BIAS: A Toolbox for Benchmarking Structural Bias in the Continuous Domain

Diederick Vermetten, Bas van Stein, *Member, IEEE*, Fabio Caraffini, *Member, IEEE*, Leandro L. Minku, *Senior Member, IEEE*, and Anna V. Kononova, *Member, IEEE*

***Abstract*—Benchmarking heuristic algorithms is vital to understand under which conditions and on what kind of problems certain algorithms perform well. Most benchmarks are performance based, to test algorithm performance under a wide set of conditions. There is also resource- and behavior-based benchmarks to test the resource consumption and the behavior of algorithms. In this article, we propose a novel behavior-based benchmark toolbox: BIAS (Bias in algorithms, structural). This toolbox can detect structural bias (SB) per dimension and across dimension-based on 39 statistical tests. Moreover, it predicts the type of SB using a random forest model. BIAS can be used to better understand and improve existing algorithms (removing bias) as well as to test novel algorithms for SB in an early phase of development. Experiments with a large set of generated SB scenarios show that BIAS was successful in identifying bias. In addition, we also provide the results of BIAS on 432 existing state-of-the-art optimization algorithms showing that different kinds of SB are present in these algorithms, mostly toward the center of the objective space or showing discretization behavior. The proposed toolbox is made available open-source and recommendations are provided for the sample size and hyper-parameters to be used when applying the toolbox on other algorithms.**

***Index Terms*—Evolutionary computation, optimization methods, statistical analysis.**

I. INTRODUCTION

THE MODERN world has an ever-growing need for good heuristic optimization algorithms due to the large amounts of data and increasingly difficult problems to be optimized. Since one overall best heuristic does not exist [1], we have to benchmark heuristics to understand which one is better under what conditions. Benchmarking can be performance based, resources based, or behavior based. Most benchmarks for continuous optimization are performance based. An example is the well-known black-box optimization benchmark (BBOB) [2] test suite. Performance-based benchmarks aid in learning about the performance of one algorithm and the comparison between other algorithms in different situations. For example, one algorithm could perform very well on separable functions and another algorithm on uni-modal, high-conditioned functions. Resource-based benchmarks show the amount of resources (computation power/memory/energy) required under certain conditions. However, these types of benchmarks do not easily allow the analysis of the "behavior" of these algorithms under different circumstances. Behavior is, for example, how a population of candidate solutions move in a swarm optimization algorithm either dependent or independent of the function to optimize. An example of a behavior-based benchmark is RobustBench [3], an adversarial attack benchmark designed to test the robustness of image classification deep neural networks. Behavior-based benchmarks can be used to learn additional information about the behavior of an algorithm under different conditions. Here, we concentrate on a particular kind of behavior-based performance estimate, structural bias (SB). SB is a form of bias inherent to the iterative heuristic optimizer in the objective space that also affects the performance of the optimization algorithm.

Detecting whether, when, and what type of SB occurs in a heuristic optimization algorithm can provide guidance on what needs to be improved in these algorithms, besides helping to identify conditions under which such bias would not occur. In many cases, SB can be avoided by slightly redesigning an algorithm component or by using different hyper-parameters. However, SB is hard to detect when not specifically looking for these kind of issues. Therefore, in this article we propose a toolbox to automatically benchmark continuous optimization algorithms to discover a portfolio of eleven different SB scenarios. We aim to answer the following research questions.

- *RQ1:* How to determine whether a heuristic continuous optimization algorithm suffers from SB?
- *RQ2:* How to determine the type of SB suffered by a heuristic continuous optimization algorithm?
- *RQ3:* Which heuristic continuous optimization algorithms suffer from what type of SB? Under what conditions?

To answer RQ1, we evaluate a large set of statistical tests based on 1500 repetitions of 194 different artificially generated parameterized distributions containing 11 different scenarios of SB. The proposed toolbox can be used to detect potential bias issues in newly developed algorithms as well as benchmark already existing optimization algorithms. To answer RQ2, we propose a machine learning approach to identify which SB scenarios are most likely occurring in a given

Manuscript received 9 September 2021; revised 27 February 2022; accepted 28 June 2022. Date of publication 13 July 2022; date of current version 1 December 2022. *(Corresponding author: Diederick Vermetten.)*

Diederick Vermetten, Bas van Stein, and Anna V. Kononova are with the Leiden Institute of Advanced Computer Science, University Leiden, 2311 EZ Leiden, The Netherlands (e-mail: d.l.vermetten@liacs.leidenuniv.nl).

Fabio Caraffini is with the School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, U.K.

Leandro L. Minku is with the School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TEVC.2022.3189848.

Digital Object Identifier 10.1109/TEVC.2022.3189848

optimizer based on the resulting test statistics. The complete SB benchmark statistical test suite (BIAS), data generators for different SB scenarios, and the machine learning model for identifying different SB scenarios are provided open-source [4]. To answer RQ3, we adopt the proposed toolbox to detect SB in 432 different continuous optimization algorithms. We also discuss insights provided by the resulting benchmarking of these algorithms and potential future research directions.

This article is structured as follows: in Section II, SB is introduced and explained in detail. The main components of our proposed BIAS toolbox are also briefly introduced. In Section III, all considered statistical tests for uniformity are introduced, including several newly proposed tests specifically for SB. In Section IV, the SB scenarios and experiments are explained in detail. In Section V, an analysis of the statistical tests is conducted, answering RQ1. In Section VI, the machine learning approach used to identify the type of SB is introduced and evaluated, answering RQ2. In Section VII, existing heuristic continuous optimization algorithms are benchmarked, answering RQ3. Finally, we end with recommendations and conclusions in Section VIII.

II. Structural Bias

Put simply, algorithms are tools applied to particular problems. Just like a good hammer working well on many nails, we would like algorithms to ideally deliver good solutions for many problems. However, as postulated by the so-called No-Free Lunch Theorem, no best optimizer exists over *all* possible optimization problems [1]. What is possible though is algorithms specializing on *certain kind* of problems. Identifying sets of such suitable problems for each algorithm is not a trivial problem [5] and many landscape features have been investigated to understand the success of some algorithms on some kinds of functions [6]. However, what cannot practically be a feature uniting such problems is the location of optima in the domain—having an algorithm that consistently finds an optima only located in the origin is of no use. Therefore, good algorithms should not be biased toward specific locations of the search space, e.g., toward solutions at the origin, center, or in the borders of the search space. Extrapolating such reasoning, a good optimization algorithm should be able to find the optima regardless where exactly they are located within the domain. Or, even stronger, a good algorithm should ideally locate solutions anywhere in its domain with equal "effort."

In case of iterative optimization algorithms, points sampled during the initialization "move" within the domain defined by its boundaries under the influence of algorithm's operators and potentially bring improvement of the values of the objective function during the optimization run, following some kind of "survival-of-the-fittest" logic. In effect, such movement of the algorithm toward the optima gets steered by the differences in the values of the objective function in the sampled points or their derivates of some kind. Any feedback that is external to the objective function or domain knowledge might hinder such progression to the optima. Such external feedback stemming from the iterative nature of the algorithm is referred to here as SB.

Because of the high interdependence between the fitness landscape and the information on the fitness obtained from this cyclical application of the algorithm's operators, the SB contribution during the search for optima cannot be easily unveiled if not by means of a specific objective function capable of nullifying such interaction over multiple optimization runs. The f_0 function, first introduced in [7], serves this purpose and can be used to decouple these effects, thus separating the SB component, arising from algorithmic design choices, from the main driving force represented by the sampled differences in the fitness landscape. Function f_0 is a "truly" random problem, having the simple analytical representation reported in the following[1]:

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

A. Existing Methodology for Measuring SB

As explained in Section II, a heuristic optimization algorithm that does not exhibit SB should be able to find randomly placed (uniformly distributed) optima for the special objective function f_0 with equal difficulty/ease. Therefore, the problem of determining whether a heuristic optimization algorithm suffers from SB can be reduced to the problem of checking whether the best solutions found by multiple runs of this algorithm to minimise[2] f_0 are uniformly distributed. However, such uniformity check is not a trivial task due to the many forms in which SB can manifest itself for different algorithms (see Section II-B) and under different algorithmic configurations/parameter settings. To complicate things further, challenges encountered while testing for SB depends on the experimental setup used to execute optimizers, e.g., where the number of performed runs might be adequate for classic performance-based assessment but not for detecting SB. Moreover, testing for SB also depends on the allowed computational budget allocated to the algorithm and/or employed termination criterion. Indeed, the study in [8] has shown that when SB emerges during the evolutionary process (for several versions of the Differential Evolution algorithm), it only grows stronger during the remaining evaluations. This means that practitioners using different experimental settings might end up looking at biases of *different* strengths for the same algorithm, which makes direct comparisons unfair. These details can make it difficult to generate a practical and robust procedure for the SB detection.

Up until now, methods to check the uniformity of the distribution of best solutions over multiple runs included visual or statistical inspections, briefly discussed in Sections II-A1 and II-A2, respectively.

1) Visual Test: Displaying locations of the best solutions collected from multiple runs in the so-called "parallel coordinates" [9] appears to be the most effective way for

[1]Throughout this article we refer to 1-D uniform sampling within $[a, b]$ as $\mathcal{U}(a, b)$ and to sampling from the normal distribution with mean μ and standard deviation σ as $\mathcal{N}(\mu, \sigma)$.

[2]Or maximize.

visualizing SB on a multidimensional problem [7]. This approach is easily reproducible, graphically valid, and hence convenient. However, when a large number of images is generated [8], [10]–[12], visual inspection can become too laborious. Such approach is also clearly subjective and therefore not reliable for cases where graphical artefacts or unclear patterns cannot be judged by a naked eye.

2) Statistical Test: Let us consider a heuristic optimization algorithm that was run N times to solve the problem of minimizing f_0. At the end of each run i, the best solution $\mathbf{x}^{(i)}$ found by the algorithm by the end of the run is recorded, where naturally $\mathbf{x}^{(i)} \in [0,1]^n$. The random sample $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$ represents the set of best solutions retrieved by the N runs of the algorithm. Assume that $\{x_j^{(1)}, x_j^{(2)}, \ldots, x_j^{(N)}\}, j \in \{1, 2, \ldots, n\}$ was drawn from a probability distribution with a continuous cumulative distribution function $\mathcal{F}_d$. A goodness-of-fit test can be used to test the null hypothesis $H_0 : \mathcal{F}_d \sim \mathcal{U}(0,1)$.

