A hybrid ABC for expensive optimizations: CEC 2016 competition benchmark

Enrico Ampellio
Dipartimento di Ingegneria Meccanica e Aerospaziale
Politecnico di Torino
Torino, Italy
Email: enrico.ampellio@polito.it

Luca Vassio
Dipartimento di Elettronica e Telecomunicazioni
Politecnico di Torino
Torino, Italy
Email: luca.vassio@polito.it

***Abstract*—An evolution of the *Artificial Bee Colony* (ABC) optimization algorithm, called the *Artificial super-Bee enhanced Colony* (AsBeC), is presented for leading to the best improvement with a low number of analyses. AsBeC is designed to provide fast convergence speed, high solution accuracy and robust performance over a wide range of problems. It implements enhancements of ABC structure and original hybridizations with interpolation strategies. The aforementioned techniques are tested on the expensive benchmark of the *Special Session on Real-Parameter Single Objective Optimization* at CEC 2016. In this specific case, the hybridization with a quadratic trust region approach assumes a major importance. Moreover, the AsBeC results are compared to the algorithms tested on the same benchmark at CEC 2015, showing remarkable competitiveness and robustness.**

I. Introduction

Numerical optimization often concerns simulation-based problems in which computational analyses are expensive in terms of time and resources, as usual in engineering [1]. This field demands fast algorithms for solving optimization problems using as few Function Evaluations (FEs) as possible. The present work focuses on single-objective problems with relatively low dimensionality, i.e. 10 and 30 dimensions.

In the literature, there are several architectures and algorithms suitable for these tasks [2]. Many nature-based techniques have been developed, such as genetic algorithms and particle swarm optimization. One of the new most promising nature-inspired meta-heuristic methods is the Artificial Bee Colony (ABC) algorithm [3], a simple swarm-based optimizer which showed valuable and robust performance over many different problems [4]. Several technical dissertations, test case validations and improvements have been done in recent years [5]. ABC is an example of swarm intelligence: it models the collective behaviour of decentralized and self-organized systems consisting of simple agents. The algorithm reproduces the behaviour of a honey bee colony searching the best seed solution (NS) into a target area. A group of bees, called the employees, search the space near each NS, while other bees, called the onlookers, help in the most promising regions. Seed solutions that reveal themselves non-productive are abandoned in place of new ones, investigated by the scout bee which moves in the whole target area. An entire optimization sequence composed of employees, onlookers and scout is called a *cycle*.

Despite the huge amount of available literature on ABC variants, to the best of our knowledge no paper underlines a performance gain even with very few function evaluations. As well, no hybridization with methods similar to trust region has been analysed so far, despite they are recognized to be among the fastest techniques, especially when dimensionality is small [6]. This framework, joined with the great potential offered by ABC architecture, motivates the introduction and analysis of modifications specifically targeted to expensive optimization. The main idea behind the algorithm presented in this paper is to hybridize the ABC with the basic principles of derivative-free, quadratic trust region optimizers for high-cost objective functions (Sec.II-C). The new algorithm, called *Artificial super-Bee enhanced Colony* (AsBeC), shows robust performance over a wide range of problems by successfully merging the main advantages of swarm-based and interpolative methods [7].

An executable demo of the new algorithm is publicly available for allowing further tests by the community[1].

II. Improving techniques

These techniques have the purpose to achieve the fastest solution improvement, without clustering the swarm and leading to premature convergence at the same time. Further details on these techniques and their implementations can be found in [7]. A minimization problem is considered without loss of generality.

A. Biased Onlooker assignment (BO)

In ABC, onlookers are assigned to the seed solutions with a probability proportional to the solution quality, given by a fitness function [8]. However, if the objective values for all the seed solutions is similar, that fitness function returns almost identical values and the relative probabilities are alike. Hence, the standard fitness formulation is not able to distinguish the solution quality in relative terms. In order to always strengthen exploitation, the fitness is rescaled between 0 and 1 according to the distance of the seed solutions from the best one. In

[1] https://lucavassio.wordpress.com/expensive_optimization

978-1-5090-0623-6/16/$31.00

this way, the onlooker assignment takes into account also the quality difference among the NSs in relative terms, similarly to a ranking procedure.

B. Local Interpolation (LI) with Postponed onlooker Dance (PD)

In order to quickly improve the best solutions in their local neighbourhoods, the concepts of opposition principle [9] and parabolic interpolation are introduced. Whenever a pseudo-random movement for the seed solution k to a new position $x^{k,rnd}$ does not produce any improvement, the algorithm moves in the opposite direction with the same step size. If this new position $x^{k,opp}$ is better than the previous one, then the correspondent seed solution k is updated.

The parabolic interpolation estimates the local curvature of the objective function. This technique follows the opposition principle: if the opposite step is not successful, then the multi-dimensional parabola passing through three already known points, i.e., (i) starting seed solution point x^k, (ii) first random movement $x^{k,rnd}$ and (iii) opposite position $x^{k,opp}$, is calculated and its minimum is tested. Opposition principle and second order interpolation are performed in sequence, and the algorithm restarts from a random movement whenever the seed solution is improved.

