Can single solution optimisation methods be structurally biased?

Kononova, A.V.; Caraffini, F.; Wang, H.; Bäck, T.H.W.

Can Single Solution Optimisation Methods Be Structurally Biased?

Anna V. Kononova
LIACS, Leiden University
The Netherlands
a.kononova@liacs.leidenuniv.nl

Fabio Caraffini
Institute of Artificial Intelligence
De Montfort University
Leicester, UK
fabio.caraffini@dmu.ac.uk

Hao Wang
LIP6, Sorbonne Université
Paris, France
hao.wang@lip6.fr

Thomas Bäck
LIACS, Leiden University
The Netherlands
t.h.w.baeck@liacs.leidenuniv.nl

***Abstract*—This paper investigates whether optimisation methods with the population made up of one solution can suffer from structural bias just like their multisolution variants. Following recent results highlighting the importance of choice of strategy for handling solutions generated outside the domain, a selection of single solution methods are considered in conjunction with several such strategies. Obtained results are tested for the presence of structural bias by means of a traditional approach from literature and a newly proposed here statistical approach. These two tests are demonstrated to be not fully consistent. All tested methods are found to be structurally biased with at least one of the tested strategies. Confirming results for multisolution methods, it is such strategy that is shown to control the emergence of structural bias in single solution methods. Some of the tested methods exhibit a kind of structural bias that has not been observed before.**

***Index Terms*—structural bias, algorithmic design, hypothesis testing, single solution methods, constraint handling**

I. Optimisation methods: population-based vs single solution

Evolutionary computation (EC) methods draw inspiration directly from Darwinian evolution. Following this metaphor, candidate solutions to the optimisation problem play the role of individuals in a population. This population evolves through repeated application of various operators inspired by processes of mutation, recombination and selection based on the 'fitness' values of the solutions, i.e., values of the objective function.

The whole plethora of EC methods capitalises on concurrent exploration of the search domain within one run. Instead of concentrating on a single direction of search for the optimum inside a vast, usually multidimensional, domain, a selection of arbitrary directions is available to the method to choose from. This increases the exploration of the domain and decreases the need for potentially wasteful restarts of the algorithm. The success of EC methods is assumed to result from the following main assumptions:

- Great solutions are easily reachable (in terms of generating operators) from the good solutions or, in other words, there is a correlation between promising solutions.
- Algorithm's exploratory operators are able to provide new search directions.
- Information about good regions of the domain is exchanged/propagated within the population.

The extent to which these assumptions hold for a particular combination of the method and the objective function defines whether or not the method is capable of solving the problem. In any case, having a pool of good solutions spread out across the domain undoubtedly helps the search. Presence in the population of a solution located in the promising part of the domain forms a basis of the *'hedging' hypothesis* of success of the use of populations in optimisation methods [1]. The cost of maintaining the population - where not every single move of solutions leads to an increase in information about the function's landscape – is thought to be outweighed by having multiple search directions available, thus, hedging against the bad luck in choosing of the initial solution and other choices made throughout the optimisation. The use of populations provides another benefit since populations act as low-pass filters of the landscape [2]. This feature allows the search to ignore 'high-frequency disturbances' and act in a more robust manner. Following this reasoning, increasing or decreasing the population size allows to de-focus or re-focus the search [3].

The idea of using populations of solutions is not unique for the field of EC – the differences lie in the accents. Old methods such as the Nelder-Mead [4] or Solis-Wets algorithms [5] internally operate with several solutions but do not particularly focus on the 'evolving population' metaphor.

As shown by the famous No Free Lunch theorem [6], on the whole, populations do not offer any benefit against other search algorithms when performance is measured over all possible search problems. This implies that there should be problem instances for which population-based methods are better suited than methods that maintain a single candidate solution only, and vice versa.

Indeed, in the context of evolution strategies a population size of one, i.e., a single candidate solution at each iteration, has been a common strategy variant from the very beginning in the so-called (1+1) evolution strategy [7]. Today, the (1+1)–CMAES (with Cholesky decomposition) represents the state-of-the-art in the domain of single-solution variants of evolution strategies (see e.g. [8]). These methods are typically designed with the aim of reducing the number of function evaluations to a minimum[1], e.g., when a single function evaluation is a

[1]In real-world applications, this can be only a few hundred evaluations.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

time-consuming simulation or even a real-world experiment. The algorithm's search behaviour often represents a stochastic gradient search, (as shown by Schwefel [9] for the (1+1) Evolution Strategy).

This paper is organised as follows. Section II reviews the concept of structural bias, existing and newly proposed statistical procedure for testing for structural bias and existing results. Section III contains description of experimental setup and all single solution optimisation methods considered in this paper together with explanation of strategies for dealing with solutions generated outside the domain. Section IV discussed results on the distributions of final best solutions, traditional and new tests for presence of structural bias for all method configurations. Finally, Section V summarises conclusions and proposes directions for future research.

II. Structural bias

Not all algorithms are created equal. And yet any two optimisation algorithms are equivalent when their performance is averaged across all possible problems [10]. That is the *conundrum of comparing optimisation algorithms*.

As more than abundant literature [11], [12] suggests, optimisation methods can be compared in the light of *different aspects*: best/average/worst performance, complexity, universality of application, memory usage, scalability, etc. Moreover, performance can be evaluated on a class of functions (with widely varying definitions of 'class') or over all possible problems (which is practically impossible but theoretically relevant in the context of global convergence proofs).

One recently suggested aspect for comparison of performance is the presence or absence of the so-called *structural bias* (SB) [13] – the tendency of an algorithm to 'prefer' some parts of the search domain over others due to the non-trivial interaction between iteratively applied individual operators of the algorithm, regardless of the objective function. By design, all EC methods are driven by the ranking of the objective function of candidate solutions in the populations. However, it has been demonstrated that the actual movement of the population in time is a complex superposition of the 'force' of the objective function landscape and the 'force' of the SB inherent to the optimisation method itself [13]. SB steers the optimisation process away from specific areas of the domain, thus reducing the exploratory search property of the method.

