Can compact optimisation algorithms be structurally biased?

Anna V. Kononova[1][0000−0002−4138−7024], Fabio Caraffini[2][0000−0001−9199−7368], Hao Wang[3][0000−0002−4933−5181], and Thomas Bäck[1][0000−0001−6768−1478]

[1] Leiden Institute of Advanced Computer Science (LIACS), Leiden University, The Netherlands
{a.kononova,t.h.w.baeck}@liacs.leidenuniv.nl
[2] Institute of Artificial Intelligence, De Montfort University, UK
fabio.caraffini@dmu.ac.uk
[3] LIP6, Sorbonne Université Paris, France
hao.wang@lip6.fr

Abstract. In the field of stochastic optimisation, the so-called *structural bias* constitutes an undesired behaviour of an algorithm that is unable to explore the search space to a uniform extent. In this paper, we investigate whether algorithms from a subclass of *estimation of distribution algorithms*, the *compact algorithms*, exhibit structural bias. Our approach, justified in our earlier publications, is based on conducting experiments on a test function whose values are uniformly distributed in its domain. For the experiment, 81 combinations of compact algorithms and strategies of dealing with infeasible solutions have been selected as test cases. We have applied two approaches for determining the presence and severity of structural bias, namely an (existing) visual and an (updated) statistical (Anderson-Darling) test. Our results suggest that compact algorithms are more immune to structural bias than their counterparts maintaining explicit populations. Both tests indicate that strong structural bias is found only in the `cBFO` algorithm, regardless of the choice of strategy of dealing with infeasible solutions, and `cPSO` with `mirror` strategy. For other test cases, statistical and visual tests *disagree* on some cases classified as having mild or strong structural bias: the former one tends to make harsher decisions, thus needing further investigation.

Keywords: structural bias · compact algorithm · continuous optimisation · estimation of distribution algorithm · infeasible solution.

1 Introduction

Evolutionary algorithms (EAs) [9, 1] are based on a *biological metaphor* which creates an *ontological link* between a set of solutions of the optimisation problem, which iteratively approximate its optimum, and a population of biological individuals, which adapt to their environment through evolution. An essential part of this metaphor is an *individual*, an atomic part of the *population*, that has been created by some combination of one or more of its parent individuals

in an attempt to build upon previously successful approximations of the optimum. Both biological and (most) computational populations typically do not explicitly 'record' their history, thus, potentially loosing the already exploited information regarding the 'successes' in the past generations. Following the biological metaphor, a 'success' in some generation is directly translated into the individual's reproductive advantage and, therefore, an opportunity to pass on its 'achievements'.

Striving to exploit the historical information contained in the sequential populations of an evolutionary algorithm, a special class of algorithms has been been proposed in the 1990s [19, 18] which attempts to build *explicit probabilistic models of promising solutions* as the optimisation process progresses and steer the subsequent simulated evolutionary progress towards such solutions. These new algorithms, just like other heuristics [15, 20, 7], are probabilistic, iterative, and thus can suffer from undesirable algorithmic behaviours such as premature convergence, stagnation and presence of *structural bias* (SB) [15, 7]. The latter is the *focus of this paper*.

The aforementioned class of algorithms, referred to as *estimation of distribution algorithms* (EDAs) [10], do not maintain explicit populations but rather have *virtual sampling populations*. They work through updating their models incrementally, starting from some uninformed prior and, ideally, leading up to the model producing only the optimum solution. Clearly, the problem of constructing such a model in itself is by far not trivial and can only be solved with some simplifications. It is the scope and extent of such *simplifications* that define the sub-classes of EDAs.

This paper addresses the question of whether a *subclass* of algorithms with virtual populations exhibit such algorithmic deficiency as structural bias – the tendency of an algorithm to 'prefer' some parts of the domain irrespective of the objective function. This paper continues the effort of the authors to investigate a wide range of heuristic optimisation algorithms for possible structural bias deficiencies [15, 5, 7, 14]. The paper is organised as follows: Section 2 discusses compact algorithms in general and the particular instances investigated in this study, Section 3 describes the experimental methodology and methods for assessing SB, Section 4 discusses results concerning SB in compact algorithms, and Section 5 provides the conclusions.

2 Compact algorithms

The term 'compact algorithm' refers to a subclass of EDAs that mimic the behaviour of established population-based algorithms [11] through a *'memory-saving'* probabilistic model where design variables are assumed to be *fully uncorrelated*. This minimalist model is fully described with a $2 \times n$ matrix (n is problem dimensionality) that defines the generating distribution[4] $\mathcal{D}_\theta$, where $\boldsymbol{\theta} = [\boldsymbol{\mu}, \boldsymbol{\sigma}]$,

[4] It is called 'probability vector' in the original publications [11]; a terminology which we find somewhat misleading in case of a continuous search space and a Gaussian generating distribution.

$\boldsymbol{\mu} \in \mathbb{R}^n, \boldsymbol{\sigma} \in (\mathbb{R}^+)^n$ are the vectors containing the chosen mean and the standard deviation values for a truncated Gaussian distribution (the optimisation process takes places in the *re-normalised domain* $[-1, 1]^n$).

