EVALUATING THE EVOLUTIONARY ALGORITHMS - CLASSICAL PERSPECTIVES AND RECENT TRENDS

Swagatam Das
Indian Statistical Institute, Kolkata, India.

Keywords: Evolutionary algorithms, real parameter optimization, benchmark functions, variable linkage, statistical hypothesis testing.

Contents

1. General Introduction
2. Classical Numerical Benchmarks
3. General Guidelines for Designing Benchmark Problems
4. Modern Benchmark Suites
5. Experimental Conditions and Performance Measures
6. Statistical Test Procedures
7. Issues Related to Testing Evolutionary Algorithms on Real World Problems
8. Concluding Remarks
Glossary
Bibliography
Biographical Sketch

Summary

Evolutionary Computation (EC) algorithms are expected to be good black-box optimizers. Their performance should remain statistically appreciable on a wide range of or at least on some well-defined classes of optimization problems. Before an Evolutionary Algorithm (EA) can be published in a reputed journal, it usually needs to go through a number of tests to detect its strengths and weaknesses. Such investigation also includes the problem class to which the algorithm is most applicable and the problem characteristics that may deceive it from carrying out an effective search. Since the early days of research on and with EAs for real parameter optimization, a popular approach is to investigate their performances on a number of mathematical functions, also called benchmark functions, which are expected to capture various aspects of the complexities of the real world problems.

This chapter provides a comprehensive review of benchmarking EAs by using mathematical test functions. The chapter discusses the evolution of the benchmarking procedure itself along with the complexities and downsides of the modern day’s test problems. It also elaborates on the performance measures used for comparing the search abilities of various EAs. The discussion then proceeds to focus on the statistical methods currently in use to judge the significance of the results returned by an EA. The chapter is concluded with a few potential issues that need the attention of the EC researchers. The discussions of this chapter are mainly centered on EAs for solving single-objective box-constrained function optimization problems involving continuous variables.
1. Introduction

Generally speaking, optimization involves search for a vector of the form \( \bar{x} = [x_1, x_2, x_3, \ldots, x_n] \) which contains the parameters deciding some kind of system performance. Going by the name, each component \( x_j \) of the vector is a real number for real parameter or continuous optimization. The common practice (Bäck et al., 1997) is to model an objective function (also called cost function) that determines the system behavior and based on its value obtained iteratively, we are able to judge how far we have reached in our search for the best solution. It can be simply put that our task is to perform search in solution space in order to find a parameter vector \( \bar{x}^* \) which minimizes an objective function \( f(\bar{x}) \) \( \Omega \subseteq \mathbb{R}^D \rightarrow \mathbb{R} \), i.e. \( f(\bar{x}^*) < f(\bar{x}) \) \( \forall \bar{x} \in \Omega \), where \( \Omega \) is a non-empty set representing the search domain. If the function is convex and obeys some regularity assumptions, there exists a plethora of mathematical programming techniques (see for example (Boyd and Vandenberghe, 2004)) to solve the above mentioned problem. However, real world is not so easy going and very often we have to face optimization problems where the objective functions are non-convex, non-differentiable (the gradient tricks will not work!), rugged, ill-conditioned and so on. These features prohibit one to use the exact mathematical techniques. For getting a near optimal solution (which will work within a predefined level of tolerance) population-based EAs appear to be promising methodology. An EA uses mechanisms inspired by Darwinian evolution, such as reproduction, mutation, recombination, and selection (Eiben and Smith, 2003). Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the quality of the solutions. Evolution of the population then takes place through the repeated application of the above operators. The only feedback information that an EA uses to guide its population members is the evaluation of the function to be optimized at a set of trial points. The challenge is to obtain an acceptable (the acceptability is again very much dependent on the optimization scenario!) solution by using minimum number of evaluations of the objective function. A few very prominent EAs of current interest include Differential Evolution (Storn and Price, 1997; Das and Suganthan, 2011; Mallipeddi et al., 2011), Covariance Matrix Adaptation Evolution Strategies (CMA-ESs) (Hansen and Auger, 2013) real coded Genetic Algorithm with Multi-Parent Crossover (GA-MPC) (Elsayed et al., 2011), Estimation of Distribution Algorithms (EDAs) (Dong et al., 2013) etc.

