Taj Al-Ma'rifa Journal مجلة تاج المعرفة
ISSN : 3104-8846, ISSN-L: 3104-8838

Comparison of Three Well-known Global Optimization Algorithms for Solving Black Box Problems

* Abdulbaset Saad
Janzour Technical college
Abdulbasetsaad@gmail.com

[2] Ezzeddin Bendaw
College of Renewable Energy - Tajoura
ezeskhhh@gmail.com

[3] Fatah Mohamed Shakrum
Higher institute of medical technology - Abosalim
Fatah.trip@gmail.com

*Corresponding Author: * Abdulbaset Saad

Keyword

Stochastic Global Optimization Algorithms, SA, DE, HS

Abstract

Stochastic Global Optimization Algorithms (GOA) have been employed widely due to their efficiency in handling difficult engineering optimization problems. The stochastic or heuristic methods are based on the random generation of feasible points, or sampled points, and non-linear local optimization search procedures using these points. In addition to a number of well known global optimization methods, many new GOA have been presented recently for numerous optimal design applications. In this paper, three well-known global optimization algorithms include Simulated Annealing (SA), Differential evolution (DE), and Harmony search (HS) are closely tested and compared. The historical development, special features and trends on the development of these stochastic GOA are studied. Special consideration is devoted to the performance of those GOA while the number of iterations are limited. Commonly used benchmark optimization functions are used as test examples to reveal the pros and cons of those three recognized global optimization algorithms.

Received : 20/03/2026 Accepted : 31/03/2026

DOI: https://doi.org/10.64943/jkc.2026.040205

مقارنة بين ثلاث خوارزميات عالمية معروفة لاستخدامها في ايجاد الحلول للمسائل المعقدة

* عبدالباسط الهادي سعد
كلية التقنية الهندسية / جنزور
Abdulbasetsaad@gmail.com

[2] عزالدين مصطفى بن ضو
كلية الطاقات المتجددة / تاجوراء
ezeskhhh@gmail.com

[3] فاتح علي شكروم
المعهد العالي للعلوم والتقنيات الطبية / ابوسليم
Fatah.trip@gmail.com

*الباحث المرسل: * عبدالباسط الهادي سعد

الكلمة المفتاحية	الملخص
خوارزميات التحسين العالمية العشوائية، SA، DE، HS	تُستخدم خوارزميات التحسين العالمي العشوائية (GOA) على نطاق واسع نظرًا لكفاءتها في معالجة وتحسين اداء النظم الهندسي المعقدة. حيث تعتمد الطرق العشوائية أو الاستدلالية على اختيار نقاط عشوائية لنقاط الحلول الممكنة، أو النقاط المأخوذة عينات منها، وتتم إجراءات البحث عن التحسين المحلي غير الخطي باستخدام هذه النقاط المختارة . بالإضافة إلى عدد من طرق التحسين العالمي المعروفة، تم تقديم العديد من خوارزميات التحسين العالمي الجديدة مؤخرًا لتطبيقات التصميم الأمثل المتعددة. في هذه الورقة، تم اختبار ومقارنة عدد ثلاثة خوارزميات تحسين عالمي معروفة، وهي: التلدين المحاكي (SA) ، والتطور التفاضلي (DE)، والبحث التوافقي (HS). كما تم دراسة التطور التاريخي والخصائص المميزة والاتجاهات السائدة في تطوير هذه الخوارزميات العشوائية. وتم دراسة أداء هذه الخوارزميات عندما يكون الزمن وعدد محاولات ايجاد الحل الامثل محدودًا جداً. واستُخدمت دوال التحسين المعيارية الشائعة الاستخدام كأمثلة اختبارية للكشف عن مزايا وعيوب هذه الخوارزميات الثلاث المعروفة.

تاريخ الإستقبال: 2026/03/20 تاريخ القبول: 2026/03/31

DOI: https://doi.org/10.64943/jkc.2026.040205

Introduction

In the last two ducats, significant progress has been achieved in developing capable, efficient, and robust global optimization methods. Many optimization problems that were considered tough to solve and intractable even in recent years can today be effectively solved. The resulting wide variety in global optimization methods means that many users are unaware of the latest developments, and hence cannot make an adequate informed choice of the global optimization method. Through this overview on the features, trends, pros and cons of three existing global optimization algorithms and promising directions of new GOA development for the complex engineering are revealed. Global optimization algorithms (GOA), in general, can be divided into two main groups: deterministic algorithms and stochastic (or non -gradient) algorithms. Deterministic methods solve an optimization problem by generating a deterministic sequence of points converging to a globally optimal solution, such as the branch and bound, DIRECT search, and tunnelling approaches.