All 'elitist' real-valued compact algorithms share the structure outlined in Algorithm 1 and only differ by the logic used to generate a new solution $\mathbf{x}$.

Algorithm 1 Skeleton of a *generic* elitist compact algorithm

given: objective function f, generating distribution $\mathcal{D}_\theta$ with parameters $\theta = [\boldsymbol{\mu}, \boldsymbol{\sigma}]$
initialise $\boldsymbol{\mu}, \boldsymbol{\sigma}$ with $\mu_i = 0$ and $\sigma_i \gg 1$ ▷ e.g. $\sigma_i = 10$ as in [11]
draw initial solution $\boldsymbol{x}_{\text{elite}}$ from $\mathcal{D}_\theta$ and evaluate its fitness $f_{\text{elite}} = f(\boldsymbol{x}_{\text{elite}})$
while budget condition is not met **do**
  draw i.i.d. samples $\mathcal{P} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ from $\mathcal{D}_\theta$ ▷ $|\mathcal{P}|$ depends on the specific operator (Section 2.1)
  generate a new candidate solution $\mathbf{x}$ from $\mathcal{P}$
  evaluate $f(\mathbf{x})$;
  if $f(\boldsymbol{x}) \bowtie f_{\text{elite}}$ **then** ▷ $\bowtie \in \{\leq, \geq\}$ for minimisation/maximisation
    $\mathbf{l} \leftarrow \mathbf{x}_{\text{elite}}$; $\mathbf{w} \leftarrow \mathbf{x}$; $\mathbf{x}_{\text{elite}} \leftarrow \mathbf{x}$; ▷ $\mathbf{w}$ is the winner, $\mathbf{l}$ loser
  else
    $\mathbf{l} \leftarrow \mathbf{x}$; $\mathbf{w} \leftarrow \mathbf{x}_{\text{elite}}$;
  end if
  $\boldsymbol{\mu}_{\text{old}} \leftarrow \boldsymbol{\mu}$
  $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \frac{1}{\text{V}_{\text{ps}}} (\mathbf{w} - \mathbf{l})$ ▷ user defined virtual population size V_{ps} [11]
  $\boldsymbol{\sigma} \leftarrow \sqrt{\boldsymbol{\sigma} \circ \boldsymbol{\sigma} + \boldsymbol{\mu}_{\text{old}} \circ \boldsymbol{\mu}_{\text{old}} - \boldsymbol{\mu} \circ \boldsymbol{\mu} + \frac{1}{\text{V}_{\text{ps}}} (\mathbf{w} \circ \mathbf{w} - \mathbf{l} \circ \mathbf{l})}$
end while ▷ $\circ$ is the Hadamard product
Output: $\boldsymbol{x}_{\text{elite}}$

2.1 Compact algorithms employed in this study

All compact algorithms employed in this study follow the logic described in Algorithm 1. Details on these algorithms, including their suggested and adopted parameters setting, are available in [11]. A brief description of each algorithm is given below. These algorithms are equipped with various strategies of dealing with infeasible solutions (SDIS) generated, see Section 3.4.

Configurable compact differential evolution (`cDE/x/y/z`): similar to non-compact variants of differential evolution [21], a variety of compact configurations can be obtained with the combinations `x/y/z`, where `z` is either the binary `bin` or the exponential `exp` crossover [6, 21], while the `x/y` component is taken from these options[5]: (i) `rand/1` (ii) `rand/2` (iii) `best/1` (iv) `best/2` (v) `current-to-best/1` (vi) `rand-to-best/2` (vii) `current-to-rand/1` (does not require a crossover). It must be highlighted that in a DE algorithm, the `x/y/z` operators require a number of randomly selected individuals from the population to produce $\mathbf{x}$. Due to the absence of a stored population, these individuals are drawn from the generating distribution $\mathcal{D}_\theta$ in the compact representation. This implies that logically `current-to-best/1` ≡ `rand-to-best/1`.

[5] This list clearly does not exhaust all possibilities.

Compact differential evolution light: `cDE-Light` is a DE-inspired compact algorithm that requires a smaller number of computationally expensive operations with respect to its predecessor algorithm `cDE`, thus being faster and lighter in terms of memory consumption. This algorithm employs a specific mutation referred to as `mutation-light`, which mimics the behaviour of the `rand/1` mutation, and specific crossover operator referred to as `crossover-light`, which emulates the `exp` crossover without the need of looping through the solutions to exchange their variables.