Design of an EA primarily involves devising useful genetic operators (various kinds of mutation, recombination, and selection). Proposals for devising a new EA or deriving the improved version of an EA are numerous in literature and they are primarily based on some kind of intuitionistic reasoning instead of mathematical proofs. Although there are a few studies (Jägerskühper, 2007; Akimoto et al., 2012) that show the local linear convergence or global convergence, obtaining the speed of convergence of an EA mathematically is quite difficult and sometimes impossible. Moreover, studying the transition phase of the algorithm is important in practice, while investigating the transition phase mathematically seems more complicated. So what may be the way out to establish the effectiveness of an EA? The only way seems to perform numerous experiments by running the EA on various classes of functions and comparing the results against the state-of-the-art EAs. Since the early days of research on EAs, a
popular practice is to run an EA on a set of numerical functions that may be relied on as a representative set to capture most of the complex features arising from real life optimization scenarios. These functions usually stem from very old mathematical functions existing in the literatures on numerical methods. An EA is usually run on such a function till the exhaustion of a fixed budget of the number of Function Evaluations (FEs) and then the best-of-the-run function value is noted. As will be discussed shortly, there may be other stopping criteria for an EA and these may also serve as the basis of comparison between two different EAs. Since EAs start with a randomly initiated population and incorporate stochastic search operators (those coming with various random numbers), results of running an EA repeatedly on the same problem are not identical in practice. Hence a set of independent runs are taken on each function and the mean best-of-the-run values and standard deviations are usually reported. Less standard deviation indicates greater robustness of the EA as the results do not deviate much over the repeated runs.

A possible shortcoming with such empirical procedure is that the resulting conclusions depend quite heavily on what problems are used for testing and nevertheless on the algorithms that are being compared. This could result in a set of algorithms that are designed and tuned to perform well on a particular test suite. However, the resulting specialization may or may not translate into improved performance on other problems or applications. Thus, it is imperative that benchmarking suites should contain problems that are both challenging and diverse. Test problems can be designed to be easy to describe, understand, and visualize. They are also easy to implement, fast, and their optima are often known in advance. The need is to have a class of benchmarking functions that are well-understood. In addition, appropriate performance measures should be employed and suitable statistical methods should be used for drawing solid conclusions.

In this chapter we begin with a review of the classical and modern numerical benchmark functions used to evaluate EAs and various nature inspired metaheuristics. We then discuss on various performance measures used in literature to judge the merits of an EA. We also briefly elaborate on the statistical test procedures adopted for comparing among various EAs. Finally we point out some issues that demand attention from EC researchers to meet the challenges of the ever-growing and rapidly changing field of engineering optimization.