These approaches converge rapidly to the global optimal solution; however, they need the optimization problem to have certain mathematical characteristics. The stochastic or heuristic methods are based on the random generation of feasible points, or sampled points, and non-linear local optimization search procedures using these points. Typical stochastic optimization methods include genetic algorithms (GAs) [1], simulated annealing (SA) [2], Tabu Search [3], ant colony optimization (ACO) [4], and particle swarm optimization (PSO) [5]. Genetic algorithms are search methods that are inspired by the natural selection and survival of the fittest in the biological world, and are different from traditional optimization techniques since GAs involve a search from a population of solutions, rather than a single point. Simulated annealing is a generalization of the Monte Carlo method for examining the equations of state and frozen states of n-body systems. The algorithm emulates the annealing process on how liquid freezes or metal re-crystallizes in cooling. Simulated annealing is easy to implement, although the method converges slowly and it is difficult to find an appropriate stopping rule. Statistical algorithms employ a statistical model of the objective function to bias the selection of a new sample point. One of the challenges in using statistical methods is the verification on the appropriateness of the statistical model for the class of problems under consideration. Tabu search, as described by Glover, is a meta-heuristic superimposed on another heuristic. The approach is characterized by forbidding or penalizing moves which take the solution in the next iteration to points in the solution space previously visited to avoid entrainment in cycles [6]. As a relatively new search method, Tabu search has traditionally been used on combinatorial optimization problems. Under active research, the method continues to evolve and improve. Ant colony optimization is a swarm intelligence technique which was originally proposed for combinatorial problems and lately for difficult and discrete optimization problems. Particle swarm optimization is a population-based stochastic optimization technique, inspired by social behaviour of bird flocking or fish schooling and developed by Kennedy and Eberhart [7]. Stochastic optimization methods are capable of solving complex optimization problems, but the algorithms usually suffer from certain inefficiency in dealing with continuous multimodal functions due to two major drawbacks, premature convergence and weak exploitation capability. Occurrence of premature convergence often leads to a local optimum instead of global optimum, while weak exploitation capability often causes slow convergence.

Related Work

Many Global optimization techniques have been around for many years and have gained excellent reputation due to their outstanding performance and capability to recognize the design optimum, whenever used properly, in different optimization problems. An overview of these stochastic optimization methods is presented to prepare for the later performance comparison of various optimization methods. Many optimization problems in engineering, science, and even in medicine can be expressed as global optimization (GO) problems. Thus, over the past three decades, significant progress has been made in introducing and developing efficient and robust GO algorithms. Classical optimization techniques (gradient-based) such as Newton's method, the steepest descent method, and the simple method, are widely used for finding a solution for many optimization problems; however, these classical methods have difficulties with obtaining a global solution particularly when the functions have numerous local minima. Global optimization algorithms can replace the classical optimization methods and, because of their capabilities, have become commonly used when dealing with many challenging and complex engineering optimization problems. Nature-based global optimization (NBGO) [8] algorithms are a branch of the GO methods, and found to be efficient, flexible, and easy to implement. For instance, Genetic Algorithm (GA) , Simulated Annealing (SA) , Particle Swarm Optimization (PSO) , Artificial Bee Colony (ABC) [9], Firefly Algorithm (FFA) [10], Cuckoo Search (CS) [11], Bat Algorithm (BA) [12], Flower Pollination Algorithm (FPA) [13], and Grey Wolf Optimizer (GWO) [14], have been introduced with improved ability to deal with complex and high-dimensional global optimization problems [15]. GO algorithms require no gradient information and can provide high accuracy with reasonable time complexity. As well, the global optimization algorithms are capable of solving complex optimization problems and are particularly useful when evaluation of the objective function is black box.

Simulated Annealing (SA)

The Simulated Annealing (SA) algorithm is the most popular GO search approach and draws its name from the metallurgical process, "Annealing". Annealing is a process used to change the properties of a metal wherein the metal is heated to a certain high temperature and then allowed to slowly cool with a specific cooling rate [16]. Similarly, SA explores the design space controlled by two parameters: SA method is the most popular GO search approach and draws its name from the metallurgical process, “Annealing”. Annealing is a process used to change the properties of a metal wherein the metal is heated to a certain high temperature and then allowed to slowly cool with a specific cooling rate [16].

