MANCHESTER 1824
The University of Manchester

The University of Manchester Research

Global Optimization with Derivative-free, Derivative-based and Evolutionary Algorithms

DOI:
10.1109/SMC.2014.6973891

Link to publication record in Manchester Research Explorer

Citation for published version (APA):
Bashir, H. A., & Neville, R. S. (2014). Global Optimization with Derivative-free, Derivative-based and Evolutionary Algorithms. In *The 2014 International Conference on Systems, Man, and Cybernetics* (pp. 100-105). IEEE. https://doi.org/10.1109/SMC.2014.6973891

Published in:
The 2014 International Conference on Systems, Man, and Cybernetics

Citing this paper
Please note that where the full-text provided on Manchester Research Explorer is the Author Accepted Manuscript or Proof version this may differ from the final Published version. If citing, it is advised that you check and use the publisher's definitive version.

General rights
Copyright and moral rights for the publications made accessible in the Research Explorer are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Takedown policy
If you believe that this document breaches copyright please refer to the University of Manchester's Takedown Procedures [http://man.ac.uk/04Y6Bo] or contact uml.scholarlycommunications@manchester.ac.uk providing relevant details, so we can investigate your claim.

OPEN ACCESS

Download date:08. Jun. 2022

2014 IEEE International Conference on Systems, Man, and Cybernetics
October 5-8, 2014, San Diego, CA, USA

Global Optimization with Derivative-free, Derivative-based and Evolutionary Algorithms

Hassan A. Bashir
School of Computer Science
University of Manchester, UK
Email: habash1600@gmail.com

Richard S. Neville
School of Computer Science
University of Manchester, M13 9PL, UK
Email: r.neville@manchester.ac.uk

***Abstract*—This paper investigates global optimization methods from the perspective of population-based and restarted point-based heuristics. We examine the performance of a standard evolutionary computation (EC) methodology, a derivative-based sequential quadratic programming (SQP) algorithm and a novel derivative-free stochastic coordinate ascent (SCA) algorithm. All methods are analyzed by random sampling of the feasible search space. A comparison was made to evaluate the three algorithms, in the light of newly updated IEEE CEC2013 benchmarks, on a set of *multimodal* and *composite* test cases. Results revealed that while the standard EC algorithm is generally more robust, on the basis of convergence efficiency both the restarted SCA and SQP algorithms have shown remarkable performance on some of these benchmarks. The results further suggest that depending on the nature of the problem landscape and dimensionality the three algorithms, chosen from different optimization frameworks, perform complementary to each other.**

I. Introduction

Optimization enables searching for optimal solutions to facilitate effective decision making in practice. Evolutionary optimization in particular has diverse application areas [1], including problems in but not limited to continuous, discrete (i.e. combinatorial), temporal and multiobjective domains.

Optimization problems involve multiple and often conflicting design requirements. Consequently, the search for an optimum solution is challenging. Fundamental differences exist in the nature of the typical solution space (landscape) of a local optimization problem and that of a global one (cf. Fig 1). In a local landscape (Fig. 1a), only one optimal solution exists, but such problems are mostly idealistic. A global landscape (Fig. 1b), however, involves a combination of many local and global optimal solutions. In fact, the majority of practical optimization problems are of the global type. Regardless, optimization methods are expected to find the global optimum solution with minimum possible computational cost.

Several solution techniques exist for the different types of the aforementioned optimization problems. Two main categories are the numerical and heuristic methods. This paper aims to investigate the behavior of *three* different categories of optimization methods, namely restarted derivative-free, restarted derivative-based and evolutionary algorithms, on global optimization benchmarks. Based on robustness and convergence efficiency, we empirically compare and analyze the performance characteristics of three algorithms:

1) a standard *evolutionary computation* (EC) algorithm;
2) a *sequential quadratic programming* (SQP); and
3) a novel *stochastic coordinate ascent* (SCA) algorithm

on a variety of optimization landscapes.

On a broad scope, the first contribution of this paper is a 'high-level' classification – in form of a *research relevance tree* – of the various global optimization methodologies in Section II. In Section III we first introduce the standard genetic algorithm (GA) which features the standard elitism replacement method, Section III then introduces the derivative-based SQP algorithm. Also in Section III, this paper proposes a novel stochastic coordinate ascent derivative-free algorithm. In Section IV this paper empirically compares and analyzes the three optimization methodologies on two sets of global optimization benchmarks built for the CEC2013 competition on real-parameter optimization. The paper concludes with remarks on further research in Section V.