The Kolmogorov–Smirnov test [13] was first employed with a sample of size $N = 50$ and significance level $\alpha = 0.01$ in [7]. Subsequently, following the "power analysis" performed in [14] across three common tests, namely, Kolmogorov–Smirnov, Cramér-Von Mises [15], and Anderson–Darling [16] tests, the latter test was chosen and used in combination with the Benjamini–Hochberg [17] correction method for multiple comparisons to achieve higher statistical power. However, it was noted that the original sample size was not adequate for testing all algorithms under investigation. Hence, a higher number $N = 100$ of runs had to be used for some algorithms in order to achieve a satisfactory level of statistical power. Similar problems were encountered in a further study on SB in a subclass of Estimation of Distribution Algorithms [18], even when using an aggregated measure of SB defined as the sum of the statistically significant (across all dimensions) test statistics of the Anderson–Darling test.

It was concluded that the described statistical approaches can effectively detect most cases of "strong" SB but are deficient on other scenarios, including clear "mild" SB that can be identified with the visual approach. Reasons for these discrepancies between the two methodologies might be a conservative nature of the employed tests combined with the relatively low sample size. Indeed, more accurate SB detection results could be obtained with $N = 600$, as used in [8], instead of $N = 100$.

Using a large sample size ($N = 600$) indeed seems to catch bias more often, but still gives no guarantee to detect *all* different kinds of SB, at any significance level [8]. Even larger sample sizes are necessary for smaller levels of significance, higher desired power, and smaller sizes of the deviations to be identified [19]. However, given the limited computational resources to run heuristic optimization algorithms, it is not always possible to obtain (very) large sample sizes. Therefore, tests better able to detect significant deviations from uniformity given limited sample sizes are desirable for detecting SB. We study this in more detail in Section V-B.

Regardless of the above, there is a clear need for a better automated statistical testing procedure to detect SB readily available to the community as a software package.

B. Known Deviations From Uniformity Due to SB

Formulating a good statistical measure of SB has turned out to be difficult (see Section II-A2) due to a wide range of potential deficiencies of distributions of locations of final best points produced by the algorithm in a series of independent runs—i.e., *in what aspect* samples from such distributions deviate from uniformity.

The following deviations have been observed for different kinds of iterative heuristic optimization algorithms [7], [8], [10], [14], [18], [20]:

1) in terms of proportion of occupied continuous domain:
    a) sample values do not span the whole domain—the sample is uniform on an interval smaller than [0, 1] ("centre-bias");
    b) values in a sample cover discrete set of values (regularly spaced) within [0, 1] ("discretization-bias");
2) in terms of locations of points, sample points exhibit:
    a) local clustering within the domain or clustering around several clusters within the domain ("cluster-bias");
    b) clustering in certain parts of each dimension: on the bounds, in one or both sides of each dimension ("bound-bias");
    c) one or more large empty gaps consistently identified in all dimensions ("gap-bias");
3) in terms of correlations between different dimensions of the sample points: present or not.

C. BIAS Toolbox

Our proposed BIAS (bias in algorithms, structural) toolbox fills in the gap in terms of the need for a better-automated procedure to detect SB and identify its type. In this article, we discuss the different components which constitute the proposed BIAS toolbox. These components include 39 statistical tests and a procedure to aggregate them for detecting SB (Section III), a random forest (RF) model to identify the type of SB (Section VI), the code for the used f_0 (which was described in Section II) and the code for the generation of the statistical bias decisions and plots (of which an example is illustrated in Fig. 3).

The BIAS toolbox is available as a python package at [4]. It provides a clear decision on whether or not SB is present in the sample of final positions provided by the user and, in case some SB is found, an assessment on the possible type of SB observed in the sample using an RF model. In addition to the SB detection, the toolbox also contains the needed functionality to sample data from the scenarios in Table I, thus providing the means to benchmark other statistical test for detecting SB.

III. Tests

As explained in Section II-A2, the goodness-of-fit test previously adopted for detecting SB (`Anderson-Darling`) is deficient on several cases where SB is clearly visible even with a sample size of 100. More powerful tests are therefore desirable. This section introduces existing and newly proposed goodness-of-fit tests evaluated in this article for the purpose of detecting SB in samples of a limited size. This wide selection of tests includes both classical statistical tests and other tests that have demonstrated competitive power for testing uniformity in the statistical literature [21], [22]. All of these tests are used to test the *null hypothesis* $H_0 : \mathcal{F}_d \sim \mathcal{U}(0, 1)$ introduced in Section II-A2.

A. Tests Per Dimension

The following tests are designed to work on an individual dimension. However, by aggregating all data we can run these test on a sample size which is effectively n times larger. If these tests are run on a per-dimension basis, correction strategies need to be applied to deal with the multiple-comparison problem. For this purpose, the Benjamini–Holberg (BH) method was proposed originally, since it is less stringent than the standard Bonferroni method. However, in this work, we also investigate the effects of other multiple comparison correction methods, which is discussed in Section V-D.

1) 1-Spacing-Based Test (`1-Spacing`) [4], [23][3]: To detect cases where the bias takes the form of large spacing differences or clustering in each dimension, we can test the distribution of distances between consecutive points (or points with m neighbors between them, $m \in \{1, 2, 3\}$ here and in two subsequent tests). Such distribution can then be compared to the distribution from a large sample of truly uniform random samples using a 2-sample KS test. This and two subsequent tests in the list belong to a wider class of tests called *`m-spacing`*.

2) 2-Spacing-Based Test (`2-Spacing`) [4], [23][3]: See Section III-A1.

3) 3-Spacing-Based Test (`3-Spacing`) [4], [23][3]: See Section III-A1.

4) Sample Range-Based Test (`Range`) [4][3,4]: The range taken up by the samples in each dimension can be useful in detecting SB where the points are far removed from the boundaries. The same can be done using the sample extrema (next two items).

5) Sample Minimum-Based Test (`Min`) [4][3,4]: See Section III-A4.

6) Sample Maximum-Based Test (`Max`) [4][3,4]: See Section III-A4.

7) Anderson–Darling (`AD`) [16], [24]: Is the most commonly used test for checking uniformity, and as such the test proposed to detect SB in [18].

8) Transformed AD Test (`tAD`): To boost the power of the *`AD`* test in some of the most common cases of SB, where the bias occurs near either the boundaries or the center of the space, a transformation can be applied to the samples by taking their distance to the nearest boundary [20], [24]. This can then be tested using an *`AD`* test with domain $[0, (1/2)]$.

9) Shapiro Test (`Shapiro`): Instead of testing the samples for uniformity directly, we can transform them to their normal counterpart, and check for normality using tests like the Shapiro test [25], [26].

10) Jarque–Bera Test (`JB`): This test, obtained through the use of the Lagrange multiplier test on the Pearson family of distributions [27], [28], is also applied to transformed samples to test for their normality.

11) Minimum Linear Distance-Based Test (`LD-Min`) [4][4]: The minimum distance between a set of samples and a linearly uniform distribution using the same range and sample size.

12) Maximum Linear Distance-Based Test (`LD-Max`) [4][4]: The maximum distance between a set of samples and a linearly uniform distribution using the same range and sample size.

13) Kurtosis-Based Test (`Kurt`): This test[3,4] measures how differently the tails of a distribution are shaped, compared to the tails of the normal distribution. Kurtosis is defined as the fourth standardized central moment, of the random variable of the probability distribution [26], [29]. Note that this test is applied after transforming the data to its normal counterpart.

14) Minimal Minimum Pairwise Distance-Based Test (`MPD-Min`) [4][3,4]: The minimum pairwise distances between neighbors. If the minimum distance between neighbors is too small, this can indicate dense clusters.

15) Maximal Minimum Pairwise Distance-Based Test (`MPD-Max`) [4][3,4]: Same as *`MPD-min`* but now comparing the maximum instead. If the maximum distance between neighbors is too large, this indicates uncovered areas of the objective space.

16) Wasserstein Distance-Based Test (`Wasserstein`) [4][3]: Originally proposed for optimal allocation problems [30] and rigorously formulated by Kantorovitch in [31], this metric is used to measure the distance between two probability distributions defined on a metric space. This result is exploited in this study to build the *`Wasserstein`* test for uniformity.

17) Neyman-Smooth Test (`NS`): Neyman [32] and Biecek and Ledwina [33] constructed his smooth tests specifically to test for the continuous uniform distribution.

18) Kolmogorov–Smirnov Test (`KS`) [13], [24]: Is a nonparametric test based on the maximum distance between the empirical cumulative distribution function (ECDF). While this test is most commonly used to compare two samples, it can also be used for goodness-of-fit testing.

19) Cramer-Von Mises Test (`CvM`) [22], [34]: Is similarly based on the expected squared difference between the ECDF and true CDF function of the null distribution.

20) Durbin Test (`Durbin`): To test for uniformity, the Durbin C_n test [35] compares the cumulated periodogram graph of the residuals from a least-squares regression with the Kolmogorov–Smirnov limits. This is based on previous results showing that when the employed test statistic is calculated with a generic number $2m + 1$ of samples then it is distributed as the mean of $m - 1$ independent uniform variables [22], [36].

21) Kuiper Test (`Kuiper`): This is an extension of the standard Kuiper test obtained by replacing the original distribution with Pyke's modified empirical distribution function as proposed in [37] and implemented in [22]. This makes the test suitable for showing that a population has a prescribed continuous distribution function, including uniform.

22) 1st Hegazy-Green Test (`HG1`): This test refers to the variant of the Hegazy-Green tests for uniformity proposed

[3]For calculation of critical values, see Section III-C.

[4]Test proposed here.

in [38] and implemented in [22] employing the T_1 test statistic defined as the average of the absolute differences between samples and their expected value.

23) 2nd Hegazy-Green Test (`HG2`): This is the second variant of the Hegazy-Green test in [38] and [22] employing the T_2 test statistic defined as the average of the squared differences between samples and their expected value.

24) Greenwood Test (`Greenwood`): This test is based on a spacing statistic that can be used to evaluate events in time or locations in space by testing how the intervals between them are distributed [22], [39].

25) Quesenberry–Miller Test (`QM`): This test [22], [40] is a modification of the Greenwood test [39] which additionally considers the co-occurrence of extreme squared spacing distances.