In practice, SA has successfully solved the famous traveling salesman problem (TSP) [17]. SA has been found to be very powerful for several types of optimization problems; however, the most noticeable drawbacks are that this algorithm requires a long time to find the global optimum solutions, and the trade-off between the global solution and the CPU time needed to obtain the global solution is very high.

Differential Evolution

Differential evolution (DE) [17] is one of the most powerful evolutionary GO algorithms used at present. DE functions use the same computational steps as the evolutionary algorithms (EA). The DE is considered to be similar to GA since it uses comparable operators; crossover, mutation, and selection. The main difference between DE and GA is that DE is based on mutation operation while GA is based on crossover operations to select a better solution. This method was presented by Storn and Price in 1997 [17]. Since this approach is based on mutation, it employs the mutation as a search tool and takes advantage of the selection process in order to direct the search towards the promising regions in the design space. Target Vector, Mutant Vector, and Trail Vector are the three main steps that DE applies for creating a new population in each iteration. The target vector is the vector that holds the candidate solution for the design space; the mutant vector is the mutation of the target vector; and the trail vector is the outcome vector after applying the crossover process between target vector and mutant vector. DE begins with population initialization followed by evaluation to define the fittest candidates of the population. Then, the mutation operator is applied and new parameter vectors are created by adding the weighted difference of the two population vectors with the third vector. Later, the crossover operator is used, and the DE reaches a final stage of selection.

Due to its flexibility and simplicity, the DE has been commonly used in many engineering applications. The characteristics that make DE a powerful algorithm is its capability of handling non-differentiable, nonlinear, and multimodal objective functions. It has been used to train neural networks having real and constrained integer weights [18]. There are a handful of results showing that the DE is often more effective and more efficient than genetic algorithms while it suffers from slow convergence or being trapped in local optimum.

Harmony search (HS)

Harmony search (HS) [19] algorithm is the most popular meta-heuristic algorithms optimization algorithm. The HS algorithm uses a stochastic random search, instead of a gradient search. Harmony Search Algorithm (HSA) has recently been developed based on music improvisation process, HS algorithm uses harmony memory considering rate and pitch adjustment rate for finding the solution vector in the search space. The HS algorithm uses the concept, how aesthetic estimation helps to find the perfect state of harmony, to determine the optimum value of the objective function. The HS algorithm is simple in concept, few in parameters and easy in implementation. The harmony is stored in the harmony memory (HM) as an experience where each instrument player can refer to. A Harmony Memory Size (HMS) indicates the maximum number of harmonies that instrument players can refer to. Whenever a new harmony is generated, each instrument player decides their own note considering a Harmony Memory Considering Rate (HMCR), a Pitch Adjusting Rate (PAR), and a Fret Width (FW). HMCR is the probability that each player refers to a randomly selected harmony in the Harmony Memory (HM). PAR is the probability of adjusting a harmony that was picked from HM. FW is the bandwidth of the pitch adjustment. The overall HS process iterates the Maximum Improvisation (MI) number of consecutive trails. HSA has been successfully applied to various optimization problems.

benchmark optimization test problems

In order to evaluate the efficiency of DE, SA and HS, a number of well-known benchmark problems are used. The dimension of these problems' ranges from 2 to 16. There are, five two-dimensional cases (Banana, Beak, SC, Shub, Leon), three four-dimensional cases (Shekel, and Levy), one six-dimensional problem (HM6D), one ten-dimensional case (Sphere) and one sixteen-dimensional case F16 [20]. All of them have their own structures and characteristics and can better simulate most of the situations appearing in engineering problems. The details of these functions can be seen in Table1. the above-mentioned benchmark problems. The above-mentioned global optimization algorithms are compared and the statistical results from them are used to reflect their efficiency and robustness.