The raw approximation offered by LI is balanced by the fact that just one and two additional FEs are respectively required for the opposite movement and the parabolic interpolation, regardless of the problem dimensionality.

LI is used in the ABC in the onlooker phase, considering each seed solution. In order to give more chances to exploit LI to the onlooker group, another technique, i.e., Postponed onlooker Dance (PD), is combined with it. This modification enables the onlooker groups to seek nectar near the best regions for a longer time. Each onlooker bee performs multiple movements instead of just one to exploit its seed solution. Anyway, the number of movements should be maintained small (set to 3 in this paper), in order not to affect exploration and convergence. This modification acts like enlarging the onlooker group, but repeating their movements instead of increasing their number. Hence, the best seed solutions have more probability to improve, to be selected for mutation by the others and to use the local interpolation.

C. Quadratic Prophet (QP)

The Quadratic Prophet (QP) hybridizes the ABC with a local search method to strongly improve the convergence speed, especially in the early phases. The key idea is to exploit the function evaluations already performed to create multiple quadratic polynomial interpolators around the best regions. Hence, these quadratic models are built in the neighborhood of the active seed solutions by using their closest samplings in the pool of all the solutions analysed by the AsBeC algorithm so far. The global minima of the quadratic models, if exist, are then tested. QP does not require additional FEs to populate the samplings for the interpolations; its final cost is at the most equal to the number of seed solutions if all quadratic models succeeded. Whenever the QP finds a better solution, the related seed solution is immediately updated.

The QP method adapts its search region in accordance to the recorded samplings: if they are dense around a given solution, the algorithm is exploiting that area; instead, if they are sparse, the exploration of the whole region is empowered. This self-adaptation resembles the mechanism used to adjust the searching region in CMA-ES algorithm [10].

The implementation of the QP is based on quadratic polynomial surfaces, whose complete form, Q_C, and reduced form, Q_R, for a generic problem in D dimensions can be found in [7]. The reduced model is not able to capture rotated functions, but it is enough to cope with bowl shaped regions. Q_C needs $C = 0.5 \cdot (D+2) \cdot (D+1)$ samplings to be exactly defined. The reduced quadratic surface Q_R, without any mixed term, only needs $2 \cdot D+1$ samplings. In order to activate this technique as soon as possible, reduced models Q_R are built starting when the algorithm provides at least $2 \cdot D+1$ solutions, ending up with complete models Q_C when more than C overall samplings are available.

The model coefficients are obtained by solving exactly the linear system of equations for the C samplings nearest to a given seed solution. If the Hessian matrix of the model is diagonal positive, then the point in which the Jacobian determinant is zero is tested. Notice that the model could misinterpret the local behaviour of the objective function if the sampling is ill-conditioned or if f is not quadratic or noisy in that area. In order to increase the likelihood of a well-conditioned sampling, some modifications, reported in [7], are introduced.

This QP technique is used at every cycle, after the onlooker phase, and once during the initialization, by setting the initial number of random solutions to $max(NS, 2 \cdot D+1)$. It must be clear that the QP is conceived to deal with problems where the objective function evaluations represent the bottleneck of computational times. In fact, the cost of this technique is dominated by the resolution of the linear system, that in its general form require $O(C^3)$ operations at each cycle, for each seed solution, and therefore the computation becomes numerically heavy when addressing high dimensionality problems.

D. Systematic Global Optimization (SGO)

The S.T.E.P. algorithm [11] shows impressive convergence rate, simplicity and robustness for highly multi-modal and complex functions of one variable. It is derivative-free, not population-based, without parameters to tune and it does not assume any property for the function to optimize. The domain boundaries and a random point between them are evaluated, creating two *partitions*. For each partition is computed the curvature a parabola should have to enclose the best so far optimum inside it. The S.T.E.P. method iterates by sampling the center of the partition with smallest curvature, i.e. the one identified with the greatest chance of improving the best solution found so far, until a given tolerance is reached. The multivariate extension proposed by Pošík and Baudiš [12] interleaves the steps of the univariate solvers such that all dimensions are concurrently optimized. Each time, the dimension to investigate is chosen following a round-robin scheduling.

S.T.E.P. based solvers are very effective only on additively separable or quasi-separable problems, in which the correlations among variables are weak or only few variables are correlated. As a consequence, inside the AsBeC algorithm this technique is activated if a test for quasi-separability is successfully passed. The used test investigates the separability in the cheapest possible way, giving just a necessary but not sufficient condition. For an additively separable function, each first-order partial derivative $\partial f/\partial x_i$ is independent from other variables $x_j \neq x_i$. This is checked in just two points, estimating two partial derivatives. If the two estimations vary below a given tolerance, here chosen as 10^{-3}, the two variables could be quasi-separable. If the conditions on the derivatives holds for D couples of variables, the test is passed. The procedure is interrupted whenever a variable is found to be correlated. This test needs a total number of $FEs \leq 2 \cdot D + 1$.

If this test is successfully passed, the interleaved solver by Pošík and Baudiš [12] is run as a one-shot technique after the initialization phase. The best so-far solution is used as starting point and all the variables are investigated in deterministic sequence. For this reason, the proposed hybridization is called Systematic Global Optimization (SGO). The maximum number of evaluations per variable is set to 50, according to the results by Swarzberg et al. [11]. At the end of the SGO phase, the best seed solution is moved to the best solution found.