2. Classical Numerical Benchmarks

One of the first attempts to establish a set of problems to test the performance of the EAs was due to De Jong in his Ph.D. dissertation (De Jong, 1975). The De Jong test suite contained a set of five problems with varied characteristics which were used to test the effectiveness of the Genetic Algorithms (GAs). GAs at that time were mostly based on binary encoding for the search variables. In those days, real numbers were represented by bit strings comprising sequences of ‘0’ and ‘1’ and this would invariably incur into quantization errors in context to function optimization. The functions and their characteristics are summarized in Table 1. The first function of De Jong’s or Sphere function \( f_1 \) is one of the most simple test functions available in literature. This continuous, convex, unimodal and additively separable function can be scaled up to any
number of variables. It belongs to a family of functions called quadratic functions and has only one global minimum at \( \hat{X} = [0,0,...,0]^T \). The second function, called the generalized Rosenbrock’s function \( (f_2) \) is a challenging one, especially in higher dimensions. It has a very narrow ridge. The tip of the ridge is very sharp, and it runs around a parabola. Some classical EAs are not able to discover good directions and consequently underperform in this problem. This function behaves as unimodal in two dimensions but becomes multimodal for more than 3 dimensions (Shang and Qiu, 2006). The third function is called the step function \( (f_3) \) and it represents the problem of flat surfaces. Flat surfaces are obstacles for optimization algorithms, because they do not give any information as to which direction is favourable. Unless an algorithm has variable step sizes, it can get stuck on one of the flat plateaus. The fourth function, known as quartic \( (f_4) \), is a simple unimodal function padded with noise. The Gaussian noise ensures that the algorithm never gets the same value on the same point. Algorithms that do not perform well on this test function are expected to yield poor results on other functions mixed with noise terms. The fifth one, called the Shekel’s foxholes function \( (f_5) \) is an example of many (in this case 25) local optima. Many standard EAs get stuck in the first peak they find.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Range</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1(\hat{X}) = \sum_{i=1}^{D} x_i^2 )</td>
<td>( x_i \in [-5.12,5.11] )</td>
<td>Unimodal function, originally with ( D = 3 )</td>
</tr>
<tr>
<td>( f_2(\hat{X}) = \sum_{i=1}^{D} \left( 100 \left( x_i^2 - x_{i+1} \right) + (1-x_i)^2 \right) )</td>
<td>( x_i \in [-2.048,2.047] )</td>
<td>Nonlinear function, originally defined in 2 dimensions</td>
</tr>
<tr>
<td>( f_3(\hat{X}) = \sum_{i=1}^{D} \left</td>
<td>x_i \right</td>
<td>)</td>
</tr>
<tr>
<td>( f_4(\hat{X}) = \sum_{i=1}^{D} x_i^4 + \text{Gauss}(0,1) )</td>
<td>( x_i \in [-1.28,1.27] )</td>
<td>Noisy function, Gauss(0,1) denoting a normally distributed random number in (0,1). ( D ) is usually 30.</td>
</tr>
<tr>
<td>( f_5(\hat{X}) = \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})} )</td>
<td>( x_i \in [-65.536,65.536] )</td>
<td>Multimodal 2-dimensional function with several local optima with coefficients ( (a_i) = [-32, -16, 0, 16, 32, -32, -16, 0] )</td>
</tr>
</tbody>
</table>

Table 1. De Jong’s Test Suite (De Jong, 1975). \( D \) denotes the dimensionality of the problems.

Other test sets have been subsequently proposed in (Ackley, 1987), (Davidor, 1991), (Forrest and Mitchell, 1993), (Mühlenbein, 1991), and (Schaffer et al., 1989). Five other popular test functions have been listed in Table 2. Among these the Rastrigin’s function \( (f_6) \), Schwefel’s function \( (f_7) \), Ackley’s function \( (f_8) \), and Griewank’s function \( (f_9) \) can be scaled to any number of variables.
The Rastrigin’s Function is a typical example of non-linear multimodal function. It has several local optima arranged in a regular lattice, but it has only one global optimum. It was first proposed by Rastrigin (Törn and Zilinskas, 1989) as a two-dimensional function and was further generalized for higher dimensions by (Mühlenbein et al., 1991). This function poses a fairly difficult problem due to its large search space and its large number of local minima.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Range</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_6(\vec{X}) = (D*10) + \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i))$</td>
<td>$x_i \in [-5.12, 5.11]$</td>
<td>Rastrigin’s Function: scalable, additively decomposable and multimodal</td>
</tr>
<tr>
<td>$f_7(\vec{X}) = \sum_{i=1}^{D} (-x_i \sin[\sqrt{</td>
<td>x_i</td>
<td>}])$</td>
</tr>
<tr>
<td>$f_8(\vec{X}) = -20 \exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20$</td>
<td>$x_i \in [-30, 30]$</td>
<td>Ackley’s function: scalable, huge number of local minima with one sharp and symmetrically located global minimum</td>
</tr>
<tr>
<td>$f_9(\vec{X}) = 1 + \sum_{i=1}^{D} \frac{x_i^2}{4000} + \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right)$</td>
<td>$x_i \in [-5.12, 5.11]$</td>
<td>Griewank’s Function: scalable multimodal function</td>
</tr>
<tr>
<td>$f_{10}(\vec{X}) = (x_i^2 + x_j^2)^{0.25}\left(\left[\sin(50(x_i^2 + x_j^2)^{0.1})\right]^2 + 1\right)$</td>
<td>$x_i \in [-100, 100]$</td>
<td>two-dimensional multimodal and non separable function</td>
</tr>
</tbody>
</table>