26) Read-Cressie Test (`RC`): This test [22], [41] belongs to a family of goodness-of-fit tests based on the power divergence statistic.

27) Moran Test (`Moran`): This test [22], [42] operates via the distribution of the sum of squares of intervals into which the domain is divided, using a homogeneity of variances test.

28) 1st Cressie Test (`Cressie1`): This test [22], [43] is based on the logarithm of m-spacing gaps.

29) 2nd Cressie Test (`Cressie2`): This test [22], [44] operates via the family of statistics based on m-spacing gaps, obtained by summing a suitably regular function of each spacing gap.

30) Vasicek Test (`Vasicek`): As entropy can be used to characterize distributions, this test uses an estimate of entropy based on higher order ($m > 1$) m-spacings to test for uniformity [22]. Entropy had been originally adopted by Vasicek [45] for testing the hypothesis of normality.

31) Swartz Test (`Swartz`): Instead of using entropy, de Micheaux and Tran [22] and Swartz [46] use the Kullback–Leibler information to derive a statistic to test for uniformity. This statistic is also based on m-spacings.

32) Morales Test (`Morales`): This test [22], [47] compares an empirical and a hypothetical distribution based on the limit laws for a statistic based on ϕ-disparity [48], [49] of m-spacings.

33) Pardo Test (`Pardo`): Informational Energy is a measure of certainty, related to the Shannon entropy, which is a measure of uncertainty. This test uses m-spacings to estimate Informational Energy as a test statistic for the hypothesis of uniformity [22], [50].

34) Marhuenda Test (`Marhuenda`): This test [21], [22] uses a quantile-based divergence statistic based on Cressie and Read's power divergence statistics [41].

35) 1st Zhang Test (`Zhang1`): Zhang [51] proposed a general parameterized test statistic that can be used to derive both classical goodness-of-fit test statistics, such as Anderson–Darling [16] and Kolmogorov–Smirnov [13] and other new goodness-of-fit test statistics that were demonstrated to be more powerful [51]. `Zhang1` is derived based on this parameterized test statistic using likelihood ratio statistics [22], [51].

36) 2nd Zhang Test (`Zhang2`): This test statistics [22], [51] is derived in a similar way to `Zhang1`, but is based on an approximation.

B. Tests Across Dimensions

In addition to the tests which work on a per-dimension basis, we can also perform tests on the full set of 30-D data at once. This can be done by grouping together the samples or distances and performing the same test as the per-dimension testing on the aggregated data. For this purpose, we use all tests discussed above, with the exception of the sample limits-based tests, `LD`, `MPD`, and `Wasserstein` tests. It can also be done by using across-dimension tests. In this work, we use the following across-dimension tests.

1) Mutual Information-Based Test (`MI`): This test[3,4] is based on the fact that mutual information [52], [53] between variables of a random distribution should be close to zero. If the MI between two variables is higher, this suggests bias toward specific values in the domain shared by these two variables (dimensions). The median mutual information is the median over all dimensions.

The mutual information between two random variables U and V is defined as follows:

$$\mathrm{MI}(U, V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|}.$$

2) Maximum Minimum Pair-Wise Distance-Based Test (`MMPD`) [4][3,4]: This test is the maximum distance between two neighboring points in a (multi) dimensional space. This distance should not be significantly different from the same distance metric over a random uniform sampling. A significant higher distance means that there is a region in the search space where the algorithm is more attracted to (and subsequently a region that is avoided). In another words, bias.

3) Maximum Difference Per Dimension Between Linear Uniform Distribution-Based Test (`MDDLUD`) [4][3,4]: This test is the multidimensional equivalent of the `LD-max` test, where we aggregate either using the maximum or median across all 30 dimensions.

C. Critical Values for All Tests

To determine rejection based on the test statistic, we need to either calculate the corresponding p-values, or check if the test statistic exceeds the corresponding critical value. Several of the test we include calculate the p-value by default, but for the others we will use the critical values. To get accurate estimates of these values, we use a 100 000 samples Monte Carlo simulation of the test statistic under the uniform distribution, from which we determine the α-quantiles for $\alpha \in \{0.01, 0.05\}$. The Monte Carlo test is a well-known procedure for implementing hypothesis tests [54]. It enables calculating the critical values when the true (sampling) distribution of the test statistic is unknown. The resulting critical values calculated using this procedure are available at [55].

IV. Methodology

To effectively judge the performance of the proposed tests for different types of SB, we have defined a large portfolio of bias scenarios according to which we can generate an arbitrary number of samples. This set of scenarios is chosen in such a way that all common types of SB discussed in Section II-B are

TABLE I
OVERVIEW OF PARAMETERIZED DATA SAMPLING SCENARIOS IN [0, 1], THE TOTAL OF 194/249 SCENARIO SETTINGS (PER DIMENSION/ACROSS DIMENSIONS), PER CONSIDERED SAMPLE SIZE

scenario name	how sampled	parameter 1	parameter 2	settings	diagnosis[5]
`Uniform`[6]	sample full sample size via $\mathcal{U}(0,1)$	–	–	1	no bias
`Cut Uniform`[7]	subscenario 1: sample full sample size via $\mathcal{U}(z_c,1)$, subscenario 2: sample full sample size via $\mathcal{U}(\frac{z_c}{2}, 1-\frac{z_c}{2})$	fraction cut $z_c \in \{0.01, 0.025, 0.05, 0.1, 0.2\}$		10/10	centre-bias
`Cut Normal`[8]	sample full sample size via $\mathcal{N}(\mu,\sigma)$, remove all points outside $[0,1]$ and repeat until full sample size	$\sigma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$	$\mu \in \{0.5, 0.6, 0.7\}$	15/15	centre-bias
`Inverse Cut Normal`	same as above, but transform to have most mass at bounds	same as above	same as above	15/15	bound-bias
`Cut Cauchy`	similar to `Cut Normal` but for Cauchy distribution	same as above	same as above	15/15	centre-bias
`Inverse Cut Cauchy`	same as above, but transform to have most mass at bounds	same as above	same as above	15/15	bound-bias
`Clusters`	sample n_c cluster centre points c_i via $\mathcal{U}(0,1)$, sample remaining points around them via $\mathcal{N}(c_i,\sigma)$[9]	number of clusters $n_c \in \{1,2,3,4,5\}$	$\sigma \in \{0.01, 0.025, 0.05, 0.1, 0.2, 0.3\}$	30/30	cluster-bias
`Consistent Clusters`[10]		same as above	$\sigma \in \{0.01, 0.025, 0.05, 0.1, 0.2, 0.3\}$	0/30	cluster-bias
`Loose Clusters`	sample z_u portion of sample size via $\mathcal{U}(0,1)$. For each remaining point, select an existing point x_i and sample $\mathcal{N}(x_i,\sigma)$	fraction of uniform samples $z_u \in \{0.1, 0.25, 0.5\}$	$\sigma \in \{0.01, 0.02, 0.05, 0.1\}$	12/12	cluster-bias
`Gaps`	sample full sample size via $\mathcal{U}(0,1)$, select n_c sampled points x_i, remove all sampled points in $[x_i - r_g, x_i + r_g]$, resample missing points outside gaps via $\mathcal{U}(0,1)$ until full sample size[11]	number of centres $n_c \in \{1,2,3,4,5\}$	gap radius $r_g \in \{0.01, 0.02, 0.03, 0.04, 0.05\}$	25/25	gap-bias
`Consistent Gaps`[12]		same as above	same as above	0/25	gap-bias
`Spikes`	randomly sample integers in $[1, n_s]$, rescale as uniformly placed spikes in $[0,1]$	number of spikes $n_s \in \{25, 50, 100, 150, 200, 250, 500, 1000\}$	–	8/8	discretization-bias
`Noisy Spikes`	same as above, but spike locations are independently shifted according to $\mathcal{N}(0,\sigma)$	same as above	$\sigma \in \{0.005, 0.01, 0.02, 0.03, 0.04, 0.05\}$	48/48	discretization-bias

[5] See Section II-B
[6] Sanity check, excluded from the tests
[7] Two subscenarios: 1. modify only min (equivalent to varying only max, so don't do both to save time); 2. modify both min and max at the same time, with the same parameter setting (half cut on both sides)
[8] Vary μ only to one side since it is equivalent to the other side
[9] For across-dimension tests, need to differentiate between same cluster-centres in each dimension (`Consistent Clusters`) vs. different gaps (`Clusters`)
[10] Only used in across-dimension tests, as it is equivalent to `Clusters` per-dimension
[11] For across-dimension tests, need to differentiate between same gaps in each dimension (`Consistent Gaps`) vs. different gaps (`Gaps`)
[12] Only in across-dimension tests, as it is equivalent to `Gaps` per-dimension

represented. Additionally, these scenarios are parameterized to control the level of bias, which enables us to better judge the robustness of tests.

A. Portfolio of Scenarios

We choose the scenarios to include based on the most common types of SB (see Section II-B) as observed in previous papers.

1) Bias toward the center of the search space.
2) Bias toward the bounds of the search space.
3) Bias toward certain parts of the search space forming clusters.
4) Bias toward avoiding certain parts of the search space and creating gaps.
5) Strong discretization.

In total, this gives us 11 distinct scenarios (13 for the across-dimension case)—the full list these scenario definitions and their parameters used in this article is shown in Table I. An example of the density of several parameterizations of scenario `Cut Normal` is shown in Fig. 1.

In Fig. 2, we show that there exists minimal overlap between the densities of the different parameterizations of these scenarios by collecting 10 000 samples and performing pairwise *KS* tests. Note that these samples are aggregations of multiple independent dimensions, so for `Gaps` this means that each set of 100 samples has its own gap-centers, which explains why it is seen as similar to `Uniform`.