A *structurally unbiased optimisation algorithm* should be able to locate equally well the optimum regardless of its position in the domain. Meanwhile populations in *structurally biased algorithms* are being pulled towards some areas of the domain more than to others, thereby potentially reducing the chances of finding certain optima. In other words, a strongly structurally biased algorithm is expected to have more difficulties in finding solutions located in areas of the domain less 'preferred' due to the SB.

Strength and direction of such SB can hardly be deduced due to the non-additive contributions of each operator and the interplay between the structure of the algorithm and the objective function [13]. This commands the need for a test function capable of separating the aforementioned 'forces'. The answer to this nontrivial question has been rather trivial: *eliminate the 'force' of the landscape of the objective function altogether* by using maximally uncorrelated function, namely *uniform noise*.

Testing for SB is thus *complementary* to the traditional performance benchmarking. In fact, algorithmic design is *'multiobjective'* – 'good traditional performance' and 'structurally unbiasedness' design objective *can be conflicting*.

A. Testing for structural bias

The procedure proposed for testing for presence of SB is based on a simple *theoretical result* presented in [13] that *true optima of* $f_0 : [0,1]^n \rightarrow [0,1]$ *where* $\forall x \ f_0(x) \sim \mathcal{U}(0,1)$ *are distributed uniformly in its domain*. Therefore, the degree to which the distribution of the locations of the optima identified by the method differs from the true uniform distribution allows meaningful conclusions regarding presence of SB in the method.

The procedure for testing for SB works as follows [13]:

- run the method under investigation for a statistically significant number of times minimising[2] the special objective function f_0;
- record the position of the final best solution found in each run in the series;
- plot locations of all final best solutions in parallel coordinates [14] and analyse the distributions of these locations[3].

B. Previous results on structural bias

A number of methods has been investigated in the past in the light of SB: Genetic Algorithms and Particle Swarm Optimisation in [13], [15], Nelder-Mead and Rosenbrock algorithms [15], Differential Evolution (DE) in [15]–[17], advanced DE variants and hybrids such as jADE, SHADE [18] and L-SHADE-EpSin and UMOEA-II in [19], and Across Neighbourhood Search in [20]. The majority of investigated methods exhibit SB. It has been shown that mechanisms of emergence of SB differ per algorithm and are difficult to be identified without explicit examination by means of methodology described in Section II-A. Following such analysis certain algorithmic parameters have been shown to exacerbate SB.

In [17], the choice of constraint handling techniques – another aspect, usually overlooked during the algorithmic design – is found to be a contributor to the formation of SB in DE. The *current study continues this direction* of investigation. Just like in many mathematical problems, it turns out that, it matters what happens on the domain boundaries.

Interestingly enough, in the case of Differential Evolution, it has been possible to pinpoint the particular mutation scheme

[2]As usual in iterative heuristic optimisation, no strict mathematical checks are made whether the final solution is a true minimum. Optimisation process runs for a predefined number of fitness evaluations and the best solution found by then is used as the best available approximation of the true minimum.

[3]This is easier done when visually comparing such plots for several methods or algorithmic configurations.

responsible for the emergence of SB [17]. Thus, it has been shown that tests on SB can be used as an additional *orthogonal goal* in algorithm design or selection.

Up until now, it has been difficult to judge the strength of SB automatically due to the purely visual nature of comparison used in the procedure described above. A simple Kolmogorov-Smirnov test has been applied in [13] and the strength of this test has been deemed unsuitable. Thus, *no single universal numerical estimate for the strength of SB* has been proposed so far. Furthermore, structurally biased algorithms can generate drastically different distributions of the final solutions, which complicates the automatic identification of SB additionally.

C. Estimating structural bias

As explained in the previous section, a structurally biased optimisation method yields non-uniformly distributed optima on test function f_0. Here, we propose to test such non-uniformity in $[0,1]^n$ by applying statistical procedures on each dimension of the obtained final points. Specifically, the *null hypothesis of our test* is that the coordinate of the obtained final points in each dimension is uniformly distributed in $[0,1]$. Given the sample of points $\{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(N)}\}$ (N being the number of independent runs), our treatment entails:

1) choosing a proper test procedure for the goodness-of-fit (to the uniform distribution),
2) setting a significance level $\alpha = 0.001$,
3) applying the chosen test to the sample on each dimension $\{x_i^{(1)},\ldots,x_i^{(N)}\}$ for $i = 1,2,\ldots,n$,
4) correcting the resulting p-values (since this is a multiple testing procedure) and rejecting the null hypothesis if $p < \alpha$. Instead of using the standard Bonferroni correction method, we choose the so-called *Benjamini–Hochberg* (BH) [21] procedure to control the false discovery rate of the statistical test because the Bonferroni procedure is considered too stringent and might yield many false negatives [22].

For choosing the goodness-of-fit procedure, several tests have been suggested in [23], e.g., Kolmogorov-Smirnov, Cramér-Von Mises, and Anderson-Darling test. Here, the *Anderson-Darling test* has been chosen based on the following simple *power analysis*: by simulating the alternative hypothesis using beta distributions with different parameters[4], we simulate the statistical power (a.k.a. true positive rate) as a function of the sample size. The power curves are plotted in Fig. 1, from which it is evident that the Anderson-Darling (AD) test gives rise to a much higher statistical power when the *sample size is larger than* 40.

The AD test statistic for the uniform distribution is $A^2 = N\int_0^1(\hat{F}_N(x)-x)^2/(x(1-x))\mathrm{d}x$, where $\hat{F}_N$ is the empirical cumulative distribution on the sample. A^2 quantifies the distance between the empirical distribution function and the uniform distribution. However, the test statistic must be accompanied by the p-value since a large test statistic value might be caused by random fluctuations, if the sample size is small.