Table 2. Other common test functions for evaluating evolutionary algorithms

The Schwefel’s function (Schwefel, 1981) is symmetric, separable, and multimodal. This function is deceptive in that the global minimum is geometrically distant, over the parameter space, from the next best local minima. Therefore, the search algorithms are potentially prone to convergence in the wrong direction. In addition, it is less symmetric than the Rastrigin’s function and has the global minimum at the edge of the search space. Additionally, there is no overall guiding slope towards the global minimum like in Ackley's, or less extremely as in the Rastrigin's function.

The Ackley’s function is a continuous, multimodal, and separable function. It is obtained by modulating an exponential function with a cosine wave of moderate amplitude (Ackley, 1987). Originally this problem was defined for two dimensions, but the problem has been generalized to higher dimensions later (Bäck, 1996). The Ackley’s is a highly multimodal function that has huge number of local minima but only one global minimum. Its topology is characterized by an almost flat (due to the dominating exponential) outer region and a central hole or peak where the modulations by the cosine wave become more and more influential. The functional landscape is riddled with several local optima that, for the search range, look more like noise, although they are located at regular intervals.
Figure 1. 3-D map for 2-D functions of $f_1 - f_5$ (De Jong’s functions) of Table 1.
(a) Rastrigin’s function $f_6$

Figure 2. (a) Rastrigin’s function $f_6$ 3-D map for 2-D functions $f_6 - f_{10}$ of Table 2.

(b) Schwefel’s function $f_7$

(c) Ackley’s function $f_8$

(d) Griewank’s function $f_9$

(e) Stretched sine wave function $f_{10}$

Figure 2(c) Ackley’s function $f_8$ 3-D map for 2-D functions $f_6 - f_{10}$ of Table 2.

(d) Griewank’s function $f_9$ 3-D map for 2-D functions $f_6 - f_{10}$ of Table 2.

(e) Stretched sine wave function $f_{10}$ 3-D map for 2-D functions $f_6 - f_{10}$ of Table 2.
Griewank’s function (Griewank, 1981; Locatelli, 2003) is similar to the function of Rastrigin. It has many widespread local minima regularly distributed. The function interpretation changes with the scale; the general overview suggests convex function, medium-scale view suggests existence of local extrema, and finally zooming on the details indicates complex structure of numerous local extrema. It has been found that the summation term of the function induces a parabolic shape while the cosine function in the product term creates waves over the parabolic surface; these waves give rise to local optima (Whitley et al., 1996). A quick study of the low-dimensional versions of this function shows that the basin of attraction containing the global optimum appears to encompass a larger percentage of the total space as the search volume expands. The contribution of the product term becomes smaller as the dimensionality of the search space is increased and the local optima induced by the cosine term become less severe. Thus, as the dimensionality of the search space enhances, this function becomes easier for numeric real-valued representations. From a testing point of view, this characteristic of the Griewank’s function makes it undesirable at higher dimensions. The function labeled as \( f_{10} \) in Table 2 is known as the stretched sine wave function (Schaffer et al., 1989). Functions \( f_1, f_3, f_5, f_6, \) and \( f_7 \) are seen to be examples of separable functions. On the other hand, although \( f_4 \) is separable, the addition of noise might prevent an EA from locating the optimal solution. The coordinate wise search algorithms (Box et al., 1969) can be used to solve such functions as they exploit the separability by solving for each parameter independently. Three dimensional maps of the functions listed in Tables 1 and 2 have been shown in Figures 1 and 2 respectively.