III. AsBeC algorithm

The combination of all the previous techniques, applied to the ABC, gives the AsBeC algorithm. Its pseudo-code is presented in Algorithm 1. BO and PD are enhancements of ABC regarding bees re-organization, while LI, QP and SGO are different hybridizations. Thanks to these hybridizations, the bees assume new extra abilities and are called super-bees, explaining the name of the new algorithm.

The main three parameters common to the ABC architecture, i.e., the limit parameter, the number of cycles and the colony size, are chosen. The limit parameter is set to $D \cdot NS$, as advised for the original ABC [13], while the number of cycles depends on FEs and on the colony size. The overall number of agents N, is set equal to 8 for 10D and 32 for 30D, following the robust standard setting defined in [7] and according with the intuition that larger dimensions need more bees.

The language used for coding the algorithm and testing it in the competition is Matlab R2015a.

IV. Benchmark and metrics

The expensive case benchmark for the *Special Session and Competitions on Real-Parameter Single Objective Optimization* at CEC conference 2016[2] is developed by Qu et al. [14] and consist of 15 different functions. The tolerance used is 10^{-8} and the number of repetitions is set to 20. The functions are tested with 10 dimensions and 500 FEs and with 30 dimensions and 1500 FEs. The functions of the benchmark are shifted rotated, thus analytically not separable, and it includes highly ill-conditioned and very complex problems, also with extremely jagged shapes. Further details about the settings and functions can be found in the work by Chen et al. [15].

[2]http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2016/CEC2016.htm

A metric score S, depending of the number of dimensions D and test function TF is defined as:

$$S(TF,D) = mean[f_a^{TF}]_D + median[f_a^{TF}]_D$$

where the mean and the median operators are over repetitions and where:

$$f_a^{TF} = 0.5 \cdot (f_{MaxFEs}^{TF} + f_{0.5MaxFEs}^{TF})$$

where f is the residual objective value with respect to the adopted tolerance.

The total score TS used for the competition is defined as:

$$TS = \sum_{TF=1}^{TF=15} S(TF, D=10) + \sum_{TF=1}^{TF=15} S(TF, D=30)$$

For both TS and S, a lower value means better optimization results. This metric favor those algorithms that can solve the most complicated problems; if the residuals have different orders of magnitude, the total score TS is mainly driven by the function value furthest from the optimal. Moreover, notice that with just 20 repetitions and few FEs available, the results could not be stable.

V. Comparisons and results

The techniques proposed in Section II are tested on the Benchmark. They are applied one-by-one to the reference ABC configuration. In Table I and Table II are reported the scores S for each technique and the basic ABC for each function, respectively for 10 and 30 dimensions. All the techniques have positive effects on the algorithm performance. The QP clearly takes a dominant role with respect to the other techniques, being able to gain many orders of magnitude in some test functions. The other techniques show a more limited impact: BO and LI+PD slightly improve the classic ABC in most of the functions; SGO technique often performs a little worse than the ABC, because the quasi-separability test costs few function evaluations and is almost never passed on this benchmark. However, function 4 passed the quasi-separability test and, as can be seen from the tables, the SGO largely improves the ABC and it is the overall best.

Then the AsBeC algorithm that includes all the techniques is tested and the results are also shown in Table I and Table II. Best overall results for each function among the algorithms are highlighted in gray in the tables and AsBeC is the best on almost all the function, being very close to the best results in the remaining ones. This means that combining the techniques improves even further the algorithm. The AsBeC configuration is the one with lowest TS, reaching a value of around $4.6 \cdot 10^5$,

Algorithm 1 The AsBeC pseudo-code

```
Generate max(NS, 2·D+1) random solutions in the search area (QP, Sec. II-C)
Evaluate their quality
Identify seed solutions as the best solution and assign employees to them
Find minimum of the quadratic model near the best seed solution (QP, Sec. II-C)
if a new best solution is found then update best seed solution end if
if quasi-separability test is passed then use interleaved S.T.E.P. solver (SGO, Sec. II-D) end if
if  a new best solution is found then update best seed solution end if
repeat
    for all employees do
        Generate pseudo-random solution near its seed solution
        Evaluate the quality of solution
        if it is better than current employee's seed solution then update seed solution end if
    end for
    Assign onlookers to the seed solutions depending on their quality (BO, Sec. II-A)
    for all onlookers do
        for all postponed dance iterations (PD, Sec. II-B) do
            Generate new solution near its seed solution, using local interpolation (LI, Sec. II-B)
            Evaluate the quality of solution
            if it is better than current onlooker's seed solution then update seed solution end if
        end for
    end for
    for all seed solutions  do
        Collect the samplings nearest to the current seed solution
        Build quadratic model in its neighbourhood and find its minimum (QP, Sec. II-C)
        if it is better than current seed solution then update seed solution end if
    end for
    if a seed solution is not improved for a limited time then replace it with a new random solution end if
    if new best solution found in this cycle then update global best end if
until requirements are met
```