Compact particle swarm optimisation (`cPSO`): generates novel candidate solutions $\mathbf{x}$ through the simple `PSO` perturbation logic based on a weighted sum of the currently available solution and the so-called 'velocity' vector $\mathbf{v}$, i.e. $\mathbf{x} \leftarrow \gamma_1\mathbf{x} + \gamma_2\mathbf{v}$. Before perturbing the position of $\mathbf{x}$ in the search space with the previous formula, $\mathbf{v}$ must be be updated through the standard method $\mathbf{v} \leftarrow \phi_1\mathbf{v} + \phi_2\mathbf{u_1} \circ (\mathbf{x_{lb}} - \mathbf{x}) + \phi_3\mathbf{u_2} \circ (\mathbf{x_{gb}} - \mathbf{x})$, in which $\mathbf{u_1}$ and $\mathbf{u_2}$ are two n-dimensional vectors containing uniformly drawn random numbers; $\mathbf{x_{lb}}$ is the 'local best' solution, which is not present in the compact representation and therefore has to be drawn from $\mathcal{D}_\theta$ and evaluated; $\mathbf{x_{gb}}$ is the 'global best' solution, i.e., $\mathbf{x_{gb}} \leftarrow \mathbf{x_{elite}}$. It must be pointed out that $\mathcal{D}_\theta$ is updated with $\mathbf{w}$ and $\mathbf{l}$ obtained by comparing the objective function values $\mathbf{x_{lb}}$ and $\mathbf{x}$ while the $\mathbf{x}_{\text{elite}}$ solution is subsequently updated.

Compact bacterial foraging optimisation (`cBFO`) reproduces the same search logic of the original BFO algorithm [8] with the difference that, at each iteration, a candidate solution $\mathbf{x}$ is drawn from $\mathcal{D}_\theta$ rather than being taken from a population. Such solution undergoes a series of perturbations to perform the so called 'chemotaxis', 'tumble' and 'swim' moves in the search space by means of the operator $\mathbf{x} \leftarrow \mathbf{x} + \frac{\mathbf{c} \circ \boldsymbol{\Delta}}{\sqrt{\boldsymbol{\Delta}^T\boldsymbol{\Delta}}}$, where $\mathbf{c}$ is an n-dimensional vector whose components are the so-called 'run-length' unit parameters [8], which control the step-size, and $\boldsymbol{\Delta}$ is an n-dimensional vector whose components are uniformly sampled in the interval $[-1, 1]$ as indicated in [8] for each one of the three moves.

Compact genetic algorithm: the real-valued compact genetic algorithm `rcGA` [11], or `cGA` here, is the simplest example of compact algorithm as it only draws a new solution from $\mathcal{D}_\theta$ (i.e. $\mathcal{P} = \{\mathbf{x}\}$) to produce a new candidate solution.

3 Methodology

3.1 Structural bias

The field of EAs is saturated with a multitude of nature inspired algorithms [2, 4]. For practical reasons, these algorithms need to be compared and characterised. Amongst *dimensions* over which the quality of an optimisation algorithm can be measured are: (i) values of the best or average improvement of the objective function attained over a series of independent runs on some function or class of functions; (ii) best or average ranking of the algorithm among other algorithms on some function or class of functions; (iii) the distance from the found solution to the known optima; (iv) whether the algorithm has stagnated or converged

prematurely; (v) typical or peak memory consumption required by the algorithm to solve the problem; (vi) scalability of the algorithm; (vii) proportion of the previously-non-visited solutions; etc.

In the EA/EC community, variations of the first two of the aforementioned dimensions are traditionally used. However, most performance measures come with a difficulty: dependence on the objective function [23]. Moreover, in practice, classes of objective functions are typically hard to be defined exhaustively and extensively and benchmarking over a set of diverse functions strongly depends on the choice of such functions.

In an attempt to characterise the performance of optimisation algorithms *from a different angle*, an additional fitness-free comparison 'dimension' has been suggested in [15]: the so-called *structural bias* (SB) has been defined as an *intrinsic deficiency* of a probabilistic iterative algorithm dictated solely by its structure. An algorithm is said to possess SB when it is unable to explore all areas of the search space to a *uniform* extent, *irrespective* of the objective function.

In other words, characterising the algorithm in terms of SB allows one to judge how much *general-purpose* the algorithm is, since a fully general-purpose optimisation algorithm is expected to be able to locate the optima regardless of where they are located in the search space. It has been established [15] that for a *general* objective function, the movement of solutions in the populations evolving over time is dictated by the superposition of two forces: the gradient formed by the values of objective function in the current population and the force originating from the structure of algorithm. These *two forces are not necessarily in agreement in terms of direction and strength.* The problem with the existence of the second force is that it can potentially pull the search away from some areas of the domain, thus limiting the algorithm's ability to find the optima therein.

It must be remarked that due to the *stochastic nature* of the utilised test function f_0 (see Section 3.2), there is no sense in tracking objective function improvements over time. The goal of tests on f_0 is *only* to establish deficiencies in movements of the populations during the optimisation process and *not to rank* the methods according to their 'objective-function-improvement' on f_0.

3.2 Structural bias via visual tests

The procedure for testing for presence of SB is based on a *theoretical result* [15] that *true minima/maxima of*

$$f_0 : [0,1]^n \to [0,1] \mid \forall \mathbf{x}\ \mathbf{f_0}(\mathbf{x}) \sim \mathcal{U}(\mathbf{0},\mathbf{1}) \tag{1}$$

are distributed uniformly in its domain (where $\mathcal{U}(0,1)$ denotes a scalar random value sampled independently from the uniform distribution on $[0,1]$). Thus, through examination of the distribution of locations of the optima of f_0 identified by the algorithm and its subsequent comparison to the true uniform distribution across the domain, one can establish whether the algorithm exhibits any SB [15]. To date, such comparison has been done *visually* due to the lack of a good *'all-encompassing'* measure, see Section 3.3 for more discussion. Plotting locations

of final best solutions in a series of independent runs in parallel coordinates [12] is an established technique that facilitates the analysis.