In total, this gives us 194 scenarios to consider in the per-dimension case, and 249 scenarios in the across-dimension case. For each of these scenarios, we generate data with sample sizes {30, 50, 100, 600}. In the per-dimension case, we collect

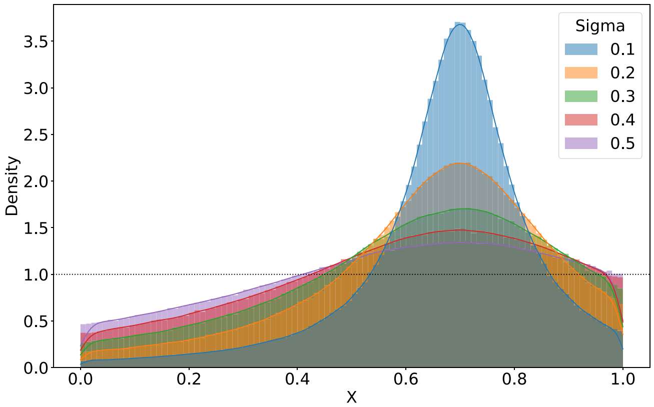

Fig. 1. Empirical density distribution for the `Cut Cauchy` scenario, with varying σ, and $\mu = 0.7$ (based on 1 000 000 samples each). The dotted line shows the theoretical density of the uniform distribution on [0, 1].

1500 independent sets of samples for each use-case, while the across-dimension cases all use 100 sets of 30-D samples.

For each of the generated sets of samples, we apply the corresponding test battery from Section III with $\alpha \in \{0.05, 0.01\}$: 36 tests for per-dimension case and 32 for the across-dimension case. Using this setup, we thus collect $194 \cdot 1\,500 \cdot 4 \cdot 36 = 4.19 \times 10^7$ test statistics/p-values for the per-dimension tests, and $249 \cdot 100 \cdot 4 \cdot 32 = 3.19 \times 10^6$ for the across-dimension tests.

We show an example of the set of statistical tests applied to an instance of the `Cut Normal` scenario in Fig. 3. This figure shows the rejections for each dimension individually, as well as the corresponding sample on which this decision is based. This visualization is available as part of the toolbox as described in Section II-C, and provides a visual way to inspect the SB present in the scenario.

For the analysis on the per-dimension test, we do not directly apply multiple correction methods. However, in Section V-D, we will analyze the effect of different correction methods on the overall false positive rate and select which method to use in practice.

V. Analysis of Statistical Tests Applied to Scenarios

This section analyses the statistical tests presented in Section III in terms of their suitability to be included in the BIAS toolbox for the purpose of detecting SB.

A. Robustness of Tests to Scenario Parameters

To analyze the effect of different parameterizations of the selected scenarios on the difficulty of detecting bias, we can consider the overall number of rejections for a single test across all parameter settings. An example for the `Inverse Cauchy` scenario is shown in Fig. 4. In this figure, we see clearly that extreme parameter settings (highly shifted mean and low variance) are always detectable by the *AD* test, while the *tAD* test performs much better when the distribution is centerd.

While this granular view of results can give important insights, it is impractical to use this low-level view to investigate the different impacts of our experiment settings on the final rejections. For the sake of completeness, the figures for all scenarios and all experimental settings have been uploaded at [56].

B. Sample Size

To study the impact of the available sample size on the overall performance of different statistical tests, we can aggregate the number of rejections over all parameterizations of each scenario. This allows us to show the fraction of cases of a scenario which are rejected by each test given a certain sample size. Fig. 5 shows this for the `Cut Normal` scenario. From this figure, we can see that the effect of sample size is not the same across all tests. As an example, the *AD* test has a relatively high number of rejections at 30 samples, but does not reach the same precision as other tests when increasing sample size to 600. This indicates that analysis of the performance of the tests should take the number of available samples into account, as this will influence which tests are more distinguishing.

From Fig. 5, we can also see that the *Moran* test significantly outperforms any others on this scenario, but even this test does not reject all cases when the sample size is small. This reinforces the notion that if possible, increasing the sample size is beneficial to the ability to detect less clear cases of SB. However, we also note that for most scenarios, a sample size of 50 seems to be sufficient to detect the presence of SB. While increasing the sample size would increase the ability to detect less obvious cases of SB, $N = 50$ should be able to correctly identify the most blatant ones.

C. Overall Analysis

With the rejection data, we can investigate the interplay between statistical tests and the scenarios, in order to find what set of tests is more suitable to each kind of SB. For this analysis, we make use of the concept of Shapley values [57] to assess the contribution of each test to a portfolio of tests for finding bias in each type of scenario. In particular, we define the marginal contribution of a test t to a portfolio of tests $T' \subset T$ on scenario S as follows:

$$c\big(t, T', S, n, \alpha\big) = \sum_{s \in S} \sum_{i=0}^{n} \max_{t' \in (T' \bigcup \{t\})} \mathbb{1}_{t'(s_i) < \alpha} - \sum_{s \in S} \sum_{i=0}^{n} \max_{t' \in T'} \mathbb{1}_{t'(s_i) < \alpha} \quad (2)$$

where n is the number of realizations s_i of scenario s. The indicator function $\mathbb{1}$ corresponds to the test t rejecting the null hypothesis with significance α on the data from realization s_i.

Based on this definition of marginal contribution, we can compute approximate Shapley values by sampling random permutations calculating the marginal contribution for each test at each position within this permutation [58], [59]. This can

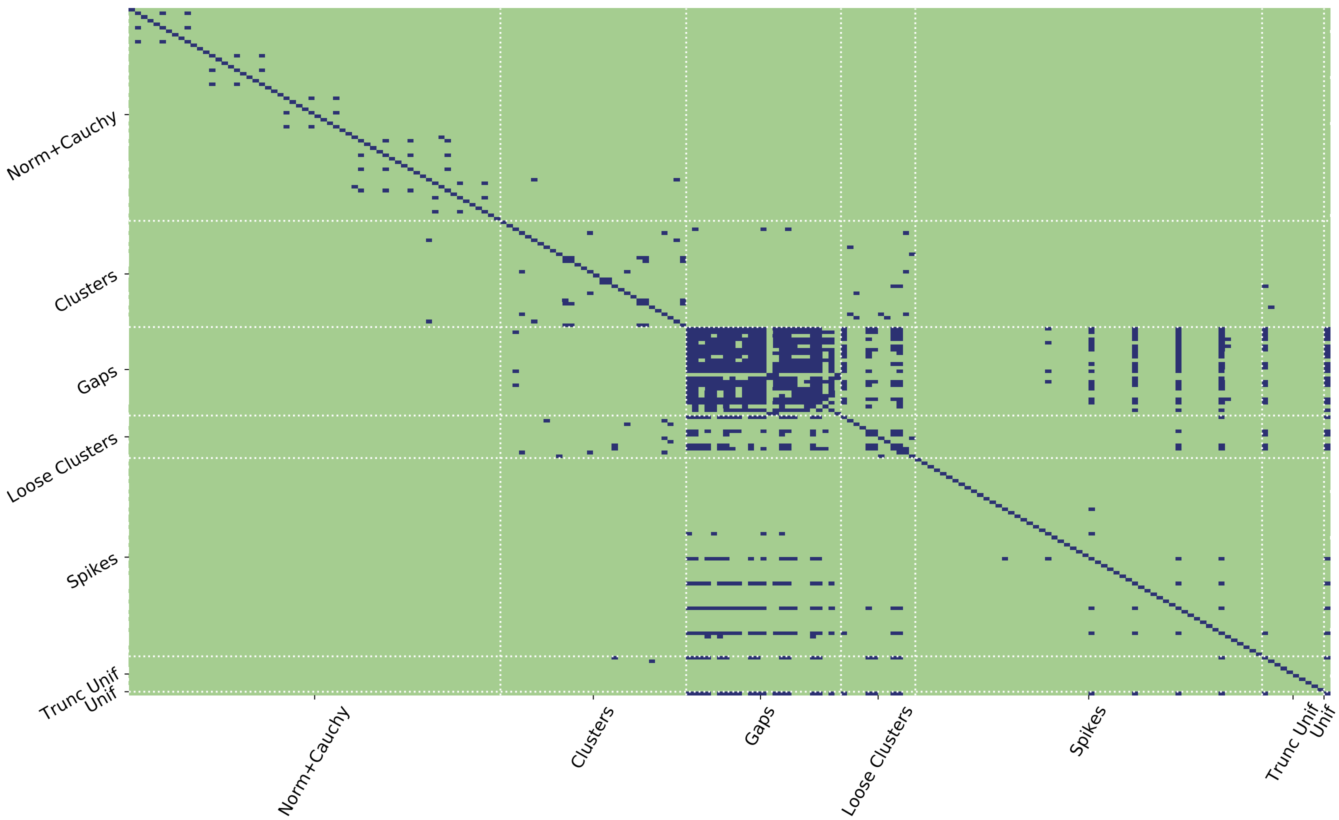

Fig. 2. Confusion matrix based on decisions from the KS-test between all scenarios. Blue indicates combinations which are not distinguishable based on the 2-sample KS-test for $\alpha = 0.01$. Used sample size is 100, with 1000 sets of samples generated for each scenario, aggregated into a 1-D sample. The white lines divide the types of scenario. `Norm+Cauchy` group contains all `Cut` and `Inverse` scenarios based on normal and Cauchy distributions.

be formulated as follows:

$$S(t, S, n, \alpha) = \sum_{r} \sum_{i=0}^{m} c\big(t, T', S, n, \alpha\big) : T' \subset T, |T'| = i \quad (3)$$

where r is the number of repetitions used and m is the maximum size of these permutations, which is introduced to ease with computations and because the impact of larger permutations on the total sum is relatively minor—in this article, we set $m = 10$.

Since the used definition of marginal contribution is commutative, we can sum the individual Shapley values for each scenario to get the overall Shapley values across scenarios. These values are then shown in Fig. 6, where we can see that most tests have a very comparable contribution to the overall rejections, with relatively few outliers in both positive (e.g., *Moran*) and negative (e.g., *MPD-min*) sense. Because of this, we consider all test to have their uses in the portfolio, and refrain from removing any tests from consideration.

Additionally, we can also consider the per-scenario Shapley values to find a relation between the statistical tests and the scenarios which it can most effectively distinguish. This is visualized in Fig. 7. There are some clear patterns visible in this figure, e.g., for the scenarios which mimic poor discretization, where the *Moran* and *Cressie1* tests have a very high contribution, while their impact on the other scenarios seems to be relatively minor. This highlights the benefit of having a large portfolio of different statistical test to identify SB.

