Assessing Ranking and Effectiveness of Evolutionary Algorithm Hyperparameters by Global Sensitivity Analysis Methodologies

Varun Ojha*†[1], Jon Timmis†[2,3], and Giuseppe Nicosia†[4,5]

[1] *University of Reading, Reading, United Kingdom*
[2] *University of York, York, United Kingdom*
[3] *The University of Sunderland, Sunderland, United Kingdom*
[4] *University of Catania, Catania, Italy*
[5] *University of Cambridge, Cambridge, United Kingdom*

Abstract

We present a comprehensive global sensitivity analysis of two widely used single-objective and two multi-objective global optimization evolutionary algorithms (EAs). This work investigates the *quality of influence* hyperparameters have on the performances of EAs. Our methodology involves four EAs: covariance matrix adaptation evolutionary strategy (CMAES), differential evolution (DE), non-dominated sorting genetic algorithm III (NSGA-III), and multi-objective evolutionary algorithm based on decomposition (MOEA/D). We applied two sensitivity analysis methods, *Morris* and *Sobol*, to systematically analyze tunable hyperparameters of EAs on 23 single-objective functions and 10 multi-objective functions. Also, we studied the *tendencies* of hyperparameters' *direct effect* and *interaction effect* with other hyperparameters. We present them as a comparative matrix where the diagonal from low direct effect and low interaction to high direct effect and high interaction shows the *order and ranking* of the hyperparameters. However, high interaction (or total effect) is a crucial measure of importance. Our investigation supports the research and innovation toward making different versions of DE as the type of DE algorithms emerged as one of the most influential hyperparameters for DE. It also supports the innovation in the type of strategy used for multi-objective decomposition in MOEA/D. Moreover, our results suggest the importance of correctly setting the initial step size for population generation in CMAES and crossover probability setting in NSGA-III.

Keywords: Global sensitivity analysis; evolutionary algorithms; elementary effects; variance-based sensitivity analysis; multi-objective optimization

*Corresponding Author: Varun Ojha, email: v.k.ojha@reading.ac.uk

†Authors have equal contributions to this article.

1 Introduction

Single-objective and multi-objective evolutionary algorithms (EAs) are global optimization algorithms. They operate on a population of candidate solutions and iteratively guide the population towards a final population. The population's fitness is subjected to objective(s) and the algorithm's hyperparameters such as population size and the number of iterations. Single objective algorithms solve one objective, while multi-objective algorithms solve two or more objectives simultaneously. The hyperparameter values play a significant role in the accuracy of the solutions obtained (De Jong, 2007). Understanding the hyperparameter's sensitivity to an algorithm's performance can inform its tuning. For example, the significance of optimal hyperparameter selection is described in (Moschitti, 2003) for text classification and in (Crossley et al., 2013) for bio-inspired algorithms.

Since hyperparameters tuning is crucial in achieving high-quality performance in solving optimization problems, methods such as manual tuning, grid search, and Bayesian search optimization are used. Bergstra and Bengio (2012) have shown the importance of *random search* instead of a *grid search* in sampling hyperparameter values of its tuning and performance analysis. In addition, *manual tuning* without proper knowledge of hyperparameters can lead to too many trial-and-errors, and grid search and Bayesian search optimization are computationally expensive approaches that are often infeasible for such population-based optimization algorithms.

Works of Bergstra and Bengio (2012) suggest that tuning some hyperparameters is more necessary than the others: an algorithm (and its output) is more sensitive to some of its hyperparameters than other hyperparameters. Hence, our objective in this research is to assess the ranking and effectiveness of four well-known EAs: covariance matrix adaptation evolutionary strategy (Hansen and Ostermeier, 1996), differential evolution (DE) (Storn and Price, 1997), non-dominated sorting genetic algorithm III (NSGA-III) (Deb and Jain, 2013), and multi-objective evolutionary algorithm based on decomposition (MOEA/D) (Zhang and Li, 2007).

We developed a framework for comprehensive sensitivity analysis of these EAs using global sensitivity analysis methodologies: elementary effects (Morris, 1991) and variance-based sensitivity analysis (Sobol and Kucherenko, 2005). Using these methodologies, we assess the effectiveness of EA hyperparameters. Such sensitivity analysis methods investigate a model's parameter (or an algorithm hyperparameters) influence on its output (Iooss and Saltelli, 2016). Such an analysis informs the hyperparameters effect on the model's output (Brooks et al., 2001). Thus, it minimizes the number of critical tunable hyperparameters to improve a model's performance (Conca et al., 2015, Hill et al., 2016).

In our framework, performances of single-objective EAs were assessed as per *best solution*, while performances of multi-objective EAs were assessed using three measures. The multi-objective measures were *generational distance* (Veldhuizen and Lamont, 1998, Veldhuizen, 1999), *inverse generational distance* (Deb and Jain, 2013), and *hyper-volume indicator index* (Zitzler and

Thiele, 1998). To evaluate EAs, we use state-of-the-art optimization problems belonging to diverse families: for single-objective optimize, we use a set of 23 problems (Yao et al., 1999), and for multi-objective optimization, we use a set of 10 problems (Deb and Jain, 2013).

Our framework assesses each algorithm on three sensitivity analysis methods: Morris Latin Hypercube sampling, Morris One-at-a-time Sampling, and Sobol. For each sample drawn from hyperparameter search space, we ran each algorithm on 30 independent runs and presented results using elementary effects and Sobol indices. These indices roughly have two categories of measures: direct effect of a hyperparameter and interaction other hyperparameter effects. Moreover, these measures form a comparative matrix of low effects to high effect, where the diagonal from low direct effect and low interaction to high direct effect and high interaction shows the *order and ranking* of the hyperparameters. For obtaining comprehensive results, we ran algorithms on a sufficiently large sample set. These experiments were computationally expensive as they took about 6 months to complete on MATLAB. Computation of these sensitive analysis indices is expensive, but they are a one-time effort, and once the ranking is determined, results are informative to researchers and practitioners for solving optimization problems.

Our results show that type of different versions of the DE algorithm has the strongest influence on the performance of DE, followed by population size and crossover probability. Our results also support the innovation in the type of strategy used for multi-objective decomposition in MOEA/D as the hyperparameter Mode of decomposition was found the most influential. For MOEA/D, mutation operation has a significant role in the MOEA/D algorithm. We found that correctly setting the initial step size for population generation in the CMAES algorithm and crossover probability setting in the NSGA-III algorithm has a high influence. Moreover, we present a ranking of hyperparameters of each algorithm.

In summary, our paper has the following three main novel contributions:

1. We create a framework to assess effectiveness and ranking of EA hyperparameters to inform sequence of hyperparameters tuning.
2. We assess effectiveness of EA hyperparameters and their effective tuning range using global sensitivity analysis methodologies.
3. We present ranking and tuning order of hyperparameters and their quality of effect on EA using a comparative matrix of sensitivity analysis indices.

The rest of the paper is structured as follows. First, we present related work in Sec 2. Then, Section 3 and Section 4 respectively describe EAs and sensitivity analysis methods. Section 5 summarizes experiments. The results are discussed in Section 6, followed by conclusions in Section 7.

2 Related Work

Hyperparameters tuning is a crucial subject that has continuously been reported in the literature since the past decades (De Jong, 2007). This is because, an appropriate hyperparameters setting is challenging since EA's hyperparameters exhibit linear and non-linear effects (Lima and Lobo, 2004). That is, they show interaction among them (De Jong, 2007, De Jong et al., 2004, Hansen and Ostermeier, 2001). Abundant literature is available on hyperparameters tuning of EAs (De Jong et al., 2004, Lima and Lobo, 2004, Greco et al., 2019). Majority of which focus on *static* or *dynamic* setting the hyperparameters (Eiben et al., 2007, Kramer, 2010). However, a systematic study of the EA hyperparameters influence is rare (Pinel et al., 2012). This is largely attributed to the computationally expansive nature of EAs and empirical evaluation requirement for tuning their hyperparameters (Maturana et al., 2010).

For example, a package *Irace* experimentally evaluates optimal hyperparameters for an optimization algorithm (López-Ibáñez et al., 2016). Similarly, Iglesias et al. (2007) analyzed genetic algorithm hyperparameters sensitivity on a water distribution network design problem by systematically varying the hyperparameters. De Jong (2007) posed questions like (1) what EA hyperparameters are useful for improving performance, and (2) how do changes in a hyperparameter affect the performance of an EA? Sensitivity analysis answers questions like *how uncertainty in each of the hyperparameters influences the uncertainty in the output of a model* (Saltelli et al., 2004). Hence, sensitivity analysis is useful in answering De Jong (2007) questions. However, sensitivity analysis is a computationally expansive method since hyperparameters are sampled from a vast hyperparameter search space. Therefore, the sensitivity analysis of EAs has very high computational (time) as well as memory (space) overhead. This has resulted in very few reported works available in the literature despite its advantages in suggesting a ranking of hyperparameters importance.

The dynamic hyperparameters tuning requires hyperparameters to adapt during an EA run, while static tuning informs which hyperparameters to tune before EA run (Kramer, 2010). A systematic approach, like sensitivity analysis, is a static hyperparameter tuning approach. Paul et al. (2011) offered an introductory work on the usage of *local* and *global* sensitivity analysis. However, they used a simple test case, and they mainly performed a sensitivity analysis of EAs from a theoretical perspective. Pinel et al. (2012) performed a comprehensive sensitivity analysis of a parallel asynchronous cellular genetic algorithm on a scheduling problem. They comprehensively evaluated EAs population size, mutation probability, crossover probability, and other cellular genetic algorithm-related hyperparameters using the Fourier amplitude sensitivity test (Fast99) Saltelli et al. (1999). Pinel et al. (2012) reported a ranking of hyperparameters on scheduling problem instances. On this scheduling problem instance, the crossover probability was ranked first, and in another instance, it was ranked third.

Our work takes an experimental approach to systematically analyze the importance of hyperparameters of state-of-the-art EAs on a testbench of state-of-the-art problems by applying Morris (Morris, 1991) and Sobol (Sobol and Kucherenko, 2005) sensitivity analysis methodologies. Our methodology comprises both single-objective and multi-objective EAs. Our framework offer rankings of hyperparameters and insights into their effectiveness of EA performance.

3 Evolutionary Algorithms

EAs are population-based evolution-inspired algorithms. EAs iteratively find solutions to a problem by applying evolutionary operators to candidate solutions. Selection, recombination, and mutation are among evolutionary operators which we apply to candidate solutions to generate new solutions in a generation. The natural selection principle guides a sequence of generations from an initial population of candidate solutions to a final population. Four different EAs are investigated in this research: two single-objective and two multi-objective algorithms. Each of these EA has its own version of evolutionary operators. This section briefly describes each of these EAs and their performance measures matrices.

3.1 Single-objective Evolutionary Algorithms

A single-objective optimization (SOO) algorithm (single solution-based or population-based) *minimizes an objective function* (a cost function or a problem) as:

$$\begin{aligned} f: \quad & \mathbb{R}^n \to \mathbb{R} \\ & \mathbf{x} \mapsto f(\mathbf{x}), \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is a candidate solution (a search point), and we want $f(\mathbf{x})$ to be as minimum as possible. An SOO algorithm converges to a solution $\mathbf{x}^*$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x}),\ \forall \mathbf{x} \in X$. The solution $\mathbf{x}^*$ therefore is a *global minimum* (global optimum). However, if for $f(\mathbf{x}^*) \leq f(\mathbf{x})$ there exist some $\delta > 0$ such that $|\mathbf{x} - \mathbf{x}^*| \leq \delta$ for any $\mathbf{x} \in X$, the solution $\mathbf{x}^*$ is a *local minimum* (near-optimum).

We study two population-based single-objective global optimization algorithms: CMAES (Hansen and Ostermeier, 1996) and DE (Storn and Price, 1997). The basic steps and operators of CMAES and DE are as follows.

3.1.1 Covariance Matrix Adaptation Evolution Strategies (CMAES)

CMAES is a population-based *evolutionary strategy* optimization algorithm (Hansen and Ostermeier, 1996). CMAES algorithm generates new candidate solutions during its search by sampling solutions from a *multivariate normal distribution*, $\mathcal{N}(\mathbf{m}, C)$ uniquely determined by its mean $\mathbf{m} \in \mathbb{R}^n$ and its symmetric positive definite covariance matrix $C \in \mathbb{R}^{n \times n}$. The initial population

(at generation $g = 0$) of λ candidate solutions, therefore, are sampled as:

$$\mathbf{x}_k^g \sim \mathbf{m}^g + \sigma^g \mathcal{N}(\mathbf{0}, C^g) \quad \text{for } k = 1, \ldots, \lambda \tag{2}$$

where $\mathcal{N}(\mathbf{0}, C)$ is a multivariate normal distribution with zero mean and covariance matrix $C^g \in \mathbf{I}$, and $\sigma^g \in \mathbb{R}_{>0}$ is an initial step size.