[4] $(\alpha,\beta) \in \{(0.5,0.5),(5,1),(1,3),(2,2),(2,5)\}$

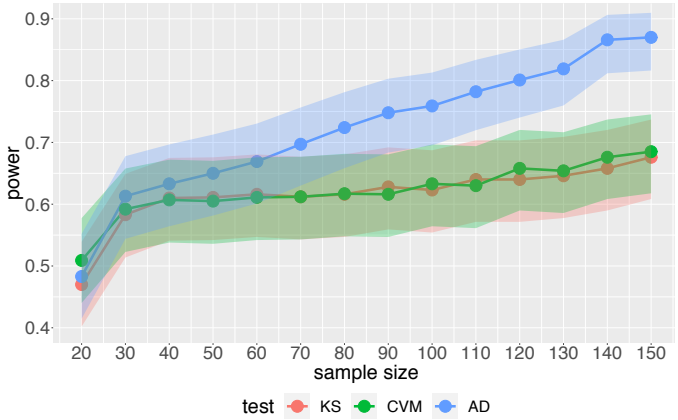

Fig. 1: The estimated statistical power of the Kolmogorov-Smirnov (KS), Cramér-Von Mises (CVM) and Anderson-Darling (AD) test against the sample size. The 95% confidence interval of the estimation is shown as the shade band.

Moreover, we validate the chosen method on the pseudo-random (using the same generator used for all methods in this study) and true uniform data generated from atmospheric noise[5]. As those statistical tests are designed for true random data, it is necessary to check if results on pseudorandom data are consistent. By using a significance level of 0.05, the validation procedure is performed by repeatedly executing the Anderson-Darling test for some preset times (the number of repetitions in the horizontal axis of Fig. 2) and then calculating the empirical Type-I error rate[6] as the number of rejections divided by the number of repetitions. In Fig. 2, we plot the empirical Type-I error rate (with the standard error shown as the shaded area) against the number of repetitions. The Type-I error rate obtained on pseudorandom numbers (the red one) is consistent with the one obtained on true uniform numbers (the blue one): as the number of repetitions increases, both of them converge to the predetermined significance level $\alpha = 0.05$ with the same convergence rate (as indicated by the rate at which the standard error decreases).

Result of the procedure for testing the uniformity outlined above are shown in Sec. IV-B for all experiments performed in this study.

III. Methods used, experimental setup

To cover all aspects of single solution optimisation, 13 diverse methods, amongst metaheuristics and their variants, are considered in this investigations. Similarly to what has been done for the population-based methods in [13], [16], [17], the aforementioned 13 methods are investigated for presence of SB with the experimental setup outlined in Section III-F and for each correction strategy amongst those described in Section III-E.

[5] https://www.random.org/decimal-fractions/

[6] The rate at which the null hypothesis is rejected when it is true.

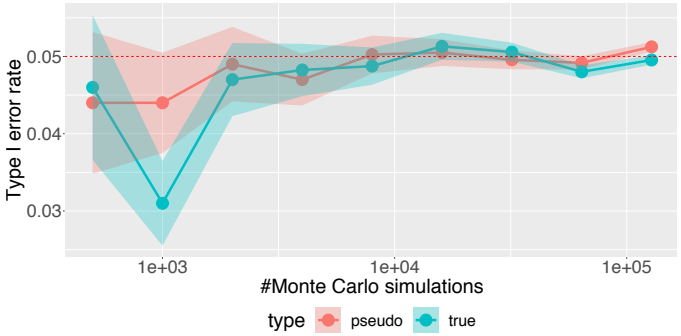

Fig. 2: Validation of the Anderson-Darling test on pseudorandom data with a significance level 0.05: the empirical type-I error rate obtained on pseudorandom uniform numbers is consistent with the one obtained from true uniform numbers.

A brief description of the employed methods is reported in the reminder of this section. These optimisation methods are implemented and tested, with the parameter configurations suggested in their original papers, using the software platform [11]. Details on their implementations can be obtained from the source code made available in the online repository [24], thus *facilitating the replicability of the results* presented in this study.

It is worth mentioning that most of the methods described in this section are designed for *global optimisation* but currently find their use as *local search routines* in various hybrid algorithmic structures. In particular, the deterministic methods in Section III-D are particularly suitable to this purpose due to their fast convergence speed and sensitivity to the quality of the initial solution.

A. Simulated Annealing methods

1) Standard Simulated Annealing (SA) [25]: is a well-known probabilistic metaheuristic method for approximating the global optimum of a function which draws inspiration from the physical process of annealing – heating and controlled cooling of a material to increase the size of its crystals and reduce their defect. At each step, the simulated annealing method considers some neighbouring state of the current state and non-deterministically decides whether to move. In this implementation, the new state is drawn from a uniform distributions and the cooling mechanisms is linear [26].

2) Non-Uniform Simulated Annealing (nuSA) [26]: is a variant of general Simulated Annealing algorithm which makes use of a non-uniform operation for generating new state (i.e. candidate solution) whose working mechanism is borrowed from adaptive evolutionary algorithms as this operator progressively reduces the neighbourhood size to control the range of search of the algorithm.

B. Evolutionary Computing and Swarm Intelligence methods

1) Intelligent Single Particle Optimizer (ISPO) [27], [28]: is a 'degenerative' variant of the popular Particle Swarm Optimisaton (PSO) method [29] in which the swarm size is reduced to a single particle. In this light, this minimalist version lacks interaction between global and local bests solutions, but displays an adaptive heuristic to update an additive 'velocity' vector used to perturb the the position of the particle within the search space.

2) Re-sampled Inheritance Search (RIS) [30]: is a very simple yet efficient Memetic Computing iterated local search algorithm based on a modification of the Hooke-Jeeves direct search method [31], referred to as 'short distance exploration' (operator S) to perform the local search. Operator S moves forward and, if needed, backward along each axis and adjusts the exploratory step length after a complete round (i.e. all axis have been perturbed). It works on a greedy logic according to which at each step the exploration stops if a fitter location is visited and then perturb the next design variable. Such search continues until a given precision is met, and never longer than a prefixed amount of fitness function evaluations (usually expressed as a percentage of the total budget). When a restart occurs, a new start point for the next local search is uniformly sampled within the domain, and subsequently crossed over with the best ever found solution, by means of the exponential crossover operator from Differential Evolution [16], [17], to retain some promising components.