II. Background: Optimization Methods

The research relevance tree in Fig. 2 classifies the global optimization methods into two broad categories, namely population-based and restarted point-based methods. Note that the restarted methods are also called *multi-start* approaches [2]. While the population-based approaches span the nature and non-nature inspired heuristics, majority of the restarted methods are numerical and have their origin in mathematical programming (MP) and operations research (OR) [3].

The evolutionary algorithms (EAs), such as the genetic algorithms (GA), genetic programming (GP) and evolutionary strategy (ES), are only a few examples of the many naturally inspired population-based heuristics. See [4], [5] for extensive survey on evolutionary computation methods and applications.

For the population-based methods (cf. Fig. 2), since this study is focused on evolutionary computation, a genetic algorithm which is one of the most widely used EAs is utilized. Detail principle of GA and its mathematical model is presented in Section III-A. For the restarted point-based methods (cf. Fig. 2) however, a sequential quadratic programming algorithm and a stochastic coordinate ascent algorithm are used. While the SQP algorithm is a 2nd order derivative-based method (Section III-B), the newly proposed SCA algorithm is derivative-free (Section III-C). Therefore, this paper investigates global optimization with these three algorithms as representatives of the various optimization frameworks depicted in Fig. 2.

978-1-4799-3840-7/14/$31.00 ©2014 IEEE

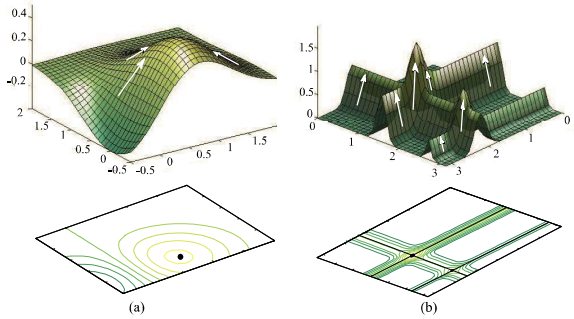

Fig. 1. An illustration of local and global optimization landscapes. (a) A local optimization problem (top) has only a single optimal solution (bottom). (b) Global optimization problem (top) with many optimum solutions and ridge-like plateaus (bottom).

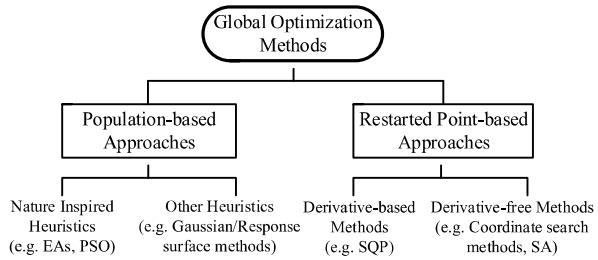

Fig. 2. A Research Relevance Tree showing a Classification of Global Optimization Frameworks.

III. Population-based and Restarted Point-based Heuristics

This section presents the three algorithms which are later evaluated on global optimization problems in Section IV.

A. An Evolutionary-based Method – The Genetic Algorithm

The standard genetic algorithm (GA) [6] is an iterative procedure (Algorithm 1) that evolves a pool of candidate solutions across generations t. GA begins with a fixed sized initial population $P(t) : |P(t)| = N$ (line 2). The sample solutions in the initial population are usually created randomly within the feasible search space. At every generation (lines 4-11), a stochastic selection process (line 5) is applied on the initial population to choose better solutions following an evaluation that is based on some measures of fitness. Evolutionary variation operators (crossover and mutation) are then applied (lines 6-7), at their respective probabilities P_c and P_m, to create an offspring pool; fitter samples are then selected from the parent and offspring pools and the process is repeated until termination condition is reached (line 4).

Notice from Algorithm 1 that, at any generation t, the parameters: $P(t), Q_s(t), Q_r(t)$ and $Q_m(t)$ respectively represent the data structures for the population at the initial generation, at the end of selection, and after the recombination and mutation operations. $\mathcal{G}$ and $\mathcal{P}$ denote the genotype and phenotype spaces respectively. Specific parameterizations for the GA used in this paper are provided later in Section IV.

