In Proceedings of the 22nd IEEE International Conference on Automation & Computing, University of Essex,Colchester city, UK, 7-8 September 2016
DOI: 10.1109/IConAC.2016.7604927

Benchmarking Heuristic Search and Optimisation Algorithms in Matlab

Wuqiao Luo
School of Engineering
University of Glasgow
United Kingdom
w.luo.1@research.gla.ac.uk

Yun Li
School of Engineering
University of Glasgow
United Kingdom
Yun.Li@glasgow.ac.uk

***Abstract*—With the proliferating development of heuristic methods, it has become challenging to choose the most suitable ones for an application at hand. This paper evaluates the performance of these algorithms available in Matlab, as it is problem dependent and parameter sensitive. Further, the paper attempts to address the challenge that there exists no satisfied benchmarks to evaluation all the algorithms at the same standard. The paper tests five heuristic algorithms in Matlab, the Nelder-Mead simplex search, the Genetic Algorithm, the Genetic Algorithm with elitism, Simulated Annealing and Particle Swarm Optimization, with four widely adopted benchmark problems. The Genetic Algorithm has an overall best performance at optimality and accuracy, while PSO has fast convergence speed when facing unimodal problem.**

Keywords—benchmarks; evolutionary algorithms; numerical optimization; single-objective

I. INTRODUCTION

During the past four decades, many heuristic algorithms have been developed. They are applicable to a wide range of real-world problems and are made available conveniently in Matlab, which is used daily by a significant number of practising engineers and scientists. With the development of these tools, it has become challenging to choose the most suitable ones for an engineer's application at hand. To provide a practical review of these tools, this paper conducts and reports their benchmarking tests.

Benchmarks used in this paper are consistent with but exceed those used in measuring conventional optimisation algorithms [1]. The tested algorithms from Matlab are the Nelder-Mead simplex method, the Genetic Algorithm (GA), a Genetic Algorithm with elitism (GAE), Simulated Annealing (SA) and Particle Swarm Optimization (PSO), which are the most representative and popular.

Section II of this paper presents the aforementioned benchmarks and Section III introduces the test functions. Benchmarking results are presented and analysed in Section IV. Conclusions are drawn in Section V.

II. UNIFORM BENCHMARKS FOR HEURISTIC TESTING

A. Optimisality

For an optimization problem with single objective, suppose that its objective function (performance index, cost function, or fitness function) is:

$$f(\boldsymbol{x}): \boldsymbol{X} \to F$$

where $\boldsymbol{X} \subseteq \boldsymbol{R}^n$ spans the entire search or possible solution space in n dimensions, $\boldsymbol{x} \in \boldsymbol{X}$ represents the n collective variables or parameters to be optimised, $F \in \boldsymbol{R}$ represents the space of all possible objective values.

The theoretical solve is usually reach at the minimum or maximum value of the objective function as:

$$f_0 = min\ or\ max\ \{f(\boldsymbol{x})\} \in F \quad (1)$$

An $\boldsymbol{x_0} \in \boldsymbol{X}$ that satisfies:

$$f(\boldsymbol{x_0}) = f_0 \quad (2)$$

is said to be a corresponding theoretical solution to the optimisation problem.

The uniform benchmarks used in this paper are defined as in [1]. Optimality represents the relative closeness (or, inversely, distance) of an objective found, $\hat{f}_0$, to the theoretical objective, f_0. It is defined as:

$$Optimality|_a = 1 - \frac{\|f_0 - \hat{f}_0\|_a}{\|\overline{f} - \underline{f}\|_a} \in [0,1] \quad (3)$$

where $\overline{f}$ and $\underline{f}$ are the lower and upper bounds of f.

B. Accuracy

Accuracy represents the relative closeness of a solution found, $\hat{\boldsymbol{x}}_0$, to the theoretical solution, $\boldsymbol{x}_0$. This may be particularly useful if the solution space is noisy, there exist multiple optima or "niching" is used. It may be defined as [1]:

$$Accuracy|_a = 1 - \frac{\|\boldsymbol{x_0} - \hat{\boldsymbol{x}}_0\|_a}{\|\overline{\boldsymbol{x}} - \underline{\boldsymbol{x}}\|_a} \in [0,1] \quad (4)$$

where $\underline{\boldsymbol{x}}$ is the lower bound of $\boldsymbol{x}$ and $\overline{\boldsymbol{x}}$ is the upper bound, representing the search range.

