**2013 IEEE Congress on Evolutionary Computation**
**June 20-23, Cancún, México**

# Benchmark Results for a Simple Hybrid Algorithm on the CEC 2013 Benchmark Set for Real-parameter Optimization

Tianjun Liao
IRIDIA, CoDE,
Université Libre de Bruxelles
Brussels, Belgium
Email: tliao@ulb.ac.be

Thomas Stützle
IRIDIA, CoDE,
Université Libre de Bruxelles
Brussels, Belgium
Email: stuetzle@ulb.ac.be

*Abstract*—**In this article, we benchmark a new hybrid algorithm for continuous optimization on the 28 functions for the CEC 2013 special session and competition on real-parameter optimization. Our algorithm makes a loose coupling of (i) IPOP-CMA-ES, an advanced evolution strategy with covariance matrix adaptation integrated with an occasional restart strategy and increasing population size, and (ii) an iterated local search (ILS) algorithm that repeatedly applies a different local search from CMA-ES to perturbations of previous high-quality solutions. The central idea of the hybrid algorithm is to let IPOP-CMA-ES and ILS compete in an initial competition phase and then the winner of the two algorithms is deployed for the remainder of the computation time. A cooperative element between the two algorithms is implemented through a solution exchange from IPOP-CMA-ES to ILS. Hence, one may classify this algorithm as a loosely coupled cooperative–competitive algorithm for continuous optimization.**

**We compare the computational results of this hybrid algorithm to the default version and a tuned version of IPOP-CMA-ES to illustrate the improvement that is obtained through this hybrid algorithm. This comparison is interesting since IPOP-CMA-ES is a state-of-the-art algorithm which somehow has become a standard benchmark to compare against for any new algorithmic proposals for continuous optimization. Our computational results show that the proposed hybrid algorithm performs significantly better than the default and tuned IPOP-CMA-ES variants on the problems of dimension 30 and 50. Thus, these results also indicate that the hybrid algorithm reaches very high performance on the CEC 2013 benchmark set.**

## I. Introduction

The development of algorithms for continuous optimization has a long tradition in the field of evolutionary computation. Over the years, many new, increasingly sophisticated algorithms have been proposed so that several of these have become state-of-the-art algorithms for black-box continuous optimization [1], [2], [3], [4]. A particular role in this development have played effective local search algorithms. In fact, the covariance matrix adaptation evolution strategy (CMA-ES) [5], [6], [7] is such an effective local optimizer that adapts the search step size and the search direction in dependence of the characteristics of the particular function to be optimized and in dependence of the search state. The extension of CMA-ES with occasional restarts and an increasing population size has proved to be surprisingly effective and the resulting IPOP-CMA-ES algorithm was identified as the top performer in the CEC 2005 special session on real parameter optimization [8]. Since that time, several newly developed algorithms in continuous optimization have made use of CMA-ES as a local search routine in hybrid algorithms. One class of approaches embeds CMA-ES into hybrid algorithms, where other population-based methods play the role of a global optimizer and occasionally solutions are improved by CMA-ES. Examples are the integration of CMA-ES into evolutionary algorithms [2], particle swarm optimization [9], or differential evolution [10]. Alternative approaches try to apply techniques from algorithm portfolios or algorithm selection. For example, Peng et al. [11] include four continuous optimizers in an algorithm portfolio and report some improvements over IPOP-CMA-ES in some classical benchmark functions.

In our research, we have developed a novel hybrid algorithm that consists of the loose coupling of IPOP-CMA-ES and a new iterated local search (ILS) [12] algorithm for continuous optimization. This coupling is based on an initial competition phase, where the two algorithms compete for further deployment in a successive deployment phase. In a sense, the central idea of our hybrid is to perform algorithm selection [13], [14] by short runs of the algorithms itself and then deploying the better performing one for the remainder of the computation time. Instead of pure competition, our hybrid is based also on some cooperative elements. In fact, first IPOP-CMA-ES is run and the best solution identified by IPOP-CMA-ES is given as input to the ILS algorithm [15]. In this paper, we benchmark this new algorithm on the CEC 2013 benchmark set [16]. We also compare the performance of our iCMAES-ILS algorithm to two versions of IPOP-CMA-ES, one using the default parameter settings as in [1] and one with the parameter settings tuned by an automatic algorithm configuration method [17]. Our analysis shows that iCMAES-ILS performs better than both competitors on the CEC 2013 benchmarks of dimension 30 and 50.

The remainder of this article is structured as follows. In Section II we present the hybrid algorithm and in Section III we explain the experimental setup we used. The experimental results and some analysis of the algorithm is given in Section IV. We end with some concluding remarks in Section V.

**Algorithm 1** Outline of IPOP-CMA-ES

```
Require: Candidate solution $s$ and termination criterion
Ensure: the best found solution
  while termination criterion is not satisfied do
    while stopping criterion for restart is not satisfied do
      {CMA-ES iterations}
      Sample a new population
      Selection
      Adapt step size and covariance matrix
    end while
    Increase population size and sample an initial solution
  end while
```

## II. iCMAES-ILS

This section gives an overview of the hybrid iCMAES-ILS algorithm, which is composed of IPOP-CMA-ES and an ILS algorithm.

### A. *IPOP-CMA-ES*

CMA-ES essentially is an $(\mu, \lambda)$-evolution strategy. It uses a multivariate normal distribution to generate at each iteration new solutions. The covariance matrix of this normal distribution is adapted after each iteration to direct the generation of new solutions. The way this covariance matrix is adapted and the way how solutions are generated makes CMA-ES invariant against linear transformations of the search space. This is a particularly useful feature on functions that are rotated, as are several of the CEC 2005 and 2013 functions. We refer to Hansen [18] for a detailed description of the main principles and details of CMA-ES.

Unfortunately, CMA-ES may be trapped in local optima regions of the search space and it is to some extent sensitive to the initial population size that is used. Auger and Hansen [1] therefore have proposed to extend CMA-ES by a restart mechanism that occasionally reinitializes the search and that uses an increasing population size between successive executions of CMA-ES. The resulting IPOP-CMA-ES algorithm was identified as the best performing algorithm in the special session on real parameter optimization at CEC'05. It therefore is considered nowadays as one main representative of the state-of-the-art in black-box continuous optimization.