Algorithm 1 A Standard Model of Genetic Algorithm

1: set counter $t \leftarrow 0$;
2: initialize $P(t) : P(t) = \{x_i | x_i \in \mathcal{P}\}$;
3: evaluate the fitness of $P(t)$;
4: **while** *not termination* **do**
5: $Q_s(t) \leftarrow$ select from $P(t)$;
6: $Q_r(t) \leftarrow$ recombine $Q_s(t)$;
7: $Q_m(t) \leftarrow$ mutate $Q_r(t)$;
8: evaluate the fitness of $Q_m(t)$;
9: $P(t+1) \leftarrow$ select from $\{Q_m(t) \cup P(t)\}$;
10: $t \leftarrow t + 1$;
11: **end while**

B. A Derivative-based Method – The SQP Algorithm

Besides the above evolutionary computation methods, many other numerical optimization methods [7] with origin in MP are also in common use. In this section we introduce the sequential quadratic programming (SQP) algorithm which is inherently a local optimization method, but in this paper it is put into a restart framework to enable global search. SQP is a line-search based quasi-Newton method that is typically applied to nonlinear optimization problems of the form

$$\text{maximize} \quad f(x) : x \in \quad ^n; n \geq 1, \tag{1}$$

where n is the dimensionality of the problem's search domain. In SQP, the iterative progression relied upon the computed search direction $d_t \in \quad ^n$ and the evaluated step size parameter $\alpha_t : \{\alpha_t \in \quad : 0 < \alpha_t \leq 1\}$, such that

$$x_{t+1} = x_t + \alpha_t d_t, \tag{2}$$

where the search direction d_t is obtained by setting the gradient of the 2nd order Taylor approximation of (1) to zero at every iteration t, and solving for d such that:

$$d = -\frac{\nabla f(x)}{\nabla^2 f(x)^T} = -[H(x_k)]^{-1} \nabla f(x). \tag{3}$$

Note that $H = \nabla^2 f \in \quad ^{n \times n}$ is the Hessian of the objective function $f(x)$ in (1).

SQP algorithm works in two phases which consist of an outer (linearization) phase and an inner (optimization) phase. At any given search point x_t, the linearization phase approximates the original (nonlinear) objective function (1) with a quadratic model; and the nonlinear constraints (if any) with their approximate linear expressions. This linearization transforms the original problem (1) into a sequence of quadratic programming (QP) subproblems as in (4)

$$\max_{x \in \quad ^n} f(x)^T x + \frac{1}{2} x^T H x : H \in \quad ^{n \times n}. \tag{4}$$

This QP subproblem is then optimized using any QP solver.

Notice from the iterative model in (2) that the step size parameter α is a positive scalar that is expected to substantially improve the function value. It is normally the optimizer of the merit function defined as:

$$\varphi(\alpha) = f(x_t + \alpha_t d_t) : \alpha > 0. \tag{5}$$

Algorithm 2 The Sequential Quadratic Programming (SQP)

1: set counter $t \leftarrow 0$;
2: $x_t \leftarrow x_0$; // x_0 is the starting point
3: $d_t \leftarrow d_0$; // d_0 is the initial search direction
4: **while** $\nabla f(x_t) > \text{Tol}_1$ **and** $t <$ MaxIter **do**
5: linearize (1) into a QP subproblem (4)
6: $H(x_t) \leftarrow \nabla^2 f(x_t)$
7: evaluate search direction d_t (3)
8: **if** $\alpha = 1$ satisfies Wolf conditions [7] **then**
9: $\alpha_t \leftarrow 1$
10: **else**
11: evaluate $\alpha_t : \alpha_t > 0$
12: **end if**
13: $x_{t+1} \leftarrow x_t + \alpha_t d_t$;
14: $t \leftarrow t + 1$;
15: **end while**

Algorithm 3 Stochastic Coordinate Ascent Algorithm (SCA)

1: Problem's dimension n;
2: Problem's search bounds $[\bar{x}_i, \underline{x}_i] : i = 1$ to n;
3: Starting point x_t, and search neighborhood ;
4: Search neighborhood radius:
$\delta_R^i \leftarrow 0.01 \times (\bar{x}_i - \underline{x}_i) : i = 1$ to n;
5: Minimum neighborhood radius $\delta_{min} \leftarrow 10^{-8}$;
6: **repeat** until termination condition: $\delta_R < \delta_{min}$
7: $GenerateAndEvaluateSearchPoints(x_t, \delta_R, n)$;
8: **if** $\max(f(\mathcal{X}\ )) > f(x_t)$ **then** // better solution found
9: $x_{t+1} \leftarrow x^j \mid f(x^j) = \max(f(\mathcal{X}\ ))$; // replace x_t
10: **else** // x_t is better than all its neighbors
11: $\delta_R^i \leftarrow \frac{1}{2}\delta_R^i, \forall i = 1 : n$; // shrink the neighborhood
12: **end** if
13: **end** (repeat)
14: **return** $(x_t, f(x_t))$