Overall, we have seen that no single test is clearly preferable over all others. Moreover, an analysis of the Kendall–Tau [60] correlations between the rejections of tests across all scenarios show that very few tests are highly correlated. Fig. 8 shows the correlation heatmaps for two representative settings of sample size and α. We can observe relatively higher correlations among some of the tests listed from *NS* to *Greenwood*, and among some of the tests listed from *QM* to *Pardo*, in all settings. However, these higher correlations involve very few of the tests, and overall these correlations tend to get slightly weaker as the sample sizes get smaller. Moreover, the correlations among the other tests are very low for the largest sample size of 600. Therefore, it is likely that different tests are best suited to recognize different types of deviations from uniformity. For this reason, we will include *all considered tests* in the BIAS toolbox.

D. Multiple Comparison Correction Methods

For the per-dimension tests, we should take into account the fact that multiple tests are being done, and thus the p-values should be changed using a correction procedure. For this purpose, we consider three methods: 1) BH [61]; 2) Benjamini–Yekutieli (BY) [62]; and 3) Holm–Bonferroni (Holm) [63].

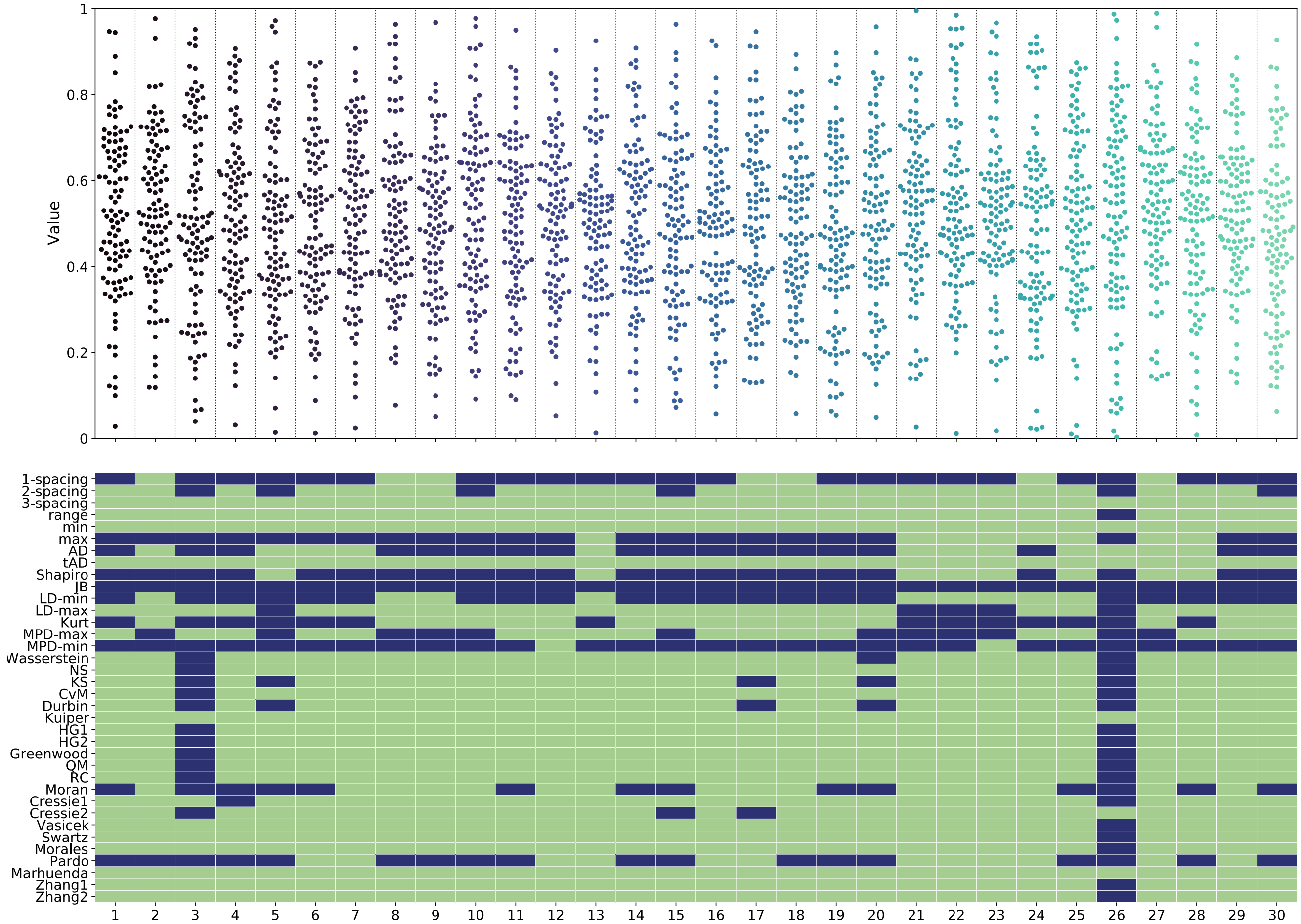

Fig. 3. Example of an instantiation of the `Normal Cut` scenario with $\mu = 0.5$ and $\sigma = 0.2$, with 100 samples in each of 30 dimensions. The top figure shows the assumed distribution of the final positions potentially returned by an optimization algorithm in each dimension. Jitter is applied here to reveal vertically overlapping points. The color scheme is used to highlight different dimensions. The binary heatmap in the bottom figure shows in green which tests reject the null-hypothesis of uniformity per dimension with $\alpha = 0.01$ (no multiple comparison correction applied).

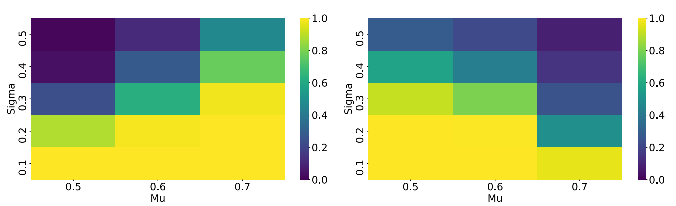

Fig. 4. Fraction of rejected cases for the *AD* (left) and *tAD* (right) tests on the `Cut Cauchy` scenario, based on its parameterizations. The used sample size is 100, with $\alpha = 0.01$ and no multiple comparison correction applied.

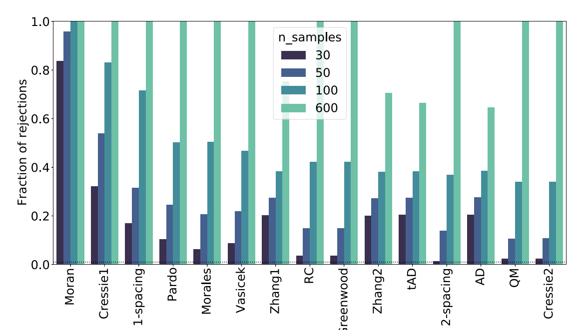

Fig. 5. Fraction of rejections for each test on the `Shifted Spikes` scenario, with $\alpha = 0.01$ and no multiple comparison correction method applied. Data is aggregated over all parameterizations of the scenario, as described in Table I. This figure shows 15 tests with the most rejections (when aggregated over the different sample sizes). Note that the negative space over each bar $(1 - x)$ is equivalent to the false negative rate of the test.

For each of these methods, we consider their impact on 30-D samples of the uniform distribution to judge the *false positive rate*. In Fig. 9 on the left, we see the false positive relative the used α values (defined as "at least 1 rejection in the 30-D sample"). This figure clearly shows the need for a multiple comparison correction method, given that the false positive rate is way over α (shown as dotted lines) when using no correction methods. We also note minor differences between the correction methods considered, with BY leading to the lowest false positive rate.

Similarly, we can also consider the overall *false negative rate* as an aggregation over all scenarios, as is done in the right part of Fig. 9. This figure shows that the impact of the choice of the multiple comparison correction method on the false negatives is relatively minor. Because of this, we set the BY-method as the default, since it is better able to produce a false positive rate lower than α.

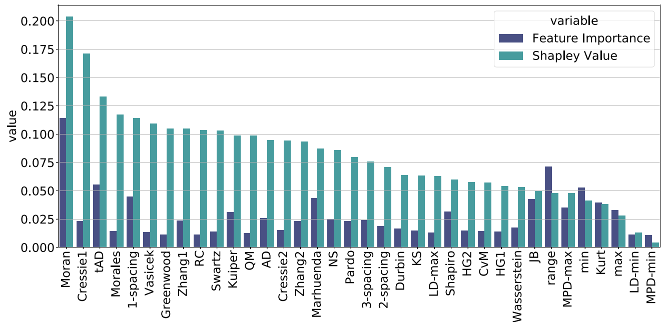

Fig. 6. Approximated Shapley values for all 36 per-dimension tests, based on marginal contribution to the total number of rejections across all scenarios (sample size 100, $\alpha = 0.01$). Additionally, the feature importance of these tests in the RF model discussed in Section VI is also shown.

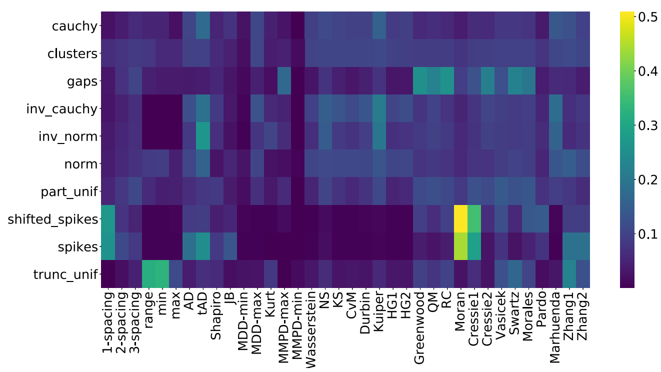

Fig. 7. Approximated Shapley values of all per-dimension tests, based on marginal contribution to the total number of rejections across all parameterizations of the used scenarios, with a sample size of 100, $\alpha = 0.01$ and no multiple comparison correction applied. These Shapley values are approximated based on 3600 random permutations.

E. Results on Correlated Samples

Within the set of scenarios, there are two sets which are enabled only for the across-dimension tests. These scenarios are the `Consistent` versions of existing scenarios `Gaps` and `Clusters`. In Fig. 10, we show the difference in test rejections between these two versions of the scenario. It is clear from the figure that aggregating samples from clusters with different intiantiations on each dimension removes the ability of many tests to detect SB, while the across-dimension tests such as *MDDLUD* do not have this issue. Additionally, since the spacing tests aggregate spacings per dimension instead of samples, their effectiveness is not reduced when the clusters are inconsistent. For the sake of brevity, the other results on the across-dimension tests are not discussed here, but the relevant figures and data is made available at [56].