For generation $g = 1, 2, \ldots$, multivariate normal distribution $\mathcal{N}(\mathbf{m}, C^{g+1})$ is generated (updated) with mean $\mathbf{m} \in \mathbb{R}^n$ and covariance matrix $C \in \mathbb{R}^{n \times n}$ updated with scalar factor $\sigma^g \in \mathbb{R}_{>0}$. Selection and recombination operation in CMAES is equivalent to computing moving mean m^{g+1} a weighted average of selected points λ_{ratio} from generation g. Adding a random vector with zero mean acts as a mutation in CMAES during the offspring generation step. The steps size control and covariance matrix adaptation (learning rate α_μ) are another two necessary steps in a generation (Hansen and Ostermeier, 1996).

3.1.2 Differential Evolution (DE)

DE is a *gradient-free* EA originally proposed by Storn and Price (Storn and Price, 1997). DE iteratively searches for a solution. For an initial population $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\lambda]$ of size λ, DE repeats its steps *selection*, *mutation*, and *recombination* until an optimum solution vector $\mathbf{x}^*$ is obtained or until a maximum iteration is reached. At each generation $g = 1, 2, \ldots$, DE randomly selects three distinct candidate solutions $\mathbf{x}_{r1}^g$, $\mathbf{x}_{r2}^g$, and $\mathbf{x}_{r3}^g$ from X such that $\mathbf{x}_{r1}^g \neq \mathbf{x}_{r2}^g \neq \mathbf{x}_{r3}^g$, selection of a base vector $\mathbf{x}_{r1}^g$ plays a crucial in DE.

A mutation operation is performed on a base vector $\mathbf{x}_{r1}^g$ to generate a donor vector $\mathbf{v}^{g+1}$ is generated using a method $\mathbf{b}_{\text{type}}$, a difference vector $(\mathbf{x}_{r2}^g \mathbf{x}_{r3}^g)$, and acceleration coefficient β_{ext}. A mutation method $\mathbf{b}_{\text{type}}$ = "DE/rand/1" is performed as:

$$\mathbf{v}^{g+1} = \mathbf{x}_{r1}^g + \beta_{\text{ext}}(\mathbf{x}_{r2}^g \mathbf{x}_{r3}^g) \tag{3}$$

A crossover operation using a crossover method {bin, exp} is performed to generate a trial vector $\mathbf{u}^{g+1}$ which takes its elements from a donor vector $\mathbf{v}^{g+1}$ using a crossover probability P[X]. If the fitness of $f(\mathbf{u}^{g+1})$ is better than the target vector $f(\mathbf{x}_t^{g+1})$, trial vector $\mathbf{u}^{g+1}$ replaces the target vector $\mathbf{x}_t^{g+1}$.

3.2 Multi-objective Evolutionary Algorithms

A multi-objective optimization (MOO) algorithm *minimizes two or more objective functions* simultaneously as per:

$$F(\mathbf{x}) \equiv (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})), \text{ i.e., } F : \mathbb{R}^n \to \mathbb{R}^k \text{ for } k \geq 2 \tag{4}$$

such that no one objective of the problem can be improved without a simultaneous detriment to at least one of the other objectives. Each $f_i(\mathbf{x})$ is a scalar objective, and MOO optimizes objective vector $F(\mathbf{x})$ where $\mathbf{x} \in \mathbb{R}^n$ is its feasible solution. More specifically, a MOO algorithm produces a set of non-dominated solutions $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\lambda\}$, also known as the Pareto-optimal solutions set (Deb, Pratap, Agarwal and Meyarivan, 2002), which is defined as follows.

A solution $\mathbf{x}_i$ dominates other solution $\mathbf{x}_j$ if for all objective $f_k = 1, 2, \ldots, k$, $f(\mathbf{x}_i) \preccurlyeq f(\mathbf{x}_j)$, where $\preccurlyeq$ should be read as "better off." On the contrary, a solution $\mathbf{x}_i$ is non-dominated if for at least one objective $f(\mathbf{x}_i) \preccurlyeq f(\mathbf{x}_j)$ does not hold. For each $\mathbf{x}_j$, a set of such non-dominated solutions is called a Pareto-optimal set of solutions.

We study population-based multi-objective global optimization algorithms NSGA-III (Deb and Jain, 2013) and MOEA/D (Zhang and Li, 2007) and investigate their algorithmic hyperparameter setting such that we obtain Pareto-optimal set of solutions for a problem.

3.2.1 Non-Dominated Sorting Genetic Algorithm–III (NSGA-III)

NSGA-III is a population-based MOO algorithm (Deb and Jain, 2013). NSGA-III uses fast non-dominated sorting and niching operations to guide an initial population X of size λ candidate solutions through a predefined number of generations to a final population while simultaneously optimizing trade-offs of multiple objectives. In each step of NSGA-III, crossover, mutation, and non-dominated sorting is performed.

The fast non-dominated sorting sorts the λ candidate solutions into several sets (called Fronts) of non-dominated solutions: $F_1, F_2, \ldots, F_s$ such that the Front F_1 contains all the non-dominated candidate solutions of population X. That is, no one solution in F_1 is dominated by any other solutions. From all the remaining solutions (i.e., except the ones already in F_1), a new Front F_2 that contains all the next non-dominated solutions of X is determined. Similarly, Front F_3 and other Fronts are subsequently obtained using non-dominated sorting. Thus, it is possible to assign a rank to the candidate solutions such that those in the Front F_1 have rank 1, solutions in Front F_2 have rank 2, and so on.

NSGA-III performs *niching* as its selection operation on non-dominated sorting solutions. Niching takes advantage of a predefined set of reference points placed on a normalized hyperplane of a k-dimensional objective-space (Das and Dennis, 1998), where each individual $\mathbf{x} \in X$ in the population is associated with reference points (Deb and Jain, 2013). The total number of reference points depends on the predefined number of divisions associated with each objective axis.

NSGA-III repeats its operations selection, crossover, mutation, and recombination until a maximum iteration is researched. The performance of NSGA-III is measured in terms of the quality of solutions it produces in its iteration and in the final population (cf. Section 3.3).

3.2.2 Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D)

MOEA/D solves a MOO problem by decomposing the MOO problem into many single (scalar) objective sub-problems (Zhang and Li, 2007). Tchebycheff approach (Miettinen, 2012) or normal boundary interaction approach (Das and Dennis, 1998) are typically used for decomposing MOO problem into (say) N scalar sub-problems. A uniform spread of N weight vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ and reference point $\mathbf{z}^* = (z_j^1, \ldots, z_j^k) = \min\{f_i(\mathbf{x}) | \mathbf{x} \in X\}$, for $i = 1, \ldots, k$ is used for computing $j = 1, 2, \ldots, N$ scalar objectives $y^{te}(\mathbf{x}|\mathbf{w}_j)$.

The scalar objective in Tchebycheff decomposition method is $y^{te}(\mathbf{x}|\mathbf{w}_j) = \max_{1 \leq i \leq k}\{\mathbf{w}_j^i | f_i(\mathbf{x}) - \mathbf{z}^*\})$, where weight vector $\mathbf{w}_j = (w_j^1, \ldots, w_j^k)$. The optimal solution of $y^{te}(\mathbf{x}|\mathbf{w}_i)$ for weight vector $\mathbf{w}_i$ should be close to a solution $y^{te}(\mathbf{x}|\mathbf{w}_j)$ for weight vector $\mathbf{w}_j$. Hence, in MOEA/D, a neighborhood of weight vector $\mathbf{w}_i$ is defined with many closest points in $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$. The neighborhood plays a vital role in MOEA/D. The sensitivity of which is investigated in this work.

Moreover, each objective is optimized as a single (scalar) objective problem. That is, ith objective is optimized such that it minimizes its distance from a reference point on a $k-$ objective space. Thus, all decomposed sub-problems move towards the reference point $\mathbf{z}^*$. MOEA/D maintains T closest solution vectors (Neighbor) for each candidate solution in successive steps. In each iteration, MOEA/D generates a new solution from select two solution vectors using genetic operators and evaluate it to update and its neighborhood and the best solution $\mathbf{x}^*$. Details of MOEA/D algorithm are available in (Zhang and Li, 2007).

3.3 Performance metrics

3.3.1 Single Objective Matrices

A population-based EA applied to solve a single-objective problem offered the best solution in its final population. The best solution $\mathbf{x}^*$ is the one that has the lowest $f(\mathbf{x})$ value among all solutions of all generations of a single-objective EA. Hence, the *Best Solution* obtained in fewer generations in a lesser wall clock time measures the quality of a single-objective EA.

3.3.2 Multi-Objective Matrices

Multi-objective EAs applied to a MOO problem typically offer a set of solutions that meets trade-offs between the objectives. This set of solutions are non-dominated solutions called the Pareto-front. A Multi-objective EA is, therefore, guides a population of candidate solutions from *current Pareto-front* $\mathbf{A}$ toward a *true Pareto-front* $\mathbf{Z}$.

In such a setting, three indicators are used to measure and compare the performance of EAs on MOO problems: generational distance, inverse generational distance, and hyper-volume indicator:

Generational Distance (GD) Generational distance G_i at an iteration i measures the generational distance between *current Pareto-front* and *true Pareto-front* of a multi-objective problem (Veldhuizen and Lamont, 1998, Veldhuizen, 1999). Generational distance G_i is a measure of error *current Pareto-front* and *true Pareto-front* as

$$G_i \triangleq \frac{\sqrt{\sum_{i=1} d_i^2}}{n} \tag{5}$$

where d_i^2 is the distance of ith solution in *current Pareto-front* $\mathbf{A}$ with *true Pareto-front* $\mathbf{Z}$ (Veldhuizen, 1999). Typically, the error is the average of n solutions considered or available in the current and true Pareto-front.

Inverse Generational Distance (IGD) Inverse generational point provides combined information on the solutions diversity and convergence quality. It makes use of a set of target reference points in k-dimensional objective-space. Like GD, IGD compares solutions in the *current Pareto-front* $\mathbf{A}$ with *true Pareto-front* $\mathbf{Z}$. However, IGD uses a single reference and computes the average Euclidean distance between all solutions that are nearest to the target reference points (Deb and Jain, 2013) as follows:

$$IGD(\mathbf{A}, \mathbf{Z}) \triangleq \frac{1}{|\mathbf{Z}|} \sum_{i=1}^{|\mathbf{Z}|} \min_{j=1}^{|A|} d(\mathbf{z}_i, \mathbf{a}_j) \tag{6}$$

where $d(\mathbf{z}_i, \mathbf{a}_j) = ||\mathbf{z}_i, \mathbf{a}_j||_2$.

Hypervolume Indicator (HV) Hyper-volume indicator, H_i measures the dominance of Pareto-front solutions on a geometric space (area for a 2D objective space) framed by the k-dimensional objective-space with respect to a positive semi-axle (Fig. 1). Hence, H_i measures the quality Pareto-optimal solutions set (Fonseca et al., 2006), and it is an indicator of the quality of the solutions obtained by two algorithms with respect to the same reference frame. We want the hyper-volume indicator index H_i to be maximized. A greater value indicates that the algorithm's overall performance is better with respect to another algorithm associated with a smaller hyper-volume value. Moreover, the greatest contributing point in a hyper-volume indicator analysis is the point that covers the largest area, and that can be considered as the best solution (Zitzler et al., 2003).

4 Global Sensitivity Analysis

The goal of the *sensitivity analysis* is to study how the uncertainty of a model's output depends on the uncertainty of its inputs (Saltelli, 2002, Saltelli et al., 2008). The *elementary effects* analysis known as a "Morris method" (Morris, 1991) and *variance-based sensitivity analysis*

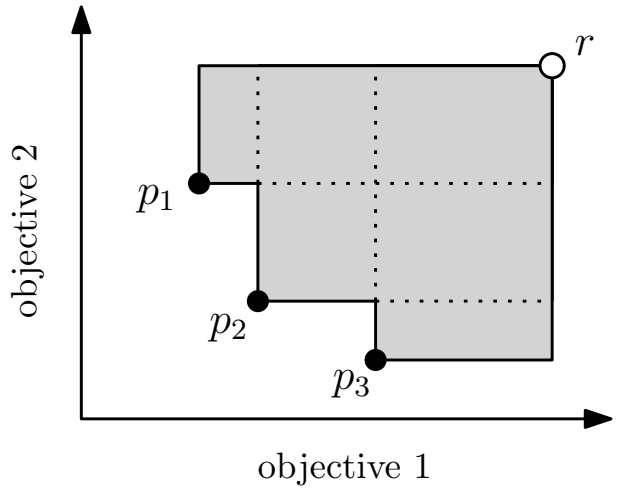

Fig. 1: Example of a 2D objective space. Candidate solution p_1, p_2, and p_3 are framed by a reference point r. H_i is the area framed by point (Pareto-front) and reference point r.

known as a "Sobol method" (Sobol and Kucherenko, 2005) were used in this research for the global sensitivity analysis of the hyperparameters of four EAs.