We give an outline of the IPOP-CMA-ES algorithm in Algorithm 1. Starting with some initial generation, the algorithm repeatedly runs CMA-ES (inner while loop in Algorithm 1). At each iteration, CMA-ES first generates a solutions, selects the $\mu$ best solutions and adapts the step size and the covariance matrix (thus, also the search direction). The default parameter settings of IPOP-CMA-ES are the following. The initial population size is $\lambda = 4 + \lfloor 3\ln(D) \rfloor$ and the number of search points selected for the parent population is $\mu = \lfloor 0.5\lambda \rfloor$. The initial step-size is $\sigma^{(0)} = 0.5(B - A)$, where $A$ and $B$ are the lower and upper bounds of the search space range. IPOP-CMA-ES increases the population size by a factor of $d = 2$ at each restart. Three parameters are used to decide when to restart CMA-ES. The parameters are *stopTolFunHist*, *stopTolFun* and *stopTolX*, which refer to the amount of improvement of the value of the best solution found during the last $10 + \lceil 30D/\lambda \rceil$ generations, the function values of the recent generation, and the standard deviation of the normal distribution in all coordinates, respectively.

### B. *ILS*

Iterated local search (ILS) [12] is a metaheuristic that repeatedly starts local search algorithms at solutions that are obtained through a perturbation of an incumbent solution. The goal of the solution perturbation is to allow the search process to escape from local optima and to identify new, promising starting solutions for local search. An acceptance criterion further implements a scheme as to which solution is used as the incumbent, that is, from which candidate solution the search process is continued. While ILS is a rather widely used method in combinatorial optimization, applications to continuous optimization problems are rather recent. Promising results have been obtained by Gimmler in his Master thesis [19] and where also reported by Kramer [20]. For our hybrid algorithm, we developed a new iterated local search algorithm (labeled henceforth as ILS). In our ILS algorithm, we use as a high performing local search algorithm Mtsls1 [21], [22]. The choice of Mtsls1 was inspired through the excellent results that were obtained in various hybrid algorithms for continuous optimization that made use of this local search algorithm [3], [22].

As a starting solution for Mtsls1, some candidate solution $s = (s_1, s_2, \ldots, s_D)$ is generated, for example, uniformly at random inside the search range. The initial step size $ss$ is set to $0.5 \times (B - A)$ as default [21]. In each iteration of Mtsls1, one dimension is searched at a time and the dimensions are visited in a fixed order. In a dimension $i$, the search works as follows. First, the value $s_i$ is modified as $s'_i \leftarrow s_i - ss$ and the resulting solution $s'$ is evaluated. If $f(s') < f(s)$, $s_i \leftarrow s'_i$ and the search continues in dimension $i+1$; otherwise, $s''_i \leftarrow s_i + 0.5 \times ss$ and the candidate solution $s''$ is evaluated. If $f(s'') < f(s)$, $s_i \leftarrow s''_i$ and the search is continued in dimension $i + 1$. If both tests fail, $s_i$ remains unchanged and the same process is applied to dimension $i+1$. If no improvement is found in any of the dimensions during one iteration of Mtsls1, the next Mtsls1 iteration uses halve the search step size. In our implementation, a parameter *LSIterations* determines the maximum number of iterations in one execution of Mtsls1.

Algorithm 2 present the proposed ILS algorithm in pseudocode form. It uses two solutions $s$ and $s_{\text{best}}$ as input for purposes we explain later. A standard ILS algorithm [12] would directly be obtained again by setting $s_{\text{best}} = s$. In the ILS algorithm, first a local search is run and a new solution $s_{\text{new}}$ is generated. If the new solution $s_{\text{new}}$ improves upon the best-so-far solution $s_{\text{best}}$, we apply a refinement local search to $s_{\text{new}}$ (in other words, we do not perturb $s_{\text{new}}$ before applying to it the local search procedure). Otherwise, a perturbation is used to generated a modified solution to start the local search from. The perturbation is biased towards the best-so-far solution $s_{\text{best}}$ by using the equation $s = s_{\text{rand}} + r \times (s_{\text{best}} - s_{\text{rand}})$, where $r$ is a random number chosen uniformly at random in $[0, 1)$ and $s_{\text{rand}}$ is a solution that is generated uniformly at random in the search range.

### C. *iCMAES-ILS*

iCMAES-ILS combines IPOP-CMA-ES and ILS into a hybrid algorithm. It consists of two phases, a competition phase and a deployment phase. In the competition phase, IPOP-CMA-ES and ILS use the same initial candidate solution

**Algorithm 2** Outline of the ILS algorithm

```
Require: Candidate solution $s$, $s_{\text{best}}$ and termination criterion
Ensure: the best found solution
  while termination criterion is not satisfied do
    $s_{\text{new}} \leftarrow$ LS ($s$, $ss$, LSIterations)            {Local search procedure}
    if $f(s_{\text{new}}) < f(s_{\text{best}})$ then
      $s \leftarrow s_{\text{new}}$
      $s_{\text{best}} \leftarrow s_{\text{new}}$
    else
      $s \leftarrow s_{\text{rand}} + r \times (s_{\text{best}} - s_{\text{rand}})$ {Perturbation}
    end if
  end while
```

**Algorithm 3** Outline of iCMAES-ILS

```
Require: Candidate solution $s$ and termination criterion (TotalBudget)
Ensure: the best found solution
  $CompBudget \leftarrow comp_r \times TotalBudget$
  $s_{\text{best}} \leftarrow$ IPOP-CMA-ES ($s$, CompBudget)
  $s'_{\text{best}} \leftarrow$ ILS ($s$, $s_{\text{best}}$, CompBudget)
  {Heuristic deployment }
  if $f(s'_{\text{best}} < f(s_{\text{best}})$ then
    {ILS' solution is better than that of IPOP-CMA-ES }
    ILS ($s'_{\text{best}}$, $s'_{\text{best}}$, $TotalBudget - 2 \times CompBudget$)
  else
    {IPOP-CMA-ES' solution is better than that of ILS }
    IPOP-CMA-ES ($s_{\text{best}}$, $TotalBudget - 2 \times CompBudget$)
  end if
```

and are executed one after another with a same fixed budget of function evaluations, *CompBudget*. After this competition phase, the algorithm that found the better solution between these two is deployed for the remaining budget of function evaluations. Specifically, if ILS finds a better solution than IPOP-CMA-ES, iCMAES-ILS applies ILS for the remaining budget. Otherwise, iCMAES-ILS restarts IPOP-CMA-ES from the best-so-far solution with its initially defined parameter settings (thus, also with the same population size as initially defined). Note that a possible restart of IPOP-CMA-ES is then triggered by the end of the competition phase, that is, not because of the termination criterion of IPOP-CMA-ES.