3.3 Structural bias via statistical tests

To identify SB, we build on the previous studies [15, 14] where Kolmogorov-Smirnov test has been used for hypothesis testing. Here, we propose a different statistical approach which *tests the uniformity* of final points per dimension *via a non-parametric goodness-of-fit* test – the Anderson-Darling (AD) test is chosen given its high statistical power [22]. The motivation behind this approach is two-fold: first, testing the multivariate uniformity is known to be a *challenging* task [13]; second, it is methodologically *erroneous to merge samples* from all dimensions to perform one univariate good-of-fit test as the design variables could be correlated and not identically distributed, thus resulting in a potential loss of information on each dimension.

Hence, for each dimension $i \in [1..n]$ the AD test is applied to the i^{th} component of final points $\{x_i^{(1)}, \ldots, x_i^{(N_r)}\}$ obtained over N_r independent runs ($N_r = 50$ here). When testing the uniformity of the sample distribution along each dimension, the AD test-statistic is formulated as: $A^2 = \int_0^1 (\widehat{F}_{N_r}(t) - t)^2 / t(1-t) \mathrm{d}t$, where $\widehat{F}_{N_r}(t) = \sum_{k=1}^{N_r} \mathbb{1}(x_i^{(k)} \leq t)$ is the empirical cumulative distribution function (ECDF) of the i^{th} component. Intuitively, A^2 quantifies the proximity between the ECDF and the theoretical distribution function of the uniform distribution. We shall denote the resulting test statistics and p-values as $\{A_i^2\}_{i=1}^n$ and $\{p_i\}_{i=1}^n$ respectively. The significance level $\alpha = 0.01$ is used to reject the null hypotheses H_0. Whenever H_0 is rejected we conclude that the ECDF differs from the uniform distribution by an amount of A^2, with an error rate of α. The SB 'degree' is then determined by counting the *rejected* dimensions.

Moreover, we propose an *aggregated measure of SB* over results from all dimensions, defined as the sum of A_i^2 test statistics that are associated with a statistical significance over all dimensions: $\mathrm{SB} = \frac{1}{n} \sum_{i=1}^{n} A_i^2 \mathbb{1}(p_i \leq \alpha)$, where $\mathbb{1}$ stands for the indicator function. We shall contrast this new measure of SB with the visual test shown in Section 4.2. Note that methods to combine p-values (e.g., Kost's method [16]), which performs a test on the results from several tests, is not suitable here since we also interested in combining the statistical effect from several dimensions.

3.4 Strategy of dealing with infeasible solutions as operator

Practical optimisation problems to be solved via computer simulations are defined in *bounded domains* whose most typical shape is hyperrectangular. Research into the algorithmic design of optimisation methods from the field of computational intelligence [7] has shown that the chosen strategy of dealing with the solutions generated outside such domain – the *infeasible solutions* (ISs) – is an *essential part of the algorithm* that to a large extent decides the success of the optimisation method. Unfortunately, in the majority of papers in the field,

the choice of such strategy is overlooked or omitted from the publications, thus limiting the reproducibility of the results and lowering the overall impact of such studies.

To highlight the importance of this algorithmic operator, we employ five different strategies of dealing with ISs (SDIS):

1. Complete One-tailed normal correction strategy (`COTN`) [7] – only in infeasible dimensions, moves an infeasible solution inside the domain to a position resampled from the rescaled one-sided Normal distribution;
2. `dismiss`[6] – dismisses an infeasible solution and replaces it with one of the parent/generating points);
3. `mirror` [7] – mirrors the position of an infeasible solution in infeasible dimensions only inwards off the closest boundary;
4. `saturation` [5, 7] – moves an infeasible solution onto the closets boundary only in infeasible dimensions;
5. `toroidal` [5, 7] – reflects an infeasible solution inwards off the opposite boundary.