4.1 Elementary Effects

The *elementary effects* (EE) technique, known as a "Morris method" as it was originally introduced by Morris (1991), is an effective way to analyze the effects (sensitivity) of input variables on the outputs of a model or a system. In our case, the Morris method assesses the EE of the algorithmic hyperparameters on the performances of an EA. This is useful in analyzing the sensitivity of EA's hyperparameters as the Morris method determines whether the effects of a hyperparameter on a model's outputs (EA performances on functions) is (a) insignificant and negligible, (b) linearly correlated, or (c) non-linearly correlated or involved in an interaction with other hyperparameters (Saltelli et al., 2008).

We briefly introduce the computation of EE as follows. Let's $Y = f(\mathbf{X})$ or simply $Y(\mathbf{X})$ be the output of a model $f(\cdot)$ (an algorithm) that takes k hyperparameters $\mathbf{X} = \{X_1, X_2, \ldots, X_k\}$ from a *p-level* grid hyperparameter space Ω. Then we compute the *elementary effect* EE_i of ith hyperparameter X_i as

$$\mathrm{EE}_i = \frac{Y(X_1, \ldots, X_{i-1}, X_i + \Delta, \ldots, X_k) - Y(X_1, \ldots, X_{i-1}, X_i, \ldots, X_n)}{\Delta}, \tag{7}$$

where Δ is a value in $\left(\frac{1}{p-1}, \ldots, 1 - \frac{1}{p-1}\right)$ which is an incremental change in the values of hyperparameter X_i when X_i is sampled from *p-level* grid hyperparameter space Ω. In this scenario, for k hyperparameters and p discrete levels, $\Delta = p/2(p-1)$ indicates distance (length) between two levels in the hyperspace Ω along ith axis. The total points in the hyperparameter

space Ω, therefore, is $p^{k-1}[p-\Delta(p-1)]$ grid points, which increase exponentially as the number of hyperparameters k increases. However, we use a *one-at-a-time* (OAT) sampling technique for generating r sample points from this space to compute r EEs for each hyperparameter.

In the OAT sampling technique, hyperparameter X_i value is changed from a grid point $X_i^{(j)}$ to the adjacent grid point $X_i^{(j\pm1)}$ by a length of Δ while all other hyperparameters (say $X_{\sim i}$) remain as it is. Then a next hyperparameter X_{i+1} is chosen, its value is changed while others remain fixed. This way of sampling is a random walk-through grid of hyperspace Ω. For sampling points (a set $\mathbf{X}$ of hyperparameters) from the hyperspace Ω, we use Latin Hypercube Sample (LHS) based Morris method (*Morris LHS*) (Campolongo et al., 2007), which is a stratified sampling approach to cover all region of the hyperspace Ω. Another approach we use is a *Morris* sampling method (we call it *Morris*) (Morris, 1991), which is a uniform non-repeating random walk within hyperparameter space Ω approach.

Here, we typically select r sample points for each hyperparameter X_i. Hence, both OAT-based Morris LHS and Morris sampling methods give us $r(k+1)$ samples points in total for k hyperparameters.

We measure two indices μ_i and σ_i indicate *mean* (central tendency) and *standard deviation* of EE_i of ith hyperparameter X_i. The measure

$$\mu_i = \frac{1}{r}\sum_{j=1}^{r} EE_i^j. \tag{8}$$

indicates the overall influence of a hyperparameter X_i where larger the μ_i measure is, larger is its *overall* individual ability to influence the outputs of an algorithm. We also measure the standard deviation σ_i of EE_i as

$$\sigma_i = \sqrt{\frac{1}{r-1}\sum_{j=1}^{r}(EE_i^j - \mu_i)^2}, \tag{9}$$

where a large measure of σ_i indicates that a hyperparameter has high interaction with other hyperparameters. The measure σ is an ensemble influence. That is, if σ_i has a high value, it means the computed r elementary effects EE_i^r of ith hyperparameter X_i varied a lot because of the variation in the values of other hyperparameters as well. Whereas a low value of σ_i means small differences in the computed r elementary effects EE_i^r of the ith hyperparameter X_i. This indicates that the influence of a hyperparameter on a model's output is independent of the choice of other hyperparameters values. However, to understand the influence of a hyperparameter, both μ and σ measures need to be seen together (see Fig. 2). We normalized the values of μ_i and σ_i between 0 and 1 to effectively show results as per Fig. 2.

4.2 Variance-Based Sensitivity Analysis

The *variance-based sensitivity analysis* is known as the "Sobol method" (Sobol and Kucherenko, 2005), and it shows how much variance of a model's output depends on its inputs. It is an in-depth sensitivity analysis method that uses two sensitivity indices: (a) *first-order effect* S_i to indicate a direct effect of a hyperparameter X_i on a model's output $Y = f(\mathbf{X})$ and (b) *total effect* ST_i to indicates a hyperparameter X_i interaction with its complementary parameters $X_{\sim i}$.

The direct effect S_i, irrespective of the hyperparameter interaction ST_i, indicates that, on average, how much model's variance $V[Y(\mathbf{X})]$ could be reduced if the hyperparameter X_i is fixed to a value. Meaning, a low value of S_i shows that the variance of model's output $Y(\mathbf{X}|X_i = x_i^*)$ does not depend on X_i, and fixing X_i to a value does not have much impact on the model's output, while for a high value of S_i, it strongly does. Indeed, a low value of S_i indicates that ith hyperparameter's influence is negligible. Similarly, the interaction effect or total effect $ST_i = 0$ indicates that the model's output $Y(\mathbf{X}|X_i)$ does not depend on X_i, and it is a non-influential parameter. The large values of interaction effect or total-effect ST_i show proportionally strong interactions between the hyperparameter X_i and its complementary parameter $X_{\sim i}$. The difference $ST_i - S_i \geq 0$ shows how much ith hyperparameter is involved in interaction with other hyperparameters. We normalized the values of S_i and ST_i between 0 and 1 for lucid interpretation of their influence (see Fig. 2).

The first-order effect S_i and total effect ST_i of Sobol method are computed as

$$S_i = \frac{V(E(Y|X_i))}{V(Y)} = \frac{y_A \cdot y_{C_i} - f_0^2}{y_A \cdot y_A - f_0^2} = \frac{\frac{1}{N}\sum_{j=1}^{N} y_A^j y_{C_i}^j - f_0^2}{\frac{1}{N}\sum_{j=1}^{N} (y_A^j)^2 - f_0^2} \tag{10}$$

and

$$ST_i = 1 - \frac{V(E(Y|X_{\sim i}))}{V(Y)} = 1 - \frac{y_B \cdot y_{C_i} - f_0^2}{y_A \cdot y_A - f_0^2} = 1 - \frac{\frac{1}{N}\sum_{j=1}^{N} y_B^j y_{C_i}^j - f_0^2}{\frac{1}{N}\sum_{j=1}^{N} (y_A^j)^2 - f_0^2}, \tag{11}$$

where N is the number of random samples, $y_A = f(A)$, $y_B = f(B)$ and $y_{C_i} = f(C_i)$ are model output vectors on sample matrix A, B and C_i respectively; and the estimated mean f_0^2 is

$$f_0^2 = \left(\frac{1}{N} \sum_{j=1}^{N} y_A^j \right)^2. \tag{12}$$

Matrices $A_{N\times k}$ and $B_{N\times(k-2k)}$ are random sample points (hyperparameter values), and each matrix C_i is formed by taking all columns of matrix B except ith column, which is taken from ith column of matrix A. Such a sampling is similar to OAT sampling, except its rows are not sorted in any specific order, and all elements in a row differ from the elements in the following row.

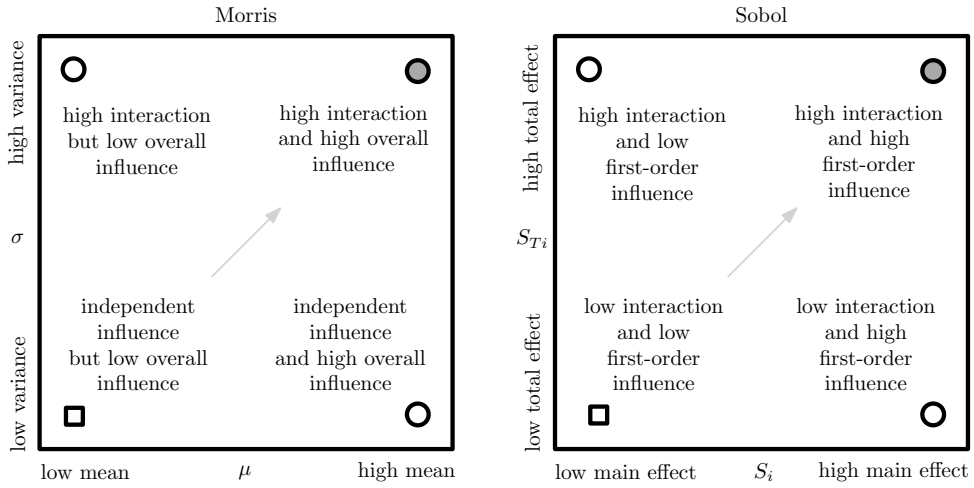

Fig. 2: Morris (*left*) and Sobol (*right*) methods measures interpretation. Top right corner circle in dark gray is the ideal case where a hyperparameter has high individual influence and high interaction (or total effect). Circles in white at top left and bottom right corners are cases where has high importance in at least one direction. Bottom left square in white shows the least ideal case where hyperparameters are non-influential, and fixing them at any values within their defined domain will not influence the algorithm's performance. Arrow along the diagonal direction indicates the order of the hyperparameters' importance and influence.

5 Experiments

Our sensitivity analysis framework's workflow has four essential structural components:

1. The setup of EAs tunable hyperparameters and optimization problems.
2. The sampling of hyperparameters from hyperparameter space of respective sensitivity analysis methods for respective algorithms.
3. The evaluation of EAs on optimization problem (testbench) for all sampled hyperparameter points and for each hyperparameter sampled, the evaluation of respective EAs over 30 independent instances to obtain stable results and to observe expected (average-case) performance of algorithms over performance measures.
4. The computation of Morris and Sobol indices.

We discuss EA's sensitivity to its hyperparameters in Section 2. Section 3 referred to EAs tunable hyperparameters and their role in EAs. All EAs start with a population of initial candidate solutions (uniformly randomly drawn from $\mathbb{R}^n$, n being dimensionality of problem). Other commonalities among EAs are evolutionary operators like "selection," "mutation," and/or "crossover" for generating new (offspring) population and their evaluation. EAs repeat this process for a number of generations until a *termination condition* is met. We set the *termination condition* to be the desired number of *function evaluations*, and we set this to a value of $100,000$ for all four algorithms for all problems. The other hyperparameters setting for our experiments were as follows:

Table 1: hyperparameter domain range of CMAES (Hansen and Ostermeier, 1996) and DE (Storn and Price, 1997). For both algorithms, termination condition was $10,000$ function evaluations.

Algo	Params	Domain	Description
CMAES	λ	$[10, 1000]$	Population size
	α_μ	$[0, 4]$	Learning rate
	σ_0	$[0.1, 2]$	Initial step size
	$\sigma_{0-scale}$	{False, True}	Re-scaling of σ_0: convergence speed controller
	$\mu\lambda_{\text{ratio}}$	$[0.1, 1]$	Percentage of population's elements usage in co-variance matrix estimation and update
DE	λ	$[10, 1000]$	Population size
	X	{bin, exp}	Crossover methods: Binomial and Exponential
	P[X]	$[0, 1]$	Crossover probability
	$\beta_{\min}$	$[0, 1]$	Minimum Acceleration coefficient
	$\beta_{\max}$	$[0, 2]$	Maximum Acceleration coefficient
	$\mathbf{b}_{\text{type}}$	{"best," "target-to-best," "rand-to-best," "rand"}	Base vector selection methods
	$\mathbf{b}\lambda_{\text{ratio}}$	$[0.01, 0.5]$	Percentage of base vectors (solution) to be used for difference vectors computation

5.1 Single-Objective Algorithm Hyperparameters

We analyzed two single-objective EAs over 23 optimization problems (*testbench*) introduced in (Yao et al., 1999). An EA needs to find a single optimal solution for an SOO problem in a few generations at the expanse of some wall-clock computation time. Hence, the *Best Solution* was used for SOO evaluation. Table 1 lists the hyperparameter tuning space of CMAES (Hansen and Ostermeier, 1996) and DE (Storn and Price, 1997) algorithms.

