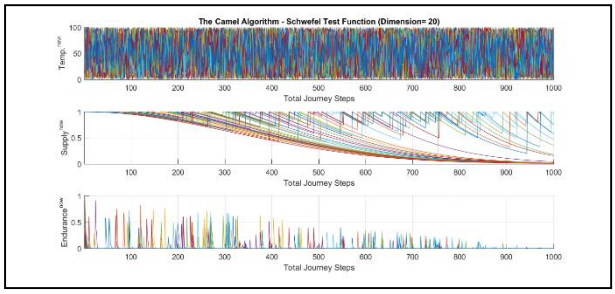


(a)

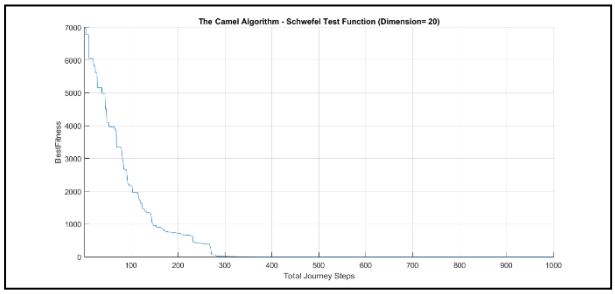


(b)

Fig. 14 Convergence of CA for Schwefel function with twenty dimensions with total journey steps =100, (a) the variation of temperature, supply, and endurance (b) best fitness.



(a)



(b)

Fig. 17 Convergence of CA for Schwefel function with twenty dimensions with total journey steps =1000, (a) the variation of temperature, supply and endurance (b) best fitness.

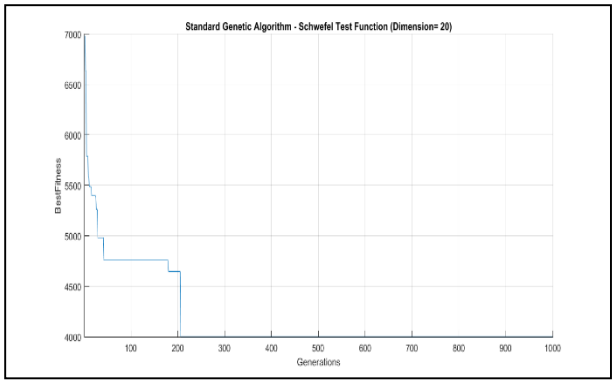


Fig. 18 Best fitness convergence of GA for Schwefel function with twenty dimensions and 1000 generations

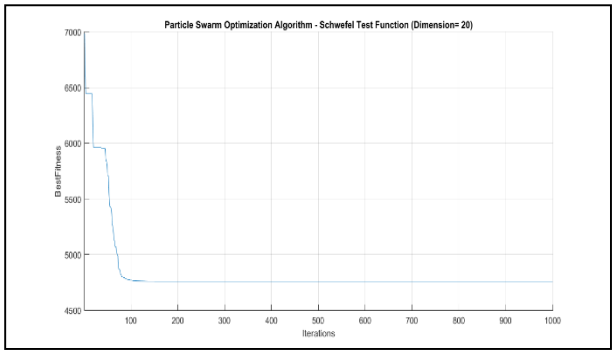


Fig. 19 Best fitness convergence of PSO for Schwefel function with twenty dimensions and 1000 iterations

V. CONCLUSIONS

The results indicate that the novel proposed camel algorithm is a very promising algorithm. The camel algorithm simple structure along with its efficient search ability allow it to deal effectively with unimodal and multimodal test functions to find an optimal solution even with difficult ones. The oasis effect gives a conditional boost to search depending on the quality of solution obtained and visibility of camel, which offers balanced between escaping optimum local areas in one hand and search effort on the other side. Also, the results show that camel algorithm can obtain excellent results in the early stage of search journey for most of the test functions, which suggest that camel algorithm would be very suitable to work with real-time applications and time sensitive optimization problems.

Even though the work results is confirm the ability and superior performance of the camel algorithm comparing to PSO and GA over different test benchmark problems and ranges, yet it is not indicate which type of problems that camel algorithm might struggle to solve or should not apply to. Taking into account the no free lunch (NFL) theorem perspective, this should be further investigate to explore the set of problems that most suitable for camel algorithm especially in real life applications.

Besides, further investigations on camel algorithm parameters values might lead to further improvements in its performance and convergence speed.

REFERENCES

[1] J. Holland, Adaptation in natural and artificial systems, Ann Arbor, MI: University of Michigan Press, 1975.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization", in Proceeding of IEEE Int. Conf. Neural Networks, vol. 4, 1995, pp. 1942-1947.

[3] Y. Volkan Pehlivanoglu, "A New Particle Swarm Optimization Method Enhanced With a Periodic Mutation Strategy and Neural Networks", IEEE Transactions on Evolutionary Computation, Vol. 17, No. 3, 2013.

[4] Hao Gao and Wenbo Xu, "A New Particle Swarm Algorithm and Its Globally Convergent Modifications", IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol. 41, No. 5, 2011.

[5] M. Dorigo and T. Stutzle, Ant Colony Optimization, Cambridge, MA: MIT Press, 2004.

[6] X. S. Yang, Engineering Optimization: An

Introduction with Metaheuristic Applications, Wiley & Sons, New Jersey, 2010.

[7] S. Deb, S. Fong, and Z. Tian, "Elephant search algorithm for optimization problems", 10th International conference on Digital Information Mangement, South Korea, Oct. 21-23, 2015, pp. 249-255.

[8] Alireza Askarzadeh, " A Novel meta-heuristic method for solving constrained engineering optimization problems: Crow search algorithm", Computers and Structures, Elsevier, 169, 2016, pp.1-12.