Generally, the ideal value for the step size parameter is the global optimizer of the merit function (5) and requires several evaluations of the objective function f and its gradient ∇f. Thus, at every iteration t, inexact line search algorithms are typically used to try out a sequence of candidate values for α_t; based on some pre-defined termination conditions a suitable value for α_t is accepted. A simple condition that ensures α_t provides a meaningful improvement in f entails:

$$f(x_t + \alpha_t d_t) < f(x_t). \tag{6}$$

The SQP model adopted in this paper is described in Algorithm 2. Note that this SQP model utilizes interior point method (IPM) [8], [9] to solve its QP subproblems (line 5). Also, the inexact line search algorithm used for estimating a suitable value of the step size parameter is based on the popular Wolf conditions [7] (lines 8 and 11). Further details on this SQP formulation can be found in [9].

C. A Derivative-free Method – The Proposed SCA Algorithm

The coordinate ascent (CA) algorithm is an inexpensive multidimensional optimization method. It also has its origins in MP. However, unlike the SQP algorithm, the CA is a derivative-free method and works by decomposing an n-dimensional optimization problem into n one-dimensional subproblems. Then, the CA algorithm, which was shown to have linear time complexity [10], cycles through the different coordinate directions during the search.

In this paper, given any continuous optimization problem (see equation (1)), we propose a novel stochastic coordinate ascent (SCA) algorithm as outlined in Algorithm 3. Unlike the traditional CA methods, at every iteration this SCA algorithm locally optimizes (1) by randomly[1] searching a set of coordinates around the current solution point x_t.

As can be seen from Algorithm 3 (lines 1-5), SCA operates only within a defined neighborhood ( ) of the initial search point x_t. The initial neighborhood size (i.e. the *step size*), defined by $\delta_R^i \mid i \in [1, n]$, is set to 1% of the width of the problem's search domain across all n dimensions (line 4). Note that the 1% initial radius is empirically decided so as to constrain the SCA algorithm to focus around a limited area of the starting point x_t. This restricted form of local search also mitigates the risk of creating infeasible solutions (i.e. points outside the feasible bounds) by the SCA algorithm. During the optimization cycle, the size of this neighborhood ( ) is expected to continuously shrink over iterations (line 11). Therefore, as a stopping criterion, a minimum threshold for the neighborhood (i.e. step) size δ_{min} is empirically set to 10^{-8} (line 5).

Having set its initial parameters, the main SCA algorithm (Algorithm 3) loops through lines 6-13 until the stopping condition is satisfied (line 5). In line 7, the algorithm invokes the $GenerateAndEvaluateSearchPoints$ sub-function (Algorithm 4) which generates and evaluates the fitness of a new set of fixed-size search points $\mathcal{X}$ within the neighborhood ( ) of the current point x_t. Now, if the fitness of the best sample in the set $\mathcal{X}$ is greater than that of x_t, the best sample becomes the next solution point x_{t+1} (lines 8-9). Otherwise the radius of the neighborhood is halved (lines 10-11). The SCA algorithm returns the locally optimum solution point with its fitness value $(x_t, f(x_t))$ in the region of the initial starting point (line 14).

The sub-function in Algorithm 4 generates and evaluates the appropriate search points in the neighborhood ; Fig. 3 further illustrates this process. The neighborhood of the current search point x_t is sampled in a circular (or spherical) manner; and the number of the sample points depends on the dimensionality n of the search domain (see Algorithm 4, lines 3 and 7). From Fig. 3, note that the size of the set of the sample points ($\mathcal{X}$) in the neighborhood of x_t is given by:

$$|x\ | = \begin{cases} 2n, & \forall n \leq 3; \\ 6, & \text{otherwise.} \end{cases} \tag{7}$$

Hence for higher dimensions $(n > 3)$ the proposed SCA algorithm essentially conducts a *block coordinate ascent* search [11], i.e. the algorithm searches the multi-dimensional neighborhood by successively optimizing any three *randomly* chosen dimensions at every iteration – hence the naming convention *stochastic coordinate ascent*. In a series of experiments, the next section evaluates the performance of the above three algorithms.