Sensitivity analysis method setup for single-objective optimization was as follows. We used $p = 30$ grid levels to form the hyperparameter space Ω for respective single objective EAs. From this hyperparameter space, we select $r = 170$ sample points for each hyperparameter of CMAES and $r = 130$ for DE in the cases of *Morris LHS* and *Morris* methods (see Equations 8 and 9). This gave us 1020 and 1040 sample points in total for CMAES and DE algorithms, respectively. The Sobol analysis is $2 + k$ times more expensive than Morris methods since it evaluates hyperparameter matrices A, B, and C_i, $i = 1, 2, \ldots, k$. Hence, we use $N = 100$ and this gave us 700 and 900 sample points in total for CMAES and DE algorithms respectively for their matrices A and B from which C_i matrices were creates.

5.2 Multi-objective Algorithms Hyperparameters

We analyzed multi-objective EAs over a testbench consisting of four families of optimization problems: (1) DTLZ1, DTLZ2, DTLZ3, and DTLZ4 (Deb, Thiele, Laumanns and Zitzler, 2002); (2) IDTLZ1 and IDTLZ2 (Deb, Thiele, Laumanns and Zitzler, 2002); (3) CDTLZ2 (Deb and Jain, 2013); and (4) WFG3, WFG6, and WFG7 (Huband et al., 2006). EAs were evaluated

and analyzed for each listed MOO problem for 3 objectives, and each problem was solved as a 10-dimensional problem. This setting was chosen based on the computation effort required for these MOO problems. Some problems ran for weeks on MATLAB, and all MOO experiments were completed in about 6 months.

Since the goal of the multi-objective EAs is to obtain a set of solutions where no one objective dominants over the other objectives (Deb, Pratap, Agarwal and Meyarivan, 2002, Zhang and Li, 2007), we use GD, IGD, and HV as the measures of EAs performances (see Section 3.3.2). These metrics result in higher values for large population size λ compared to a small population size λ. Thus, for a *population-fair* performance analysis, the metrics were calculated from a union of populations of all generations of EAs and from not only the population of the last generation of the EAs. Moreover, the values were averaged over 30 independent runs for each sampled set of hyperparameters.

NSGA-III and MOEA/D have a few common tunable hyperparameters in addition to their subjective tunable hyperparameters. Table 2 shows the domain setting of these common and subjective tunable hyperparameters of NSGA-III and MOEA/D.

Sensitivity analysis method setup for multi-objective optimization was as follows. We used $p = 10$ grid levels to form the hyperparameter space Ω for respective single objective EAs. From this hyperparameter space, we select $r = 20$ sample points for each hyperparameter of CMAES and DE in the cases of *Morris LHS* and *Morris* methods (see Equations 8 and 9). This gave us 140 and 160 sample points in total for NSGA-III and MOEA/D algorithms, respectively. In the Sobol analysis, we use $N = 30$ and this gave us 240 and 270 sample points in total for NSGA-III and MOEA/D algorithms respectively for their matrices A and B from which C_i matrices were creates. The number of sampling points in this work is sufficiently large for good sensitivity analysis (Campolongo et al., 2007, Saltelli, 2002, Saltelli et al., 2008).

We perform our sensitivity analysis experiments on MATLAB platform. Hence, the implementation of the algorithms and methods were in MATLAB. Moreover, the implementations of individual components were taken from their well-known libraries.

We used *safe toolbox* (Pianosi et al., 2015) for the implementation of sensitivity analysis sampling methods, indices calculations, and workflows. Single objective optimization algorithms were implemented using ypea library (Heris, 2019), and the implementation of multi-objective optimization problems and evaluation measure metrics related to optimization algorithms were used from PlatEMO library (Tian et al., 2017). The entire workflow framework was synchronized with the help of inbuilt functions of MATLAB. Our implementation of this framework for sensitivity analysis of EAs and experiments set up is available at (Ojha et al., 2021).

Table 2: hyperparameter domain range of NSGA-III and MOEA/D and their shared (*Common*) hyperparameter domain. For both algorithms, termination condition was $10,000$ function evaluations.

Algo	Params	Domain	Description
Common	λ	$[10, 1000]$	Population size.
	P[X]	$[0, 1]$	Simulated binary crossover (SBX) probability
	X_{DI}	$[1, 200]$	SBX distribution index
	P[PM]	$[0, 1]$	Polynomial mutation (PM) probability
	PM_{DI}	$[1, 200]$	PM distribution index
NSGA-III	K	$[2, 10]$	Tournament size
	Selection	Tournament	Parents selection for offspring generation
MOEA/D	$Mode$	{"penalty based boundary intersection (PBI)," "Tchebycheff," "Tchebycheff with normalization," "modified Tchebycheff"}	Method for MOO decomposition into many SOO subproblems
	ϵ_N	$[0.05, 0.5]$	Neighbors: percentage of the population considered as neighbors for each sub-problem generation

6 Results and Discussion

Each algorithm's sensitivity analysis results over testbench were collected and processed to produce three performance visualizations: (1) sensitivity analysis indices matrix as per Fig. 2, (2) ordered bar plot arranged from low to high sum of the normalized sensitivity analysis indices values, and (3) mean score (average performance) of each hyperparameter over select performance measures. This section describes hyperparameters' performance *influence*, *ranking*, and *quality* through these three visualizations.

Each sensitivity analysis method marginally varied how they sample hyperparameter sets. However, they use similar strategies like OAT, LHS, and Uniform distribution. Therefore, they are similar in terms of their distribution but slightly varied in how they draw the samples. Morris LHS and Sobol methods used LHS strategy, which means they stratified the hyperspace to draw samples to cover most of the sample space. Morris used OAT sampling, whereas Sobol slightly varied from OAT by allowing all hyperparameters to be taken at a time. In summary, each method may present its own ordering of hyperparameters that influence ranking and interpretation. Hence, we are also interested in the commonality of results among these three methods, and we present interpretations for a combined influence found in these three methods in this section.

6.1 Single-Objective EAs

6.1.1 CMAES Analysis

CMAES results are shown in Figs. 3, 4, and 5, where Fig. 3 is a scatter plot that presents sensitivity analysis indices as per Fig. 2. It shows the tendency of the *quality of influence* a hyperparameter has on CMAES performance on all 23 problems in the testbench. For instance, σ_0, an initial step size used for the formation of covariance matrix for the initial population generation in CMAES has high *overall* influence and high *interaction* influence in all three sensitivity analysis methods. Hence, σ_0 clearly the most significant hyperparameter of CMAES algorithm and this must be the first hyperparameter one must select to tune for the performance improvement when CMAES is applied to solve a problem.

Initial step size σ_0. Fig. 4 confirms the significance of σ_0 (initial step size) influence as this emerged as the best hyperparameter in Morris LHS and Morris methods, and for Sobol, it is a close second to population size λ. Morris LHS which is a stratified sampling method that covers the most region of suggests that σ_0 is taken from most regions of its possible values and the CMAES performance had varied because of such sampling. Examining Fig. 5, we may observe that only for a particular range of σ_0 values, CMAES mean performances were better. That is for small σ_0 values (close to 0.1) and large σ_0 values (1.5, or above), CMAES did not perform as good as for the values in $[0.2, 1.5)$. Hence $[0.2, 1.5)$ is the best range for σ_0.

Convergence controller $\sigma_{0-scale}$. CMAES convergence controller hyperparameter $\sigma_{0-scale}$, a hyperparameter meant for switching the re-scaling of initial step size σ_0 on and off, is the least influential in both Morris and Sobol methods (cf. Fig. 3). This result is supported by both Fig. 4 and Fig. 5. In Fig. 4 (Morris LHS column), $\sigma_{0-scale}$ is a distant second to $\sigma_{0-scale}$, and it is the last one in Morris and Sobol methods. Hence, an agreement between all three methods of its low influence and low interaction suggests that the re-scaling of initial step size $\sigma_{0-scale}$ which controls the CMAES convergence speed is a non-influential hyperparameter when we take a sufficiently large number of function evaluations for optimizing problems. In this case, we had $10,000$ function evaluations, and CMAES performance, poor or worse on all problems, was not linked with the setting of $\sigma_{0-scale}$ to *True* or *False* (cf. Fig. 5).

Learning-rate α_μ. Similar to $\sigma_{0-scale}$, learning-rate α_μ was found to be non-influential. However, it was found to be non-influential. However, since the performance of CMAES was consistent for its chosen values across all three methods, learning-rate α_μ was better than re-scaling $\sigma_{0-scale}$. Moreover, learning-rate α_μ shows more interaction with other hyperparameters than convergence speed controller $\sigma_{0-scale}$. This is also evident as gray Bar are larger than black Bar in Fig. 4 and high fluctuation it in α_μ scores (yellow dots in Fig. 5).

Population size λ. Population size λ is an important factor in CMAES algorithms. Both Morris and Sobol methods show a strong overall influence and high interaction, which is similar

to the initial step size σ_0 influence. Morris LHS ranked it very low on interaction, which suggests that the spectrum of values chosen for λ was unaffected by the choice of other hyperparameters and its influence on the performance on CMAES was independent, and it has only a direct influence on CMAES. This is evident from λ upward performance trend (lower to higher scores) for increasing λ size in Fig. 5. In comparison to λ, σ_0 shows higher interaction than the direct effect. This means that it is straightforward to increase population size to enhance the performance of CMAES on a problem, but σ_0 tuning needs a trail and error strategy.

Covariance matrix size controller $\mu\lambda_{\text{ratio}}$. Hyperparameter $\mu\lambda_{\text{ratio}}$, which controls the percentage of population λ to be used for the covariance matrix estimation and update, has high interaction than a direct influence on CMAES performance. This is rightly so, as its values and the choice of λ are closely linked, and the choice of this ratio will increase or decrease the size of the covariance matrix. The $\mu\lambda_{\text{ratio}}$ is the third most influential hyperparameter across all three methods.

CMAES hyperparameters ranking. In summary, we may provide a *ranking of hyperparameters* for CMAES from the most to least influential hyperparameter as σ_0, λ, $\mu\lambda_{\text{ratio}}$, α_μ, and $\sigma_{0-scale}$. One may ignore tuning α_μ and $\sigma_{0-scale}$ completely as setting a sufficiently large function evaluation number would neutralize their importance in CMAES algorithm.

6.1.2 DE Analysis

DE versions $\mathbf{b}_{\text{type}}$. DE results are shown in Figs. 3, 4, and 5, where Fig. 3 shows the quality of influence each hyperparameter has on DE performance over 23 problems. As per the results of the three sensitivity analysis methods, it is clear that the type of DE base vector selection method $\mathbf{b}_{\text{type}}$ (DE version) is by far the most significant hyperparameter. Examining Fig. 5, we observe that DE type *best* and *rand* have lower scores, fluctuating. Whereas the base vector selection methods *target-to-bin* and *best-to-bin* performed more consistently with high scores. Therefore, the average performance of DE over the testbench was highly sensitive to the selection of $\mathbf{b}_{\text{type}}$. We also observe that the sphere of influence of $\mathbf{b}_{\text{type}}$ in Fig. 3 remains at $(1, 1)$ corner of the plot, meaning it had both high *direct effect* and high *interaction effect*. High interaction effects suggest that $\mathbf{b}_{\text{type}}$ performance was also subjected to changes in other hyperparameters values.