The competition phase of iCMAES-ILS, however, includes also some cooperative aspects. In particular, the input $s_{\text{best}}$ of ILS is set to the best-so-far solution returned by the IPOP-CMA-ES. Thus, a limited cooperation during this initial phase of iCMAES-ILS is implemented through a solution transfer from IPOP-CMA-ES to ILS. This transferred solution can influence the behavior of ILS as it will be used in the acceptance decision and in the perturbation. In an independent ILS algorithm, the input $s_{\text{best}}$ would be set to the same as the input candidate solution $s$.

In iCMAES-ILS we need to set the parameter *CompBudget*, which defines the length of the initial short runs of IPOP-CMA-ES and ILS. We set *CompBudget*$=$ $comp_r \times$*TotalBudget*, where *TotalBudget* is the maximum number of function evaluations in each algorithm run and the actual parameter to be defined, thus, becomes $comp_r$. An algorithmic outline of iCMAES-ILS is given in Algorithm 3. Initially, the computational budget of IPOP-CMA-ES and ILS for the initial competition phase is defined. Then, IPOP-CMA-ES and ILS are executed in sequence. The winner of the two is then deployed for the remainder of the computation time until the total computational budget is exhausted.

TABLE I. GIVEN IS THE INFORMATION ON THE COMPUTATIONAL COMPLEXITY OF THE ALGORITHM. FOR DETAILS ON THE REQUESTED MEASUREMENTS SEE [16]. $T0$ IS THE TIME TO EXECUTE A SPECIFIC `FOR` LOOP; $T1$ IS THE TIME TAKEN OF 200 000 FUNCTION EVALUATIONS FOR FUNCTION $f_{14}$, AND $\hat{T2}$ THE AVERAGE TIME TAKEN BY ICMAES-ILS TO RUN FOR 200 000 FUNCTION EVALUATIONS ON FUNCTION $f_{14}$.

| | $T0$ | $T1$ | $\hat{T2}$ | $(\hat{T2} - T1)/T0$ |
|---|---|---|---|---|
| $D$=10 | 0.16s | 1.44 | 1.62 | 1.13 |
| $D$=30 | 0.16s | 4.94 | 5.36 | 2.65 |
| $D$=50 | 0.16s | 15.5 | 16.2 | 4.25 |

## III. EXPERIMENTAL SETUP

The CEC'13 test-suite consists of 28 benchmark functions of dimensions 10, 30 and 50. We followed the protocol described in [16] for the CEC'13 test-suite, that is, the maximum number of function evaluations is $10\,000 \times D$. The investigated algorithms were independently run 51 times on each function. We report error values defined as $f(\boldsymbol{s}) - f(\boldsymbol{s}^*)$, where $\boldsymbol{s}$ is a candidate solution and $\boldsymbol{s}^*$ is the optimal solution. Error values lower than $10^{-8}$ are clamped to $10^{-8}$, which is used as the zero threshold. To ensure that the final solution obtained by iCMAES-ILS is inside the bounds, the bound constraints of the benchmark functions are enforced by clamping the variable of each generated solution that violates the bound constraints to the nearest value on the bounds before evaluating a solution. The same bound handling mechanism is used in ILS.

For the implementation of IPOP-CMA-ES, we have used the `C` version available from the website of Hansen (https://www.lri.fr/~hansen/cmaes_inmatlab.html) extended with a mechanism to clamp solutions outside the search range to the closest feasible solution on the bounds. In addition, we compare the performance of our iCMAES-ILS to the default version of IPOP-CMA-ES, that is, to the version that uses the parameter settings as proposed in [1], as well as to a version of IPOP-CMA-ES that has undergone the same tuning process as iCMAES-ILS [17]. Note that for these two variants we use the same `C` version that is also used in our implementation of iCMAES-ILS (and also the same bound handling mechanism).

All the experiments have been executed on cluster nodes each being equipped with two Intel Xeon E5410 quadcore CPUs running at 2.33 GHz with $2 \times$ 6MB L2 cache and 8 GB RAM. However, each run of iCMAES-ILS uses only a single core. In Table I we give the requested information by the CEC'2013 special session on the speed of computation on our machines and the execution time of iCMAES-ILS.

### A. Parameter configuration

For benchmarking the iCMAES-ILS algorithm on the CEC 2013 benchmark set, we do not apply any parameter tuning that is specific to this particular benchmark set. In fact, we use directly the parameter values that we recommended in a study of the iCMAES-ILS algorithm, which is currently submitted for publication [15]. In the following, we shortly describe the configuration process that we had applied in that study. For the tuning of the parameters of iCMAES-ILS, we applied iterated F-race [23], a racing algorithm for algorithm configuration that is included in the publicly available `irace` package [24]. (For details on the working of iterated F-race we refer to [23] and the user guide in [24].) In total, we exposed eleven parameters that directly control internal parameters of iCMAES-ILS.

These eleven parameters are given in Table II, together with the internal parameter of iCMAES-ILS controlled by each of them and the range considered for tuning. The setting of iterated F-race we used are the default [24]. The budget of each run of iterated F-race is set to 5 000 runs of iCMAES-ILS. The performance measure is the fitness error value of each instance. Iterated F-race handles various parameter types such as categorical (c), ordinal (o) and continuous (r). The inputs to iterated F-race are the parameter ranges and a set of training instances.