C. Methods based on Evolution Strategies

1) (1+1)–Evolution Strategy with 1/5 Success Rule [32]: is based on adjusting a single standard deviation σ of a normal distribution according to the fraction p_s of successful mutations. If $p_s < 1/5$, σ is decreased by multiplying by a factor $0 < c < 1$, otherwise increased by dividing by c. The theoretical derivation of the optimal standard deviation $1/5$ is given in [7]. The two variants in [32], one accepting the new solutions only if fitter, while the second one also if it displays the same fitness value of the previous one, are both tested and referred to as (1+1)–ESv1 and (1+1)–ESv2 respectively.

2) (1+1)–'Cholesky' Covariance Matrix Adaptation ES ((1+1)–CMAES) [33]: introduces a method for adapting the covariance matrix implicitly, using a so-called Cholesky decomposition. It is based on a theorem proving that an update of the Cholesky factor $\mathbf{A}$ is possible without explicit knowledge of the covariance matrix $\mathbf{C} = \mathbf{A}\mathbf{A}^\top$.

D. Methods that do not belong to nature-inspired computing

1) Nelder-Mead algorithm (NMA) [4]: is a popular direct search method which progressively approximates the unknown optimum solution through a simplex[7]. The algorithm keeps a list of vertices sorted according the values of objection function. Throughout optimisation, the algorithm attempts to replace the worst vertex with a new solution, which depends on the worst solution and the centre of the best vertices.

2) Solis-Wets algorithm (SWA) [5]: is another direct search method that generates a randomised perturbation vector from the Gaussian distribution centred on the only candidate solution with an adaptive standard deviation which decreases when

[7]Generalisation of a triangle to an arbitrary dimensionality

the search is no longer prolific. This vector is then added to the candidate solution to move it within the problem's domain and, if a fitter position is not found, it is also subtracted to explore in the opposite direction.

3) Powell method (PM) [34]: is a method designed to build a set of 'non-interfering' directions to be optimised with a line-minimisation approach as e.g. the suggested Brent method [35]. Initially, these are a set of conjugated directions which are then updated so that the rank of the corresponding matrix is always full (i.e. the directions are linear independent vectors). In the most recent variant of this algorithm [35], this is done via a heuristic logic replacing the direction which contributed most to the new direction (i.e. the one along which the fitness function showed the largest decrease). This means that the procedure is *not mathematically accurate* and the *risk* of having linearly dependent directions is present.

4) Rosenbrock method (RM) [36], [37]: is a classical deterministic metaheuristic for real-valued optimisation. Initially, it probes each axis with an exploratory step which is increased in case of success and decreased otherwise. Similarly to SWA and RIS, if a step along an axis is not successful, a second step is taken in the opposite direction. When a new successfully position is found, the described process is repeated but only after using Gram–Schmidt ortho-normalisation to rotate the coordinate system towards the newly generate fitter solution. As a result, the new perturbation round does not longer move along the axis of the original search space but diagonally across it.

5) Simultaneous Perturbation Stochastic Approximation (SPSA) [38], [39]: is an attempt to replace classic finite-difference gradient approximation with a stochastic equivalent to reduce the computational cost. Unlike finite-difference based methods, which perturb one design variable at a time, SPSA acts on all the dimensions simultaneously by means of a perturbation vector whose components must be drawn from a zero mean distribution as e.g. Bernoulli ± 1 with probability 0.5 (as suggested in [39]). Before applying the perturbation vector, two candidate solutions must be generated to evaluate an incremental ratio and move toward the most promising direction accordingly. In the original method, if ten consecutive iterations are not capable of guaranteeing significant improvements in terms of fitness value, the search is stopped.

To perform a fair analysis, a version deprived of the stop mechanism, referred to as SPSAv2, is also tested in this study to make sure that SPSA results are not affected by a premature arrest due to the random fitness value assigned by f_0.

E. Strategies of dealing with solutions generated outside the domain

12 out 13 methods discussed in the previous section can be combined with 5 popular strategies (1 to 5 in the list below). The remaining method, SA, makes sense only with inherent strategy (number 6 in the list below). Thus, 61 combinations of methods and strategies are considered in this study.

1) Dismiss strategy [16], [17]: if a newly generated solution is outside the domain, it is dismissed and replaced by one of the parent/generating solutions.[8]

2) Saturation correction strategy [16], [17]: if newly generated solution is outside the domain, move only those values of coordinates that are outside the domain to the domain boundary; keep unchanged the original fitness value. This is a superficial correction/repair strategy.

3) Toroidal correction strategy [16], [17]: if newly generated solution is outside the domain, reflect only those values of coordinates that are outside the domain off the opposite domain boundary inwards; keep unchanged the original fitness value[9]. This is a superficial correction/repair strategy.

4) Mirror correction strategy [17]: if newly generated solution is outside the domain, move only those values of coordinates that are outside the domain by reflecting the value outside off the boundary inwards the domain; keep unchanged the original fitness value. This is a superficial correction/repair strategy.

5) Complete One-tailed normal correction strategy (COTN) [17]: if newly generated solution is outside the domain, for all the dimensions with coordinates outside the domain, resamples (iteratively, until the point is inside the domain) coordinates from $|\mathcal{N}(0, \frac{1}{3})|$ for dimensions where coordinate is smaller than the lower bound of the domain or from $1-|\mathcal{N}(0, \frac{1}{3})|$ if the coordinate is greater than the upper bound of the domain. In other words, this strategy nondeterministically maps solutions outside the domain to the area close to the boundary. This is a probabilistic complete correction strategy.

6) Inherent correction strategy: Generating operators of some optimisation methods explicitly prohibit generating solutions outside the domain – e.g. SA where positions of the newly sampled points are explicitly truncating to be inside the search domain. This superficial correction/repair strategy is inherent to the method and rarely encountered in the field.