Acceleration coefficients $\beta_{\min}$ and $\beta_{\max}$. Acceleration coefficients hyperparameters $\beta_{\min}$ and $\beta_{\max}$ have a vital role in DE performance. Specifically, the setting of an appropriate range is vital for DE performance. This is evident from $\beta_{\min}$ being the most influential hyperparameter in Morris LHS methods (cf. Fig. 4). Since Morris LHS method uses a stratified sampling approach, it had forced it to select values of $\beta_{\min}$ across its whole range, and the performance of DE impacted negatively by the higher values of $\beta_{\min}$. Examining Fig. 5, we observed that $\beta_{\min}$ scores for values in $[0.0, 0.5]$ performed consistently with better scores than the values in

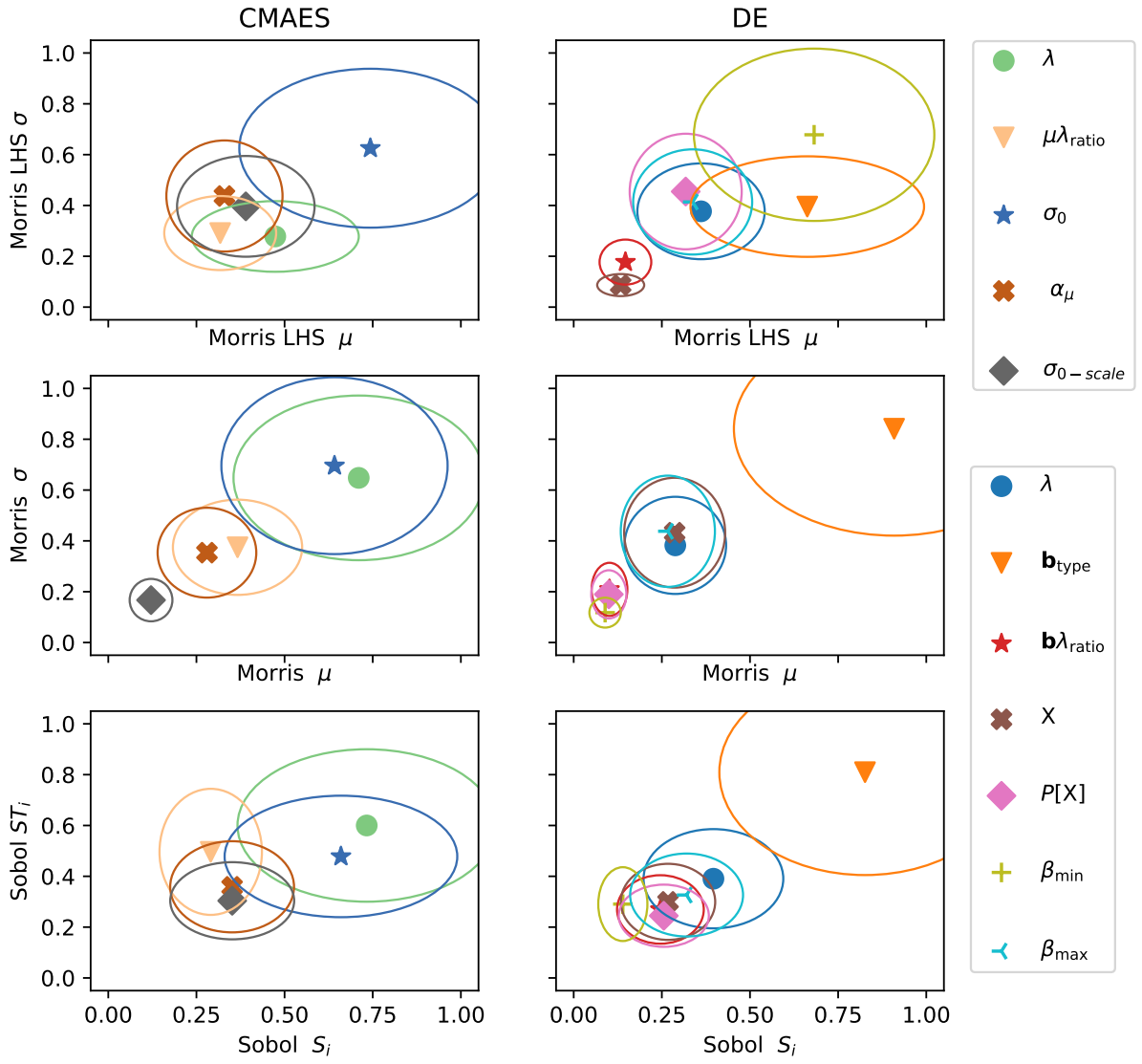

Fig. 3: Single objective algorithms sensitivity analysis. CMAES and DE hyperparameters sensitivity analysis are shown in column 1 and column 2, respectively. Rows 1, 2, and 3 respectively indicate Morris LHS, Morris, and Sobol methods. The upper right legend belongs to CMAES and the lower right to DE. A symbol and a color represent each hyperparameter. An eclipse centered at a hyperparameter is the span of that hyperparameter influence, and the shape of the eclipse shows the direction of its influence. A larger width of the eclipse of a hyperparameter in the x-axis direction means more direct dominance of that hyperparameter, and a larger height in the y-axis direction means its dominant total (or interaction) influence. In each sub-plots, further apart a hyperparameter in the diagonal direction from the origin $(0, 0)$ is, higher is its importance to the algorithm. CMAES hyperparameter λ, $\mu\lambda_{\text{ratio}}$, σ_0, α_μ, and $\sigma_{0-scale}$ respectively are population size, percentage of population for covariance matrix, initial step size, learning rate, and convergence speed controller. DE hyperparameters λ, $\mathbf{b}_{\text{type}}$, $\mathbf{b}\lambda_{\text{ratio}}$, X, P[X], $\beta_{\min}$, and $\beta_{\max}$ respectively are population size, base vector selection type, percentage of population for base vector selection, crossover type, crossover probability, minimum acceleration coefficient, and maximum acceleration coefficient. The hyperparameters definition and domain are given in Table 1.

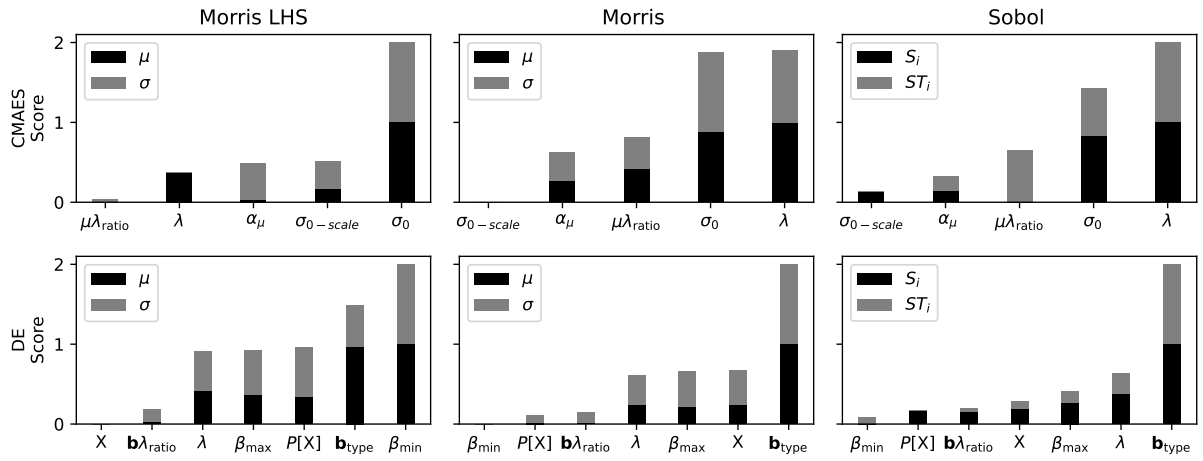

Fig. 4: Ordering (small to larger) of the sum of sensitivity analysis indices of single objective algorithms. CMAES (row 1) and DE (row 2) algorithms hyperparameters performance across all problems (functions). Columns 1, 2, and 3 respectively show performance evaluated using Morris LHS, Morris, and Sobol methods. Darker (black) color portion of a bar is the direct influence normalized value in $[0, 1]$ and gray color portion is interaction (total) influence value in $[0, 1]$. Larger height bar implies a higher influence. The hyperparameters name, definition and domain are given in Table 1.

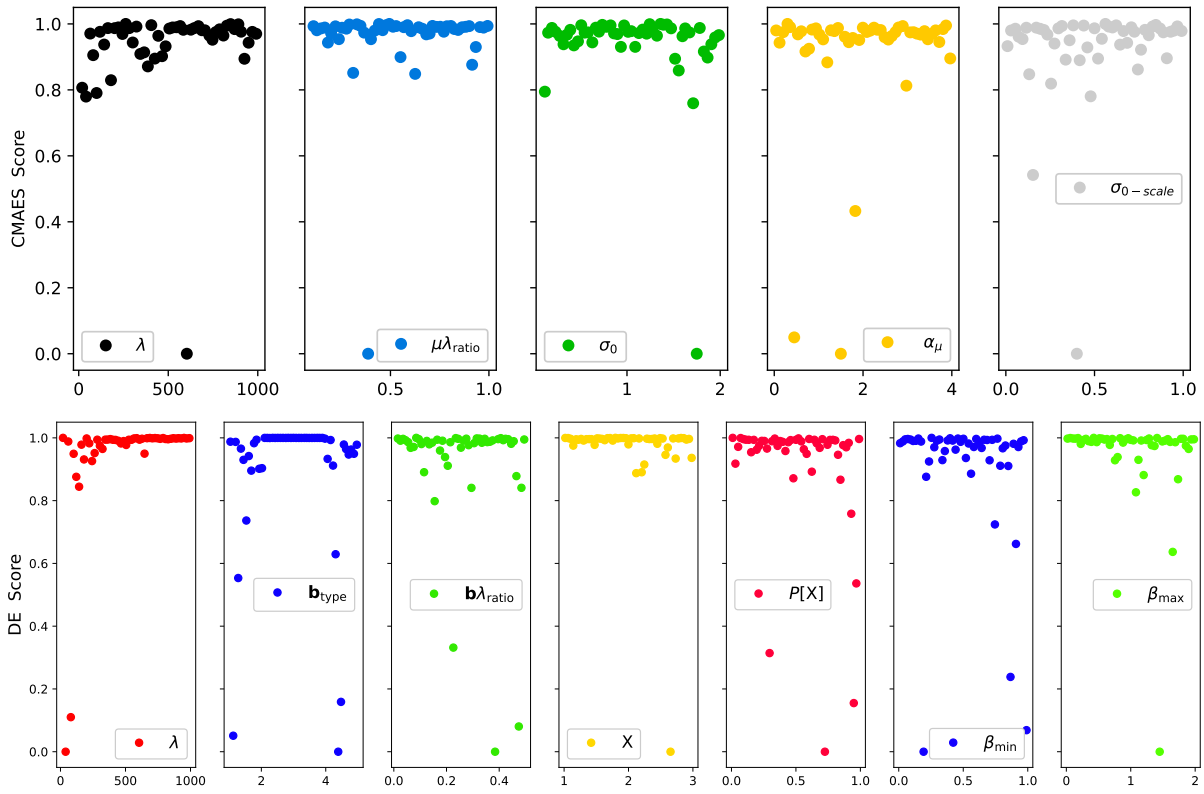

Fig. 5: CMAES and DE algorithms average performance on 30 runs of each hyperparameter set. CMAES and DE algorithms respectively had $2,740$ and $2,980$ hyperparameter sample sets evaluated in total by Morris LHS, Morris, Sobol methods. Each dot in a subplot is a mean performance of a bin of such samples. Along x-axis there are 50 bins from lower to higher values of each hyperparameter which is plotted against y-axis. Each hyperparameter is independently normalized between a score 0 and 1, where 1 is the best score.

$(0.5, 1.0]$. However, Morris and Sobol had a uniform distribution and show that the influence of this hyperparameter is non-influential; therefore, setting these values somewhere in $[0.0, 0.5]$ will suffice, and one may not need to exhaustively tune this hyperparameter.

Similarly, $\beta_{\max}$ was found sensitive to its range selection. Fig. 5 offers us the ways to investigate which range had a positive influence and which had a negative. We observe that the lower values had higher scores than the larger values of $\beta_{\max}$ (cf. Fig. 5). Investigating closely, we found that scores in $[0.2, 0.9]$ are by far better than the scores for other values. This means tuning $\beta_{\max}$ values within range $[0.2, 0.9]$ for a problem is an appropriate strategy. We clearly see that since we experiment with a range $[0, 2]$, $\beta_{\max}$ emerged rank third in both Morris and Sobol methods (cf. Fig. 4), and overall it ranked second most influential hyperparameter.

Population size λ. Overall population size λ is the third most influential hyperparameter in DE (cf. Figs. 3 and 4). Comparatively, it had produced better scores for larger population size than the smaller population size (cf. Figs. 4). However, the size of the population of DE was a distant third influential hyperparameter. This indicates that for apart a very small population size (less than 200), DE's performance was invariable when population size was increased from 200 to 1000 (cf. Fig. 5). This was when the number of function evaluations was the same for each population size. That is, a population size 10 or 1000 had the same number of function evaluations.

Base vectors selection pool $\mathbf{b}\lambda_{\text{ratio}}$. A related hyperparameter to population size is the $\mathbf{b}\lambda_{\text{ratio}}$, which defines the percentage of the population used for the selection of base vectors for DE. Essentially, $\mathbf{b}\lambda_{\text{ratio}}$ is a base vectors selection pool. We found that $\mathbf{b}\lambda_{\text{ratio}}$ has negligible influence on the performance of DE. This fact is agreed by all three sensitivity analysis methods (cf. Figs. 3 and 4).