C. Convergence

a) Reach-time

The most frequently used benchmark for convergence is the number of evaluation times of function. The stop condition is usually set to when there is little change, less than 1e-6. But it could convergence very quickly ending at local extreme point. Hence the optimality is also evaluated in convergence as reach-time [1].

$$Reach - time|_b = C^b \quad (6)$$

Wuqiao Luo is grateful to the China Scholarship Council and the University of Glasgow for a CSC scholarship.

where represent the total number of 'function evaluations' conducted by which the optimality of the best individual first reaches $b \in [0, 1]$.

To estimate the order of the polynomial, $C^{0.999999}$ may be plotted against the number of parameters being optimized, *n*, as revised in:

$$NP - time(n) = C^{0.999999}(n) \quad (7)$$

b) Total number of evaluations

$$\min\{C^{0.999999}(n), 400n^2\} \quad (8)$$

which implies that a benchmark test should terminate either when the goal has been reached or *20n* generations with a size of *20nm* have been evolved.

D. Optimizer overhead

Alternative to or in addition to the 'total number of evaluations', the 'total CPU time' may be used in a benchmark test. More quantitatively, the optimizer overhead may be calculated by [1]:

$$Optimizer\ Overhead = \frac{Total\ time\ taken - T_{PFE}}{T_{PFE}} \quad (9)$$

where T_{PFE} is the time taken for pure function evaluations.

III. Benchmark Testing Functions

A. Experimental set-up

In this experiments, we investigated the performance of Simplex, GA, GAE, SA and PSO which are all searching methods provided in Matlab Toolbox.

Each experiment was repeated 10 times to get a mean value of benchmarks. For NM Simplex and SA, initial point is needs. A random starting point is given for all 10 runs in these two algorithms. In all runs, we kept a same criterial of stop searching as description in equation 8 to provide a fair performance comparison.

In order to get T_{PFE} for each tested function, all the benchmark functions (as in Table 2) are run 400n2 time and the time is takes as average value from 10 runs in Matlab. These data are given in Section IV as theoretical value of optimizer overhead.

B. Algorithmic settings

All the algorithms are tested in Matlab. For NM Simplex, reflection coefficient is 1, the expansion coefficient is 2, the contraction coefficient is 0.5 and the shrink coefficient is 0.5. For GA, the population size is set as 20n, which n is the dimension number. The max generation times is 20n, so that the maximum function evaluation is $400n^2$ as described in Section II about reach-time. The crossover function is using scattered which creates a random binary vector and selects the genes from two parents based on that binary vector. Crossover fraction is set as 0.8. Probability rate of mutation is 0.01. For GA with elitism, 5 out of 100 population are guaranteed to survive to the next generation. For SA, the initial temperature is set to 100, and the temperature is lowered following function of $T = T_0 \times 0.95^{50}$. For PSO, swarm size is 10n, and maximum iteration is set as 40n so that the maximum function evaluations is $400n^2$.

C. Benchmark functions

For comparison, four non-linear functions used in [2, 3] are used here in Table 1. All the functions are tested in 10 and 30 dimensions.Difference types of tested function are chosen for a comprehensive comparison. The first function is Quadric function, which is a unimodal function. The rest three functions are multimodal ones. The n-D Varying Lanscape problem, which is also the only maximization problem in the four tested functions, that was introduced by Michalewicz [4] and further studied by Renders and Bersini [5].

The objective function $f_2(x)$ is, in effect, de-coupled in every dimension represented by $f_i(x_i) = \sin(x_i)sin^{2m}(\frac{ix_i^2}{\pi})$. Every such member function is independent. The larger the product *mn* is, the sharper the landscape becomes. There are *n*! local maxima within the search space $[0, \pi]^n$. The theoretical benchmark solution to this *n*-dimensional optimization problem may be obtained by maximizing *n* independent uni-dimensional functions, $f_i(x_i)$, the fact of which is however unknown to an optimization algorithm being tested.