Yao et al. compiled a set of 23 functions in their 1999 study on the Fast Evolutionary Programming (FEP) (Yao et al., 1999) which included all the previously mentioned test functions. This set has been used either in parts or as a whole later on in numerous papers to compare among various EAs, see for example (Leung and Yuping, 2001), (Lee and Yao, 2004), (Das et al., 2009).

It is often possible that the algorithms tested on the above test problems can become customized for that particular see the particular benchmark suite. This can raise some serious concerns when the test suite contains problems that do not reflect nature of the problems that EAs normally used to solve.

3. General Guidelines for Designing Benchmark Problems

The problems contained in test suites should be indicative of the types of applications for which an EA is appropriate. For example, it would be inappropriate to test heuristic search algorithms on a test suite made up of only linear functions, since most of the real-world problems are rarely linear in nature. There are certain characteristics that a good benchmark suite should possess. A test suite must contain problems with diverse ranges of difficulty and structure. Such a suite should also comprise of problems that are multimodal, sparse, and non-separable. In addition, suitable performance evaluation metrics should be employed to reveal the applicability of the algorithm in different situations. The presence of a few unimodal instances helps to test the convergence speed of the algorithm (Chapter 14, Eiben and Smith, 2003).
Differential Evolution (DE): A powerful derivative free optimization algorithm that works by maintaining a population of candidate solutions and creating new candidate solutions by combining the existing ones through a simple difference vector based formulae, and then keeping whichever candidate solution has the best score or fitness on the optimization problem at hand.

Particle Swarm Optimization (PSO): A metaheuristic optimization algorithm that imitates the collectively intelligent behavior of the group of social creatures like school of fish or flock of birds; works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position.

Hypothesis testing: Refers to the process of choosing between competing hypotheses about a probability distribution, based on observed data from the distribution. It is a core topic in mathematical statistics, and indeed is a fundamental part of the language of statistics.

Bibliography


Barrera J., Coello C. A. C. (2011). Test function generators for assessing the performance of PSO algorithms in multimodal optimization, Handbook of Swarm Intelligence, ALO 8, pp. 89–117. Springer-Verlag Berlin Heidelberg [This chapter reviews several test function generators that have been used over the years for assessing the performance of the particle swarm optimizers].

pp. 666-685 [This paper proposes a bi-objective differential evolution algorithm to identify different niches on a multimodal fitness landscape].


C. Y. Lee and X. Yao, “Evolutionary programming using the mutations based on the L'evy probability distribution”, *IEEE Transactions on Evolutionary Computation*, 8(1), pp.1-13, Jan. 2004 [This paper presents a novel evolutionary programming variant based on the L'evy probability distribution and tests the same on a number of numerical benchmarks].


Das S., Suganthan P. N. (2011). Differential evolution – a survey of the state-of-the-art, *IEEE Transactions on Evolutionary Computation* 15(1): 4-31 [This paper provides a comprehensive and exhaustive survey on the DE family of algorithms, their several variants, and their applications to diverse domains]


De Jong K. (1975) *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*, Ph.D. Thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI [This thesis, the first of its kind, describes the behavior of a class of genetic adaptive systems analytically].

Derrac J., Garcia S., Hui S., Herrera F. (2013). Statistical analysis of convergence performance throughout the evolutionary search: A case study with SaDE-MMTS and Sa-EPSDE-MMMS. *IEEE Symposium on Differential Evolution*, pp. 151 - 156, Singapore [This work investigates a non-parametric procedure, the Page test, to detect significant differences between the evolutions of the error of two evolutionary algorithms as the search continues].

and Evolutionary Computation, vol. 1, pp. 3–18 [This paper provides a survey of the complete set of nonparametric procedures developed to perform both pair-wise and multiple comparisons, for multi-problem analysis with respect to swarm and evolutionary computing algorithms].