3.5 Experimental setup

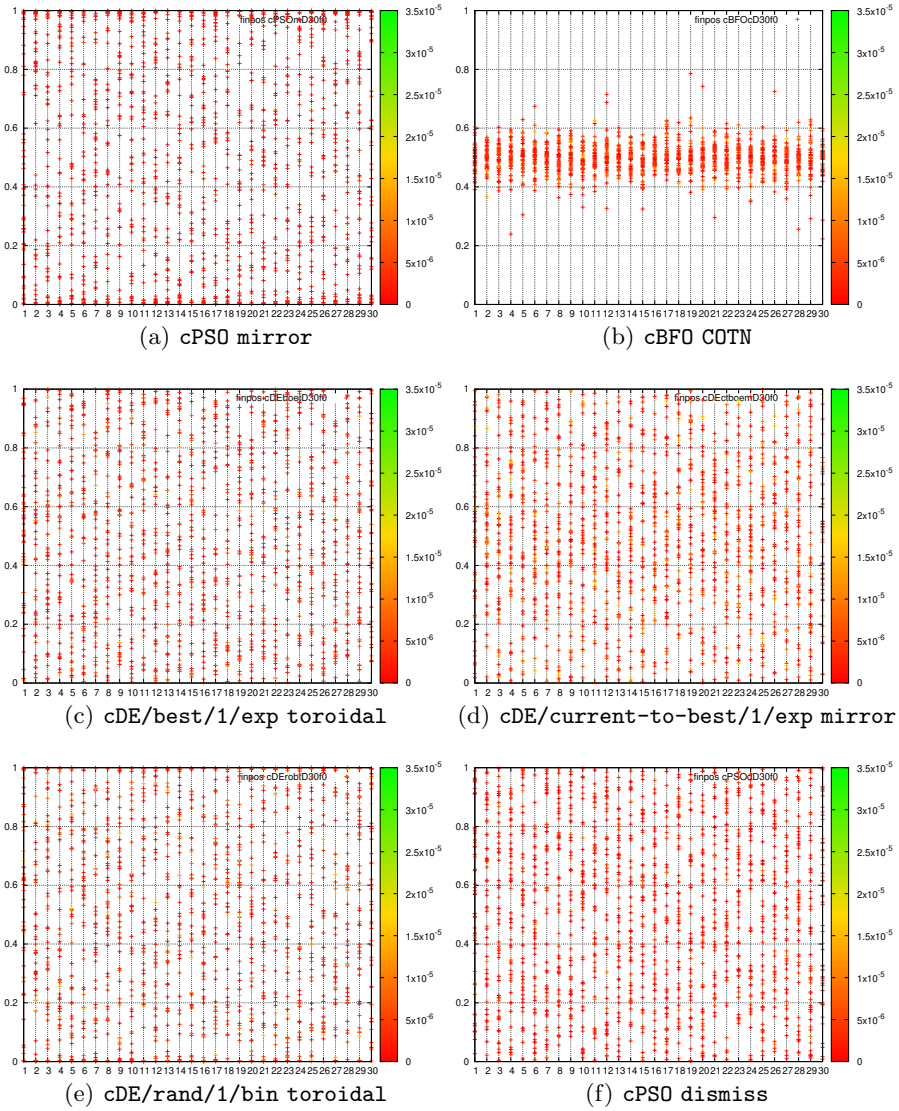
This experimentation involves 13 `cDE/x/y/z` variants and the 4 other algorithms described in Section 3.4. All of them but cGA (which generates only feasible solutions) are considered with 5 SDIS – the total of $16 \times 5 + 1 = 81$ configurations considered.

Results on the SB presented in this paper are based on *experiments*: (i) *minimising* the test function f_0 (see Section 3.2) for $n = 30$ (ii) by 81 algorithmic configurations described in Sections 2.1 and 3.4; (iii) each configuration is run 50 times; (iv) each run has independently seeded *Java random.utils* pseudorandom generator – seed is initialised with the current time since January 1, 1970 in milliseconds via Java's `System.currentTimeMillis`; (v) each run is budgeted in terms of the number of objective function evaluations as $10000n$.

All algorithms refer to their *persistent elitist* variants. All experiments are executed on a standard desktop using the SOS platform [4] implemented in Java (algorithms' source code is available online [3]). It is worth mentioning that the aforementioned pseudorandom generator used here for all experiments is considered *on the better side of the scale for linear congruential generators* [17]

4 Discussion of results

Using the approaches described in Sections 3.2 and 3.3, all 81 configurations have been investigated. Results in these figures are shown in *parallel coordinates* [12] and *should be read as follows*: final positions attained in a series of 50 independent runs of each configuration are shown with 50 '+' markers on each of the $n = 30$ parallel vertical 'axes'. Positions of these 'axes' identify dimensions and are shown on the traditional horizontal axis; meanwhile, the traditional vertical axis



(a) `cPSO mirror` (b) `cBFO COTN`

(c) `cDE/best/1/exp toroidal` (d) `cDE/current-to-best/1/exp mirror`

(e) `cDE/rand/1/bin toroidal` (f) `cPSO dismiss`

Fig. 1. Distribution of locations of final best solutions, *example* configurations that exhibit *strong* SB in Figs. 1(a), 1(b), *mild* SB with: *local clustering* in Fig. 1(c), *clustering across domain* in Fig. 1(d), *clustering on boundaries domain* in Fig. 1(e) and *large gaps* in Fig. 1(f). See Section 4 for explanation on how to read this figure.

shows the range of the dimension ($[0, 1]$ here). Values of f_0 attained by the final solutions are shown in colour (a recap: this is a minimisation problem). Due to the page limit in this publication, only a few figures are shown in Fig. 1. All results can be obtained from [6].