F. Experimental setup

For each of the possible combinations between methods and strategies, 50 runs of minimisation of $f_0 : [0,1]^n \rightarrow [0,1]$, where $\forall x \ f(x) \sim \mathcal{U}[0,1]$ and $n = 30$ have been carried out. Each run had a budget of $10000 \times n$ fitness evaluations. All simulations are carried out in the SOS platform [11]. The source of all algorithms and the necessary experimental setup for reproducing the presented results are available in [24]. For all methods, default parameters are used as recommended in the original publications.

IV. Results

A. Traditional testing for structural bias

All methods and strategies for dealing with solutions generated outside discussed in Section III have been tested for

[8]In elitist single solution methods this correction strategies has the same effect of a penalty function and could be problematic if the algorithm is based on deterministic perturbation logic as it will keep re-assigning the same previous feasible solution leading to an infeasible one.

[9]As if the boundaries are connected and the domain forms a ring.

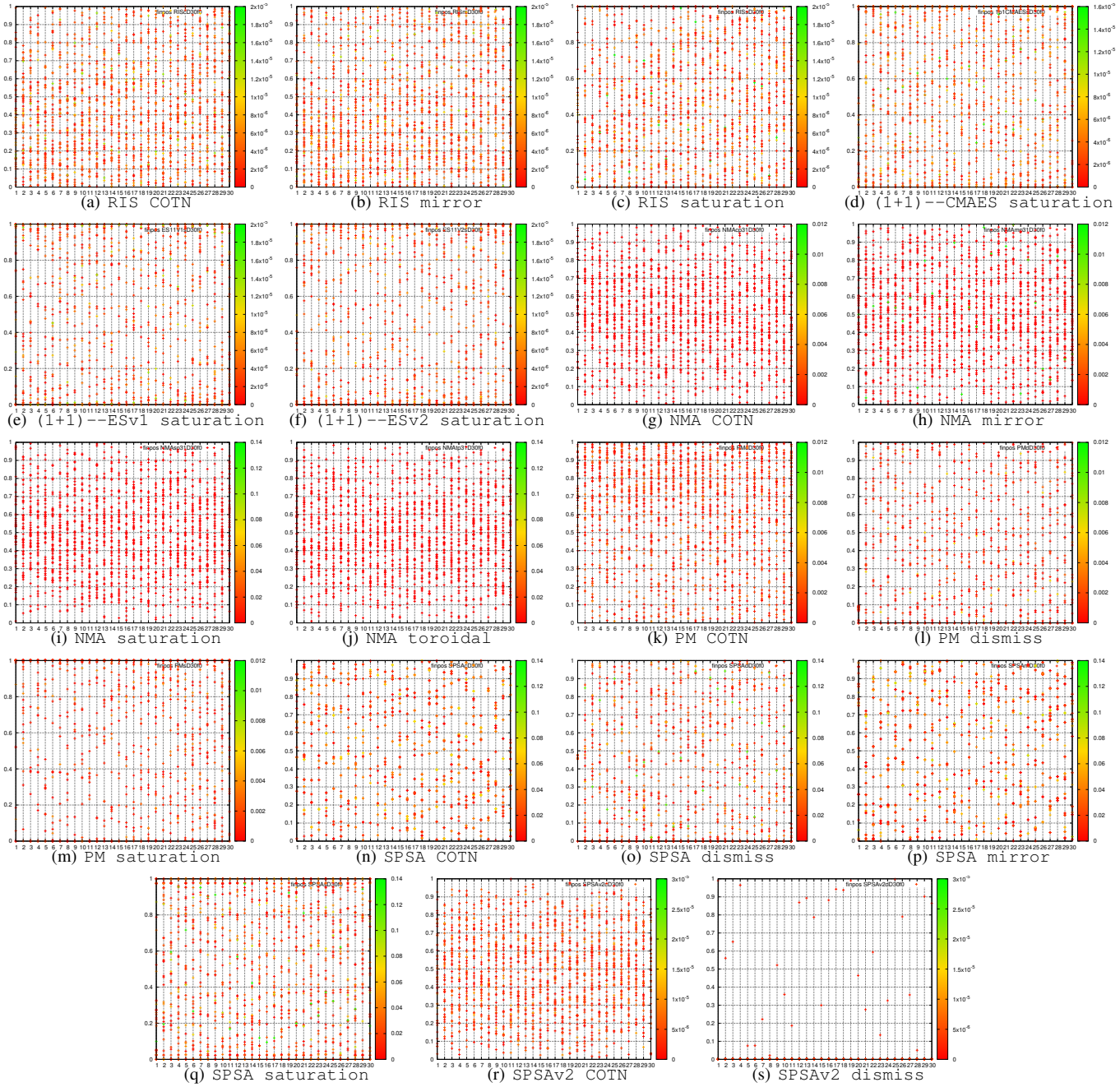

Fig. 3: Distribution of locations of final best solutions for *methods found to be structurally biased* shown in parallel coordinates where horizontal axis shows ordinal number of the dimension and vertical axis shows the range of this dimension; fitness value attained by the final solution is shown in colour. *These distributions deviate significantly from uniform.*

presence of SB as described in Section II-A and results for *structurally biased methods only* are shown in Fig. 3 and summarised in Table I. All results can be found in [40].

1) Results per strategy: `COTN` strategy appears to be the second worst in terms of SB – only 5 out of 13 are clearly unbiased. `Dismiss` strategy shows a slight improvement with 7 out of 13 clearly unbiased methos. `Mirror` strategy is second best. `Saturation` strategy seems to be the least successful one on the whole in terms of SB in single-solution methods – the majority of methods considered here are strongly structurally biased when equipped with saturation strategy. Distribution of locations of final best solutions from a series of independent runs for methods with `saturation` strategy varies from avoiding domain corners (NMA) to preferring some corners (RIS) or preferring all domain corners ((1+1)–CMAES, (1+1)–ESv1, (1+1)–ESv2). `Toroidal` strategy appear to be the *best strategy for mitigating SB in single solution methods* – only NMA is strongly structurally biased when

TABLE I: Results of traditional test for SB: '++' stands for presence of strong SB (see the configurations in Fig. 3), '+' for minor SB and '-' for no SB; absence of any sign means this combination of method and strategy is not possible. The cell in grey shades indicates the cases where the statistical decision in Fig. 4 agrees with this table.