TF	ABC	BO	LI+PD	QP	SGO	**AsBeC**
1	1.8E+09	8.6E+08	8.4E+08	0	2.9E+09	0
2	8.0E+04	7.7E+04	6.3E+04	0	8.6E+04	0
3	2.3E+01	2.1E+01	2.2E+01	1.9E+01	2.4E+01	1.8E+01
4	1.7E+03	1.0E+03	2.0E+03	2.3E+03	1.6E+02	2.0E+02
5	5.3E+00	4.3E+00	4.8E+00	5.9E+00	5.1E+00	4.4E+00
6	6.0E+00	2.9E+00	4.1E+00	3.1E+00	5.3E+00	1.7E+00
7	4.0E+01	2.0E+01	3.9E+01	1.7E+00	4.5E+01	9.5E-01
8	7.8E+02	6.6E+02	4.5E+02	2.0E+03	2.5E+03	7.0E+01
9	8.2E+00	8.3E+00	8.4E+00	8.5E+00	8.3E+00	8.1E+00
10	1.6E+06	2.6E+06	1.3E+06	2.1E+03	1.3E+06	2.7E+03
11	3.4E+01	2.8E+01	2.6E+01	2.5E+01	3.8E+01	1.8E+01
12	5.5E+02	6.0E+02	6.2E+02	5.1E+02	5.9E+02	3.9E+02
13	7.9E+02	7.5E+02	7.3E+02	7.3E+02	8.1E+02	6.8E+02
14	4.3E+02	4.3E+02	4.4E+02	4.2E+02	4.4E+02	4.1E+02
15	1.0E+03	9.5E+02	9.8E+02	8.9E+02	9.9E+02	7.6E+02

TABLE I: Scoring S for ABC, ABC with different techniques and AsBeC, 10D, 500FEs, with best achievements highlighted in gray

TF	ABC	BO	LI+PD	QP	SGO	**AsBeC**
1	8.8E+10	8.6E+09	7.0E+10	0	9.0E+10	0
2	2.3E+05	2.4E+05	2.4E+05	0	2.4E+05	0
3	8.4E+01	7.8E+01	8.3E+01	7.8E+01	8.5E+01	7.0E+01
4	1.1E+04	4.6E+03	1.2E+04	1.3E+04	4.0E+02	5.7E+02
5	7.1E+00	5.7E+00	6.8E+00	8.1E+00	6.9E+00	7.5E+00
6	1.1E+01	3.7E+00	1.1E+01	2.7E+00	1.1E+01	1.4E+00
7	2.4E+02	3.4E+01	2.0E+02	1.1E+00	2.2E+02	1.1E+00
8	2.6E+06	4.4E+05	8.4E+05	5.1E+05	3.5E+06	8.0E+04
9	2.8E+01	2.8E+01	2.8E+01	2.8E+01	2.8E+01	2.7E+01
10	4.8E+07	4.5E+07	4.8E+07	1.1E+06	6.2E+07	3.7E+05
11	4.9E+02	2.5E+02	3.7E+02	3.1E+02	5.1E+02	1.6E+02
12	2.2E+03	1.9E+03	2.1E+03	2.1E+03	2.2E+03	1.8E+03
13	2.1E+03	1.1E+03	1.6E+03	1.4E+03	2.2E+03	9.2E+02
14	7.6E+02	6.4E+02	7.5E+02	7.2E+02	7.5E+02	5.8E+02
15	3.0E+03	2.7E+03	2.8E+03	2.5E+03	2.9E+03	2.3E+03

TABLE II: Scoring S for ABC, ABC with different techniques and AsBeC, 30D, 1500FEs, with best achievements highlighted in gray

while the TS of the ABC reaches just $9.0 \cdot 10^{10}$ (see Table VI). Moreover, in Figure 1 and Figure 2 are plotted the function-by-functions mean residual results as function of FEs. The lines with circle markers represent the performance of the AsBeC, the ones with starred markers represent the ABC and the other four lines (not distinguishable) report the four techniques (Section II) applied one-by-one to the ABC. The AsBeC shows remarkable results in the whole FEs envelope. Notice that similar results for AsBeC and its techniques have been obtained in [7] with another benchmark, showing the robustness of the techniques and their combinations.

Since the CEC 2016 competition is based on the metric TS, AsBeC is then further tuned solely on it. In the used benchmark all the functions are shifted rotated, then the SGO is deactivated, expecting worse performance only on test function 4. Moreover, different colony sizes are tried obtaining an improvement of the total score TS using 40 agents for 30D and maintaining 8 agents for 10D. In this way, the new algorithm configuration, called AsBeC_tuned, lowers the TS to $3.7 \cdot 10^5$ (see Table VI). Detailed statistical results about the 20 independent runs of the AsBeC_tuned algorithm are reported in Table III and Table IV.