As the training instances, we used the 10-dimensional benchmark functions from a recent special issue of the Soft Computing journal (labeled as SOCO) [25], [26] on large-scale continuous optimization. We originally did so to separate between the functions on which we actually search for high-performance parameter settings of iCMAES-ILS and the test functions on which we actually benchmark the algorithm. Note that the SOCO benchmark set consists of 19 functions. Four of these functions were the same as in the older CEC'05 benchmark set and we therefore removed them from the training set. Training functions are then sampled in a random order and possibly repeatedly fed into the iterated F-race, in case a race would require to see more than the available 15 functions. The maximum number of different runs of iCMAES-ILS is set to $5\,000$ and each run of iCMAES-ILS is given a maximum of $5\,000 \times D$ function evaluations. Note that one run of iCMAES-ILS corresponds to the execution of one specific parameter configuration on a single function. The tuned settings of iCMAES-ILS are presented in the last column of Table II.[1] By separating in this way the training set for parameter tuning and the test set on which to evaluate the algorithm, we avoid a potential overtuning of iCMAES-ILS to the specific set of CEC 2013 benchmark functions. Also note that the parameter tuning of iCMAES-ILS on the SOCO benchmark functions is rather quick: the overall tuning process on the 10-dimensional SOCO benchmark functions took less than one hour of CPU time on a single core of our machines, that is, the tuning time is actually negligible when compared to the time spent for implementation. In fact, this also illustrates the advantage of automatic configuration of algorithms when compared to the traditional, manual way of determining reasonable parameter settings.

When compared with the version of iCMAES-ILS described in [15], we introduced two minor additional modifications. The first is that the initial solution for CMA-ES around which the sampling of new candidate solutions occurs is the best of an initial sample of $D$ solutions generated according to a uniform distribution in the feasible search range (or initialization range). The second is that we limited ad hoc the maximum population size of IPOP-CMA-ES to 200, to avoid a too strong growth of the population size between restarts.

TABLE II. GIVEN ARE THE PARAMETER SETTINGS OBTAINED AFTER SOME LIMITED TUNING FOR ICMAES-ILS. GIVEN ARE THE PARAMETER, A SHORT DESCRIPTION OF THE EFFECT OF THE PARAMETER, AND THE TUNED VALUES.

| Alg Components | **Parameter** | **Internal parameter** | | | **Values** |
|---|---|---|---|---|---|
| IPOP-CMA-ES | $a$ | Init pop size: $\lambda_0$ | $=$ | $4 + \lfloor a \ln(D) \rfloor$ | 9.687 |
| | $b$ | Parent size: $\mu$ | $=$ | $\lfloor \lambda / b \rfloor$ | 1.614 |
| | $c$ | Init step size: $\sigma_0$ | $=$ | $c \cdot (B - A)$ | 0.6825 |
| | $d$ | IPOP factor: *ipop* | $=$ | $d$ | 3.245 |
| | $e$ | *stopTolFun* | $=$ | $10^e$ | $-9.023$ |
| | $f$ | *stopTolFunHist* | $=$ | $10^f$ | $-10.82$ |
| | $g$ | *stopTolX* | $=$ | $10^g$ | $-16.26$ |
| ILS | $i_r$ | *LSIterations* | $=$ | $i_r \times D$ | 1 |
| | $ss_r$ | *ss* | $=$ | $ss_r \times (B - A)$ | 0.6703 |
| | $bias_e$ | *BiasExtent* | $=$ | $bias_e$ | 0.01910 |
| Competition | $comp_r$ | *CompBudget* | $=$ | $comp_r \times TotalBudget$ | 0.15 |

## IV. EXPERIMENTAL RESULTS

In this section, we present the computational results with iCMAES-ILS. To obtain an indication of the performance of our hybrid algorithm, we decided to first compare it to the default version of IPOP-CMA-ES. In Figure 1 we give correlation plots where on the $x$-axis is given the average error observed with iCMAES-ILS and on the $y$-axis the average error obtained by IPOP-CMA-ES, each result averaged across 51 independent runs of each algorithm and for each dimension $D \in \{10, 30, 50\}$. Since most results focus around the diagonal, we provided also a zoom into this area. In the plots are also given the number of times the average error of iCMAES-ILS is smaller than that of IPOP-CMA-ES (win), the same (draw) or worse (lose). We use the two-sided Wilcoxon matched-pairs signed-rank test to determine whether the differences observed are statistically significant. In fact, for dimensions 30 and 50, the differences observed are significant, while for dimensions 10 the null hypothesis of equal performance could not be rejected. In fact, observing the number of times our hybrid algorithm gives lower average errors than IPOP-CMA-ES, we can observe that its advantage increases with increasing dimensionality.

One may ask whether the usage of an automatic algorithm configuration method for setting the parameters of iCMAES-ILS is the main responsible for its significantly improved performance over IPOP-CMA-ES with default parameter settings. To examine this question, we compared iCMAES-ILS also to a tuned version of IPOP-CMA-ES to which we refer as IPOP-CMA-ESt. In fact, it was observed that tuning may further improve the performance of IPOP-CMA-ES [17], [27]. For the tuning of IPOP-CMA-ESt, we applied the same setup as for iCMAES-ILS (see also [17] for a careful analysis of the performance of the tuned IPOP-CMA-ES). The results of this comparison are given in Figure 2. As can be seen, the number of times the hybrid algorithm improves over IPOP-CMA-ESt is less than in the comparison to IPOP-CMA-ES using default parameter settings. However, for the functions of dimensions 30 and 50, statistically significant improvements of iCMAES-ILS over IPOP-CMA-ESt have been observed by the two-sided Wilcoxon matched-pairs signed-rank.

It is also interesting to examine the optimization behavior of iCMAES-ILS and, in particular, the role that is played by the ILS algorithm. In Table III, we count the number of times IPOP-CMA-ES either solves a function to an error value below the zero threshold already during the initial competition phase in all runs, in how many functions the ILS algorithm is never chosen to be run in the deployment phase, and the number of functions in which the ILS algorithm is chosen at least once for further execution in the deployment phase. Note that even if ILS is not further executed in the deployment phase, it triggers a restart of the IPOP-CMA-ES algorithm from scratch and in

[1]The tuned setting essentially imply a larger initial population size than in the default settings of IPOP-CMA-ES and a faster increase of the population size. For the ILS component, the parameter settings result in relatively short local searches.