	COTN	dismiss	mirror	saturation	toroidal	inherent
SA						-
nuSA	-	+	-	-	-	
ISPO	-	-	-	+	+	
RIS	++	+	++	++	-	
(1+1)–CMAES	-	-	-	++	-	
(1+1)–ESv1	-	-	-	++	-	
(1+1)–ESv2	-	-	-	++	-	
NMA	++	-	++	++	++	
SWA	+	-	-	-	+	
PM	++	++	-	++	-	
RM	+	-	+	+	-	
SPSA	++	++	++	++	-	
SPSAv2	++	++	-	+	-	

equipped with such strategy. Regarding `inherent` strategy, little conclusions can be made due to the lack of data.

2) Results per method: `nuSA` which exhibits SB only with dismiss strategy and three `ES-based` methods for which only saturation strategy appears to be unsuitable, are the *most-free-of-SB 'winners'* of this study. `ISPO` and `SWA` possess minor SB only for two strategies each – saturation and toroidal and COTN and toroidal, respectively. Final solutions from `RIS` tend to strongly 'prefer' one or another side of each dimension. `RM` possesses minor SB for COTN, mirror and saturation strategies – distribution of position of final solutions for all of these corrections spans the whole domain but 'leaves more gaps' than should be expected. `PM` exhibits SB for three strategies – COTN, dismiss and saturation – and always shows a strong preference to the corners of the domain including the exact zero solution in multiple independent runs. Meanwhile, `NMA` is unbiased only for dismiss strategy and strongly biased for the other four. The nature of SB in `NMA` is the same regardless of the chosen strategy – solutions tend towards the centre of the domain, avoiding all the corners. `SPSA` is unbiased only for toroidal strategy; meanwhile remaining strategies lead to the previously unobserved kind of SB – the method generates *identical solutions in a series of independent runs*. For `SPSA` COTN and mirror, multiple groups of identical solutions are generated in a series of independent runs; meanwhile dismiss and saturation result in strong 'preferences' of the region around zero or corners, respectively. Out of two structurally biased versions of `SPSAv2`, COTN is strongly biased towards the centre of domain and dismiss generates zero solution in 49 out of 50 runs, also exhibiting previously unseen behaviour.

B. Statistical testing for structural bias

In this section, result of the statistical procedure described in Section II-C are presented. Firstly, the Anderson-Darling is applied to the sample for each pair of dimension and method configuration. Note that for the most of methods, 50 independent runs are conducted (hence the sample size is 50 here), while 100 runs are performed to *achieve a good level of statistical power* for the following methods: `RIS-dismiss`, `RM-toroidal`, `RIS-mirror`, and `ISPO-mirror`.

Values of test statistic A^2 are shown in Fig. 14 of the extended results document [40] where a higher value indicates a larger effect of SB. Note that in some cases the test statistic takes an *infinite* value. When the test statistic is relatively small, the resulting value of A^2 might be caused by random fluctuations, instead of the SB. Thus, it is necessary to use the calculated p-values to decide on which pair of dimension and method configuration a significant SB is observed. Such statistical decision is shown in Fig. 4, where only methods with at least one rejection of the null hypothesis *'the sample points follow a uniform distribution"* are included.

From this figure, some methods show significant SB in all dimensions, e.g., `(1+1)-CMAES saturation`, `RIS-dimiss` and `PM-saturation`. On some other methods, for instance `SPSA-COTN` the null hypothesis is only rejected on a few dimensions. This potentially connotes that such a method works differently in each dimension, or the sample size is not big enough and therefore false negatives (where the null hypothesis fails to be rejected) are made here. This point is subject to *further investigation*.

Comparing the statistical decisions in Fig. 4 with the results of traditional test for SB discussed in Section IV-A leads to some disagreements. This comparison is illustrated in Table I, in which the cells in the grey shade indicate there is an agreement between those two methods. For the cases where disagreements arise, the authors argue that this could be affected by the insufficiency of the sample size, which should be studied in the *future work*.

C. Further observations

Our main observation is that despite considering a wide range of single-solution optimisation methods in this study, *none of those is found to be free of SB*.

It is remarkable to see that a number of methods considered in this study are indeed *highly structurally biased towards the zero region*. It is even more remarkable in the light that initial solutions of these methods are forcefully *not seeded in the zero point but sampled uniformly across the domain*. Thus, it is SB that is responsible for the solutions aggregating around zero, and not the inability of these methods to leave the region on the initial sampling.

Majority of methods considered in this study generate remarkably little solutions in experiments carried out for this study. The authors attribute this to the inability of generating operators to produce promising solutions from the current solution due to limited exploration of the domain within a single run. Lack of hedging the search through the use of a population mentioned in Section I is another reason.

Some methods considered in this study have been observed to generate *identical solutions* (or even sequences of identical solutions) in a series of independent runs. Dimensionality of

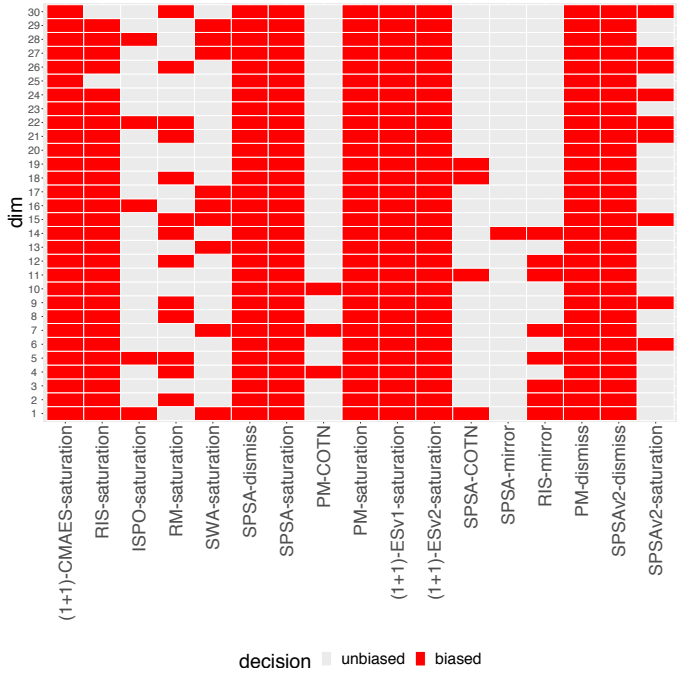

Fig. 4: The statistical decisions made in the Anderson-Darling test with a significance level $\alpha = 0.001$. The p-values used here are corrected by the Benjamini–Hochberg (BH) procedure. Only methods with SB in at least one dimension are shown (i.e. with at least one rejection of the null hypothesis overall dimensions).