Table 1 Benchmark problems for global optimization

Category	Fun.	Number of D	Design space	Analytic global optimum
Low dimensional problems (n=2-6)	Banana	2	$[-2, 2]^2$	0
	Beak	2	[-3 3] ×[-4 4]	-6.5511
	SC	2	$[-2, 2]^2$	-1.0320
	Shub	2	$[-10, 10]^2$	-186.7309
	Leon	2	$[-10, 10]^2$	0
	Shekel	4	$[0, 10]^4$	-10.1531
	Levy	4	$[-10, 10]^4$	0
	HM6	6	$[0, 1]^4$	-3.3220
High dimensional problems (n >=10)	Sphere	10	$[-5.12, 5.12]^{10}$	0
	F16	16	$[-1, 1]^{16}$	25.875

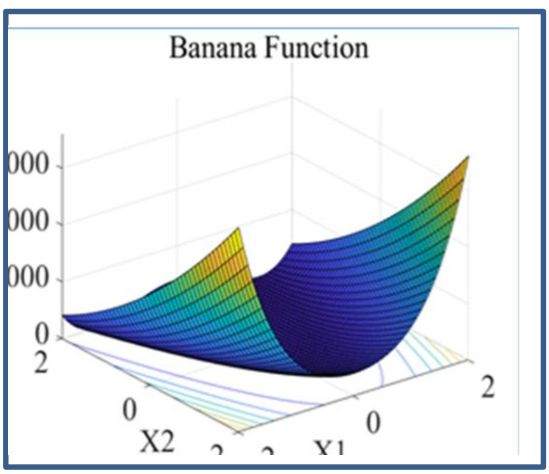


Fig. (1)

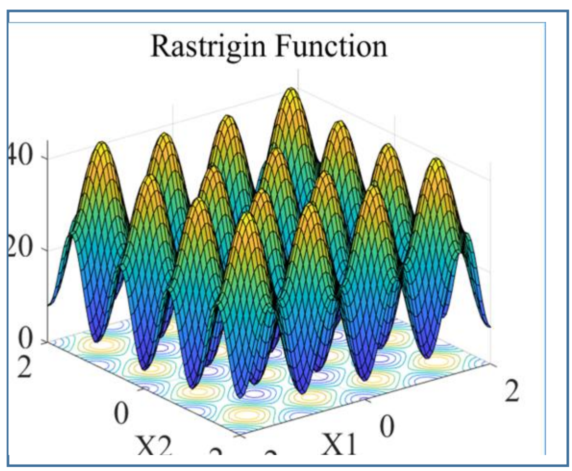
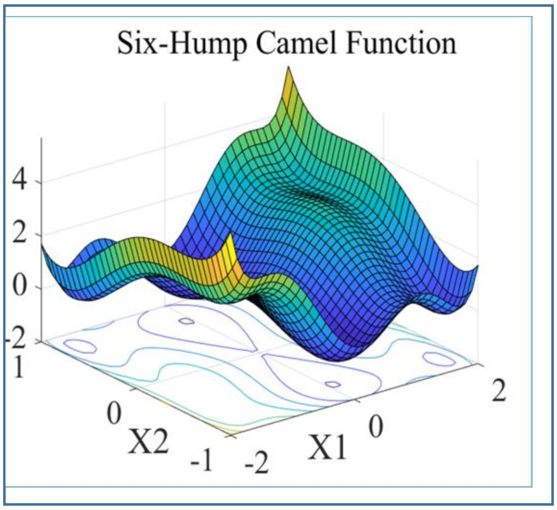


Fig. (2)



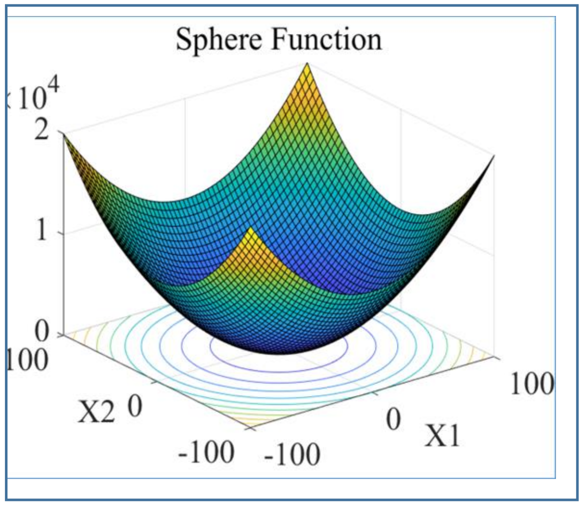


Fig. (3) Fig. (4)

Table 2 Preliminary comparison results on seven representative benchmark functions