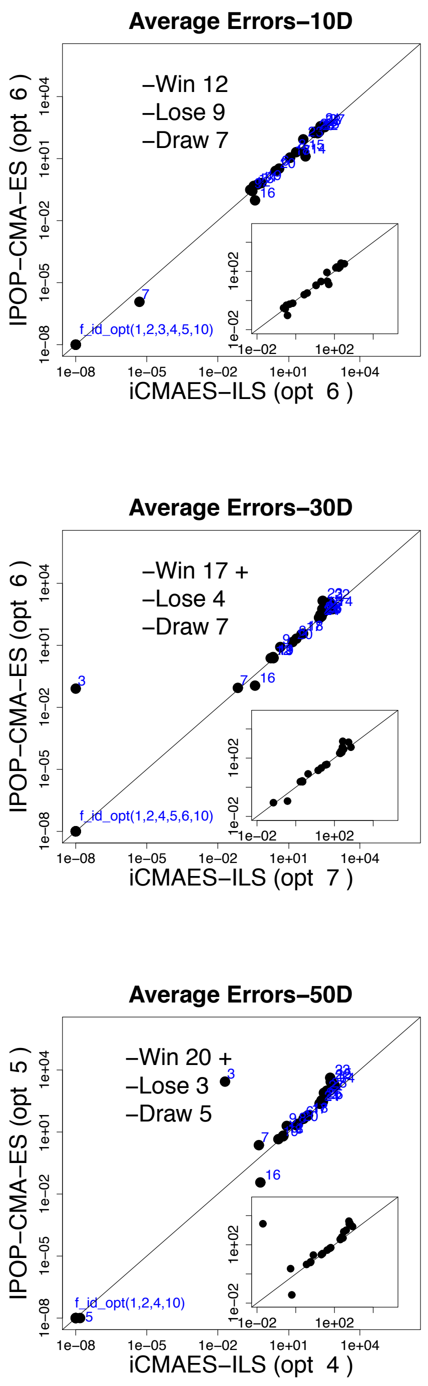

Fig. 1. Correlation plot of IPOP-CMA-ES and iCMAES-ILS on CEC13 benchmark functions of $D = 10$ (top), $D = 30$ (middle), and $D = 50$ (bottom). Each point represents the average error value obtained by either of the two algorithms. A point on the upper triangle delimited by the diagonal indicates better performance for the algorithm on the x-axis (iCMAES-ILS); a point on the lower right triangle indicates better performance for the algorithm on the y-axis (IPOP-CMA-ES). The number labeled beside some outstanding points represent the index of the corresponding function. The comparison results of the algorithm on the x-axis (that is, iCMAES-ILS) are presented in the form of -win, -draw, -lose, respectively. We marked with a $+$ symbol those cases in which there is a statistically significant difference as indicated by a two-sided Wilcoxon matched-pairs signed-rank test at the 0.05 $\alpha$-level between the algorithms. The number of opt on the axes shows the number of means that is lower than the zero threshold, obtained by the corresponding algorithm.

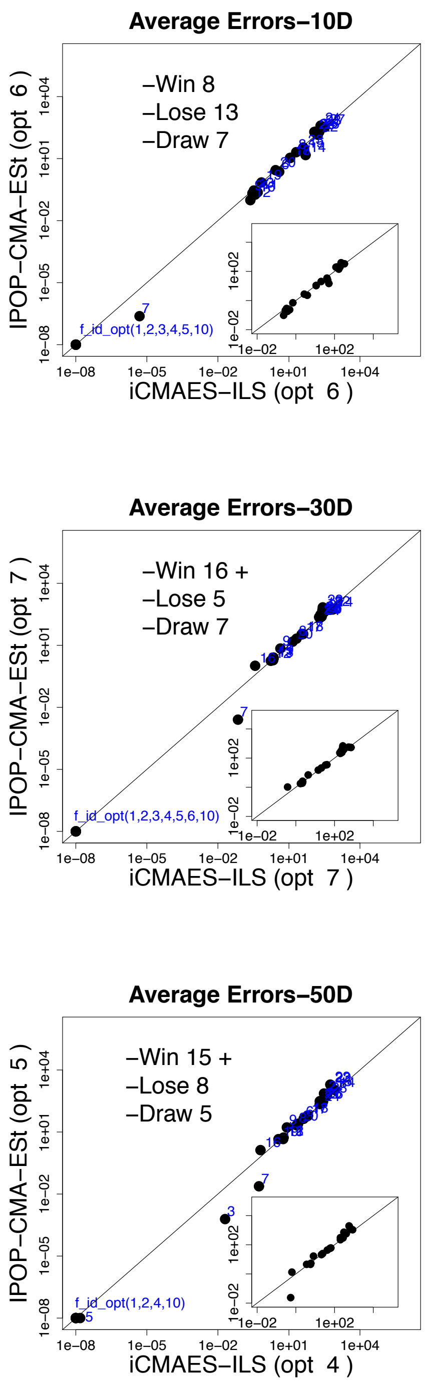

Fig. 2. Correlation plot of the tuned version of IPOP-CMA-ES (referred to as IPOP-CMA-ESt) and iCMAES-ILS on CEC13 benchmark functions of $D = 10$ (top), $D = 30$ (middle), and $D = 50$ (bottom). Each point represents the average error value obtained by either of the two algorithms. A point on the upper triangle delimited by the diagonal indicates better performance for the algorithm on the x-axis (iCMAES-ILS); a point on the lower right triangle indicates better performance for the algorithm on the y-axis (IPOP-CMA-ESt). The number labeled beside some outstanding points represent the index of the corresponding function. The comparison results of the algorithm on the x-axis (that is, iCMAES-ILS) are presented in the form of -win, -draw, -lose, respectively. We marked with a $+$ symbol those cases in which there is a statistically significant difference as indicated by a two-sided Wilcoxon matched-pairs signed-rank test at the 0.05 $\alpha$-level between the algorithms. The number of opt on the axes shows the number of means that is lower than the zero threshold, obtained by the corresponding algorithm.