Crossover type X and crossover probability P[X]. The crossover related hyperparameters, type of crossover X and probability of crossover P[X], also play a comparable role in DE's performance as of population size λ and the type of DE $\mathbf{b}_{\text{type}}$ (cf. Figs. 3 and 4). Among these two crossover-related hyperparameters, the crossover-type has noticeably produced a significant difference in DE's performance for its choice. That is, the crossover-type *binomial* offered better scores than the crossover-type *exponential* (cf. Fig. 5). The crossover probability P[X] has its usage only for binomial crossover. However, the choice of probability P[X] has an insignificant influence on DE performance for this select testbench of 23 problems. However, when we forced an equally spaced spectrum of P[X] values, it has a better a significance level equal to population size and maximum acceleration coefficient.

DE hyperparameters ranking. In summary, we *rank* DE hyperparameters from the most significant to least significant as $\mathbf{b}_{\text{type}}$, λ, X, P[X], $\beta_{\max}$, $\beta_{\min}$, and $\mathbf{b}\lambda_{\text{ratio}}$.

6.2 Multi-objective EAs

6.2.1 NSGA-III Analysis

Population size λ. Results of NSGA-III are presented in Figs. 6, 7, and 8. In Fig. 6, we present results of three measures GD, IGD, and HV; see columns in Fig. 6, and along rows in Fig. 6, we present Morris LHS, Morris, and Sobol sensitivity methods. For NSGA-III, we clearly observe that the population size λ is a clear significant hyperparameter, and the probability of crossover is the second most significant hyperparameter. Population size influence has approximately equal high direct influence and high interaction influence. This means, although population size is the most significant hyperparameter, its performance varied because of the variation of the other hyperparameters. This fact was found true across all three methods, and all three measures as the eclipse of its influence centered around coroner $(1, 1)$ in Fig. 6, and the black and gray bars have comparable lengths in Fig. 8. A further examination of the population size shows monotonically increasing scores for increasing population size values (cf. Fig. 8). Therefore, the variations in the performance of NSGA-III after a sufficiently large population size (in this case, 200) come from the variations of other hyperparameters.

Tournament size K. Hyperparameters tournament size K, probability of polynomial mutation $P[\text{PM}]$, polynomial mutation distribution index PM_{DI} and simulated binary crossover distribution index X_{DI} have comparable significance. However, they differ for different methods and measures. Among these hyperparameters, tournament size K clearly shows high influence on NSGA-III performance. Tournament size K shows more interaction influence than direct influence, except for HV measure of Sobol method. The high score of K in Fig. 8 with clear fluctuation is the evidence of its interaction with other hyperparameters, but the scores (especially in GD scores) shows an upward trend also indicating it has comparatively more direct influence than hyperparameters $P[\text{PM}]$, PM_{DI} and X_{DI}.

Crossover and mutation hyperparameters. The crossover and mutation related hyperparameters, specifically mutation related hyperparameters $P[\text{PM}]$ and PM_{DI} have negligible influence on NSGA-III performance over these continuous optimization problems. The SBX distribution index X_{DI} however, it shows more significance than PM_{DI}. Both distribution indices X_{DI} and PM_{DI} were tested within a range $[0, 200]$, and it is clear that both show good performance for lower values, and the performances started dropping as values approach closer to 100, and then start to rise again (cf. Fig. 8). This phenomenon happened roughly around a value of 100 of these indices, which is aligned with the range for these hyperparameters suggested in (Deb and Deb, 2014, Deb et al., 1995). Therefore, $(20, 100)$ is an appropriate range of these hyperparameters if one needs to tune these hyperparameters.

Probability of crossover $P[\text{X}]$ shows more linear relation between its values and NSGA-III performance measures GD, IDG, and HV. For increasing values of crossover probability, we

see decreasing GD and IGD scores and increasing scores of HV (cf. Fig. 8). This aligns with the property of crossover: for high crossover rate means the solutions in successive generations can have higher similarity, i.e., diversity of solution decreases casing GD to decrease. However, a high crossover rate may lead to better solutions along the problem's objective dimensions, i.e., increasing scores of HV. This fact is supported by high direct and interaction influence of crossover P[X] for GD and IDG measures and relatively direct influence on HV. Sobol method on P[X] does show a very high total influence compared to direct influence.

NSGA-III hyperparameters ranking. Considering the hyperparameters performance influence, we rank them from most influential to least influential hyperparameters as λ, P[X], K, X_{DI}, P[PM], and $\mathrm{PM}_{\mathrm{DI}}$.

6.2.2 MOEAD Analysis

Population size λ. MOEAD results are shown in Figs. 9, 10, and 11. In Fig. 9, the results of three sensitivity analyses for three measures of MOEAD performance are presented. Unlike NSGA-III results, population size λ is not a clear most significant hyperparameter for MOEAD multi-objective algorithm. Rather, MOEAD's hyperparameters $Mode$, the MOO decomposition method is also among the influential hype-parameters. Morris LHS method clearly shows that the population size is the most significant hyperparameter on all three measures. However, Morris and Sobol show that population size has a lower influence on GD measure, but population size strongly influences IGD and HV measures. Fig. 11 also confirms this fact as for the population size values, the GD, IGD, and HV measures show a strong correlation.

For example, HV measure in Fig. 11 shows an upward trend, but it has clear fluctuations in scores. This is due to the fact that population size has high interaction with other hyperparameters, and tuning population size alone can not compensate for the role of the other hyperparameters in the performance of MOEAD. GD measure shows that population size values above 500 lead to a phase shift in the performance of MOEAD. This result is aligned with HV measure that also shows fewer fluctuations in scores of MOEAD than the population size values less than 500.

MOO decomposition type $Mode$. The next set of hyperparameters that we observe as highly influential is $Mode$ as it shows high interaction and high overall influence in Morris LHS, Morris, and Sobol for GD and HV measures. HV measure for Sobol placed the hyperparameters on the diagonal from high overall influence to high total influence (cf. Fig. 9), which suggests that the hyperparameters either have a good high interaction or good overall influence. Hence, the sum of these to presented in Fig. 10 differs only marginally. Sobol rank $Mode$ first in GD measure as both high interaction and high overall influence and third in HV measure as it has a high direct influence. Examining $Mode$ performance in Fig. 11 we confirm that the type of MOO decomposition "Tchebycheff" had the best performance followed by "modified Tchebycheff," and "Tchebycheff with normalization" has significantly poor performance and "penalty based

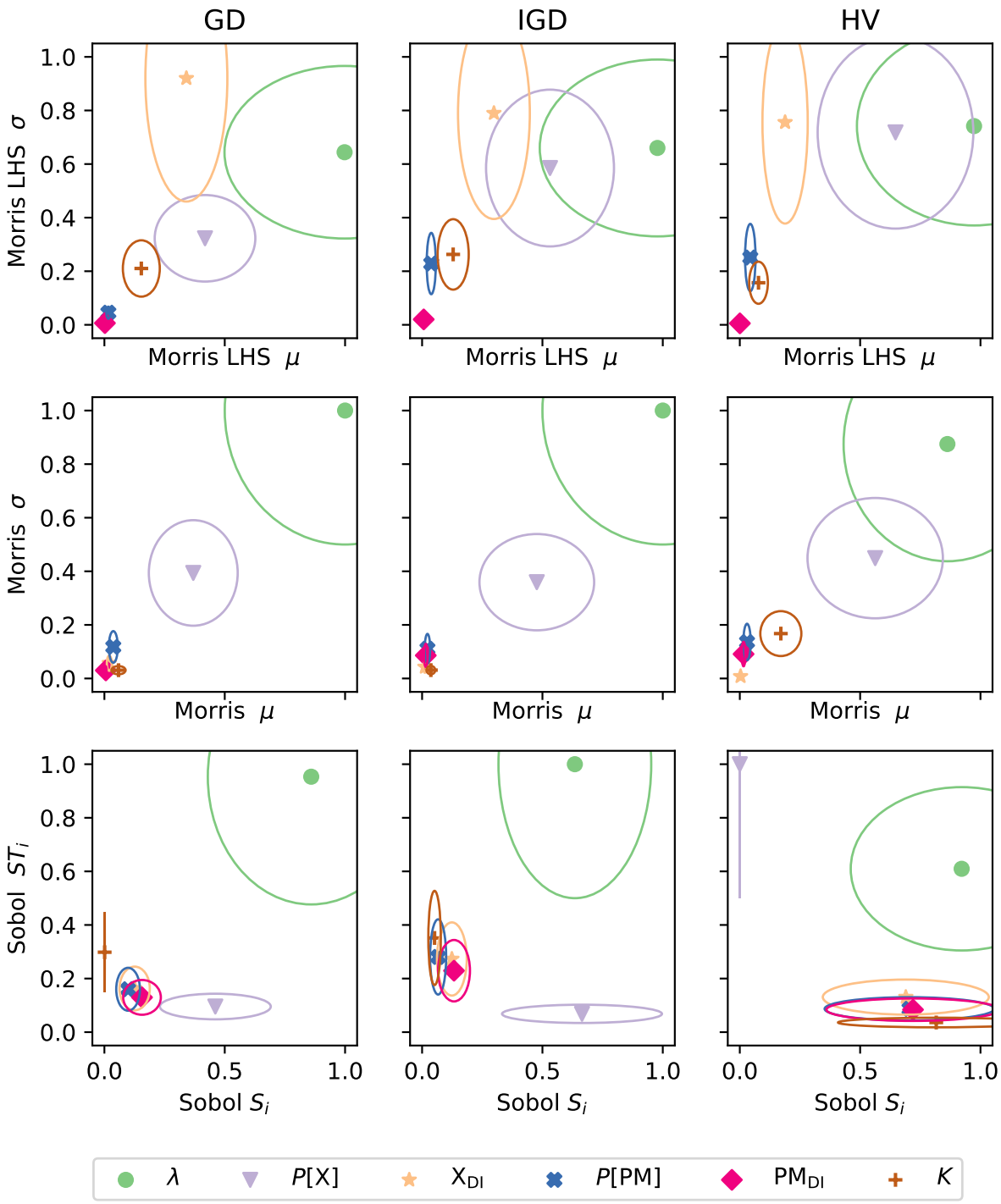

Fig. 6: NSGA-III hyperparameters sensitivity analysis. Columns 1, 2, and 3 respectively indicate performance measure metrics GD, IGD, and HV. Rows 1, 2 and 3 respectively indicate Morris LHS, Morris, and Sobol methods. Legends of hyperparameter are shown at the bottom. Each hyperparameter is represented by a symbol and a color. An eclipse centered a hyperparameter is the span of its influence and direction of its influence. Further apart a hyperparameter is in the diagonal direction from the origin $(0, 0)$, the higher is its importance to the algorithm. A larger width of the eclipse of a hyperparameter in the x-axis direction mean more direct dominance of that hyperparameter, and larger height in the y-axis direction mean its dominant total (or interaction) influence.

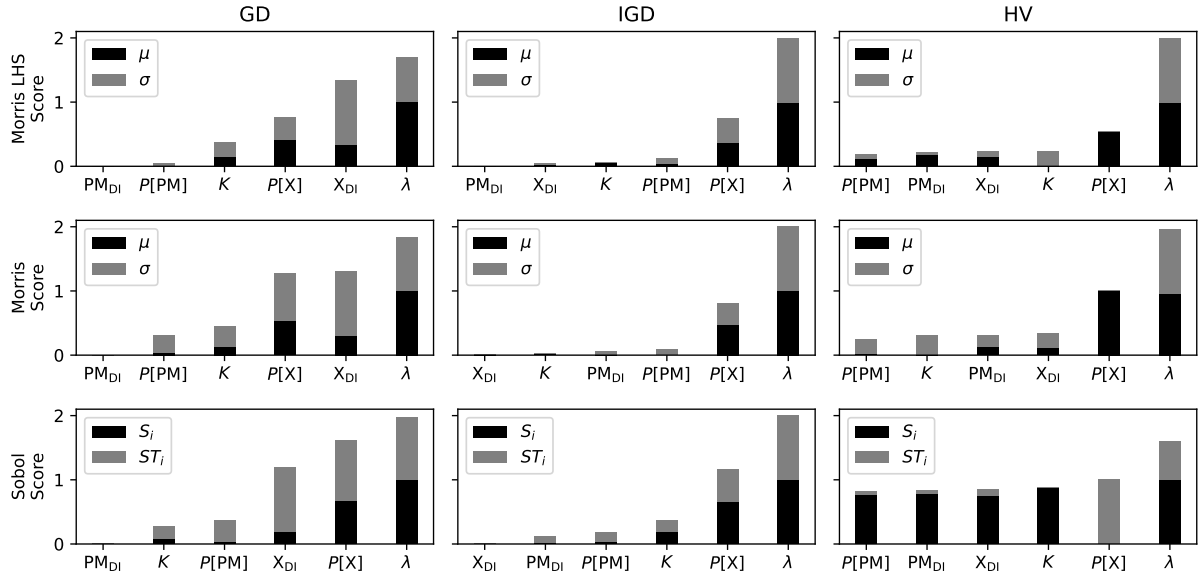

Fig. 7: NSGA-III algorithm's hyperparameters performance across all problems (functions). Rows 1, 2, and 3 respectively show performance evaluated using Morris LHS, Morris, and Sobol methods. Columns 1, 2, and 3 respectively indicate metric GD, IGD, and HV. Darker (black) color portion of a bar is direct influence normalized value in $[0, 1]$ and gray color portion is interaction (total) influence value in $[0, 1]$. Larger height bar imply higher influence. hyperparameters in each subplot is arranged from lower to a higher influencing hyperparameter.