VI. Estimation of the SB Type

Since we use the results of many statistical tests to find bias in artificially generated samples and different tests may be better at capturing different deviations from uniformity, we can use these tests to not only check if SB is present, but also to identify what the most likely form of bias is. This provides an answer to RQ2. To achieve this, we build an RF model, which takes as input the *test-rejections from all per-dimension tests*. This is done to allow scaling to arbitrary dimensions while having one model for all sample sizes. Specifically, if we use statistical test values directly, we would need one model per sample size, and a way to aggregate the resulting predictions. Instead, an RF based on rejections only needs to deal with the aggregation problem.

The data used to train and evaluate the RF model consists of the full set of scenario results (per-dimension version) on all tests, with the output being the scenario type it comes from. However, if for a specific sample no test rejects the null hypothesis, these samples are discarded, since we have no evidence of SB. This two-stage approach leaves us with 1 158 000 biased samples, on which we train the RF model with 100 trees and balanced class weights. For the sake of clarity and reproducibility, we make the training data-set available in the online repository [55]. A confusion matrix created from an 80–20 test split is shown in Fig. 11 (F1-score of 0.56). Similarly to Fig. 2, we see that the distinction between the `Cut Uniform` and the other scenarios can be challenging to accurately detect. However, this does not have to be an issue for practical detection of SB, since the scenarios misidentified as `Cut Uniform` might show similar types of bias, even tough their initial creation mechanism is different.

To provide a more practical estimation of SB in our toolbox, we create an additional model to predict the type of bias, as shown in the final column of Table I. These five categories are more distinct from each other, removing overlap between some similar classes, i.e., between `Spikes` and `Noisy Spikes`. Overall, this model gives us an improved F1-score of 0.79 on a similar 80–20 split.

To use these models in the BIAS toolbox to predict bias of the multidimensional test, we need to perform some aggregation across dimensions to transform it into a binary vector. We do this by checking the number of false positive tests in 30-D uniform samples. We run 10 000 simulations, where we record the maximum number of test rejections by each test. This gives us a total of 92 cases where a test gives two rejections, and two cases where a test gives three rejections. As such, we set the threshold for the aggregation of multidimensional data to $0.1 \cdot n$. If no test is rejected in this aggregation, we consider the samples to be nonbiased. This threshold value is then used to create the binary input vector for the RF model.

To verify that this works for other dimensionalities as well, and to gauge the overall performance of the toolbox, we simulate the false positive and false negative rates. This is achieved by sampling (with replacement) from the set of test statistics on each of our used scenarios and applying this aggregation rule. For false positives, this is done 100 000 times on the (true) uniform data, while for false negatives it is done 10 000 times on every nonuniform scenario. The results, shown in Fig. 12, indicate that while the $0.1 \cdot n$ threshold is rather conservative on higher dimensionalities, the FPR is well below the selected $\alpha = 0.01$, while the FNR is not needlessly increased.

VII. Benchmarking SB of Real Algorithmic Data

This section benchmarks a large set of heuristic optimization algorithms by applying the BIAS toolbox, answering RQ3.

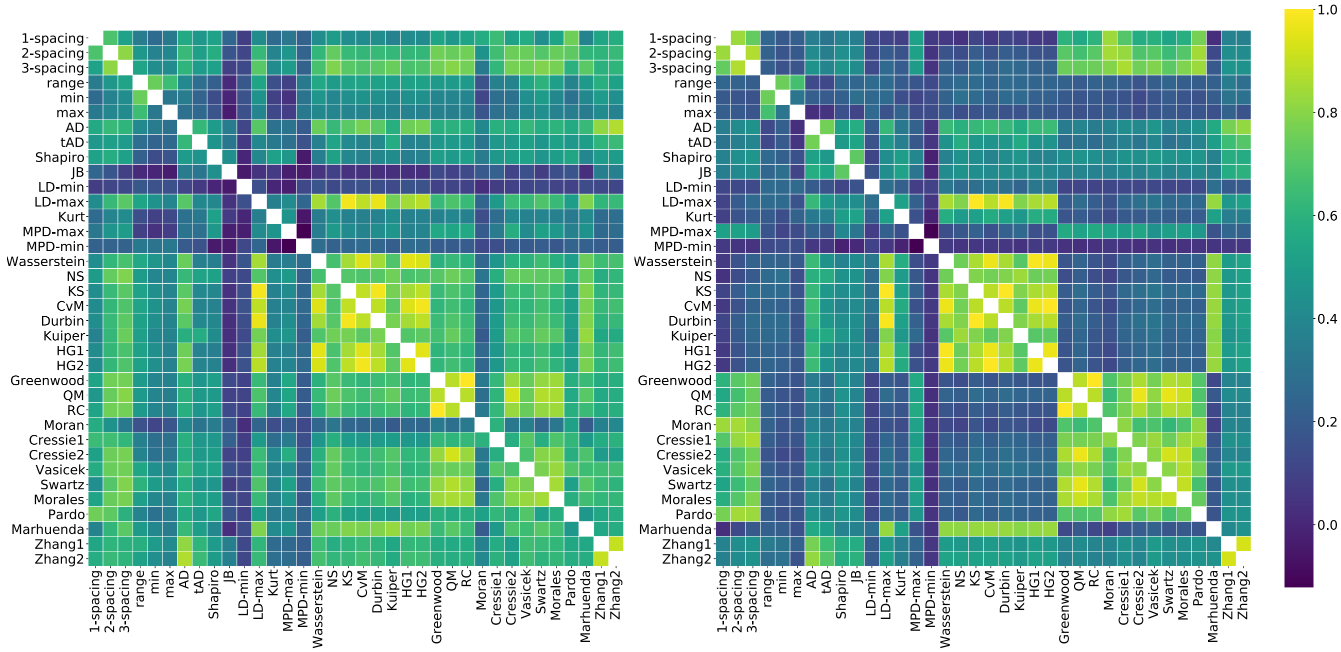

Fig. 8. Cluster plot showing the Kendall–Tau correlations [60] between test rejections on all scenarios, with a sample size 50 (left) and 600 (right), $\alpha = 0.01$.

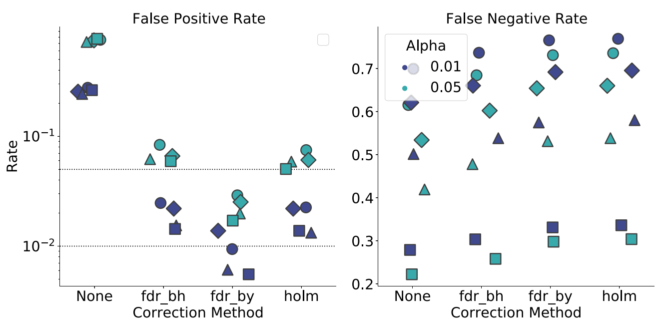

Fig. 9. Evaluation of multiple comparison correction methods: fraction of false positives in 30-D samples with different correction methods (on the left) and aggregated fraction of false negatives across all scenarios of the *six most distinguishing tests* based on Shapley values (on the right). On the left, values *above* the relevant α-thresholds (0.01, 0.05) indicate too large false positive rates. On both figures, markers identify the used sample size: ◯, ◇, △, and □ are 30, 50, 100, and 600, respectively. Here, we conclude that BY correction method should be chosen.

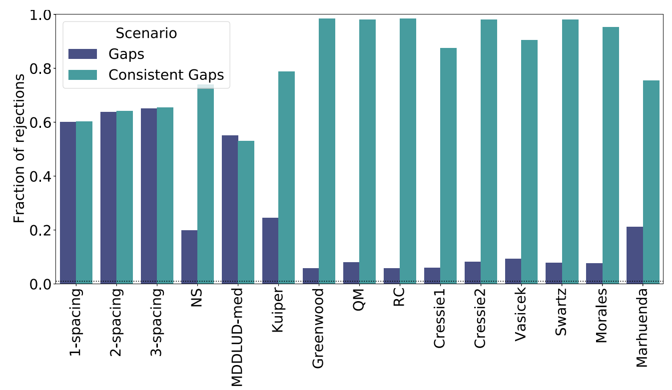

Fig. 10. Overall fraction of rejections across all sample sizes and scenario parametrization of the two versions of the `Gaps` scenario, using the across-dimension version of the statistical tests with $\alpha = 0.01$ (shown with faint black vertical line). Selection of tests shown is done as the top 15 tests with the most total rejections when combining the two bars.

A. Data Collection Setup

We use data from a heterogeneous pool of heuristics executed over f_0 at dimensionalilty $n = 30$ for a maximum of $10\,000 \cdot n$ fitness functional calls. In total, we consider 432 optimization heuristics, which fall into the following categories (all except the latter use $N = 100$, while the latter uses $N = 50$ runs each).

1) Variants of Differential Evolution (195 configurations).
2) Compact optimization algorithms (81 configurations).
3) Single-solution algorithms (60 configurations).
4) Variants of Genetic Algorithms (96 configurations).

For the sake of clarity and reproducibility, the exact composition and set-up of these algorithmic configurations is described fully in a dedicated document available from [64].

B. Results

For each of the considered algorithm configurations, we collect their final positions and feed these into the BIAS toolbox. In Fig. 13 (left side), we show the outcome from the RF predicting the type of SB present in the different GA configurations (only the biased ones are shown). This shows that there are quite some differences in the detected bias, even within this limited algorithm design space. It is also interesting to note that the population size seems to have a relatively small impact on the type of predicted bias, which seems to be mostly impacted by the operator configuration.

For the single-solution algorithms, we see in the right part of Fig. 13 that the strategy of dealing with infeasible solution seems to drastically change the type of detected bias. For example, the Powell algorithm is classified as "discretization"

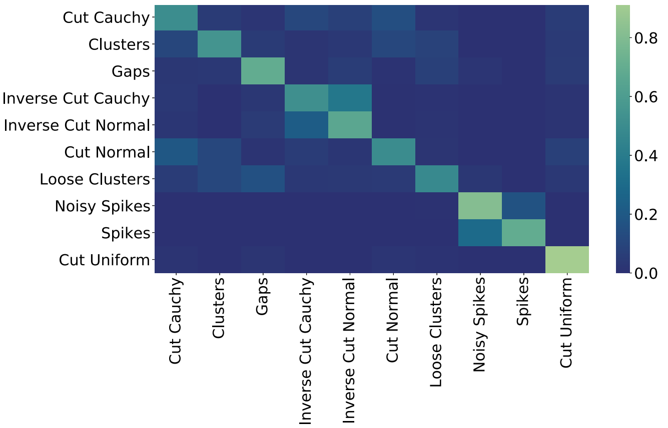

Fig. 11. Confusion matrix for the RF model trained on test rejections, aggregated over all sample sizes. The true scenario is shown on the y-axis, while the predicted scenario is on the x-axis.