[1]Random here means selection of a set of dimensions in the state space in a stochastic manner.

Algorithm 4 Neighborhood Samples Generation Function

$GenerateAndEvaluateSearchPoints(x_t, \delta_R, n)$
if (problem dimension $n \leq 3$) **then**
  // generate $2n$ samples at radius δ_R from x_t
  $\mathcal{X} \leftarrow \{[x_t^i - \delta_R^i, x_t^i + \delta_R^i], \forall i \in [1, n]\}$;
  $f(\mathcal{X}) \leftarrow$ evaluate the fitness of samples in $\mathcal{X}$;
else
  $k \leftarrow U(n, 3)$; // chose any 3 from the n-dimensions;
  // generate $2k$ samples at radius δ_R from x_t
  $\mathcal{X} \leftarrow \{[x_t^i - \delta_R^i, x_t^i + \delta_R^i], \forall i \in [1, k]\}$;
  $f(\mathcal{X}) \leftarrow$ evaluate the fitness of samples in $\mathcal{X}$;
end
return $(\mathcal{X}, f(\mathcal{X}))$

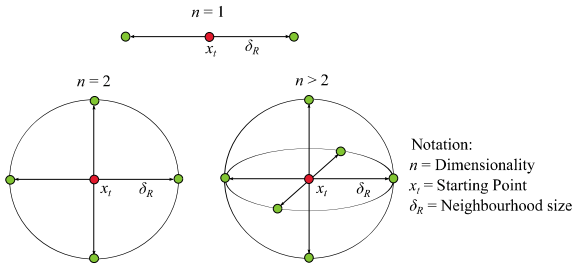

Fig. 3. The search neighborhood for the SCA algorithm for an n-dimensional search domain. The current search point x_t represented by a black (red in color) dot is at the center of the linear (for $n = 1$), circular (for $n = 2$) or a spherical (for $n > 2$) search domain. Situated at a neighborhood radius δ_R, the new search points represented by grey dots (green in color) are 2, 4 or 6 depending on the dimensionality of the search domain.

IV. Experiments

This section evaluates the three optimization frameworks described thus far. The objectives of these experiments are:

1) to examine the performance characteristics of the EC, SQP and SCA algorithms on two categories of global optimization tasks of varying complexities; and
2) to compare and analyze the fitness characteristics of the three algorithms regarding *how much progress* is been made and *how fast* (robustness and efficiency).

Detail parameter settings for the EC, SCA and SQP algorithms are described in Table I. In all experiments, the restarted algorithms (SCA and SQP) are repeatedly run on random uniform samples in the feasible search space. For the standard EC algorithm, a pool size N (cf. Table I) that yields good performance is chosen depending on the problem dimension.

A. Evaluation Benchmark Test functions

Two categories of global optimization benchmarks from the redesigned test problems used in the CEC2013 competition on real-parameter optimization [12] are used in these experiments. As shown in Table II, we compare the three algorithms on a set of *three* multimodal and composition benchmarks having varying dimensionalities over the optimization period MaxFES $= n \times 10,000$ function evaluations. Since the SQP algorithm requires derivatives, only benchmarks with no discontinuities in their search space are selected.

TABLE I. Parameter Settings for the EC, SCA and SQP Algorithms

Parameter Name	Symbol	Description/Values
EC Algorithm		
Population Size	N	$N \in \{200, 400, 750, 1000\}$
Initial Population	P_0	Uniform random distribution
Encoding	–	Real-valued
Selection Scheme	–	Binary Tournament
Operator type	$\mathcal{C}$	Crossover: Intermediate recombination operator
	$\mathcal{M}$	Mutation: BGA mutation operator
Crossover Probability	$P_{\mathcal{C}}$	1.0
Mutation Probability	$P_{\mathcal{M}}$	0.01
Replacement Scheme	–	Generational–Elitist
Termination Criterion	Max-FEs	Max. evaluations $(n \times 10,000)$
SQP Algorithm		
Minimum Gradient of f	Tol_1	$\mathrm{Tol}_1 = \nabla f < 10^{-8}$
Maximum Iteration	MaxIter	100
SCA Algorithm		
Neighborhood size	δ_R	0.01
Min. neighborhood size	δ_{min}	10^{-8}