the problem and frequency of such observations excludes the possibility of a random occurrence. This phenomena has not been observed previously in any of the studies mentioned in Section II-B. Due to the *unexpected nature* of such observations in multidimensional real-valued optimisation, the authors have paid particular attention to the correctness of the setup, generating code and analysis[10]. These results are *consistently repeated* for a significantly larger number of independent runs.

One method (SPSA) where this happens in particular uses a stochastic gradient estimates for guiding the search. Similarly, PM has been originally designed to mimic the hessian matrix behaviour and specifically address quadratic forms. In later years, what used to be a good estimate for a gradient, started being used as a heuristic for any function. As a result, the quality of the estimate is no longer guaranteed and the method is no longer guided as *originally intended*. At the same time, undoubtedly, trying to optimise f_0 using gradient information is hopeless. It must be pointed out that the lack of population can also be responsible for triggering this mechanism, in particular when the dismiss strategy is used with deterministic metaheuristic such as PM, RM and the S operator of RIS. Indeed, once the a solution is dismissed, the previous one is fed again to the method which will end up generating the same sequence of steps. This drawback does not occur in randomised methods such as those based on the ES.

Several methods, such as Rosenbrock, Powell, Solis-Wets and SPSA, are designed to be global optimisers but are currently mainly used for local search [41]–[43]. Also metaheuristics such as (1+1)–CMAES have been proven to show better results when equipped with a re-start mechanism, to move upon exploration, and run multiple times with a short budget to refine promising candidate solutions [44].

Finally, it has bee reported previously that NMA and RM are free of SB [15]. However, this does not agree with results presented in this study. Unfortunately, the authors of [15] do not indicate what happens with solutions generated outside the domain – the authors expect correction strategy to be the culprit of different results here and in [15].

V. Conclusion and Future Directions

Results presented in this paper allow a number of clear conclusions, summarised as follows:

- Despite considering a wide range of single-solution optimisation methods in this study, none of those is found to be free of SB.
- The choice of the strategy for dealing with solutions generated outside the domain (i.e., the boundary handling strategy, BHS) controls the emergence of SB in single-solution methods.
- No dominant choice of a BHS exists for the 13 methods considered.
- The choice of a BHS is not always considered in the original implementations of the methods. However, behaviour of these methods strongly depends on the chosen BHS.
- Presence or absence of SB in a method is still difficult to predict without running a thorough analysis as described in Section II-A.
- The majority of these methods generate a remarkably small number of solutions in experiments carried out for this study.
- A number of methods is highly structurally biased towards the zero region despite initial sampling being uniform in the whole domain across the series of independent runs.
- Some methods generate identical solutions in a series of independent runs. This phenomenon has not been observed previously in the SB analysis.
- Some single solution optimisation methods are too specialised by design which does not justify their wide use as *non-local optimisers*.
- Some widely cited and used methods do not behave as expected.
- Classifying the distribution of locations of final best solutions solely based on the visual inspection is difficult and highly subjective without a proper quantitative measure.
- Finding a suitable sensitive statistical test to classify methods in terms of SB is not straightforward.
- It is difficult to find one quantitative measure able to capture all patterns assumed by the distribution of final best solutions in a general case.

[10]including testing the random generator's properties and seeding it generator correctly

- Some methods exhibit SB of different strength per dimension. This requires further study.
- Despite previous speculations in [13], populations are not among the drivers of SB in optimisation methods since single solution methods are subject to SB as well.
- Overall, single-solution optimisation methods seem to be more structurally biased than popular population-based algorithms.
- Yet again [17], it matters what happens on the boundaries of the problem or, in other words, the choice of BHS used by the method truly matters in algorithmic design – neither researchers nor practitioners should neglect this aspect.

REFERENCES

[1] A. Prügel-Bennett, "Benefits of a population: Five mechanisms that advantage population-based algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, pp. 500–517, 2010.

[2] R. Poli, A. Wright, N. McPhee, and W. Langdon, "Emergent behaviour, population-based search and low-pass filtering," in *IEEE International Conference on Evolutionary Computation*, 2006.

[3] A. Caponio, A. V. Kononova, and F. Neri, "Differential evolution with scale factor local search for large scale problems," in *Computational Intelligence in Expensive Optimization Problems*, ser. Studies in Evolutionary Learning and Optimization, Y. Tenne and C.-K. Goh, Eds. Springer Berlin Heidelberg, 2010, vol. 2, pp. 297–323.

[4] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 01 1965.

[5] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Mathematics of Operations Research*, vol. 6, no. 1, pp. 19–30, 1981.

[6] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67–82, 1997.

[7] I. Rechenberg, *Evolutionsstrategie: Optimierung Technischer System nach Prinzipien des Biologischen Evolution*. Fromman-Holzboog Verlag, 1973.

[8] T. Bäck, C. Foussette, and P. Krause, *Contemporary Evolution Strategies*. Berlin, Germany: Springer, 2013.

[9] H.-P. Schwefel, *Evolution and Optimum Seeking*. New York: Wiley, 1993.

[10] D. Wolpert and W. Macready, "Coevolutionary free lunches," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 721–735, 2005.

[11] F. Caraffini, "The sos platform: designing, tuning and statistically benchmarking optimisation algorithms," MDPI Preprints, 2020030381, 2020.

[12] V. Beiranvand, W. Hare, and Y. Lucet, "Best practices for comparing optimization algorithms," *Optimization and Engineering*, vol. 18, no. 4, pp. 815–848, Dec 2017.