Fun.	DE		SA		HS	
	Min f^*	CPU time	Min f^*	CPU time	Min f^*	CPU time
Banana	2.1907E-07	0.859	4.3665E-06	0.945	3.01E-04	0.848
Beak	-6.5303	0.203	-6.5511	0.0625	-6.5509	0.0312
SC	-1.0312	0.0265	-1.0316	0.0165	-1.0309	0.0514
Shub	-186.448	0.1875	-186.7171	0.0468	-186.6716	0.0271
Leon	8.4446E-05	0.1718	0.1054	0.0317	0.10553	0.0414
Shekel	-10.303	0.5312	-10.531	0.0625	-2.4233	0.0781
Levy	8.753E-06	0.0512	7.8977	0.0468	7.0297E-04	0.0787
HM6	-3.3221	1.4063	-3.3222	0.2656	-3.2781	0.375
Sphere	8.96E-06	3.165	2.925E-04	0.1406	0.020741	0.125
F16	25.876	3.3120	25.8753	0.6562	26.0861	0.750

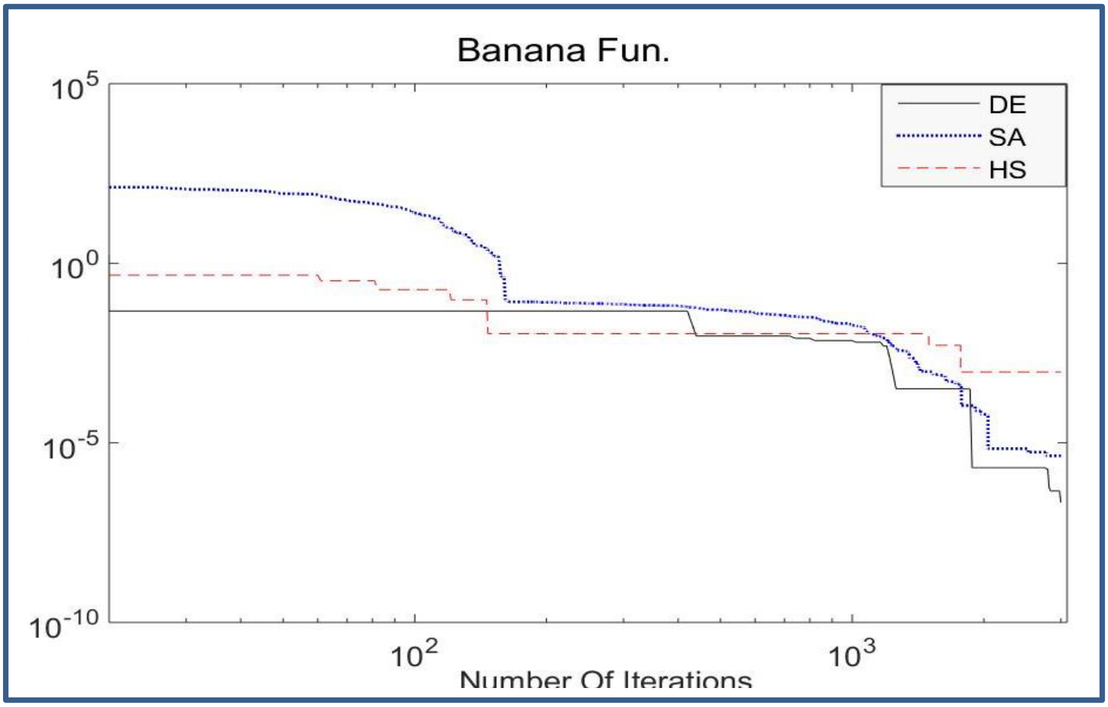

Banana Fun.
DE
SA
HS
10^5
10^0
10^{-5}
10^{-10}
10^2
10^3
Number Of Iterations

Fig. (5)