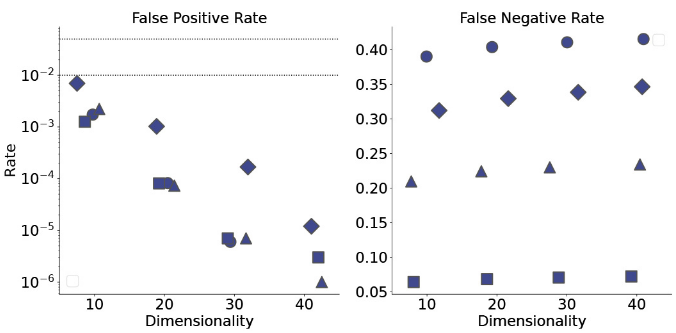

Fig. 12. Evaluation of toolbox in different dimensions at $\alpha = 0.01$: fraction of false positives (on the left) and aggregated fraction of false negatives across all scenarios (on the right). On both figures, markers identify the used sample size: ◯, ◇, △, and ☐ are 30, 50, 100, and 600, respectively.

bias when using the mirror strategy, while the classification changes completely with a COTN strategy. Such differences can give us useful insight into the effect of these SDIS methods on the optimization behavior of these algorithms.

VIII. Conclusion

Behavior-based benchmarking is a great way to better understand heuristic algorithms. We have proposed a new behavior-based benchmark, BIAS, in order to detect different scenarios of SB in optimization algorithms. The BIAS toolbox consists of 36 per-dimension tests and 32 across-dimension tests to detect SB (RQ1). These statistical tests are selected based on an elaborate literature research; some of these tests are specifically designed for this toolbox by the authors. The tests are compared and analyzed using different sample sizes and hyper-parameters to detect 11 different scenarios of SB. In addition to the tests, a generator of the SB scenarios is provided as well as two machine learning models to predict the scenario of SB using the 11 scenarios mentioned above (RQ2).

From the results of the analysis, it is clear that the toolbox performs very well in detecting SB in the generated distributions. The machine learning models show some confusion between similar SB scenarios, such as `Gaps`—`Loose clusters`, and `Spikes`—`Noisy Spikes`, but overall perform well enough to provide suggestions of the type of SB.

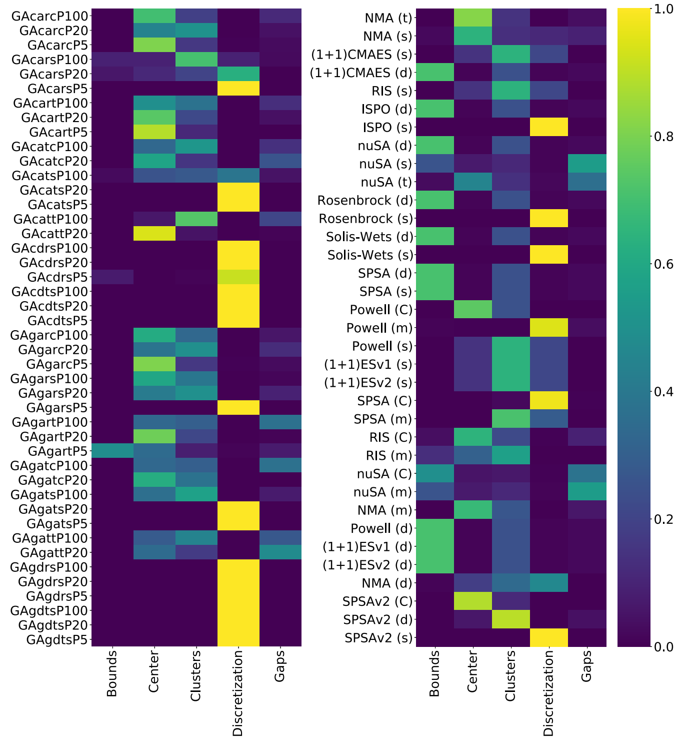

Fig. 13. Predicted SB class probabilities of the biased GA configurations (left, sorted alphabetically) and the biased single-solution algorithms (right), using the RF model. Names for the GA are structured as mutation–crossover–selection–SDIS–population size. For the single-solution algorithms, the character in brackets refers to the used SDIS.

In addition we provided the results of BIAS on a large set of optimization algorithms, including variants of Genetic Algorithm and Differential Evolution, compacts and single-solution algorithms (RQ3). The results show different kinds of SB in existing heuristic optimization algorithms. The BIAS toolbox, including the f_0 function, the statistical tests, the SB scenario generator, and the RF models are provided open-source [4]. We recommend to use the BY correction method when using BIAS, since this correction method is fast to compute and more conservative than BH. We furthermore recommend a minimum sample size of 50 since in most cases it is sufficient to detect SB for all scenarios, as can be seen from the Genetic Algorithm results. However, some mild SB scenarios could still go undetected. If time and computation power permits, a sample size of 100 would be best.

For future research, it would be interesting to explore different machine learning models to see if the predictions of the type of SB can be improved. Additionally, decreasing the number of statistical tests could improve the execution time required to run the benchmark, and may not necessarily decrease accuracy of the BIAS toolbox.

References

[1] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[2] N. Hansen, A. Auger, R. Ros, O. Mersmann, T. Tušar, and D. Brockhoff, "COCO: A platform for comparing continuous optimizers in a black-box setting," *Optim. Methods Softw.*, vol. 36, no. 1, pp. 114–144, 2021.

[3] F. Croce *et al.*, "RobustBench: A standardized adversarial robustness benchmark," 2020, *arXiv:2010.09670*.

[4] D. Vermetten, B. van Stein, F. Carafini, L. L. Minku, and A. V. Kononova. "BIAS Tooblox." [Online]. Available: https://github.com/Dvermetten/BIAS (Accessed: Jun. 2022).

[5] K. Smith-Miles, D. Baatar, B. Wreford, and R. Lewis, "Towards objective measures of algorithm performance across instance space," *Comput. Oper. Res.*, vol. 45, pp. 12–24, May 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054813003389

[6] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, and G. Rudolph, "Exploratory landscape analysis," in *Proc. 13th Annu. Conf. Genet. Evol. Comput.*, 2011, pp. 829–836.

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

[8] B. van Stein, F. Caraffini, and A. V. Kononova, "Emergence of structural bias in differential evolution," in *Proc. Genet. Evol. Comput. Conf. Companion*, Jul. 2021, pp. 1234–1242.

[9] A. Inselberg, "The plane with parallel coordinates," *Vis. Comput.*, vol. 1, no. 2, pp. 69–91, 1985.

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

[11] F. Caraffini and A. V. Kononova. "Structural Bias in Optimisation Algorithms: Extended Results." 2021. [Online]. Available: http://dx.doi.org/10.17632/zdh2phb3b4.4

[12] B. van Stein, F. Caraffini, and A. V. Kononova. "Emergence of structural bias in differential evolution—Source code & extended graphical results." 2021. [Online]. Available: http://dx.doi.org/10.17632/pb2bdp2gkp.1

[13] A. N. Kolmogorov, "Sulla determinazione empirica di una legge di distribuzione," *Giornale dell'Istituto Italiano degli Attuari*, vol. 4, no. 1, pp. 83–91, 1933.

[14] A. V. Kononova, F. Caraffini, H. Wang, and T. Bäck, "Can single solution optimisation methods be structurally biased?" in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Glasgow, U.K., 2020, pp. 1–9.

[15] S. Csorgo and J. J. Faraway, "The exact and asymptotic distributions of Cramer-von Mises statistics," *J. Roy. Stat. Soc. Ser. B, Methodol.*, vol. 58, no. 1, pp. 221–234, 1996.

[16] T. W. Anderson and D. A. Darling, "Asymptotic theory of certain 'goodness of fit' criteria based on stochastic processes," *Ann. Math. Stat.*, vol. 23, pp. 193–212, Jun. 1952.

[17] Y. Benjamini, "Discovering the false discovery rate," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 72, no. 4, pp. 405–416, 2010.

[18] A. V. Kononova, F. Caraffini, H. Wang, and T. Bäck, "Can compact optimisation algorithms be structurally biased?" in *Parallel Problem Solving from Nature (PPSN) XVI*, T. Bäck *et al.*, Eds. Leiden, The Netherlands: Springer Int., 2020, pp. 229–242.

[19] S. S. Kar and A. Ramalingam, "Is 30 the magic number? issues in sample size estimation," *Nat. J. Community Med.*, vol. 4, no. 1, pp. 175–179, 2013.

[20] D. Vermetten, A. V. Kononova, F. Caraffini, H. Wang, and T. Bäck, "Is there anisotropy in structural bias?" in *Proc. Genet. Evol. Comput. Conf. Companion*, 2021, pp. 1243–1250. [Online]. Available: https://doi.org/10.1145/3449726.3463218

[21] M. A. Marhuenda, Y. Marhuenda, and D. Morales, "Uniformity tests under quantile categorization," *Kybernetes*, vol. 34, no. 6, pp. 888–901, 2005.

[22] P. L. de Micheaux and V. A. Tran, "PoweR: A reproducible research tool to ease Monte Carlo power simulation studies for goodness-of-fit tests in R," *J. Stat. Softw. Articles*, vol. 69, no. 3, pp. 1–44, 2016.

[23] R. Pyke, "Spacings," *J. Roy. Stat. Soc. Ser. B, Methodol.*, vol. 27, no. 3, pp. 395–436, 1965.

[24] J. Faraway, G. Marsaglia, J. Marsaglia, and A. Baddeley. "goftest: Classical Goodness-of-Fit Tests for Univariate Distributions." 2021. [Online]. Available: https://CRAN.R-project.org/package=goftest

[25] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, nos. 3–4, pp. 591–611, 1965.