4.1 Visual tests

Following the methodology of visual testing described in Section 3.2, out of 81 configurations considered in this paper, only 6 configurations have been found to be structurally biased (e.g. Figs. 1(a), 1(b)), meanwhile 40 configurations exhibit only mild SB. It is worth highlighting that decisions in visual tests on whether mild SB is present are *highly subjective* and should be contrasted with results from statistical testing in Section 4.2.

The summary of results discussed in this Section can be found in Table 1 in the columns marked as 'visual SB test' for all basic compact configurations (rows) and all strategies of dealing with IS (smaller columns)[6].

Based on the *visual tests only*, overall, compact configurations appear to be *more 'immune' to the strong SB* than their equivalents maintaining explicit populations [6, 15, 7]. SB, if at all present, is more *subtle* across all configurations of compact algorithms considered. The resulting distributions of locations of final best solutions differ from the true uniform distribution in clustering of points and not in the span of the domain (with exception of all `cBFO` configurations as discussed below). It means that, on the whole, compact configurations of algorithms considered in this study *should have more exploratory potential* and be more successful in finding optima wherever they are situated in the domain. The latter one, however, is *not guaranteed* without the use of good exploitative operators (such investigation is *out of the scope* of this paper).

One of the exceptions to the above statement is all the `cBFO` configurations that have turned out to be badly biased towards the middle of the domain regardless of the choice of correction strategy, e.g. Fig. 1(b). More precisely, `cBFO` appears to be *unable* to find optima on f_0 outside the region $[0.4, 0.6]^{30}$ (with only a handful of exceptions per configuration).

Another exception to the above statement is the `cPSO mirror` configuration which exhibits strong SB towards all corners of the domain (see Fig. 1(a)) – *interestingly enough, such situation resembles the case of non-compact* `PSO` *with a small population size* [15].

When talking about *mild SB*, resulting distributions of the locations of final best solutions appear to *marginally* deviate from the uniform distribution in the following non-exclusive aspects:

1. 'higher-than-expected' clustering of points *within* the domain (e.g. Fig. 1(c));
2. 'higher-than-expected' clustering of points *across* the domain (e.g. Fig. 1(d);
3. 'higher-than-expected' clustering of points on the *boundaries*[7] (e.g. Fig. 1(e));
4. *large empty gaps* consistently identified in all 30 dimensions (e.g. Fig. 1(f)).

When analysing results for `cDE/x/y/z` only, out of 30 `bin` and 30 `exp` considered configurations, 16 and 13, respectively, appear to be *mildly biased*. Out 5 `cDE/current-to-rand/1` configurations that require *no crossover*, 3 appear

[6] To avoid complicating Table 1 further, results for `cGA` that requires no SDIS are shown as `dismiss` – it is the closest to how `cGA` deals with infeasible solutions.

[7] This is easily explained if `saturation` is used but is not trivial if `toroidal` is used.

Table 1. Comparison of results on the presence and strength of structural bias based on *visual* and *statistical* tests across all 81 configurations (see [6]). For both tests, cells with background in **black** mark configurations exhibiting strong SB, in **grey** - configurations with mild SB and in **white** - configuration with no SB identified based on the corresponding tests (i.e. *colour marks the corresponding decision of the test*). Cells containing '×' mark configurations that are not possible by design. *Symbols mark results of comparing the two tests*: symbol '=' stands for cells where results of the visual and statistical tests coincide (colour of the symbol has no meaning) and '•' - for the differences in results from visual and statistical tests (colour of the symbol has no meaning). Values shown in columns for statistical test are the *corresponding values of the statistic*. Thresholds for decisions based on these values are given in Section 4.2.

Kind of SB test:	visual					statistical				
Configuration:	COTN	dismiss	mirror	saturation	toroidal	COTN	dismiss	mirror	saturation	toroidal
cDE/rand/1/bin	=	=	•	•	=	0.00	0.00	0.02	∞	0.13
cDE/rand/1/exp	=	•	=	•	=	0.00	0.00	0.00	∞	0.02
cDE/rand/2/bin	•	=	•	•	•	0.02	0.00	0.02	∞	0.21
cDE/rand/2/exp	=	=	=	•	=	0.00	0.00	0.00	∞	0.06
cDE/current-to-rand/1	=	=	=	•	•	0.00	0.02	0.00	∞	0.00
cDE/best/1/bin	=	•	•	•	=	0.01	0.00	0.00	∞	0.00
cDE/best/1/exp	•	=	=	•	•	0.00	0.00	0.00	∞	0.00
cDE/best/2/bin	=	=	=	•	•	0.01	0.00	0.00	∞	0.01
cDE/best/2/exp	=	=	=	•	•	0.00	0.00	0.00	∞	0.01
cDE/current-to-best/1/bin	•	=	=	•	•	0.00	0.00	0.01	∞	0.02
cDE/current-to-best/1/exp	•	=	=	•	•	0.00	0.00	0.00	1.00	0.01
cDE/rand-to-best/2/bin	•	=	=	•	=	0.00	0.00	0.00	∞	0.01
cDE/rand-to-best/2/exp	•	•	•	•	•	0.02	0.00	0.00	∞	0.02
cDE-Light	•	•	•	•	•	0.00	0.00	0.00	0.01	0.00
cPSO	=	•	=	•	=	0.03	0.00	0.44	∞	0.01
cBFO	=	=	=	=	=	1.00	0.92	1.00	0.96	0.93
cGA (no SDIS, shown as dismiss)	×	•	×	×	×	×	0.01	×	×	×
Found strong SB/total cases:	1/16	1/17	2/16	1/16	1/16	1/16	1/17	2/16	15/16	2/16
Found mild SB/total cases:	8/16	6/17	4/16	13/16	9/16	5/16	2/17	3/16	1/16	10/16
Strong/mild/no SB cases:	6/40/35					21/26/59				
Agreement between visual and statistical tests (in %, calculated 'post factum'):	56	65	69	6	44	all cases				
	100	100	100	100	100	cases with strong SB only*				
	38	17	25	0	55	cases with mild SB only				
	71	90	80	0	17	cases with no SB only				