The third is Scaffer F6 function with minimum at zero. And the last one is a composition function of Schwefel's function, Rastrigin's function and High Conditioned Elliptic Function. The composition rate is 0.3, 0.3 and 0.4 respectively.

IV. Benchmarking Results and Analysis

A. Compare among algorithms

The results reported in this section are summarized in from Table 2 to Table 5 based on different tested function. All the theoretical values are given at the bottom of each table for comparison.

First function

The first test function is a unimodal function with only one minimum point in the search area $[-1.28\ 1.28]^D$. The searching results are shown in Table 2, Fig.1 and Fig.2. The algorithms are running for different dimension of 10 and 30.

All five tested algorithms had optimality up to 0.9999. In lower dimension of 10, NM Simplex and PSO showed better performance with lower optimization overhead and better accuracy. In 30 dimension problem, the optimizer overhead of GA and GAE decreased to the same level of Simplex, while the PSO had the best with 32.49%. SA has the most optimizer overhead for both 10-D and 30-D tests. Fig.1 and Fig.2 shows convergence trace for each algorithms. For f_1, all the algorithms show a fast and relatively uniformed speed. Overall, all the algorithms performant well for unimodal function. Simplex and PSO are most suitable for this kind of problem.

TABLE 1 TESTED FUNCTIONS

	Name of function	Test function	Search Space	Minimum/ Maximum

Unimodal	Quadric [6, 7]	$\min f_1(x) = \sum_{i=1}^{D} i x_i^4$	[-1.28,1.28]^D	D=10, $y \in [0, 147.6395]$ D=30, $y \in [0, 1248.2]$
Multimodal	n-D Varying Landscape [5, 8]	$max\ f_2(\boldsymbol{x}) = \sum_{i=1}^{n} f(x_i) = \sum_{i=1}^{n} \sin(x_i) sin^{2m}(\frac{i x_i^2}{\pi})$	$\boldsymbol{x} \in$ [0, pi]^D	D=10, $y \in [0, 9.6547]$ D=30, $y \in [0, 29.6252]$
	Scaffer's F6 Function [2, 3]	$minf_3(\boldsymbol{x})$=g(x_1, x_2) + g(x_2, x_3) + … + g(x_{D-1}, x_D) + g(x_D, x_1) $g(x, y) = 0.5 + \frac{(\sin^2(\sqrt{x^2 + y^2}) - 0.5)}{(1 + 0.001(x^2 + y^2))^2}$	$\boldsymbol{x} \in$ [-100,100]^D	D=10, $y \in [0, 9.9760]$ D=30, $y \in [0, 29.9280]$
	Composition Functions [2, 3]	$\min f_4 = 0.3 \times f_{41} + 0.3 \times f_{42} + 0.4 \times f_{43}$ $f_{41} = 418.9829 \times D - \sum_{i=1}^{D} g(z_i)$ $m = mod(\lvert z_i \rvert, 500)$ $g(z_i) = \begin{cases} z_i \sin(\lvert z_i \rvert)^{1/2} & if\ \lvert z_i \rvert \le 500 \\ (500 - m) \sin\left(\sqrt{\lvert 500 - m \rvert}\right) - \frac{(500 - z_i)^2}{10000D} & if\ z_i > 500 \\ (m - 500) \sin\left(\sqrt{\lvert 500 - m \rvert}\right) - \frac{(500 + z_i)^2}{10000D} & if\ z_i < -500 \end{cases}$ $f_{42} = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$ $f_{43} = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	$\boldsymbol{x} \in$ [-100,100]^D	D=10, $y \in [0, 5.5424e+08]$ D=30, $y \in [0, 1.2208e+09]$

Second function

The problem described in second function has multiple extremes in [0, pi]^D. The objective is to maximum the function value, which is different from the other three tested function.

From Table 3, NM Simplex performance are not satisfactory. It stalled at local maximum point. GA and GAE, on the other hand, showed steady performance, though it took much more time to reach the maximum point than NM Simplex. Also, GA with elitism shows a little faster than GA to reach to the stop criterial. But both GA and GA with elitism stop at local maximum point. Fig.3 and Fig.4 shows convergence trace for each algorithms. For f_2, Simplex and SA shows faster convergence speed at the beginning, but stall at local extremes. GA, GAE and PSO can get closer to global extreme with more iterations.