[13] A. V. Kononova, D. W. Corne, P. D. Wilde, V. Shneer, and F. Caraffini, "Structural bias in population-based algorithms," *Information Sciences*, vol. 298, pp. 468–490, 2015.

[14] A. Inselberg, "The plane with parallel coordinates," *The Visual Computer*, vol. 1, no. 2, pp. 69–91, Aug 1985.

[15] A. P. Piotrowski and J. J. Napiorkowski, "Searching for structural bias in particle swarm optimization and differential evolution algorithms," *Swarm Intelligence*, vol. 10, no. 4, pp. 307–353, Dec 2016.

[16] F. Caraffini and A. V. Kononova, "Structural bias in differential evolution: a preliminary study," in *LeGO 2018 - 14th International Workshop on Global Optimization, Leiden, The Netherlands*. AIP, 18-21 September 2018.

[17] F. Caraffini, A. V. Kononova, and D. W. Corne, "Infeasibility and structural bias in differential evolution," *Information Sciences*, vol. 496, pp. 161–179, 2019.

[18] A. P. Piotrowski and J. J. Napiorkowski, "Step-by-step improvement of jade and shade-based algorithms: Success or failure?" *Swarm and Evolutionary Computation*, 2018.

[19] ——, "Some metaheuristics should be simplified," *Information Sciences*, vol. 427, pp. 32–62, 2018.

[20] A. P. Piotrowski, "Across neighborhood search algorithm: A comprehensive analysis," *Information Sciences*, vol. 435, pp. 334 – 381, 2018.

[21] Y. Benjamini, "Discovering the false discovery rate," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 4, pp. 405–416, 2010.

[22] J. P. Shaffer, "Multiple hypothesis testing," *Annual review of psychology*, vol. 46, no. 1, pp. 561–584, 1995.

[23] R. B. D'Agostino and M. A. Stephens, *Goodness-of-Fit Techniques*. USA: Marcel Dekker, Inc., 1986.

[24] F. Caraffini, "The Stochastic Optimisation Software (SOS) platform," https://doi.org/10.5281/zenodo.3237023, june 2019.

[25] S. Kirkpatrick, C. D. J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, no. 220, pp. 671–680, 1983.

[26] Z. Xinchao, "Simulated annealing algorithm with adaptive neighborhood," *Applied Soft Computing*, vol. 11, no. 2, pp. 1827–1836, mar 2011.

[27] J. Zhen, Z. Jiarui, L. Huilian, and W. Qing-Hua, "A Novel Intelligent Single Particle Optimizer," *Chinese Journal of Computers*, vol. 33, no. 3, pp. 556–561, apr 2010.

[28] G. Iacca, F. Caraffini, F. Neri, and E. Mininno, "Single particle algorithms for continuous optimization," in *2013 IEEE Congress on Evolutionary Computation, CEC 2013*, 2013, pp. 1610–1617.

[29] J. Kennedy, *Particle Swarm Optimization*. Boston, MA: Springer US, 2010, pp. 760–766.

[30] F. Caraffini, F. Neri, B. N. Passow, and G. Iacca, "Re-sampled inheritance search: high performance despite the simplicity," *Soft Computing*, vol. 17, no. 12, pp. 2235–2256, Dec 2013.

[31] R. Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *J. ACM*, vol. 8, no. 2, p. 212–229, Apr. 1961.

[32] A. Auger, "Benchmarking the (1+1) evolution strategy with one-fifth success rule on the BBOB-2009 function testbed," in *Proceedings of the 11th annual conference companion on Genetic and evolutionary computation conference - GECCO '09*. New York, New York, USA: ACM Press, 2009, p. 2447.

[33] C. Igel, T. Suttorp, and N. Hansen, "A computational efficient covariance matrix update and a (1+1)-CMA for evolution strategies," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO 2006)*. ACM, 2006, pp. 453–360.

[34] M. J. D. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *The Computer Journal*, vol. 7, no. 2, pp. 155–162, Jan. 1964.

[35] R. P. Brent, *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

[36] H. H. Rosenbrock, "An Automatic Method for Finding the Greatest or Least Value of a Function," *The Computer Journal*, vol. 3, no. 3, pp. 175–184, mar 1960.

[37] J. R. Palmer, "An improved procedure for orthogonalising the search vectors in rosenbrock's and swann's direct search optimisation methods," *The Computer Journal*, vol. 12, no. 1, pp. 69–71, 1969.

[38] J. Spall, "A stochastic approximation technique for generating maximum likelihood parameter estimates," in *American Control Conference, 1987*, 1987, pp. 1161–1167.

[39] J. C. Spall, "Implementation of the simultaneous perturbation algorithm for stochastic optimization," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 34, no. 3, pp. 817–823, 1998.

[40] F. Caraffini and A. V. Kononova, "Structural Bias in Optimisation Algorithms: Extended Results," 2020. [Online]. Available: http://dx.doi.org/10.17632/zdh2phb3b4.2

[41] D. Molina, M. Lozano, and F. Herrera, "Ma-sw-chains: Memetic algorithm based on local search chains for large scale continuous global optimization," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8.

[42] M. G. Epitropakis, F. Caraffini, F. Neri, and E. K. Burke, "A Separability Prototype for Automatic Memes with Adaptive Operator Selection," in *IEEE SSCI 2014 - 2014 IEEE Symposium Series on Computational Intelligence - FOCI 2014: 2014 IEEE Symposium on Foundations of Computational Intelligence, Proceedings*, 2015, pp. 70–77.

[43] F. Caraffini, F. Neri, and M. Epitropakis, "Hyperspam: A study on hyper-heuristic coordination strategies in the continuous domain," *Information Sciences*, vol. 477, pp. 186–202, mar 2019.

[44] F. Caraffini, G. Iacca, and A. Yaman, "Improving (1+1) covariance matrix adaptation evolution strategy: A simple yet efficient approach," in *AIP Conference Proceedings*, vol. 2070, no. February, 2019, p. 020004.