TABLE II. Benchmark Test Problems – Selected from the CEC2013 Competition on Real-parameter Optimization

Category	Acronym	Dimensionality (n)	Max. Evaluations (MaxFES)
Multimodal	F6, F7 and F8	$n \in \{10, 30, 50\}$	$10,000 \times n$
Composition	F21, F22 and F23		

The evaluation criteria in the CEC2013 competition [12] require optimizing all the benchmarks for $n \in \{10, 30, 50\}$ dimensions in 51 repeated runs. Then for each run, 11 optimization error values

$$f_{error} = (f_i(x) - f_i^*(x)) \tag{8}$$

are recorded after $(0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0) \times$ MaxFES evaluations, where f_{error} is the error between the current best solution $(f_i(x))$ and the true optimal solution $(f_i^*(x))$. For each problem, we record the *mean* values of f_{error}. Note that all benchmarks are of minimization type.

B. Experiment 1: Evaluation on the Multimodal Benchmarks

In this section, we examine the performance of the three algorithms on a set of multimodal benchmarks (F6, F7 and F8) over $n = \{10, 30, 50\}$ dimensions.

Results – Analysis and Interpretation

Fig. 4 shows the performance comparison results for the EC and the restarted SQP and SCA algorithms on varying dimensions of the CEC2013 multimodal benchmarks (F6, F7 and F8); the results are averages of 51 repeated runs. As described earlier, each of these benchmarks is rotated, asymmetric, and highly multimodal with huge number of local optima [12]. Notice, from the plots labeled A and B in Fig. 4, that the performances of the EC and SCA algorithms are not significantly different[2] on the 10 and 30 dimensions of the F8 benchmark. In fact, none of the three algorithms is significantly better than any other on the 50 dimensions of the F8 benchmark (labeled C).

[2]The Kruskal-Wallis nonparametric test is used in a multiple comparison procedure with $\alpha = 0.01$.

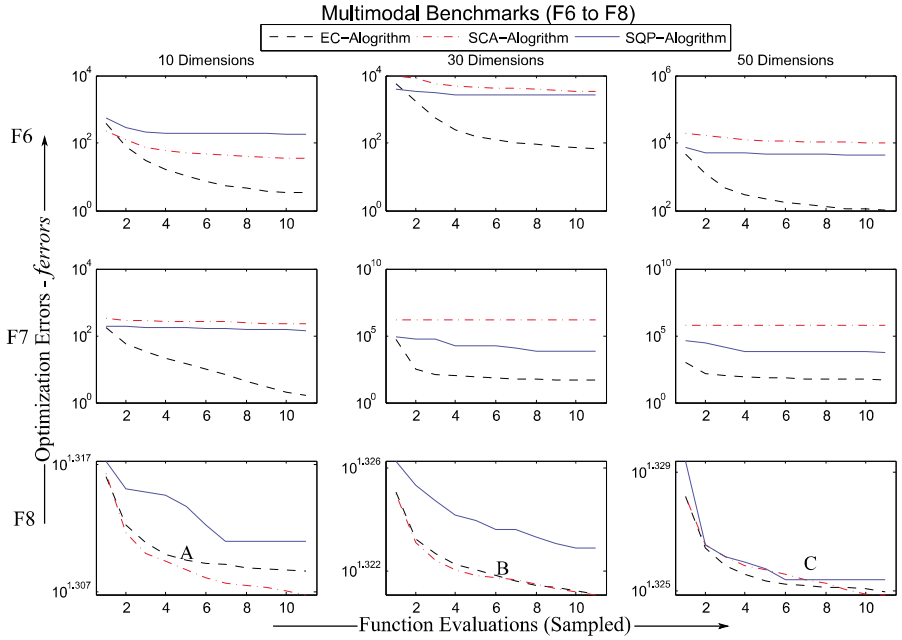

Fig. 4. Performance comparison of the EC, SCA and SQP Algorithms on the CEC2013 Multimodal Benchmarks of varying sizes. Results are averages of 51 independent runs. The vertical axes are in log scale; the horizontal axes are the 11 linearly sampled Function evaluations over the optimization period (see text).