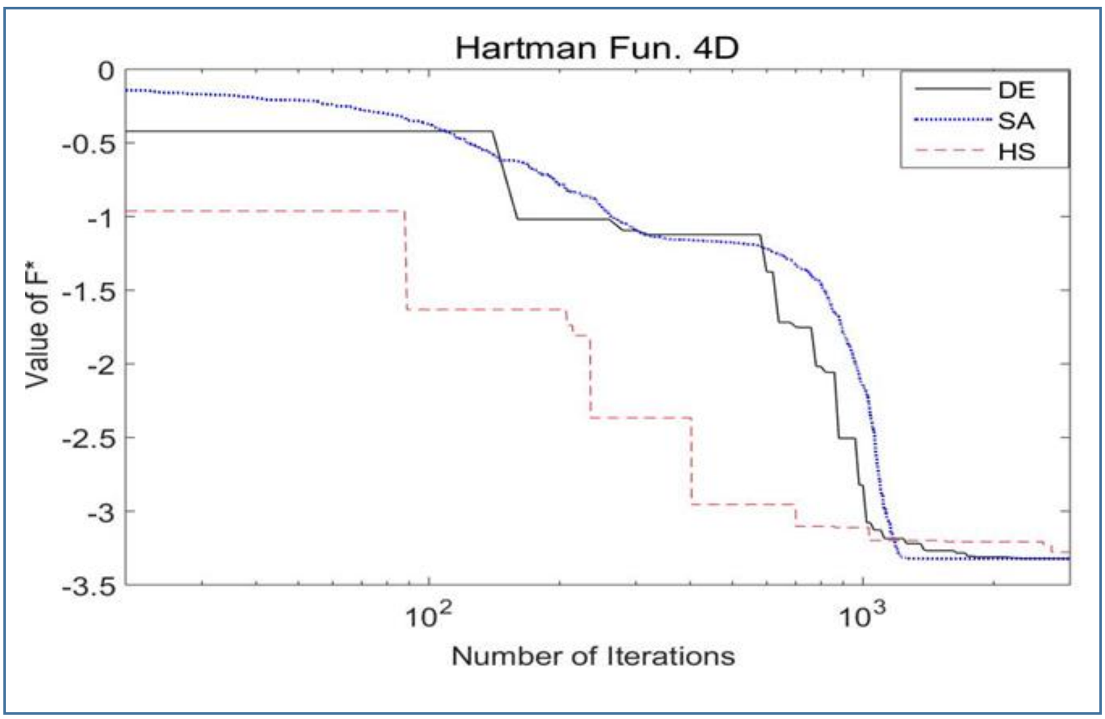

Hartman Fun. 4D
DE
SA
HS
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
Value of F*
10^2
10^3
Number of Iterations

Fig. (6)

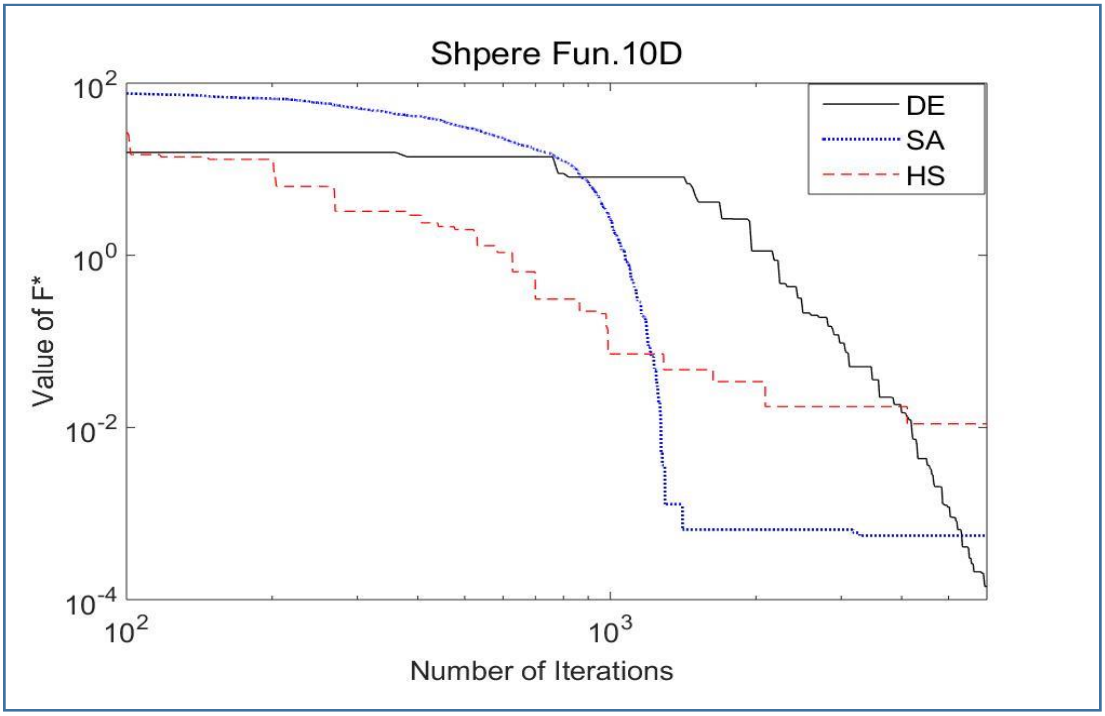

Shpere Fun.10D
DE
SA
HS
Value of F*
10^2
10^0
10^{-2}
10^{-4}
10^2
10^3
Number of Iterations