Dong W., Chen T., Tino P., Yao X. (2013). Scaling up estimation of distribution algorithms for continuous optimization, *IEEE Transactions on Evolutionary Computation*, DOI: 10.1109/TEVC.2013.2247404. [This article proposes a modified and improved estimation of distribution algorithm for solving high dimensional function optimization problems]


Eiben A. E., Smith J. E. (2003). *Introduction to Evolutionary Computation*, Springer [This is a text book covering several important topics of the theoretical and practical aspects of the evolutionary algorithms].


paper proposed a genetic algorithm called the orthogonal genetic algorithm with quantization for global numerical optimization with continuous variables].

Li J.-P., Balazs M. E., Parks G. T., Clarkson P. J. (2002). A species conserving genetic algorithm for multimodal function optimization, *Evolutionary Computation* 10 (3), pp. 207–234 [This paper presents a niching technique called species conservation for evolving parallel subpopulations on multimodal fitness landscapes].


Locatelli M. (2003). A note on the Griewank test function, *J. Global Opt.*, 25, 169-174 [This paper analyzes the behavior of the widely used Griewanks function and proposed some modifications to make this function more challenging as an optimization benchmark].


Picek S., Golub M., Jakobovic, D. (2012) On the analysis of experimental results in evolutionary computation, MIPRO 2012, pp. 1064-1069 [This paper provide an overview of the possible approaches in the experimental analysis and describes statistical methods that could be used to test and compare evolutionary algorithms].

Qu B. Y. and Suganthan P. N. (2010). Novel multimodal problems and differential evolution with ensemble of restricted tournament selection, *IEEE Cong. on Evol. Comput*. Barcelona, Spain [This paper introduced a number of scalable multimodal benchmark functions for testing evolutionary niching algorithms].


Törn A., Zilinskas. A. (1989). *Global Optimization*. Lecture Notes in Computer Science, No. 350, Springer-Verlag, Berlin [This book serves as a comprehensive treatise on the global optimization algorithms that are concerned with finding the global extremum (maximum or minimum) of a mathematically defined function (the objective function) in some region of interest].


Whitley, L. D., Rana, S., Dzubera, J., Mathias, K. E., (1996). Evaluating evolutionary algorithms, *Artificial Intelligence*, 85, pp. 245-276, 1996 [This paper was the first of its kind to review and modify the test suites available to evaluate the evolutionary search algorithms].


Yao X., Liu Y., Lin, G. (1999). Evolutionary programming made faster, *IEEE Transactions on Evolutionary Computation*, 3(2), 82-102 [This paper proposed a modified version of the evolutionary programming algorithm and also tested the proposed variant on a collection of 23 numerical benchmark functions].

**Biographical Sketch**

**Swagatam Das** is currently serving as an assistant professor at the Electronics and Communication Sciences Unit of Indian Statistical Institute, Kolkata. His research interests include evolutionary computing, pattern recognition, multi-agent systems, and image processing. Dr. Das has published one research monograph, one edited volume, and more than 150 research articles in peer-reviewed journals and international conferences. He is the founding co-editor-in-chief of “Swarm and Evolutionary Computation”, an international journal from Elsevier. He serves as associate editors of the IEEE Trans. on Systems, Man, and Cybernetics: Systems, IEEE Computational Intelligence Magazine, IEEE Access, Neurocomputing, Information Sciences, and Engineering Applications of Artificial Intelligence. He is an editorial board member of Progress in Artificial Intelligence (Springer), Mathematical Problems in Engineering, International Journal of Artificial Intelligence and Soft Computing, and International Journal...
of Adaptive and Autonomous Communication Systems. He has been associated with the international program committees and organizing committees of several regular international conferences including IEEE WCCI, IEEE SSCI, SEAL, GECCO, and SEMCCO. He has acted as guest editors for special issues in journals like IEEE Transactions on Evolutionary Computation, ACM Transactions on Adaptive and Autonomous Systems, and IEEE Transactions on SMC, Part C. He is the recipient of the 2012 Young Engineer Award from the Indian National Academy of Engineering (INAE).