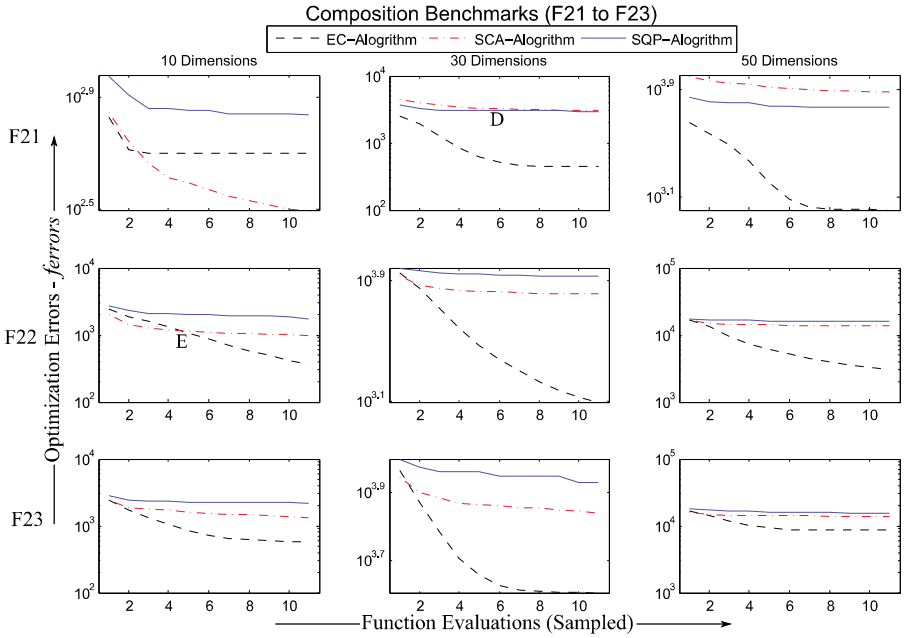

Fig. 5. Performance comparison of the EC, SCA and SQP Algorithms on the CEC2013 Composition Benchmarks of varying sizes. All results are averages of 51 repeated runs. The vertical axes are in log scale; the horizontal axes are the 11 linearly sampled Function evaluations over the optimization period (see text).

It is observed from across the various dimensions of the F6 and F7 benchmarks (Fig. 4) that, it is only the EC algorithm that mostly enjoys steady progress by continuously minimizing the optimization error (f_{error}) over the entire optimization period. This is not unexpected since, with such largely flat and rugged landscapes in F6 and especially F7 (see [12]), the EC algorithm is inherently global and can maintain sufficient diversity necessary to sustain optimization progress. This characteristic performance of the standard EC algorithm can be attributed to the provision of the carefully chosen population sizes suitable for the different sizes of these benchmarks (see the parameter settings in Table I).

On the other hand, still on the F6 and F7 benchmarks (Fig. 4), it is observed that the two restarted algorithms (SCA and SQP) tend to get stuck at some local optima after a few iterations. However, it can be noticed that the SCA algorithm only outperforms the SQP on the 10-dimensions of the F6 benchmark. This is a surprising finding given that the SQP algorithm mainly relies on gradient information – a feature that is difficult to harness in such wide and ruggedly-flat landscapes. Nevertheless, while the performances of the EC and SCA algorithms on the F8 benchmark (see Fig. 4) are tightly close (not significantly different at $\alpha = 0.01$), it is the SCA algorithm that yielded the best results in this case. Overall, although none of the compared algorithms achieves the optimum optimization error value (i.e. $f_{error} = 0$), the standard EC algorithm outperforms both the restarted SQP and SCA algorithms on this set of multimodal benchmarks (Fig. 4). This is mainly due to its ability to maintain steady and continuous progress during the optimization period by avoiding more local optimal solutions.

C. Experiment 2: Evaluation on the Composition Benchmarks

This section examines the performance of the three algorithms on the composition benchmarks F21, F22 and F23 [12].

Results – Analysis and Interpretation

The results of Experiment 2 are as shown in Fig. 5. All vertical axes (f_{error}) are in log scale to aid visualization, and all results are averages of 51 independent repeated runs. Note, however, that the performance differences between the EC and SCA algorithms on the 10 dimensions of F22 (marked as E) are not statistically significant within a 99% confidence level. Also, although the two restarted algorithms (SCA and SQP) trail the global EC algorithm on the 30 dimensions of F21 (labeled D), their performances are not significantly different in this case. Nevertheless, in all other cases (see the unlabeled plots in Fig. 5) the observed differences in the characteristics of all the three algorithms are statistically significant.