None of the tested algorithms can reach to 0.999999 optimality in N times of function evaluations. GA and GAE have provided relatively better results than other. Though PSO has a faster convergence speed.

Third function

Like the first and second functions, GA and PSO shows consistence better performance than other two algorithms. Fig.5 and Fig.6 shows convergence trace for each algorithms. For f_3, similar to the tests in f_2, Simplex and SA have quick convergence speed but soon stop at local extremes. PSO has a slower convergence speed than GA and GAE, but can reach to the same level of extreme point.

However, all the algorithms shows great overhead at this tested functions, which suggested that optimizer overhead is not only related to algorithms itself, but also related to the fitness function.

Fourth function

The hybrid function is complex which should be expected more optimizer overhead. However, except for Simplex and SA, the overhead of other three algorithms are less than 100%. The optimality all can reach to 0.9999, while the accuracy in Simplex is 64.48% in 30-D. Overall, all the five algorithms can locate the global extreme in the last tested function. Fig.5 and Fig.6 shows convergence trace for each algorithms. Fig.7 and Fig.7 shows convergence trace for each algorithms. For f_4,.convergence speed of SA is the slowest. Also because the random value for initial point, initial value of Simplex is quite large, while for GA and PSO, there are relative better point in the first population. Hence in this tested function, NM shows slower convergence speed than GA and PSO. Overall, GAE has a relative better performance.

Fig. 9 gives the average of three benchmarks, optimality, accuracy and optimizer overhead, between five algorithms. GA and GA with elitism have an overall steady and good performance. For optimality and accuracy, GA with elitism didn't show distinguished improvement. But it show steady improvement in optimizer overhead.

B. Compare between functions

Because f_1 is a unimodal function. All the algorithms are performing well. In 30-D, the optimizer overhead of PSO is only 32.49% because it only took less than 6% of set times of function evaluations to reach to stop point. For multimodal functions, heuristic methods have an overall better performance than deterministic method as NM Simplex.

From the theoretical time of pure running of fixed function evaluations, f_4 took most long time and f_3 is the shortest. However, the optimizer overhead of all the algorithms in f_3 is the biggest while small in f_1 and f_4. Moreover, for NM Simplex, the overhead is relatively stable in both 10-D and 30-D, while the optimizer overhead decreased at 30-D for heuristic methods. It is make sense that because GA and PSO are imitation of group activating in nature, which depends on big population to keep diversity. When the dimension increase, the population size is also increased.

C. Evaluations of benchmarks

The mean optimality and mean accuracy represent relative closeness to the theoretical value. It would lost meaning when the range for $\boldsymbol{x}$ and y become very big. The relative value would be too small to show difference between compared algorithms.

The minimum value for function is 0, however, the maximum value is increased expontially with dimension of n. In this case, the benchmark of optimality can't reflect the distant of searching result to theoretical value. When the distance between f_{max} and f_{min} is too big, the optimality could be very close to 1 even the absolute distance between $\hat{f}_0$ and f_0. As it is shown in Table 9 and Table 10. All the optimality can reach to 0.9998, but the mean function value is up to 10^5 scale in SA and Simple.

Though reach-Time or N evaluations of functions also has disadvantages, it gave a convergence speed reflected both convergence and optimality. But because of large range of f in f_4, PSO and GA stops before it reaches to its best solutions. Though the optimizer overhead in this test is very small, it is not the actual overhead time because algorithms stops early. For example, in Table 10, the times for function evaluation of PSO is only 45780, compare to N=360000. The results could be better if the search continue. But PSO stops search based on the reach time setting. Thus it compromised its performance to fit the testing standard.

Benchmarks of optimization, accuracy, convergence and optimizer overhead have shown some advantages for a uniformed standard to compare. Its application still has limits.

V. Conclusion

Using 4 commonly adopted tested functions, we have compared 5 algorithms available in the Matlab Optimization Toolbox. The Simplex has shown good performance for unimodal problems, but has not delivered satisfactory performance for multimodal and high dimension problems. The GA and GAE have shown overall consistently performance with all kinds of problems. The PSO has offered the highest convergence speed and relatively lower optimizer overhead, though the optimality and accuracy were not as good as the GA and GAE. If we mark the best one in each benchmarks in green in Table 2-7, the highest score in multimodal problems goes to GA and GAE. But for unimodal function, PSO has the best performance.