Fig. (7)

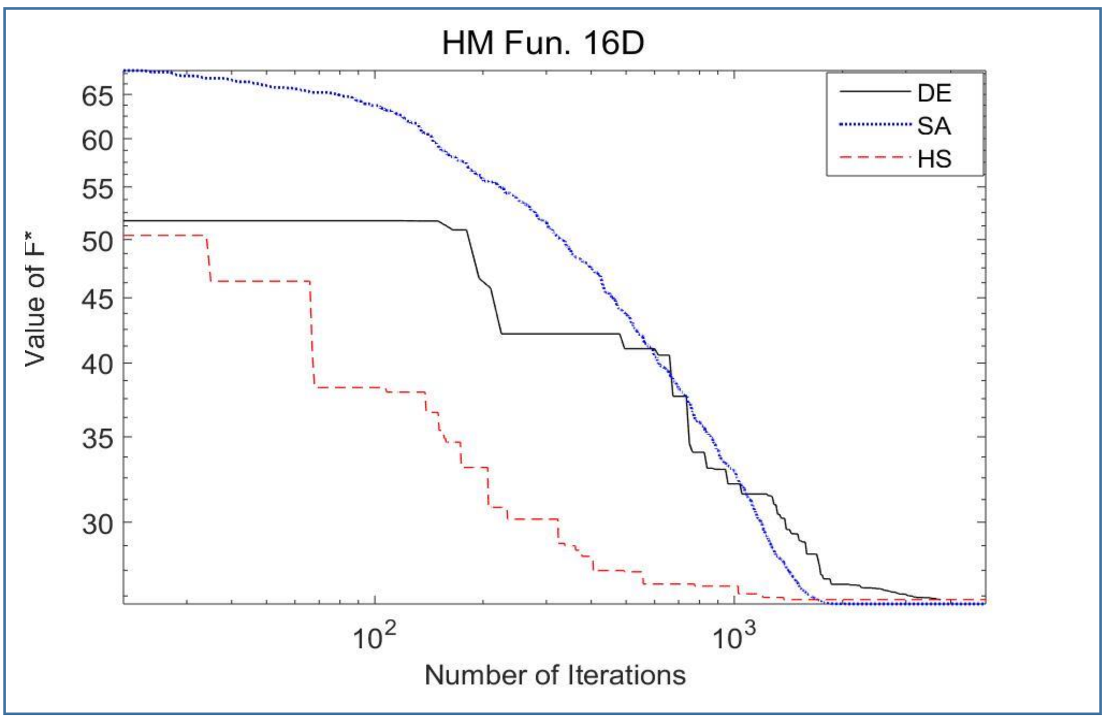

HM Fun. 16D
DE
SA
HS
Value of F*
65
60
55
50
45
40
35
30
10^2
10^3
Number of Iterations

Fig. (8)

Experiments Results

In this experiment, the impact of a limit of the number of iterations on the accuracy of the results of the tested algorithms are analyzed. Figure 5 to Figure 8 show the optimal solution found by DE, SA and HS approaches versus the iteration number for four computationally-expensive problems. The selected functions are Banana (*f1*), HM6D (*f8*) ,Sphere (*f9*)and HM16D (*f10*), which represent unimodal, multimodal and tough shape functions. The primary goal is to measure the performance of these algorithms based on different surface topologies/structures. The accuracy is measured by comparing the obtained optimum with the known actual optimum of the function. The evaluated efficiency is based on the convergence rate of these algorithms under the given conditions. The algorithm that yields accurate global solutions in a shorter computation time is the efficient and robust method.

As illustrated from the experiments results, DE, followed by SA, have shown the best performance among the examined algorithms regardless of the function form used. DE starts to converge to the global solution within approximately 100 iterations, while the other algorithms require a higher number of iterations to find the region where the global solution may exist. The SA algorithm also demonstrates its superior efficiency in terms of the accuracy of the solution. This is, perhaps, because of the DE's and SA's search mechanism that prevents them from easily being trapped in local optima. DE obtained the global solution for the most tested optimization problems with a very quick convergence rate, and SA is the second best as shown in Table 2. On the other hand, HS shows the lowest convergence rates with comparably low efficiencies. It appears that the HS algorithm requires a higher number of iterations to converge to the global solution. This may be because of the poor exploration ability of HS. This result implies that HS is potentially more efficient in solving low-dimensional optimization problems. it is clear that DE shows a superior performance and robustness in handling high dimensional optimization problems when the number of Iterations is limited.