TABLE III. SUMMARY FOR THE NUMBER OF FUNCTIONS SOLVED BY IPOP-CMA-ES IN THE COMPETITION PHASE IN ALL RUNS (NUMF-SOLVECP), THE NUMBER OF FUNCTIONS WHERE ILS IS APPLIED AS THE EXTERNAL RESTART TRIGGER OF IPOP-CMA-ES IN ALL RUNS (NUMF-ILSTRIGGER), AND THE NUMBER OF FUNCTIONS WHERE ILS IS APPLIED FOR THE DEPLOYMENT PHASE IN AT LEAST ONE RUN (NUMF-ILSDEPLOY).

| Dim | NumF-solvecp | NumF-ILStrigger | NumF-ILSdeploy |
|---|---|---|---|
| 10 | 4 | 8 | 16 |
| 30 | 2 | 15 | 11 |
| 50 | 2 | 16 | 10 |

TABLE IV. GIVEN ARE THE BEST, WORST, MEDIAN, AND MEAN FUNCTIONS VALUES AS WELL AS THE STANDARD DEVIATION OF THE FUNCTION VALUES FOR EACH FUNCTIONS OF DIMENSION 10 FROM THE CEC 2013 BENCHMARK SET.

| Func. | Best | Worst | Median | Mean | Std |
|---|---|---|---|---|---|
| 1 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 2 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 3 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 4 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 5 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 6 | 1.00E−08 | 9.81E+00 | 6.57E−05 | 3.89E+00 | 4.80E+00 |
| 7 | 1.00E−08 | 7.00E−05 | 3.56E−07 | 4.91E−06 | 1.34E−05 |
| 8 | 2.01E+01 | 2.05E+01 | 2.04E+01 | 2.04E+01 | 7.61E−02 |
| 9 | 1.00E−08 | 2.66E+00 | 4.18E−05 | 2.86E−01 | 5.38E−01 |
| 10 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 11 | 1.00E−08 | 1.99E+00 | 1.30E−07 | 4.77E−01 | 5.71E−01 |
| 12 | 1.00E−08 | 9.95E−01 | 1.00E−08 | 2.34E−01 | 4.26E−01 |
| 13 | 1.00E−08 | 9.95E−01 | 1.00E−08 | 3.33E−01 | 4.73E−01 |
| 14 | 1.87E−01 | 4.05E+02 | 1.19E+01 | 5.08E+01 | 9.99E+01 |
| 15 | 1.87E−01 | 6.86E+02 | 1.20E+01 | 4.42E+01 | 1.02E+02 |
| 16 | 5.30E−07 | 1.21E+00 | 2.36E−01 | 3.73E−01 | 3.00E−01 |
| 17 | 1.02E+01 | 1.30E+01 | 1.12E+01 | 1.12E+01 | 5.08E−01 |
| 18 | 1.04E+01 | 1.23E+01 | 1.12E+01 | 1.12E+01 | 5.01E−01 |
| 19 | 3.38E−01 | 9.98E−01 | 7.09E−01 | 6.98E−01 | 1.50E−01 |
| 20 | 1.38E+00 | 3.54E+00 | 2.84E+00 | 2.72E+00 | 5.24E−01 |
| 21 | 1.01E+02 | 4.00E+02 | 2.01E+02 | 2.18E+02 | 1.11E+02 |
| 22 | 3.07E+01 | 4.17E+02 | 1.46E+02 | 1.66E+02 | 8.10E+01 |
| 23 | 1.07E+01 | 1.07E+02 | 3.66E+01 | 4.08E+01 | 2.08E+01 |
| 24 | 1.07E+02 | 2.00E+02 | 1.19E+02 | 1.32E+02 | 3.25E+01 |
| 25 | 1.12E+02 | 2.05E+02 | 2.00E+02 | 1.92E+02 | 2.47E+01 |
| 26 | 1.03E+02 | 2.00E+02 | 1.16E+02 | 1.18E+02 | 1.30E+01 |
| 27 | 3.00E+02 | 4.00E+02 | 3.00E+02 | 3.25E+02 | 4.20E+01 |
| 28 | 2.28E+00 | 3.00E+02 | 3.00E+02 | 2.24E+02 | 1.01E+02 |

TABLE V. GIVEN ARE THE BEST, WORST, MEDIAN, AND MEAN FUNCTIONS VALUES AS WELL AS THE STANDARD DEVIATION OF THE FUNCTION VALUES FOR EACH FUNCTIONS OF DIMENSION 30 FROM THE CEC 2013 BENCHMARK SET.

| Func. | Best | Worst | Median | Mean | Std |
|---|---|---|---|---|---|
| 1 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 2 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 3 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 4 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 5 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 6 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 7 | 1.00E−08 | 7.98E−01 | 1.00E−08 | 7.01E−02 | 1.56E−01 |
| 8 | 2.08E+01 | 2.10E+01 | 2.09E+01 | 2.09E+01 | 6.23E−02 |
| 9 | 7.10E−03 | 8.06E+00 | 4.53E+00 | 4.34E+00 | 1.72E+00 |
| 10 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 11 | 1.00E−08 | 4.97E+00 | 1.99E+00 | 2.25E+00 | 1.05E+00 |
| 12 | 1.00E−08 | 4.97E+00 | 1.99E+00 | 1.72E+00 | 1.23E+00 |
| 13 | 1.00E−08 | 6.76E+00 | 1.99E+00 | 2.16E+00 | 1.30E+00 |
| 14 | 1.50E+02 | 1.18E+03 | 7.95E+02 | 7.08E+02 | 2.94E+02 |
| 15 | 1.36E+02 | 5.48E+02 | 1.91E+02 | 2.59E+02 | 1.18E+02 |
| 16 | 1.48E−02 | 1.21E+00 | 4.26E−01 | 3.75E−01 | 2.65E−01 |
| 17 | 2.66E+01 | 3.91E+01 | 3.42E+01 | 3.43E+01 | 1.86E+00 |
| 18 | 3.19E+01 | 1.68E+02 | 3.64E+01 | 4.01E+01 | 1.87E+01 |
| 19 | 6.63E−01 | 3.18E+00 | 2.28E+00 | 2.24E+00 | 5.66E−01 |
| 20 | 1.21E+01 | 1.50E+01 | 1.45E+01 | 1.44E+01 | 7.38E−01 |
| 21 | 1.00E+02 | 2.00E+02 | 2.00E+02 | 1.88E+02 | 3.25E+01 |
| 22 | 3.98E+01 | 1.14E+03 | 5.20E+02 | 5.33E+02 | 3.63E+02 |
| 23 | 1.25E+02 | 6.96E+02 | 2.02E+02 | 2.69E+02 | 1.41E+02 |
| 24 | 2.00E+02 | 2.00E+02 | 2.00E+02 | 2.00E+02 | 6.16E−04 |
| 25 | 2.29E+02 | 2.51E+02 | 2.40E+02 | 2.40E+02 | 5.12E+00 |
| 26 | 2.00E+02 | 3.00E+02 | 2.00E+02 | 2.16E+02 | 3.67E+01 |
| 27 | 3.00E+02 | 3.00E+02 | 3.00E+02 | 3.00E+02 | 9.34E−03 |
| 28 | 1.00E+02 | 3.00E+02 | 3.00E+02 | 2.45E+02 | 9.01E+01 |