We have also evaluated the benchmarks and tested how good the benchmarks can reflect the search performance. Based on the analysis, discussions and examples shown in Sections 5 and 6, it is evident that more work is required on improving the existing benchmark measures. Whether it is possible to have such a set of benchmark measures that gives an accurate or distinguishable evaluation is still unknown. It is important to note that without proper and widely accepted benchmark measures, quantifying performance of heuristic algorithms is challenging.

TABLE 2 BENCHMARK TEST RESULTS ON THE 10-D AND 30-D QUADRIC PROBLEMS

Algorithms tested	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Score (green)
	10-D				30-D				-
Simplex	**100%**	**100%**	5766.6	118.09%	99.98%	**94.80%**	93779	104.84%	3
GA	**100%**	99.88%	19800	946.16%	**100%**	99.74%	134100	113.34%	2
GA with Elitism	**100%**	99.86%	18480	833.43%	**100%**	99.71%	122040	100.46%	2
SA	99.98%	94.49%	8871	9378.15%	99.88%	91.32%	30709	1315.49%	0
PSO	99.99%	97.34%	**4400**	**77.86%**	99.99%	96.62%	**19860**	**32.49%**	**4**
Theoretical value	***100%***	***100%***	***40000 Max***	***0.0691 sec***	***100%***	***100%***	***360000 Max***	***1.6532 sec***	-

TABLE 3 BENCHMARK TEST RESULTS ON THE 10-D AND 30-D VARYING LANDSCAPE FUNCTIONS

Algorithms tested	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Score (green)
	10-D				30-D				-
Simplex	20.70%	69.22%	**40000**	**342.49%**	22.99%	69.71%	**360000**	**169.96%**	5
GA	**95.38%**	**87.58%**	**40000**	1684.73%	**93.81%**	**87.59%**	**360000**	734.99%	**6**
GA with Elitism	95.29%	86.73%	**40000**	1618.96%	92.61%	86.69%	**360000**	704.96%	2
SA	33.62%	77.02%	**40000**	18206.36%	19.02%	75.61%	**360000**	2304.87%	2
PSO	86.10%	83.53%	**40000**	612.21%	72.80%	78.12%	**360000**	493.67%	2
Theoretical value	***100%***	***100%***	***40000 Max***	***0.0786sec***	***100%***	***100%***	***360000 Max***	***2.0406sec***	-

TABLE 4 BENCHMARK TEST RESULTS ON THE 10-D AND 30-D SCAFFER FUNCTION

Algorithms tested	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Score (green)
	10-D				30-D				
Simplex	55.78%	72.42%	**40000**	**809.26%**	53.06%	73.07%	**360000**	**878.64%**	4
GA	**89.21%**	**95.79%**	**40000**	3804.32%	**87.18%**	**94.03%**	**360000**	1835.67%	**6**
GA with Elitism	88.74%	95.26%	**40000**	3532.41%	83.68%	91.07%	**360000**	1712.33%	2
SA	62.26%	74.95%	**40000**	45367.90%	54.47%	68.36%	**360000**	2961.18%	2
PSO	84.01%	89.91%	**40000**	1386.73%	70.73%	82.18%	**360000**	1138.23%	2
Theoretical value	***100%***	***100%***	***40000 Max***	***0.0324 sec***	***100%***	***100%***	***360000 Max***	***0.7300 sec***	

TABLE 5 BENCHMARK TEST RESULTS ON THE 10-D AND 30-D HYBRID FUNCTION

Algorithms tested	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Mean Optimality	Mean Accuracy	N or Reach-Time	Optimizer Overhead	Score (green)
	10-D				30-D				
Simplex	99.99%	77.03%	10816	93.15%	99.98%	64.48%	360000	209.35%	0
GA	**100%**	99.88%	10400	170.76%	**100%**	99.62%	35700	35.85%	2
GA with Elitism	**100%**	**99.98%**	11680	186.20%	**100%**	99.79%	**32460**	**32.19%**	**5**
SA	99.99%	85.59%	27019	10573.30%	99.99%	86.09%	275690	6236.73%	0
PSO	99.99%	91.33%	**6560**	**49.84%**	99.99%	94.17%	45780	41.50%	2
Theoretical value	***100%***	***100%***	***40000 Max***	***0.2131 sec***	***100%***	***100%***	***360000 Max***	***4.4989 sec***	