to be *mildly biased*. To some extent, it is fair to say that *simpler* `cDE/x/y/z` *configurations with* `y`>1 appear to be *freer of mild SB*.

4.2 Statistical tests

Here, we present the calculated values of the statistical measure of structural bias (defined in Section 3.3) in the 'statistical SB test' column of Table 1 (the meaning of symbols and colour scales are explained in the table caption). We use the 20- (0.00) and 90-quantiles (0.158)[8] of the statistical values over all combinations as *thresholds to determine the level of SB*. More specifically, zero values of statistic shall be classified as having *no* SB; ranges for *mild* and *strong* SB are $(0, 0.158]$ and $(0.158, 1] \cup \{+\infty\}$, respectively.

From results presented in the table, it is obvious that `cBFO` is exceptionally biased regardless of the SDIS. Also, the `saturation` SDIS seems to yield strong SB for all the algorithms except `cDE-Light`. For the remaining combinations, we observe either no or mild SB.

Comparing to the visual test on the same combinations, it seems that cases classified as strongly biased by the visual tests are always indicated as strongly biased as well from the statistical side – see the third line from the bottom in Table 1, marked with a *. However, since there are at most two discoveries of the strong bias from both tests, *the reliability of this agreement is questionable*. In contrast, cases with *mild* SB in the visual test are largely *mis-classified* as possessing no SB in the statistical approach. Also, most of the algorithms with the `saturation` SDIS are indicated as strongly biased by the statistical measure while those cases are considered mildly biased in the visual test. We conjecture the observed mismatches between those two approaches as follows: (i) the SB measure is calculated from a multiple testing procedure, where the p-value is corrected, thus the SB measure can suffer from a reduction of its statistical power (i.e., more false-negative decisions are made). This leads to a scenario that the Anderson-Darling test is rejected on all dimensions for those cases with mild SB in the visual test and hence the statistical measure classifies them as not biased; (ii) the SB measure is not scale-invariant and can be less informative after the performed normalisation. In this light, when no bias is displayed, we shall conclude that some SB degree is exhibited but negligible if compared to the bias shown by the most biased algorithm (i.e., `cBFO`). Such relativity in the statistical approach might be different from that in the visual test, which leads to the observed discrepancy.

5 Conclusions

The extensive experimentation presented in this piece of research has unveiled the presence of mild structural biases for most compact algorithms except `cBFO`

[8] The quantiles are chosen *ad hoc*, based on the distribution of statistical measure over all combinations of algorithms and SDISs.

and cPSO – the former one especially carries a so strong SB that can be categorically detected via the visual inspection of the generated graphs. More precisely, in cBFO, regardless of the employed SDISs, only the middle section of the search domain is populated with the found best solutions, while its peripheral areas are left completely out. This undesired algorithmic behaviour suggests that cBFO is not suitable for general-purpose optimisation, since it displays design flaws that limit its applicability to problems whose optimum/optima is/are at the centre of the search space. Similarly, also cPSO mirror displays a visible strong bias. However, it is interesting to observe that in this case, the solutions obtained over multiple runs accumulate towards the corners of the search space. This behaviour is in line with the one of the standard PSO algorithm – when employed with a small population size [15].

In a similar way, also the mild SB individuated in the remaining algorithms under study mainly reveals itself in the form of 'higher-than-expected' clustering of final solution located either across the domain or on the boundaries. However, in a few cases, uniformly distributed large empty gaps are also visible on each dimension of the generated graphs. Such gaps clearly flag the presence of SB, but final solutions do not accumulate in specific areas of the search space and thus do not seem to cause deleterious effect in terms of coverage of the whole domain. It is interesting to point out that amongst the cDE/x/y/z variants tested in this study, a mild SB is mainly visible only for those cases equipped with mutation operators using one difference vector – e.g. this is evident for the best/1 mutation, in particular when used in combination with binomial crossover bin. cDE variants equipped with mutation operators using two difference vectors, on the other hands, seem to be freer from SB – e.g. the case of rand/2, in particular when followed by exponential crossover exp.