several such cases we observed that this led to improved results with respect to the sole execution of IPOP-CMA-ES. In the case, where ILS is run at least once, it improves in the initial competition phase upon IPOP-CMA-ES. For various functions, this is rather frequently the case. For example, for dimension 50, for 5 of the 10 functions where ILS is finally deployed, it is deployed in more than halve of the cases.

Finally, the numerical results obtained after the largest number of function evaluations equal to $10\,000D$, are given in Tables IV to VI for the functions of dimension 10, 30, and 50, respectively. Given are the best, worst, median and mean function values that were observed; function values smaller than $10^{-8}$ were approximated to $10^{-8}$.

## V. CONCLUSION

In this paper, we have presented computational results with iCMAES-ILS on the benchmark set of the CEC 2013 special session on real-parameter optimization. iCMAES-ILS is a novel hybrid algorithm that consists of an initial competition phase between IPOP-CMA-ES and an ILS algorithm that uses the Mtsls1 local search procedure [21]. The algorithm of the two that gives a better final solution in this competition phase is then chosen for further deployment until the maximum computation budget. Clearly, the main role in this hybrid algorithm is played by IPOP-CMA-ES, which is known to be a high performing algorithm since the CEC 2005 benchmark competition. However, for various functions, the ILS actually is the winning algorithm in the initial competition phase and, even if it does not win, it triggers an independent restart of the IPOP-CMA-ES algorithm, which occasionally leads to improved performance [15]. When compared to IPOP-CMA-ES with default or tuned parameter settings, we found that iCMAES-ILS yields statistically significantly improved performance for dimensions 30 and 50 on the CEC 2013 benchmark set. We believe that these results are very encouraging, especially when taking into account that the hybrid algorithm is based on a rather simple interaction scheme. In future work, we would like to examine further algorithm selection and portfolio techniques to improve upon current state-of-the-art continuous optimizers.

## ACKNOWLEDGMENTS

The authors would like to thank Dr. Nikolaus Hansen for making publicly available the IPOP-CMA-ES codes. This work was supported by the Meta-X project funded by the Scientific Research Directorate of the French Community of Belgium. The authors also acknowledge support from the FRFC project "*Méthodes de recherche hybrides pour la résolution de problèmes complexes*". Thomas Stützle acknowledges support from the Belgian F.R.S.-FNRS, of which he is a Research Associate. Tianjun Liao acknowledges a fellowship from the China Scholarship Council. This research and its results have also received funding from the COMEX project within the Inter-university Attraction Poles Programme of the Belgian

TABLE VI. GIVEN ARE THE BEST, WORST, MEDIAN, AND MEAN FUNCTIONS VALUES AS WELL AS THE STANDARD DEVIATION OF THE FUNCTION VALUES FOR EACH FUNCTIONS OF DIMENSION 50 FROM THE CEC 2013 BENCHMARK SET.

| Func. | Best | Worst | Median | Mean | Std |
|---|---|---|---|---|---|
| 1 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 2 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 3 | 1.00E−08 | 7.60E−01 | 1.00E−08 | 2.01E−02 | 1.08E−01 |
| 4 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 5 | 1.00E−08 | 3.98E−08 | 1.18E−08 | 1.52E−08 | 6.96E−09 |
| 6 | 7.95E−01 | 4.34E+01 | 4.34E+01 | 4.19E+01 | 7.82E+00 |
| 7 | 1.00E−08 | 4.27E+00 | 2.46E−02 | 5.44E−01 | 1.10E+00 |
| 8 | 2.09E+01 | 2.12E+01 | 2.11E+01 | 2.11E+01 | 6.29E−02 |
| 9 | 6.52E−04 | 1.43E+01 | 8.84E+00 | 8.18E+00 | 3.28E+00 |
| 10 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 1.00E−08 | 0 |
| 11 | 9.95E−01 | 1.09E+01 | 5.97E+00 | 5.94E+00 | 1.98E+00 |
| 12 | 1.99E+00 | 1.09E+01 | 5.97E+00 | 5.77E+00 | 1.81E+00 |
| 13 | 1.99E+00 | 1.44E+01 | 5.12E+00 | 5.73E+00 | 2.64E+00 |
| 14 | 4.12E+01 | 1.98E+03 | 4.76E+02 | 8.59E+02 | 7.10E+02 |
| 15 | 1.37E+02 | 1.96E+03 | 5.74E+02 | 6.42E+02 | 2.97E+02 |
| 16 | 7.45E−03 | 1.31E+00 | 7.02E−01 | 6.28E−01 | 3.53E−01 |
| 17 | 5.41E+01 | 6.06E+01 | 5.76E+01 | 5.75E+01 | 1.57E+00 |
| 18 | 5.66E+01 | 9.11E+01 | 6.07E+01 | 6.43E+01 | 8.39E+00 |
| 19 | 1.97E+00 | 5.28E+00 | 3.48E+00 | 3.62E+00 | 8.79E−01 |
| 20 | 2.17E+01 | 2.50E+01 | 2.45E+01 | 2.44E+01 | 4.52E−01 |
| 21 | 2.00E+02 | 2.00E+02 | 2.00E+02 | 2.00E+02 | 0 |
| 22 | 1.28E+02 | 2.02E+03 | 3.50E+02 | 5.87E+02 | 5.62E+02 |
| 23 | 7.91E+01 | 1.24E+03 | 4.79E+02 | 5.57E+02 | 3.26E+02 |
| 24 | 2.00E+02 | 2.00E+02 | 2.00E+02 | 2.00E+02 | 5.39E−02 |
| 25 | 2.62E+02 | 2.91E+02 | 2.74E+02 | 2.74E+02 | 6.24E+00 |
| 26 | 2.00E+02 | 3.01E+02 | 2.00E+02 | 2.41E+02 | 4.97E+01 |
| 27 | 3.00E+02 | 4.06E+02 | 3.00E+02 | 3.02E+02 | 1.49E+01 |
| 28 | 4.00E+02 | 4.00E+02 | 4.00E+02 | 4.00E+02 | 0 |