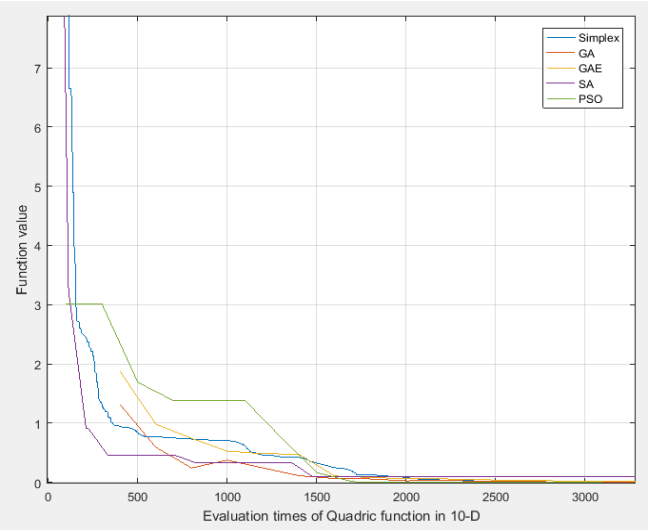

Figure 1 Convergence traces of tested algorithms in f_1, 10-D

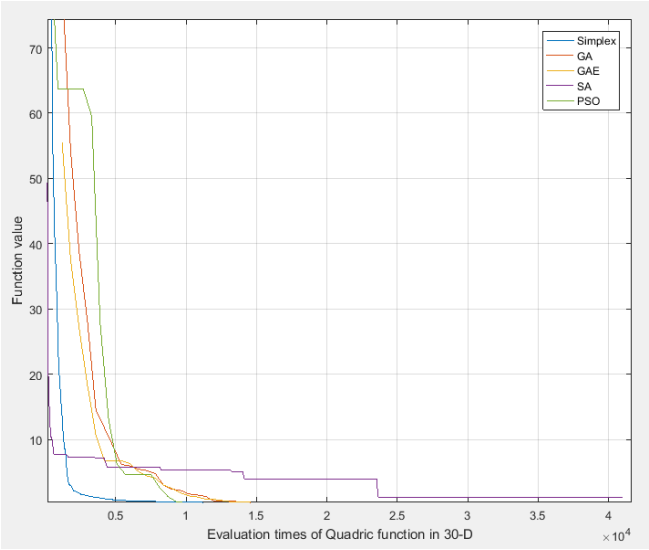

Figure 2 Convergence traces of tested algorithms in f_1, 30-D

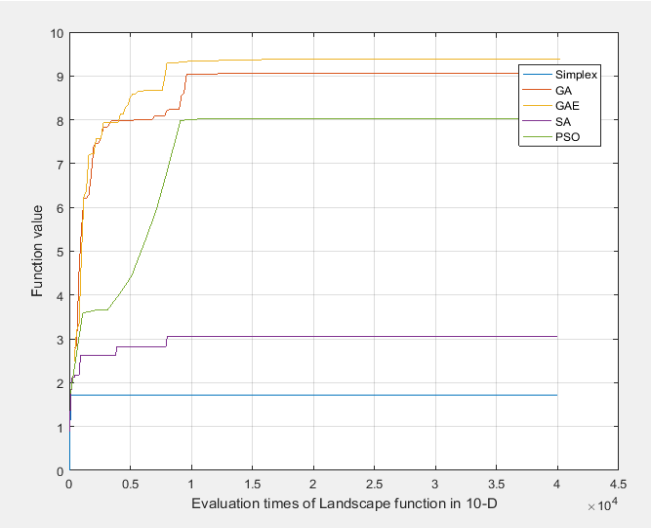

Figure 3 Convergence traces of tested algorithms in f_2, 10-D

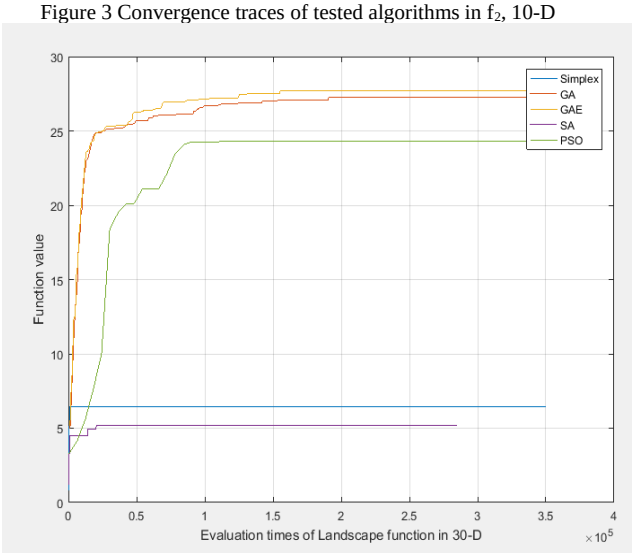

Figure 4 Convergence traces of tested algorithms in f_2, 30-D

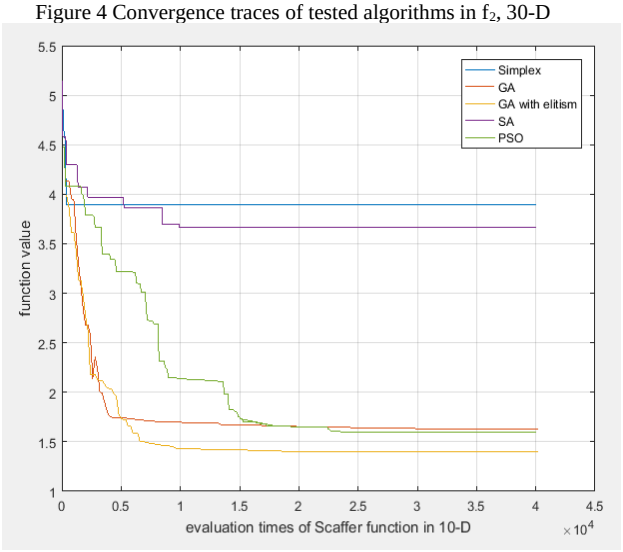

Figure 5 Convergence traces of tested algorithms in f_3, 10-D

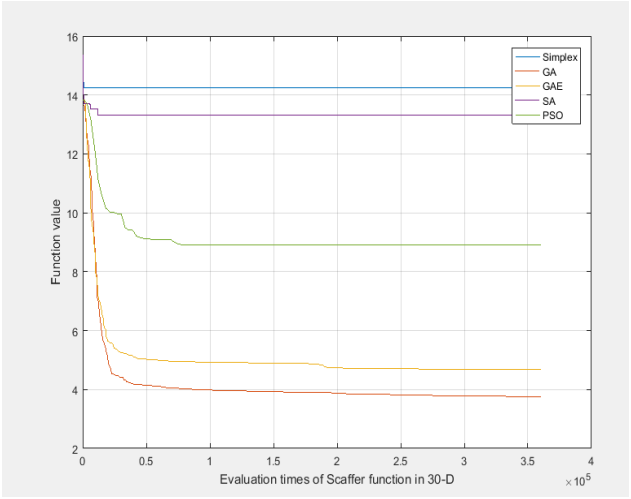

Figure 6 Convergence traces of tested algorithms in f_3, 30-D

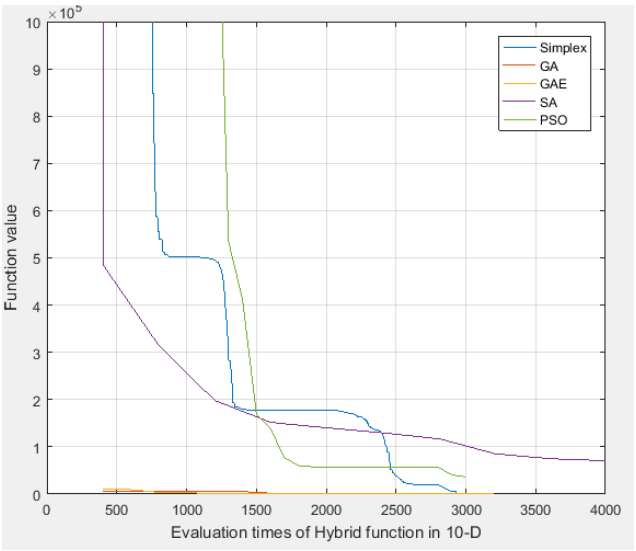

Figure 7 Convergence traces of tested algorithms in f_4, 10-D

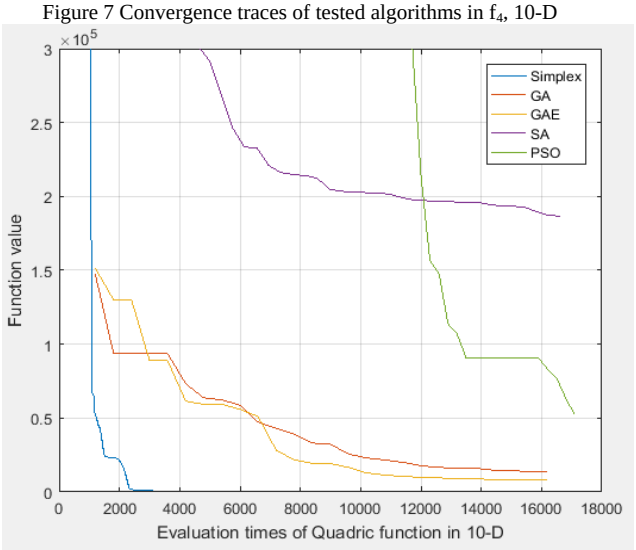