Summary

This paper compares three optimization algorithms in terms of the required computation time to find the optimum value, using various benchmark functions with different numbers of design variables. The computation time needed to converge is the most significant factor for any technique when investigating its efficiency. The number of iterations required by each algorithm on a set of benchmark test functions places the computation time of the three optimization methods under investigation. The statistical results indicate that DE and HS need the lowest and highest average CPU time respectively. SA shows good efficiency most of the time; however, it can sometimes be stuck into local minima. The DE algorithm requires the lowest number of iterations, suggesting that DE is the most promising global optimization method while the number of iterations are limited. Furthermore, HS is less sensitive to increasing the number of variables than the other methods while its accuracy is less. Its clear that the DE algorithm found to be an efficient method for solving many GO problems is proved to be capable algorithm.

Reference

[1] Yang X S. Engineering optimization: an introduction with metaheuristic applications[M]. John Wiley & Sons, 2010.

[2] Samora, I., et al., Simulated annealing in optimization of energy production in a water supply network. Water resources management, 2016. 30(4): p. 1533-1547.

[3] Saad, A.E.H., Z. Dong, and M. Karimi, A Comparative Study on Recently-Introduced Nature-Based Global Optimization Methods in Complex Mechanical System Design. Algorithms, 2017. 10(4): p. 120.

[4] Dorigo, M. and L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation, 1997. 1(1): p. 53-66.

[5] Wu, G., et al., Superior solution guided particle swarm optimization combined with local search techniques. Expert Systems with Applications, 2014. 41(16): p. 7536-7548.

[6] Sun, C., et al., A two-layer surrogate-assisted particle swarm optimization algorithm. Soft computing, 2015. 19(6): p. 1461-1475.

[7] Shin, Y.-B. and E. Kita, Search performance improvement of particle swarm optimization by second best particle information. Applied Mathematics and Computation, 2014. 246: p. 346-354.

[8] Shan, S. and G.G. Wang, Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. Structural and Multidisciplinary Optimization, 2010. 41(2): p. 219-241

[9] Scaria, A., K. George, and J. Sebastian, An Artificial Bee Colony Approach for Multi-objective Job Shop Scheduling. Procedia Technology, 2016. 25: p. 1030-1037.

[10] Ritthipakdee, A., et al., Firefly Mating Algorithm for Continuous Optimization Problems. Comput Intell Neurosci, 2017. 2017: p. 8034573.

[11] Yatim, J., A. Zain, and N.E.N. Bazin, Cuckoo Search Algorithm for Optimization Problems—A Literature Review and its Applications. Vol. 28. 2014

[12] Al-Betar, M., et al., Bat-inspired Algorithms with Natural Selection mechanisms for Global optimization. 2017.

[13] Alam, D.F., D.A. Yousri, and M. Eteiba, Flower Pollination Algorithm based solar PV parameter estimation. Vol. 101. 2015.

[14] Kamboj, V.K., S.K. Bath, and J.S. Dhillon, Solution of non-convex economic load dispatch problem using Grey Wolf Optimizer. Neural Computing and Applications, 2016. 27(5): p. 1301-1316.

[15] Yusof, Y. and Z. Mustaffa, Time Series Forecasting of Energy Commodity using Grey Wolf Optimizer. Vol. 1. 2015. 25-30.

[16]b Singh, H.K., T. Ray, and W. Smith. Surrogate assisted simulated annealing (SASA) for constrained multi-objective optimization. in Evolutionary Computation (CEC), 2010 IEEE Congress on. 2010. IEEE.

[17] Das, S. and P.N. Suganthan, Differential evolution: A survey of the state-of-the-art. IEEE transactions on evolutionary computation, 2011. 15(1): p. 4-31.

[18] El-Fergany, A. and H. Hasanien, Single and Multi-objective Optimal Power Flow Using Grey Wolf Optimizer and Differential Evolution Algorithms. Vol. 43. 2015. 1548-1559.

[19] Yang, X.-S., Bat Algorithm: Literature Review and Applications. Int. J. Bio-Inspired Computation, 2013. 5(3): p. 141--149.

[20] Gu, J., G.Y. Li, and Z. Dong, Hybrid and adaptive meta-model-based global optimization. Engineering Optimization, 2012. 44(1): p. 87-104.