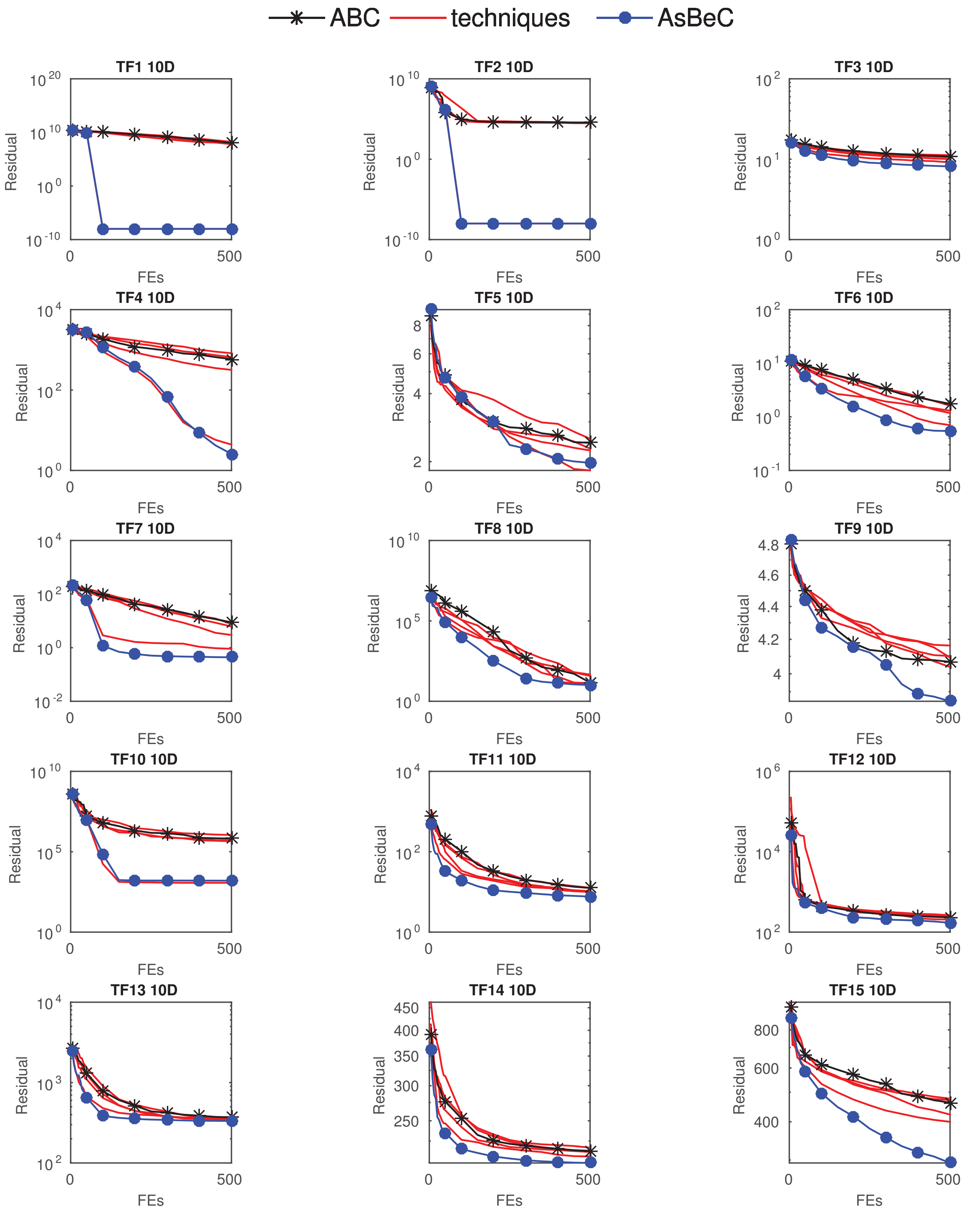

Fig. 1: Mean residual results for ABC, AsBeC and other variants of the ABC, as function of FEs, 10D and 500 Fes