Figure 8 Convergence traces of tested algorithms in f_4, 30-D

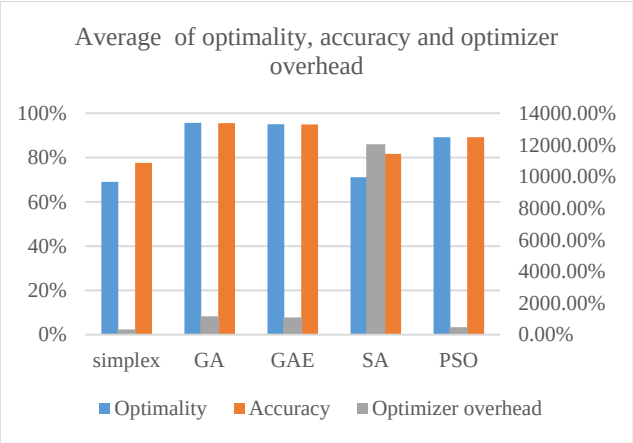

Figure 9 Average of optimality, accuracy and optimizer overhead of 4 tested functions.

REFERENCES

[1] W. Feng, T. Brune, L. Chan, M. Chowdhury, C. K. Kuek and Y. Li. "Benchmarks for testing evolutionary algorithms." in Asia-Pacific Conference on Control and Measurement. 134-138. 1998.

[2] Q. Chen, B. Liu, Q. Zhang, J. Liang, P. Suganthan and B. Qu, "Problem definitions and evaluation criteria for cec 2015 special session on bound constrained single-objective computationally expensive numerical optimization."

[3] J. Liang, B. Qu and P. Suganthan, "Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization," Computational intelligence laboratory, 2013.

[4] Z. Michelewicz, "Genetic algorithm+ data structure= evolutionary programs," New York: Springer—Verlag, 1996. 1: p. 996.

[5] J.-M. Renders and H. Bersini. "Hybridizing genetic algorithms with hill-climbing methods for global optimization: Two possible ways." in Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on. 312-317. 1994.

[6] Z.-H. Zhan, J. Zhang, Y. Li and H. S.-H. Chung, "Adaptive particle swarm optimization," Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 2009. 39(6): p. 1362-1381.

[7] X. Yao, Y. Liu and G. Lin, "Evolutionary programming made faster," Evolutionary Computation, IEEE Transactions on, 1999. 3(2): p. 82-102.

[8] Z. Michalewicz, Genetic algorithms+ data structures= evolution programs. 2013: Springer Science & Business Media.