It is observed from Fig. 5 that the SCA algorithm copes better than the SQP algorithm on these composite benchmarks (see the unlabeled plots in Fig. 5), except on the 50 dimensions of F21. However, it can be recalled from Experiment 1 above (cf. Fig. 4) that the SQP mostly outperformed the SCA algorithm. These are interesting findings because often the restarted gradient-based (greedy) methods, such as the SQP, are assumed to yield inferior performance on such highly multimodal problems (Fig. 4). However, the outcomes from these two experiments suggest that it requires such highly complex (composite) landscapes (Fig. 5) for a restarted stochastic algorithm, like SCA, to outperform a restarted SQP algorithm.

In addition, it noticed from Fig. 5 that the performance benefit exhibited by the EC algorithm over the two restarted methods is less pronounced on these composition benchmarks (see the 10 and 50 dimensions of F22 as well as all the instances of the F23 benchmarks). Nevertheless, for all the plots in Fig. 5, the slope of the optimization curves revealed that it is the EC algorithm that mostly demonstrates continuous progress during the entire optimization period.

Table III compares some possible limitations of the three algorithms based on there inherent parameterizations (cf. Table I) and the outcomes of the empirical experiments above.

TABLE III. LIMITATIONS OF THE EC, SQP AND SCA ALGORITHMS

	Complexity	Convergence rate	Sensitive Parameters	Resilience to dimensionality
EC:	Polynomial	First order	$N, P_{\mathcal{C}}, P_{\mathcal{M}}$	High
SQP:	Quadratic	Second order	min. gradient Tol_1	Low
SCA:	Polynomial	First order	σ_r, σ_{min}	Low

V. CONCLUSION

This paper investigated global optimization methods from three different perspectives using the IEEE CEC2013 benchmarks. We found that the ability of the standard EC algorithm to search global optimization landscapes of varying complexities by evolving a suitably sized pool of search points has largely led to its observed superior robustness. For the restarted approaches, however, by effectively exploiting any available gradient information, the SQP algorithm has surprisingly achieved better convergence efficiency than SCA on a set of highly *multimodal benchmarks*. Nevertheless, on the more complex *composition benchmarks*, the low-cost SCA algorithm maintained a robust search compared to the SQP.

The experimental results revealed that the three examined algorithms possess complementary performance characteristics – an essential feature for building effective hybrid frameworks. In addition, the outcomes have provided new and vital insights into *which* among the algorithms should be used to kick-start the search (which from these results is the global EC algorithm), as well as *how* and *when* to switch among the algorithms if combined in a hybrid framework.

REFERENCES

[1] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control systems engineering: A survey," *Control Engineering Practice*, vol. 10, no. 11, pp. 1223–1241, 2002.

[2] R. Martí, M. G. C. Resende, and C. C. Ribeiro, "Multi-start methods for combinatorial optimization," *European Journal of Operational Research*, vol. 226, no. 1, pp. 1–8, 2012.

[3] C. A. Floudas and C. E. Gounaris, "A review of recent advances in global optimization," *Journal of Global Optimization*, vol. 45, no. 1, pp. 3–38, 2009.

[4] D. Fogel, *Evolutionary Computation: The Fossil Record.* Wiley-IEEE Press, 1998.

[5] P. J. Werbos, "Computational intelligence for the smart grid-history, challenges, and opportunities," *Computational Intelligence Magazine, IEEE*, vol. 6, no. 3, pp. 14–21, 2011.

[6] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, no. 2, pp. 95–99, 1988.

[7] M. A. Bhatti, *Practical Optimization Methods: With Mathematica Applications.* Rensselaer, New York: Springer-Verlag Inc, 2000.

[8] F. A. Potra and S. J. Wright, "Primal-dual interior-point methods," *SIAM*, 1997.

[9] H. A. Bashir and L. Ximing, "Interior point method based sequential quadratic programming algorithm with quadratic search for nonlinear optimization," *IJCISIM*, vol. 3, pp. 51–60, 2011.

[10] I. Loshchilov, M. Schoenauer, and M. Sebag, "Adaptive coordinate descent," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. Dublin, Ireland: ACM, 2011.

[11] P. Richtárik and M. Takáč, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Mathematical Programming*, pp. 1–38, 2012.

[12] J. J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Nanyang Technological University, Singapore, Tech. Rep., 2013.