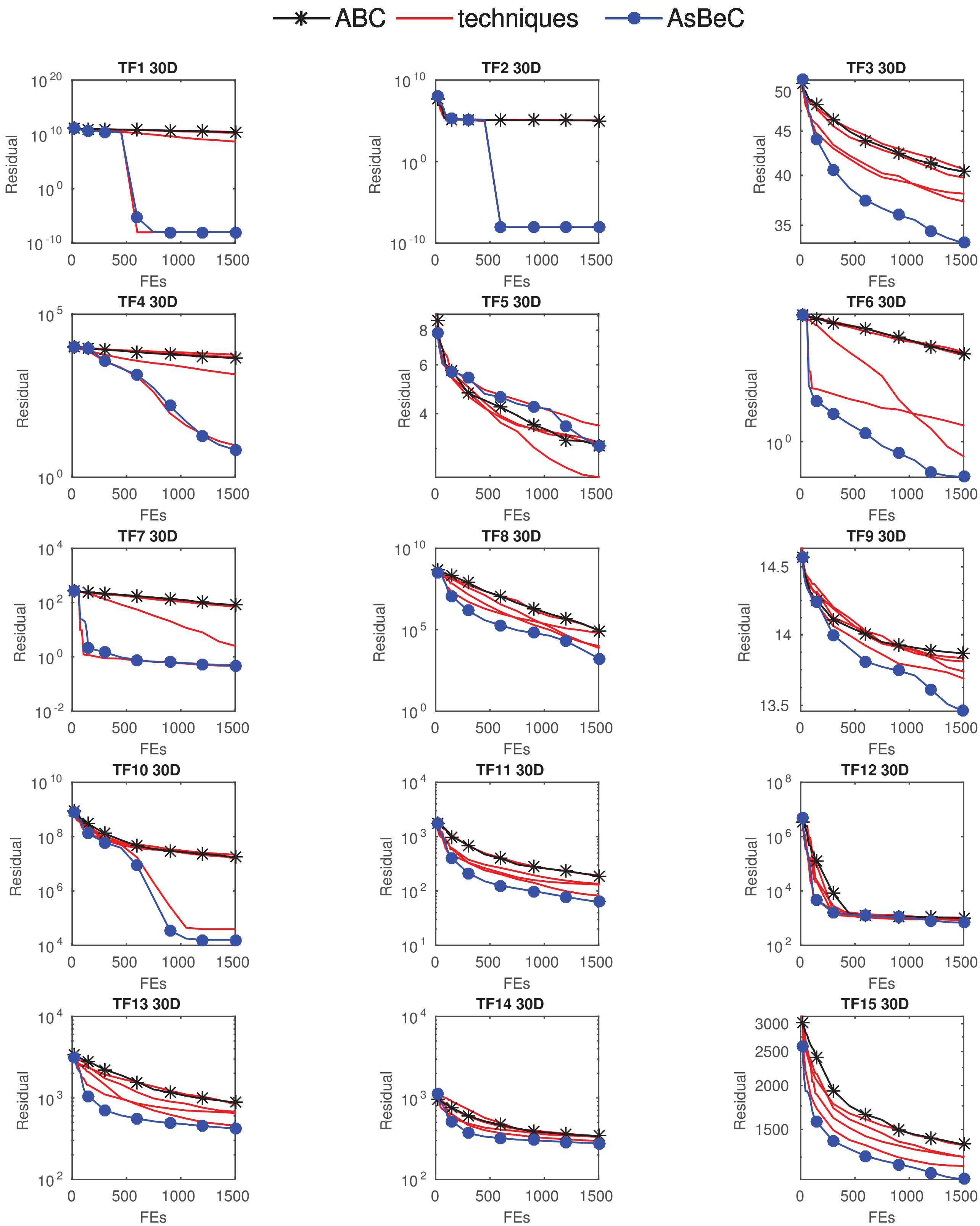

Fig. 2: Mean residual results for ABC, AsBeC and other variants of the ABC, as function of FEs, 30D and 1500 Fes

TF	Best	Worst	Median	Mean	Std
1	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.00000	0.00000	0.00000	0.00000
3	5.55440	11.52710	8.85070	8.26562	1.61658
4	109.42010	976.17300	434.04925	420.56647	190.99244
5	0.68970	3.93990	1.81955	1.97372	0.76389
6	0.29640	0.98140	0.61595	0.62122	0.18465
7	0.19330	0.76670	0.48000	0.45287	0.13160
8	4.63880	20.66150	8.98350	9.98656	4.04855
9	3.10670	4.21750	3.91185	3.84722	0.27715
10	558.98400	3005.40900	1050.25700	1281.91845	607.60702
11	3.88900	14.69400	7.62250	7.88650	2.46716
12	65.89400	329.35200	171.58150	168.61150	77.50184
13	320.27200	346.38100	330.58350	331.09825	7.80912
14	194.21400	220.00500	203.49650	204.42645	7.04848
15	18.39800	540.41500	362.91200	263.11410	184.00660

TABLE III: AsBeC_tuned final residual results, 10D, 500FEs

TF	Best	Worst	Median	Mean	Std
1	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.00000	0.00000	0.00000	0.00000
3	27.55090	39.18600	33.29380	33.29020	3.00456
4	1990.03700	4422.34300	3539.82500	3388.10730	629.96747
5	1.92640	4.13090	3.08115	2.99927	0.73286
6	0.31380	2.45000	0.46415	0.68579	0.60987
7	0.45890	0.57530	0.49610	0.50424	0.03150
8	117.69530	9093.23200	1032.31000	2446.56621	3084.45730
9	12.57800	13.99430	13.51405	13.42652	0.38201
10	3843.36700	91649.55000	12669.65000	26010.53945	26798.55863
11	27.66700	145.76500	40.12400	56.43945	32.82199
12	413.99000	1076.70800	657.60700	686.22060	168.29850
13	381.78100	513.52000	436.44750	439.13910	38.83335
14	254.00400	309.77600	273.50400	274.67910	13.72275
15	777.28600	1248.33700	1131.91950	1121.50775	111.26175

TABLE IV: AsBeC_tuned final residual results, 30D, 1500FEs

The computational complexity is another important metric for an algorithm. In Table V the median computational time is reported, according to the median adimensional definition in [15]. The algorithm complexity widely increases for the setting with 30D and this is driven by the QP technique, according with the considerations given in Section II-C.