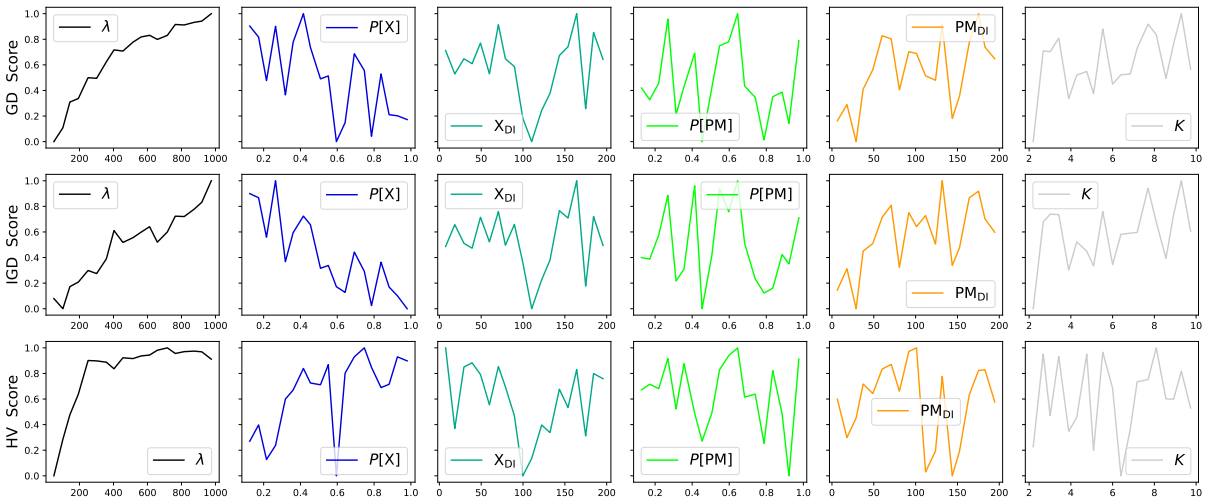

Fig. 8: NSGA-III hyperparameter (x-axis) against mean metric value ((y-axis)). Rows 1, 2, and 3 respectively are GD, IGD, and HV metrics (1 as their best score). Dominance of hyperparameter are plotted using line. CMAES and DE algorithms average performance on 30 runs of sets of hyperparameters. A total xx and xx samples respectively were evaluated for CMAES and DE algorithms for Morris LHS, Morris, Sobol methods. Each dot in a subplot is mean performance of a bin of such samples. Each hyperparameter values arranged in 50 bins (lower values to higher values) across x-axis. Against y-axis best value metric (performance of algorithms) is normalized between a score 0 and 1, where 1 being the best score.

boundary intersection (PBI)" decomposition had the worse scores among MOO decomposition methods. MOEAD hyperparameter ϵ_N about neighbor for selecting percentage of population for sub-problems selection MOEAD has an equivalent influence as the probability of mutation distribution index $\mathrm{PM_{DI}}$.

Crossover and mutation hyperparameters. Genetic operator related hyperparameters P[X], X_{DI}, P[PM] and $\mathrm{PM_{DI}}$ show significance on different measures on different sensitivity methods. For example, the probability of mutation distribution index $\mathrm{PM_{DI}}$ has a high influence on HV measures (pink diamond and eclipse in Fig. 9) and high total influence on HV measures in Sobol method. The probability of mutation $P[PM]$ is second to $\mathrm{PM_{DI}}$ in total influence on HV as per Sobol method. This suggests that *mutation has a high influence in diversifying the population* in MOEAD, helping it produces a better Pareto-front. We also observe that P[PM] and $\mathrm{PM_{DI}}$ have mirror image performance (cf. Fig. 11), which suggest that high values of P[PM] and a smaller value of $\mathrm{PM_{DI}}$ is more effective in MOEAD performance. The probability of crossover P[X] has relatively lowered influence on MOEAD population than mutation related hyperparameter and X_{DI} is the least influential among genetic operator related hyperparameters.

MOEAD hyperparameters ranking. In summary, the *ranking of hyperparameters* of MOEAD from the most influential to least influential hyperparameters is: λ, $Mode$, $\mathrm{PM_{DI}}$, P[PM], P[X], ϵ_N, and X_{DI}.

7 Conclusions

We present a framework for systematic and methodological analysis of the evolutionary algorithm's hyperparameter effectiveness, which results in recommendable rankings. We apply our methodology to state-of-the-art evolutionary algorithms: two widely used single-objective algorithms and two multi-objective algorithms. The single-objective algorithms used are covariance matrix adaptation evolutionary strategy (CMAES), differential evolution (DE), and multi-objective algorithms used are non-dominated sorting genetic algorithm III (NSGA-III), and multi-objective evolutionary algorithm based on decomposition (MOEA/D). Our methodology involves two global sensitivity analysis methods, Morris and Sobol, to analyze tunable hyperparameters of EAs on state-of-the-art testbench systematically. This methodology is computationally heavy, but it produces widely usable and effective recommendations on hyperparameters ranking. That is the order in which one can tune EA hyperparameters to achieve high performance. The initial step size, base vector selection type, probability of crossover, and mode multi-objective problem decomposition were among the most influential hyperparameters of CMAES, DE, NSGA-III, and MOEA/D algorithms, respectively. This supports the research in creating different versions of algorithms like DE and MOEA/D.

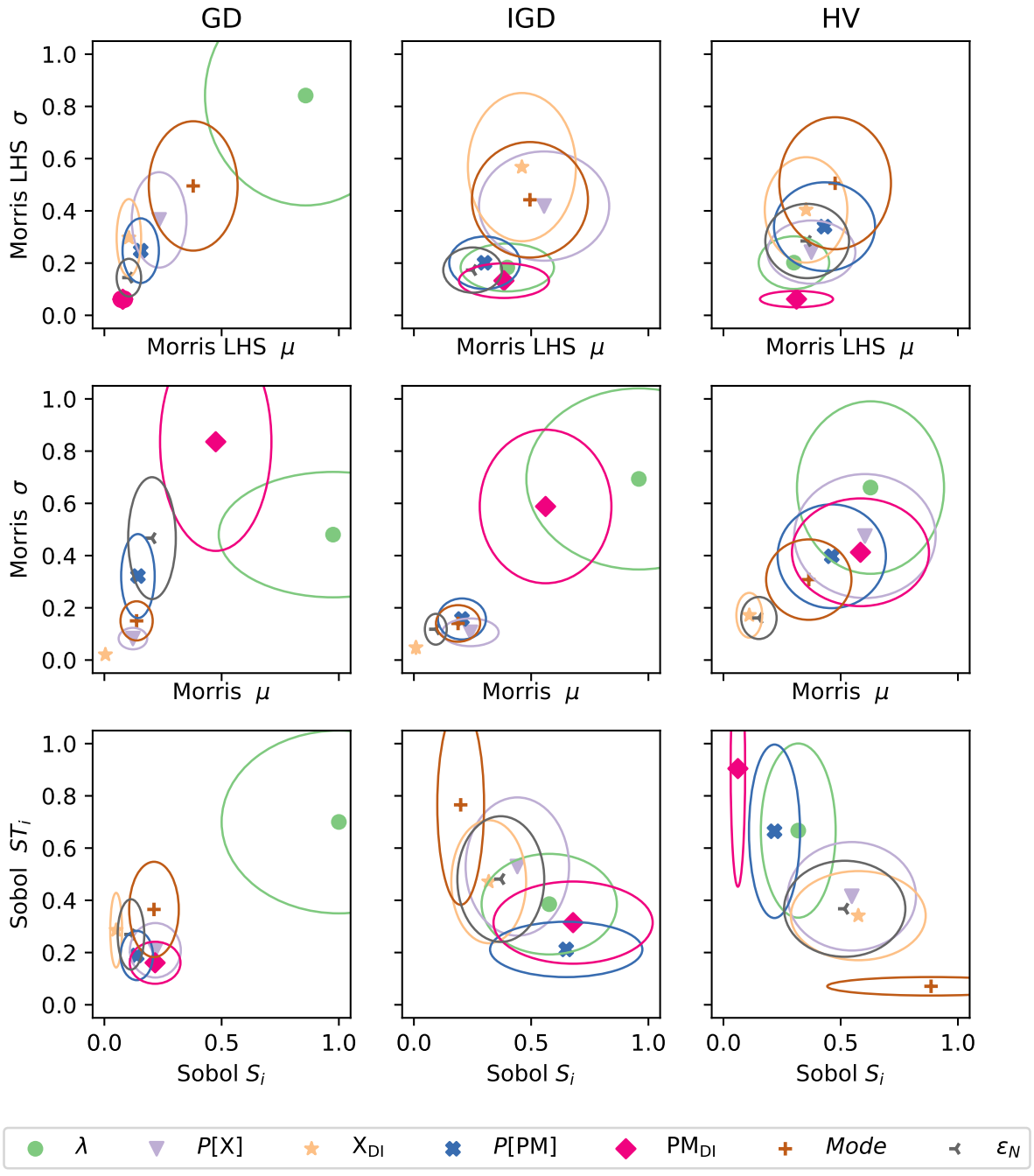

Fig. 9: MOEAD hyperparameters sensitivity analysis. Columns 1, 2, and 3 respectively indicate performance measure metrics GD, IGD, and HV. Rows 1, 2 and 3 respectively indicate Morris LHS, Morris, and Sobol methods. Legends of hyperparameter are shown at the bottom. Each hyperparameter is represented by a symbol and a color. An eclipse centered a hyperparameter is the span of its influence and direction of its influence. Further apart a hyperparameter is in the diagonal direction from the origin $(0, 0)$, the higher is its importance to the algorithm. A larger width of the eclipse of a hyperparameter in the x-axis direction mean more direct dominance of that hyperparameter, and larger height in the y-axis direction mean its dominant total (or interaction) influence.

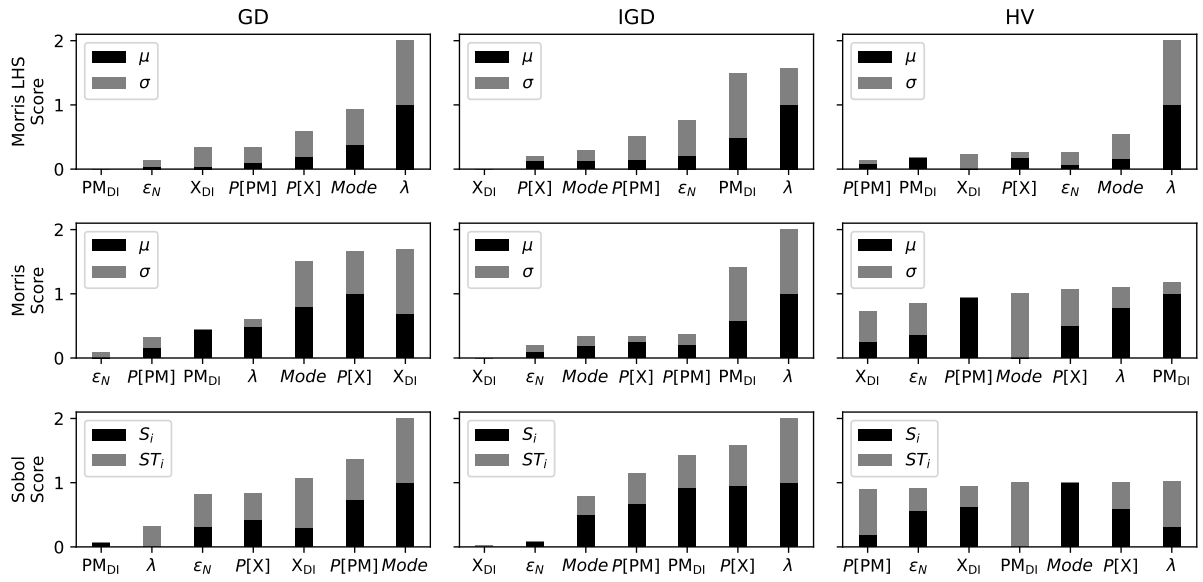

Fig. 10: MOEAD hyperparameter (x-axis) against mean metric value ((y-axis)). Rows 1, 2, and 3 respectively are GD, IGD, and HV metrics (1 as their best score). Dominance of hyperparameter are plotted using line. CMAES and DE algorithms average performance on 30 runs of sets of hyperparameters. A total xx and xx samples respectively were evaluated for CMAES and DE algorithms for Morris LHS, Morris, Sobol methods. Each dot in a subplot is mean performance of a bin of such samples. Each hyperparameter values arranged in 50 bins (lower values to higher values) across x-axis. Against y-axis best value metric (performance of algorithms) is normalized between a score 0 and 1, where 1 being the best score.