To summarise, it can be stated that the compact algorithms under investigations appeared to be more 'immune' to the SB than their population-based equivalents according to the proposed visual test. However, it is important to conclude this study by observing that the proposed statistical SB detection method agrees with the visual test on strong SB cases while disagrees on most of the visually detected mild SB cases. We speculate that this discrepancy is caused by the insufficient sample size as well as the conservative nature of this testing procedure and we commit to investigating this aspect further in our future studies. We plan to increase the sample-size in future experimentation and, most importantly, improve upon the sensitivity of the proposed statistical measure with respect to the number of independent runs.

Acknowledgments. The work of Hao Wang was supported by the Paris Ile-de-France Region.

References

1. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York, NY (1996)
2. Campelo, F., Aranha, C.: EC Bestiary: A bestiary of evolutionary, swarm and other metaphor-based algorithms (Jun 2018). https://doi.org/10.5281/zenodo.1293352

3. Caraffini, F.: The stochastic optimisation software (SOS) platform (june 2019). https://doi.org/10.5281/zenodo.3237023
4. Caraffini, F., Iacca, G.: The SOS platform: Designing, tuning and statistically benchmarking optimisation algorithms. Mathematics **8**(5), 785 (May 2020). https://doi.org/10.3390/math8050785
5. Caraffini, F., Kononova, A.V.: Structural bias in differential evolution: a preliminary study. In: LeGO 2018 - 14th International Workshop on Global Optimization. vol. 2070, p. 020005. AIP, Leiden, The Netherlands (2018)
6. Caraffini, F., Kononova, A.V.: Structural Bias in Optimisation Algorithms: Extended Results (2020). https://doi.org/10.17632/zdh2phb3b4.2, mendeley Data
7. Caraffini, F., Kononova, A.V., Corne, D.W.: Infeasibility and structural bias in differential evolution. Information Sciences **496**, 161–179 (2019). https://doi.org/10.1016/j.ins.2019.05.019
8. Das, S., Biswas, A., Dasgupta, S., Abraham, A.: Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications, pp. 23–55. Springer (2009). https://doi.org/10.1007/978-3-642-01085-9_2
9. De Jong, K.A.: An Analysis of the Behavior of a Class of Genetic Adaptive Systems. Ph.D. thesis, University of Michigan, USA (1975)
10. Hauschild, M., Pelikan, M.: An introduction and survey of estimation of distribution algorithms. Swarm and Evolutionary Computation **1**(3), 111–128 (2011). https://doi.org/10.1016/j.swevo.2011.08.003
11. Iacca, G., Caraffini, F.: Compact optimization algorithms with re-sampled inheritance. In: Kaufmann, P., Castillo, P.A. (eds.) Applications of Evolutionary Computation. pp. 523–534. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-16692-2_35
12. Inselberg, A.: The plane with parallel coordinates. The Visual Computer **1**(2), 69–91 (1985). https://doi.org/10.1007/BF01898350
13. Justel, A., Peña, D., Zamar, R.: A multivariate Kolmogorov-Smirnov test of goodness of fit. Statistics & Probability Letters **35**(3), 251–259 (1997)
14. Kononova, A.V., Caraffini, F., Wang, H., Bäck, T.: Can single solution optimisation methods be structurally biased? MDPI Preprints (2020). https://doi.org/10.20944/preprints202002.0277.v1
15. Kononova, A.V., Corne, D.W., Wilde, P.D., Shneer, V., Caraffini, F.: Structural bias in population-based algorithms. Information Sciences **298**, 468–490 (2015). https://doi.org/10.1016/j.ins.2014.11.035
16. Kost, J.T., McDermott, M.P.: Combining dependent p-values. Statistics & Probability Letters **60**(2), 183–190 (2002)
17. L'Ecuyer, P., Simard, R.: TestU01: A C Library for Empirical Testing of Random Number Generators. ACM Transactions on Mathematical Software **33**(4) (2007). https://doi.org/10.1145/1268776.1268777
18. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions i. binary parameters. In: Voigt, H.M., Ebeling, W., Rechenberg, I., Schwefel, H.P. (eds.) Parallel Problem Solving from Nature – PPSN IV. pp. 178–187. Springer (1996)
19. Pelikan, M., Goldberg, D., Lobo, F.: A survey of optimization by building and using probabilistic models. In: Proceedings of the 2000 American Control Conference. vol. 5, pp. 3289–3293 (2000)
20. Piotrowski, A.P., Napiorkowski, J.J.: Searching for structural bias in particle swarm optimization and differential evolution algorithms. Swarm Intelligence **10**(4), 307–353 (2016). https://doi.org/10.1007/s11721-016-0129-y

21. Price, K.V., Storn, R., Lampinen, J.: Differential Evolution: A Practical Approach to Global Optimization. Springer (2005). https://doi.org/10.1007/3-540-31306-0
22. Razali, N.M., Wah, Y.B.: Power Comparisons of Shapiro-wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling Tests. Journal of Statistical Modeling and Analytics **2**(1), 21–33 (2011)
23. Wolpert, D., Macready, W.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation **1**, 67–82 (1997). https://doi.org/10.1109/4235.585893