The following algorithms are all the ones used on the same benchmark[3] [14] for the competition of 2015, with public results:

[3]http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2015/CEC2015.htm

TF	T_1/T_0 , 10D	T_1/T_0 , 30D
1	1.7	43.6
2	1.9	43.5
3	3.0	54.7
4	1.9	50.4
5	2.8	51.2
6	2.4	48.0
7	2.0	50.5
8	2.8	51.7
9	2.8	50.6
10	2.3	43.9
11	2.9	49.1
12	2.9	52.3
13	2.6	48.6
14	2.7	50.2
15	3.0	52.9

TABLE V: AsBeC_tuned computational complexity

Algorithm	TS
AsBeC_tuned	375 973
AsBeC	460 542
MVMO	3 062 550
TunedCMAES	203 324 193
CMAS-ES_QR	475 807 278
iSRPSO	9 213 589 133
ABC	89 918 717 356
humanCog	106 093 535 264

TABLE VI: Total scores comparisons

- MVMO, a population-based stochastic technique with a mapping function for the offspring;
- CMAS-ES_QR and TunedCMAES, variants of the CMAES for expensive scenarios;
- iSRPSO, a PSO implementing a dynamic learning strategy for velocity updating;
- humanCog, a 3-layer architecture that mimics human cognitive behaviour.

The TS comparison among them, AsBeC and AsBeC_tuned is presented in Table VI. Moreover, function-by-function mean final residual achievements are reported in Table VII and Table VIII, with best results for each function highlighted in gray. It is evident that the AsBeC_tuned obtains a remarkable overall quality, often reaching lower values that the best method, i.e. the MVMO. Differently from the other algorithms, AsBeC_tuned is able to solve the quadratic problems, even if ill-conditioned. In almost all the other functions, AsBeC_tuned approach the best methods. Notice that TS score favors the improvements made on the most difficult function, the test function 10 in 30D. Since both the AsBeC_tuned and the MVMO are able to better minimize this problem, their scores are much lower than the others.

Lastly, the distributions of AsBeC final results have been statistically compared to the best algorithms among the others, according to their final mean. Wilcoxon-Mann-Whitney rank-sum tests [16] with significance level 0.05 are used to test both the hypothesis that AsBeC results are stochastically better and the opposite hypothesis that the other algorithm has better results. We report in Table VII and Table VIII with the symbol (S) the cases where the test is passed. The distributions of the two groups not always differ significantly, but it is worth notice that the number of repetitions (20) could be too low for assessing statistical significance.

VI. Conclusions

Achieving fast and robust improvements in single-objective optimization problems involving expensive analyses means saving precious time and resources. The AsBeC, a new swarm-based algorithm hybridized with interpolation strategies, tackles this problem showing remarkable competitiveness and robustness to different problems, also in comparison with other methods specifically tuned for the used benchmark, i.e., the expensive case of the Special Session on Real-Parameter Single Objective Optimization at CEC 2016.

These results pave the way for a useful application of the proposed algorithm, especially in engineering and in other

TF	**AsBeC_tuned**	MVMO	CMAS-ES_QR	TunedCMAES	iSRPSO	humanCog
1	0.0E+00 (S)	1.9E+02	4.4E+06	1.2E+06	7.4E+06	3.3E+09
2	0.0E+00 (S)	1.7E-02	2.6E+04	4.8E+04	3.2E+04	7.8E+04
3	8.3E+00	9.4E+00	2.8E+00 (S)	7.6E+00	6.6E+00	1.1E+01
4	4.2E+02	4.6E+02	1.7E+03	1.3E+03	9.3E+02	2.1E+03
5	2.0E+00	1.1E+00 (S)	3.2E+00	2.8E+00	2.5E+00	2.8E+00
6	6.2E-01	3.3E-01 (S)	4.2E-01	6.0E-01	5.3E-01	3.6E+00
7	4.5E-01 (S)	6.4E-01	5.5E-01	6.3E-01	5.7E-01	2.7E+01
8	1.0E+01	4.1E+01	4.7E+00 (S)	3.7E+01	5.0E+00	7.8E+03
9	3.8E+00	4.0E+00	4.0E+00	4.2E+00	4.0E+00	4.2E+00
10	1.3E+03	5.0E+02 (S)	2.2E+05	5.4E+05	3.5E+05	1.2E+06
11	7.9E+00	1.2E+01	7.6E+00	7.4E+00	7.3E+00	2.2E+01
12	1.7E+02	2.0E+02	2.4E+02	2.4E+02	1.8E+02	3.1E+02
13	3.3E+02	3.2E+02 (S)	3.3E+02	3.5E+02	3.3E+02	4.3E+02
14	2.0E+02	2.1E+02	2.0E+02 (S)	2.0E+02	2.0E+02	2.2E+02
15	2.6E+02	4.8E+02	3.8E+02	4.4E+02	3.0E+02	4.7E+02

TABLE VII: Final mean residual results, 10D, 500 FEs with best achievements highlighted in gray