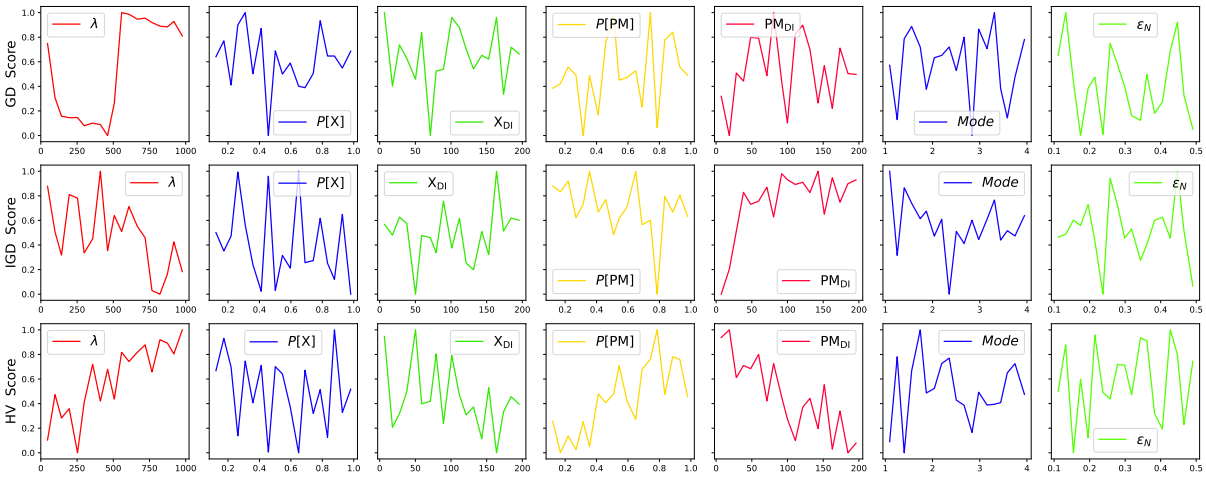

Fig. 11: MOEAD hyperparameter (x-axis) against mean metric value ((y-axis)). Rows 1, 2, and 3 respectively are GD, IGD, and HV metrics (1 as their best score). Dominance of hyperparameter are plotted using line. CMAES and DE algorithms average performance on 30 runs of sets of hyperparameters. A total xx and xx samples respectively were evaluated for CMAES and DE algorithms for Morris LHS, Morris, Sobol methods. Each dot in a subplot is mean performance of a bin of such samples. Each hyperparameter values arranged in 50 bins (lower values to higher values) across x-axis. Against y-axis best value metric (performance of algorithms) is normalized between a score 0 and 1, where 1 being the best score

References

Bergstra, J. and Bengio, Y. (2012), 'Random search for hyper-parameter optimization', *Journal of Machine Learning Research* **13**, 281–305.

Brooks, R., Semenov, M. and Jamieson, P. (2001), 'Simplifying sirius: sensitivity analysis and development of a meta-model for wheat yield prediction', *European Journal of Agronomy* **14**, 43–60.

Campolongo, F., Saltelli, A. and Cariboni, J. (2007), 'An effective screening design for sensitivity analysis of large models', *Environmental Modelling & Software* **22**, 1509–1518.

Conca, P., Stracquadanio, G. and Nicosia, G. (2015), Automatic tuning of algorithms through sensitivity minimization, *in* P. Pardalos, M. Pavone, G. M. Farinella and V. Cutello, eds, 'Machine Learning, Optimization, and Big Data', Springer, pp. 14–25.

Crossley, M., Nisbet, A. and Amos, M. (2013), Quantifying the impact of parameter tuning on nature-inspired algorithms, *in* 'The 12th European Conference on Artificial Life', MIT Press, pp. 925–932.

Das, I. and Dennis, J. E. (1998), 'Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems', *SIAM Journal on Optimization* **8**(3), 631–657.

De Jong, K. (2007), 'Parameter setting in EAs: a 30 year perspective', *Studies in Computational Intelligence (SCI)* **54**, 1–18.

De Jong, K. A., Jansen, T. and Wegener, I. (2004), 'On the choice of offspring population size in evolutionary algorithms', *HT014601767* .

Deb, K., Agrawal, R. B. et al. (1995), 'Simulated binary crossover for continuous search space', *Complex systems* **9**(2), 115–148.

Deb, K. and Deb, D. (2014), 'Analysing mutation schemes for real-parameter genetic algorithms', *International Journal of Artificial Intelligence and Soft Computing* **4**(1), 1–28.

Deb, K. and Jain, H. (2013), 'An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints', *IEEE Transactions on Evolutionary Computation* **18**(4), 577–601.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002), 'A fast and elitist multiobjective genetic algorithm: NSGA-II', *IEEE Transactions on Evolutionary Computation* **6**(2), 182–197.

Deb, K., Thiele, L., Laumanns, M. and Zitzler, E. (2002), Scalable multi-objective optimization test problems, *in* 'Proc. of 2002 IEEE Congress on Evolutionary Computation', Vol. 1, pp. 825–830.

Eiben, A. E., Michalewicz, Z., Schoenauer, M. and Smith, J. E. (2007), Parameter control in evolutionary algorithms, *in* 'Parameter setting in evolutionary algorithms', Springer, pp. 19–46.

Fonseca, C. M., Paquete, L. and López-Ibánez, M. (2006), An improved dimension-sweep algorithm for the hypervolume indicator, *in* 'IEEE International Conference on Evolutionary Computation', IEEE, pp. 1157–1163.

Greco, A., Riccio, S. D., Timmis, J. and Nicosia, G. (2019), Assessing algorithm parameter importance using global sensitivity analysis, *in* I. Kotsireas, P. Pardalos, K. E. Parsopoulos, D. Souravlias and A. Tsokas, eds, 'Analysis of Experimental Algorithms', Springer, pp. 392–407.

Hansen, N. and Ostermeier, A. (1996), Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation, *in* 'Proceedings of IEEE International Conference on Evolutionary Computation', pp. 312–317.

Hansen, N. and Ostermeier, A. (2001), 'Completely derandomized self-adaptation in evolution strategies', *Evolutionary computation* **9**(2), 159–195.

Heris, S. (2019), 'YPEA: Yarpiz evolutionary algorithms'. `https://github.com/smkalami/ypea`. Accessed on 22 September 2021.

Hill, M. C., Kavetski, D., Clark, M., Ye, M., Arabi, M., Lu, D., Foglia, L. and Mehl, S. (2016), 'Practical use of computationally frugal model analysis methods', *Groundwater* **54**(2), 159–170.

Huband, S., Hingston, P., Barone, L. and While, L. (2006), 'A review of multiobjective test problems and a scalable test problem toolkit', *IEEE Transactions on Evolutionary Computation* **10**(5), 477–506.

Iglesias, P. L., Mora, D., Martinez, F. J. and Fuertes, V. S. (2007), 'Study of sensitivity of the parameters of a genetic algorithm for design of water distribution networks', *Journal of Urban and Environmental Engineering* **1**(2), 61–69.

Iooss, B. and Saltelli, A. (2016), Introduction to sensitivity analysis, *in* R. Ghanem, D. Higdon and H. Owhadi, eds, 'Handbook of Uncertainty Quantification', Springer, pp. 1–20.

Kramer, O. (2010), 'Evolutionary self-adaptation: a survey of operators and strategy parameters', *Evolutionary Intelligence* **3**(2), 51–65.

Lima, C. F. and Lobo, F. G. (2004), Parameter-less optimization with the extended compact genetic algorithm and iterated local search, *in* 'Genetic and Evolutionary Computation Conference', Springer, pp. 1328–1339.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M. and Stützle, T. (2016), 'The irace package: Iterated racing for automatic algorithm configuration', *Operations Research Perspectives* **3**, 43–58.

Maturana, J., Lardeux, F. and Saubion, F. (2010), 'Autonomous operator management for evolutionary algorithms', *Journal of Heuristics* **16**(6), 881–909.

Miettinen, K. (2012), *Nonlinear multiobjective optimization*, Vol. 12, Springer Science & Business Media.

Morris, M. D. (1991), 'Factorial sampling plans for preliminary computational experiments', *Technometrics* **33**(2), 161–174.

Moschitti, A. (2003), A study on optimal parameter tuning for rocchio text classifier, *in* 'European Conference on Information Retrieval', Springer, pp. 420–435.

Ojha, V., Greco, A., Timmis, J. and Nicosia, G. (2021), 'Sensitivity analysis evolutionary algorithms'. `https://github.com/vojha-code/SAofEAs`. Accessed 22 September 2021.

Paul, G., Müller, C. L. and Sbalzarini, I. F. (2011), Sensitivity analysis from evolutionary algorithm search paths, *in* 'EVOLVE - A bridge between Probability, Set Oriented Numerics and Evolutionary Computation', Studies in Computational Intelligence, Springer.

Pianosi, F., Sarrazin, F. and Wagener, T. (2015), 'An effective screening design for sensitivity analysis of large models', *Environmental Modelling & Software* **70**, 80–85.

Pinel, F., Danoy, G. and Bouvry, P. (2012), Evolutionary algorithm parameter tuning with sensitivity analysis, *in* P. Bouvry, M. A. Kłopotek, F. Leprévost, M. Marciniak, A. Mykowiecka and H. Rybiński, eds, 'Security and Intelligent Information Systems', Springer, pp. 204–216.

Saltelli, A. (2002), 'Sensitivity analysis for importance assessment', *Risk Analysis* **22**(3), 579–590.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M. and Tarantola, S. (2008), *Global Sensitivity Analysis: The primer*, John Wiley & Sons.

Saltelli, A., Tarantola, S., Campolongo, F. and Ratto, M. (2004), *Sensitivity analysis in practice: a guide to assessing scientific models*, Vol. 1, Wiley.

Saltelli, A., Tarantola, S. and Chan, K.-S. (1999), 'A quantitative model-independent method for global sensitivity analysis of model output', *Technometrics* **41**(1), 39–56.

Sobol, I. M. and Kucherenko, S. (2005), 'Global sensitivity indices for nonlinear mathematical models. review', *Wilmott Magazine* **2005**, 56–61.

Storn, R. and Price, K. (1997), 'Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces', *Journal of Global Optimization* **11**, 341–359.

Tian, Y., Cheng, R., Zhang, X. and Jin, Y. (2017), 'PlatEMO: A MATLAB platform for evolutionary multi-objective optimization', *IEEE Computational Intelligence Magazine* **12**(4), 73–87.

Veldhuizen, D. A. V. (1999), Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations, PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

Veldhuizen, D. A. V. and Lamont, G. B. (1998), Evolutionary computation and convergence to a pareto front, *in* 'Late Breaking Papers on the Genetic Programmming 1998 Conference', pp. 221–228.

Yao, X., Liu, Y. and Lin, G. (1999), 'Evolutionary programming made faster', *IEEE Transactions on Evolutionary Computation* **3**(2), 82–102.

Zhang, Q. and Li, H. (2007), 'Moea/d: A multiobjective evolutionary algorithm based on decomposition', *IEEE Transactions on Evolutionary Computation* **11**(6), 712–731.

Zitzler, E. and Thiele, L. (1998), Multiobjective optimization using evolutionary algorithms—a comparative case study, *in* 'International Conference on Parallel Problem Solving from Nature', pp. 292–301.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M. and Da Fonseca, V. G. (2003), 'Performance assessment of multiobjective optimizers: An analysis and review', *IEEE Transactions on Evolutionary Computation* **7**(2), 117–132.