Science Policy Office.

## REFERENCES

[1] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *Proceeding of IEEE Congress on Evolutionary Computation, CEC'05*. Piscataway, NJ, USA: IEEE Press, 2005, pp. 1769–1776.

[2] D. Molina, M. Lozano, C. García-Martínez, and F. Herrera, "Memetic algorithms for continuous optimisation based on local search chains," *Evolutionary Computation*, vol. 18, no. 1, pp. 27–63, 2010.

[3] A. LaTorre, S. Muelas, and J.-M. Peña, "A MOS-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test," *Soft Computing*, vol. 15, pp. 2187–2199, 2011.

[4] A. Qin, V. Huang, and P. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[5] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proceedings of IEEE International Conference on Evolutionary Computation, CEC'96*. IEEE Press, 1996, pp. 312–317.

[6] ——, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[7] N. Hansen, S. Muller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)." *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[8] P. N. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Nanyang Technological University, Tech. Rep. 2005005, 2005.

[9] C. Müller, B. Baumgartner, and I. Sbalzarini, "Particle swarm CMA evolution strategy for the optimization of multi-funnel landscapes," in *Proceeding of IEEE Congress on Evolutionary Computation, CEC'09*. Piscataway, NJ: IEEE Press, 2009, pp. 2685–2692.

[10] S. Ghosh, S. Das, S. Roy, S. K. Minhazul Islam, and P. N. Suganthan, "A differential covariance matrix adaptation evolutionary algorithm for real parameter optimization," *Information Science*, vol. 182, no. 1, pp. 199–219, 2012.

[11] F. Peng, K. Tang, G. Chen, and X. Yao, "Population-based algorithm portfolios for numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 5, pp. 782–800, 2010.

[12] H. R. Lourenço, O. Martin, and T. Stützle, "Iterated local search: Framework and applications," in *Handbook of Metaheuristics*, 2nd ed., ser. International Series in Operations Research & Management Science, M. Gendreau and J.-Y. Potvin, Eds. New York, NY: Springer, 2010, vol. 146, ch. 9, pp. 363–397.

[13] J. R. Rice, "The algorithm selection problem," *Advances in Computers*, vol. 15, pp. 65–118, 1976.

[14] B. Bischl, O. Mersmann, H. Trautmann, and M. Preuß, "Algorithm selection based on exploratory landscape analysis and cost-sensitive learning," in *Proceedings of Genetic and Evolutionary Computation Conference, GECCO'12*. New York, NY, USA: ACM, 2012, pp. 313–320.

[15] T. Liao and T. Stütlze, "A simple and effective cooperative–competitive hybrid algorithm for continuous optimization," January 2013, submitted.

[16] 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," Zhengzhou University and Nanyang Technological University, Tech. Rep. TR201212, 2013.

[17] T. Liao, M. A. Montes de Oca, and T. Stützle, "Computational results for an automatically tuned CMA-ES with increasing population size on the CEC'05 benchmark set," *Soft Computing*, 2012, DOI:10.1007/s00500-012-0946-x, forthcoming.

[18] N. Hansen, "The CMA evolution strategy: A tutorial," 2010, online: http://www.lri.fr/hansen/cmatutorial.pdf.

[19] J. Gimmler, "Metaheuristiken zur kontiuierlichen, globalen optimierung," Master's thesis, Computer Science Department, TU, Darmstadt, Germany, 2005.

[20] O. Kramer, "Iterated local search with Powell's method: A memetic algorithm for continuous global optimization," *Memetic Computing*, vol. 2, pp. 69–83, 2010.

[21] L. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *Proceeding of the IEEE 2008 Congress on Evolutionary Computation, CEC'08*. Piscataway, NJ: IEEE Press, 2008, pp. 3052–3059.

[22] T. Liao, M. A. Montes de Oca, D. Aydin, T. Stützle, and M. Dorigo, "An incremental ant colony algorithm with local search for continuous optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'11*. New York, NY, USA: ACM, 2011, pp. 125–132.

[23] M. Birattari, Z. Yuan, P. Balaprakash, and T. Stützle, "F-race and iterated F-race: An overview," in *Experimental Methods for the Analysis of Optimization Algorithms*, Bartz-Beielstein *et al.*, Eds. Springer, Berlin, Germany, 2010, pp. 311–336.

[24] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, and M. Birattari, "The irace package, iterated race for automatic algorithm configuration," IRIDIA, Université Libre de Bruxelles, Belgium, Tech. Rep. TR/IRIDIA/2011-004, 2011. [Online]. Available: http://iridia.ulb.ac.be/IridiaTrSeries/IridiaTr2011-004.pdf

[25] F. Herrera, M. Lozano, and D. Molina, "Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems," 2010, uRL: `http://sci2s.ugr.es/eamhco/`.

[26] M. Lozano, D. Molina, and F. Herrera, "Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems," *Soft Computing*, vol. 15, pp. 2085–2087, 2011.

[27] S. Smit and A. Eiben, "Beating the world champion evolutionary algorithm via REVAC tuning," in *Proceedings of IEEE Congress on Evolutionary Computation, CEC 2010*. IEEE Press, Piscataway, NJ, 2010, pp. 1–8.