TF	**AsBeC_tuned**	MVMO	CMAS-ES_QR	TunedCMAES	iSRPSO	humanCog
1	0.0E+00 (S)	2.1E+03	8.5E+05	1.5E+06	7.2E+08	4.7E+10
2	0.0E+00 (S)	6.9E-03	9.2E+04	1.4E+05	7.7E+04	1.1E+05
3	3.3E+01	3.8E+01	1.2E+01 (S)	2.4E+01	2.6E+01	4.1E+01
4	3.4E+03	1.4E+03 (S)	6.7E+03	6.1E+03	5.4E+03	8.0E+03
5	3.0E+00	1.7E+00 (S)	4.6E+00	3.1E+00	4.2E+00	4.4E+00
6	6.9E-01	5.2E-01	7.3E-01	7.2E-01	6.4E-01	5.0E+00
7	5.0E-01	4.4E-01 (S)	7.5E-01	7.3E-01	5.7E-01	8.9E+01
8	2.4E+03	4.0E+02	1.7E+01 (S)	2.8E+01	6.3E+02	5.2E+06
9	1.3E+01	1.3E+01	1.3E+01	1.4E+01	1.4E+01	1.4E+01
10	2.6E+04	9.3E+04	3.2E+06	4.9E+06	6.8E+06	5.6E+07
11	5.6E+01	1.4E+02	2.5E+01	2.1E+01 (S)	5.1E+01	2.8E+02
12	6.9E+02	8.6E+02	6.3E+02	7.7E+02	7.4E+02	1.6E+03
13	4.4E+02	3.4E+02 (S)	3.8E+02	4.1E+02	4.0E+02	8.4E+02
14	2.7E+02	2.8E+02	2.4E+02 (S)	2.5E+02	2.7E+02	3.9E+02
15	1.1E+03	1.2E+03	4.9E+02 (S)	8.0E+02	9.5E+02	1.5E+03

TABLE VIII: Final mean residual results, 30D, 1500 FEs with best achievements highlighted in gray

applied sciences. Indeed, some of the basic principles driving the AsBeC already led to interesting results in the field of aeronautical design of turbines, as reported in the authors' works [17] and [18]. Furthermore, an extension of the algorithm to multi objective problems is currently under study.

REFERENCES

[1] S. S. Rao, *Engineering optimization: theory and practice*, 4th ed. John Wiley & Sons, 2009.

[2] C. Floudas and M. Pardolos, *Encyclopedia of Optimization*, 2nd ed. Springer, 2009.

[3] D. Karaboga, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39(3), pp. 459–471, 2007.

[4] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Applied Mathematics and Computation*, vol. 214(1), pp. 108–132, 2009.

[5] A. Bolaji, A. Khader, M. Al-Betar, and M. Awadallah, "Artificial Bee Colony algorithm, its variants and applications: A survey," *Journal of Theoretical and Applied Information Technology*, vol. 47(2), pp. 434–459, 2013.

[6] L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: A review of algorithms and comparison of software implementations," *Journal of Global Optimization*, vol. 56(3), pp. 1247–1293, 2013.

[7] E. Ampellio and L. Vassio, "A hybrid swarm-based algorithm for single-objective optimization problems involving high-cost analyses," *Swarm Intelligence*, no. doi:10.1007/s11721-016-0121-6, 2016.

[8] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[9] M. El-Abd, "Opposition-based artificial bee colony algorithm," in *Proceedings of the 13th annual conference on genetic and evolutionary computation*. ACM, 2011, pp. 109–116.

[10] N. Hansen, "The CMA evolution strategy: A comparing review," in *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*. Springer, 2006, pp. 75–102.

[11] S. Swarzberg, G. Seront, and H. Bersini, "S.T.E.P.: the easiest way to optimize a function," in *Proceedings of the 1st IEEE conference on evolutionary computation*. IEEE press, 1994, pp. 519–524.

[12] P. Baudiš and P. Pošík, "Global line search algorithm hybridized with quadratic interpolation and its extension to separable functions," in *Proceedings of the 2015 annual conference on genetic and evolutionary computation*. ACM, 2015, pp. 257–264.

[13] D. Karaboga and B. Basturk, "Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems," in *Proceedings of the 12th International Fuzzy Systems Association World Congress*. Springer, 2007, pp. 789–798.

[14] B. Y. Qu, J. J. Liang, Z. Y. Wang, Q. Chen, and P. N. Suganthan, "Novel benchmark functions for continuous multimodal optimization with comparative results," *Swarm and Evolutionary Computation*, vol. 26, 2016.

[15] Q. Chen, B. Liu, Q. Zhang, J. J. Liang, P. N. Suganthan, and B. Y. Qu, "Problem definitions and evaluation criteria for CEC 2015 special session on bound constrained single-objective computationally expensive numerical optimization," Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, Tech. Rep., 2015.

[16] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, vol. 18(1), pp. 50–60, 1947.

[17] F. Bertini, L. Dal Mas, L. Vassio, and E. Ampellio, "Multidisciplinary optimization for gas turbines design," in *XXII AIDAA Conference*, 2013, http://arxiv.org/pdf/1402.0420v1.

[18] E. Ampellio, F. Bertini, A. Ferrero, F. Larocca, and L. Vassio, "Turbomachinery design by a swarm-based optimization method coupled with a CFD solver," *Advances in Aircraft and Spacecraft Science*, vol. 3(2), pp. 149–170, 2016.