[26] R Core Team (R Found. Stat. Comput., Vienna, Austria). *R: A Language and Environment for Statistical Computing*. (2013). [Online]. Available: http://www.R-project.org/

[27] C. M. Jarque and A. K. Bera, "A test for normality of observations and regression residuals," *Int. Stat. Rev.*, vol. 55, no. 2, pp. 163–172, 1987.

[28] G. Sucarrat. "AutoSEARCH: General-to-Specific (GETS) Modelling." [Online]. Available: https://cran.r-project.org/package=AutoSEARCH (Accessed: Jun. 2022).

[29] K. Pearson, "Das Fehlergesetz und seine Verallgemeinerungen Durch fechner und Pearson. A rejoinder," *Biometrika*, vol. 4, nos. 1–2, pp. 169–212, Jun. 1905. [Online]. Available: https://doi.org/10.1093/biomet/4.1-2.169

[30] L. V. Kantorovich, "The mathematical method of production planning and organization," *Manag. Sci.*, vol. 6, no. 4, pp. 363–422, 1939.

[31] L. Kantorovitch, "On the translocation of masses," *Manag. Sci.*, vol. 5, no. 1, pp. 1–4, 1958. [Online]. Available: http://www.jstor.org/stable/2626967

[32] J. Neyman, "Smooth test for goodness of fit," *Scand. Actuarial J.*, vol. 1937, nos. 3–4, pp. 149–199, 1937.

[33] P. Biecek and T. Ledwina. "ddst: Data Driven Smooth Tests." 2016. [Online]. Available: https://cran.r-project.org/package=ddst

[34] H. Cramér, "On the composition of elementary errors: First paper: Mathematical deductions," *Scand. Actuarial J.*, vol. 1928, no. 1, pp. 13–74, 1928.

[35] J. Durbin, "Tests for serial correlation in regression analysis based on the periodogram of least-squares residuals," *Biometrika*, vol. 56, no. 1, pp. 1–15, Mar. 1969.

[36] J. Durbin and R. Brown, "Tests of serial independence based on the cumulated periodogram," *Bull. Int. Stat. Inst.*, vol. 42, pp. 1039–1048, 1967.

[37] H. D. Brunk, "On the range of the difference between hypothetical distribution function and Pyke's modified empirical distribution function," *Ann. Math. Stat.*, vol. 33, no. 2, pp. 525–532, 1962.

[38] Y. A. S. Hegazy and J. R. Green, "Some new goodness-of-fit tests using order statistics," *J. Roy. Stat. Soc. Ser. C, Appl. Stat.*, vol. 24, no. 3, pp. 299–308, 1975.

[39] M. Greenwood, "The statistical study of infectious diseases," *J. Roy. Stat. Soc.*, vol. 109, no. 2, pp. 85–110, 1946.

[40] C. P. Quesenberry and F. L. Miller, Jr., "Power studies of some tests for uniformity," *J. Stat. Comput. Simul.*, vol. 5, no. 3, pp. 169–191, 1977.

[41] N. Cressie and T. R. C. Read, "Multinomial goodness-of-fit tests," *J. Roy. Stat. Soc. Ser. B, Methodol.*, vol. 46, no. 3, pp. 440–464, 1984.

[42] P. A. P. Moran, "The random division of an interval–part II," *J. Roy. Stat. Soc. Ser. B, Methodol.*, vol. 13, no. 1, pp. 147–150, 1951.

[43] N. Cressie, "Power results for tests based on high-order gaps," *Biometrika*, vol. 65, no. 1, pp. 214–218, 1978.

[44] N. Cressie, "An optimal statistic based on higher order gaps," *Biometrika*, vol. 66, no. 3, pp. 619–627, Dec. 1979.

[45] O. Vasicek, "A test for normality based on sample entropy," *J. Roy. Stat. Soc. Ser. B, Methodol.*, vol. 38, no. 1, pp. 54–59, 1976.

[46] T. Swartz, "Goodness-of-fit tests using Kullback-Leibler information," *Commun. Stat. Theory Methods*, vol. 21, no. 3, pp. 711–729, 1992.

[47] D. Morales, L. Pardo, M. C. Pardo, and I. Vajda, "Limit laws for disparities of spacings," *J. Nonparametric Stat.*, vol. 15, no. 3, pp. 325–342, 2003.

[48] B. G. Lindsay, "Efficiency versus robustness: The case for minimum Hellinger distance and related methods," *Ann. Stat.*, vol. 22, pp. 1081–1114, Jun. 1994.

[49] M. L. Menendez, D. Morales, L. Pardo, and I. Vajda, "Two approaches to grouping of data and related disparity statistics," *Commun. Stat. Theory Methods*, vol. 27, no. 3, pp. 609–633, 1998.

[50] M. C. Pardo, "A test for uniformity based on informational energy," *Stat. Papers*, vol. 44, pp. 521–534, Oct. 2003.

[51] J. Zhang, "Powerful goodness-of-fit tests based on the likelihood ratio," *J. Roy. Stat. Soc.*, vol. 64, no. 2, pp. 281–294, 2002.

[52] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Oct. 1948.

[53] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825–2830, 2011.

[54] E. W. Noreen, *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. New York, NY, USA: Wiley, 1989.

[55] D. Vermetten, A. V. Kononova, F. Caraffini, B. van Stein, and L. Minku. "BIAS: A Toolbox for BenchmarkingStructural Bias in the Continuous Domain—Tests Statistics and Rejections." Sep. 2021. [Online]. Available: https://figshare.com/articles/dataset/BIAS_A_Toolbox_for_BenchmarkingStructural_Bias_in_the_Continuous_Domain_–_Tests_Statistics_and_Rejections/16546041

[56] D. Vermetten, F. Caraffini, A. V. Kononova, B. van Stein, and L. Minku. "BIAS: A Toolbox for BenchmarkingStructural Bias in the Continuous Domain—Figures." Sep. 2021. [Online]. Available: https://figshare.com/articles/figure/BIAS_A_Toolbox_for_BenchmarkingStructural_Bias_in_the_Continuous_Domain_–_Figures/16546128

[57] L. Shapley, "A value for n-Person games," in *Contributions to the Theory of Games II*, H. Kuhn and A. Tucker, Eds. Princeton, NJ, USA: Princeton Univ. Press, 1953, pp. 307–317.

[58] T. van Campen, H. Hamers, B. Husslage, and R. Lindelauf, "A new approximation method for the Shapley value applied to the WTC 9/11 terrorist attack," *Soc. Netw. Anal. Min.*, vol. 8, no. 1, pp. 1–12, 2018.

[59] J. Castro, D. Gómez, and J. Tejada, "Polynomial calculation of the Shapley value based on sampling," *Comput. Oper. Res.*, vol. 36, no. 5, pp. 1726–1730, 2009.

[60] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, nos. 1–2, pp. 81–93, 1938.

[61] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *J. Roy. Stat. Soc. Ser. B, Methodol.*, vol. 57, no. 1, pp. 289–300, 1995.

[62] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *Ann. Stat.*, vol. 29, no. 4, pp. 1165–1188, 2001.

[63] S. Holm, "A simple sequentially rejective multiple test procedure," *Scand. J. Stat.*, vol. 6, no. 2, pp. 65–70, 1979.

[64] D. Vermetten, A. V. Kononova, F. Caraffini, B. van Stein, and L. Minku. "BIAS: A Toolbox for BenchmarkingStructural Bias in the Continuous Domain—Code." Sep. 2021. [Online]. Available: https://figshare.com/articles/software/BIAS_A_Toolbox_for_BenchmarkingStructural_Bias_in_the_Continuous_Domain_–_Code/16546245

Diederick Vermetten is currently pursuing the Ph.D. degree with Leiden University (NL), Leiden, The Netherlands, where he is part of the NaCo group.

He is supervised by H. Wang and T. Bäck from Leiden University and C. Doerr from Sorbonne University, CNRS. His research interests include benchmarking of optimization heuristics, dynamic algorithm selection, and configuration as well as hyperparameter optimization. He is a part of the core development team of the IOHprofiler project for benchmarking iterative optimization heuristics.

Bas van Stein (Member, IEEE) received the Ph.D. degree in computer science from the Leiden Institute of Advanced Computer Science (LIACS), Leiden University, Leiden, The Netherlands, in 2018.

From 2018 to 2021, he was a Post-doctoral Researcher with LIACS, Leiden University and he is currently an Assistant Professor with LIACS. His research interests lie in surrogate-assisted optimization, surrogate assisted neural architecture search, and explainable AI techniques for industrial applications.

Fabio Caraffini (Member, IEEE) received the B.Sc. degree in electronics engineering and the M.Sc. degree in telecommunications engineering from the University of Perugia, Perugia, Italy, in 2008 and 2011, respectively, the first Ph.D. degree in computer science from the De Montfort University, Leicester, U.K., 2014, and the second Ph.D. degree in mathematical information technology from the University of Jyväskylä, Jyväskylä, Finland, in 2016.

He is currently an Associate Professor of Computer Science with De Montfort University, where he is a member of the Institute of Artificial Intelligence. His research interests include theoretical and applied computational intelligence with a strong emphasis on metaheuristics for optimization.

Leandro L. Minku (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2010.

He is an Associate Professor with the University of Birmingham. Prior to that, he was a Lecturer (an Assistant Professor) with the University of Leicester, Leicester, U.K. His main research interests are machine learning for nonstationary environments/data stream learning, class imbalance learning, ensembles of learning machines, dynamic optimization, and computational intelligence for software engineering.

Dr. Minku is an Associate Editor-in-Chief for *Neurocomputing*, a Senior Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and an Associate Editor for *Empirical Software Engineering Journal* and *Journal of Systems and Software*.

Anna V. Kononova (Member, IEEE) received the M.Sc. degree in applied mathematics from Yaroslavl State University, Yaroslavl, Russia, in 2004, and the Ph.D. degree in computer science from the University of Leeds, Leeds, U.K., in 2010.

She is currently an Assistant Professor with the Leiden Institute of Advanced Computer Science, University Leiden, Leiden, The Netherlands. After five years of post-doctoral experience with Technical University Eindhoven, Eindhoven, The Netherlands, and Heriot-Watt University, Edinburgh, U.K. She spent five years working as an Engineer and a Mathematician in industry. Her current research interests include analysis of behavior of optimization algorithms and machine learning.