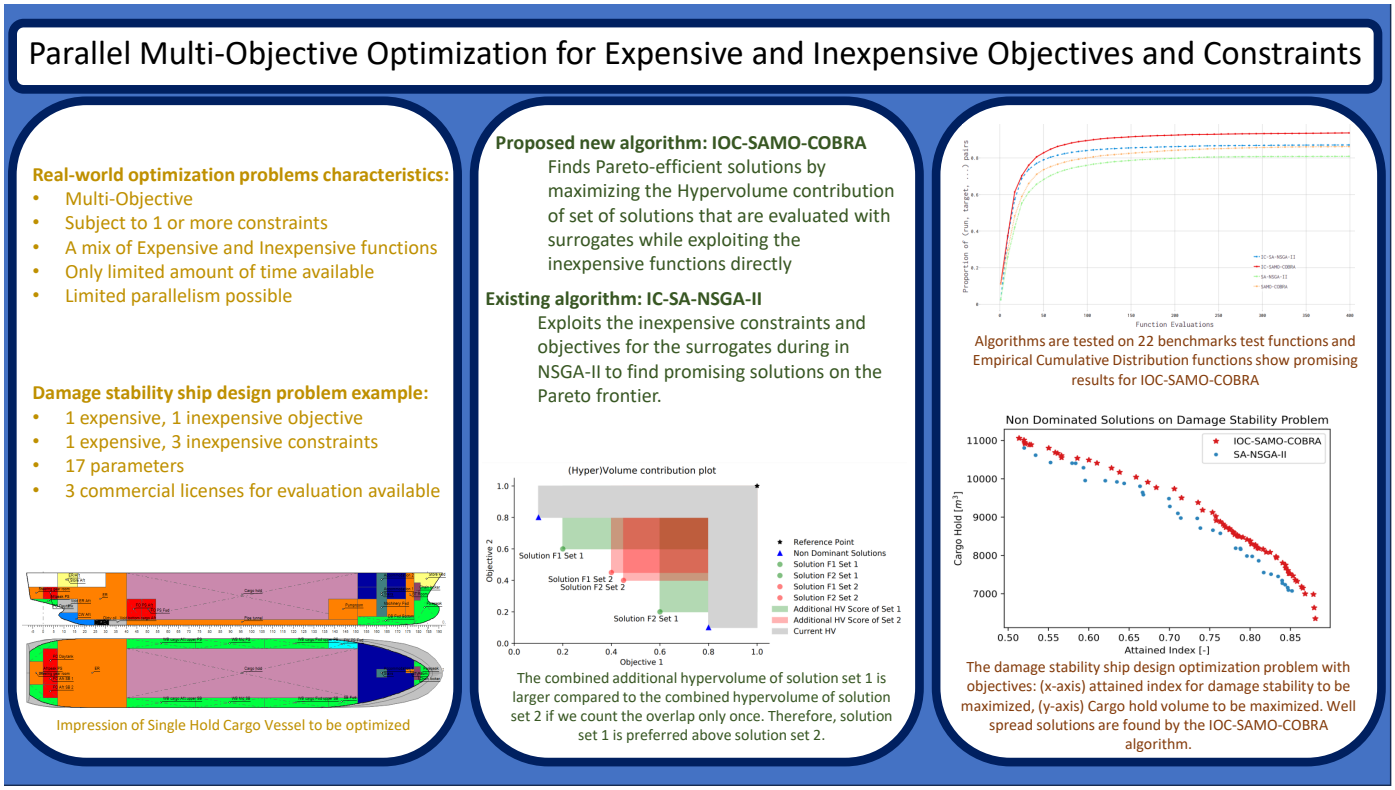
Graphical Abstract

Parallel Multi-Objective Optimization for Expensive and Inexpensive Objectives and Constraints

Roy de Winter, Bas Milatz, Julian Blank, Niki van Stein, Thomas Bäck, Kalyanmoy Deb



Highlights

Parallel Multi-Objective Optimization for Expensive and Inexpensive Objectives and Constraints

Roy de Winter, Bas Milatz, Julian Blank, Niki van Stein, Thomas Bäck, Kalyanmoy Deb

- Self-adaptive optimization algorithm for Parallel Constraint Multi-Objective problems.
- Optimization algorithm for a mix of Expensive and Inexpensive functions.
- Direct use of Inexpensive functions in the optimization algorithm.
- Extensive comparison between IC-SA-NSGA-II and IOC-SAMO-COBRA optimization algorithm.
- Real-world cargo vessel design has been optimized with IOC-SAMO-COBRA.

Parallel Multi-Objective Optimization for Expensive and Inexpensive Objectives and Constraints

Roy de Winter[a,b,*], Bas Milatz[b], Julian Blank[c], Niki van Stein[a], Thomas Bäck[a], Kalyanmoy Deb[c]

[a]*Leiden Institute of Advanced Computer Science, Leiden University, Niels Bohrweg 1, Leiden, 2333 CA, Netherlands*
[b]*Research and Development, C-Job Naval Architects, Regulusplein 1, Hoofddorp, 2132 JN, Netherlands*
[c]*Electrical and Computer Engineering, Michigan State University, 474 S. Shaw Lane, East Lansing, 48824, Michigan, United States*

Abstract

Expensive objectives and constraints are key characteristics of real-world multi-objective optimization problems. In practice, they often occur jointly with inexpensive objectives and constraints. This paper presents the *Inexpensive Objectives and Constraints Self-Adapting Multi-Objective Constraint Optimization algorithm that uses Radial Basis function Approximations* (IOC-SAMO-COBRA) for such problems. This is motivated by the recently proposed Inexpensive Constraint Surrogate-Assisted Non-dominated Sorting Genetic Algorithm II (IC-SA-NSGA-II). These algorithms and their counterparts that do not explicitly differentiate between expensive and inexpensive objectives and constraints are compared on 22 widely used test functions. The IOC-SAMO-COBRA algorithm finds significantly better (identical/worse) Pareto fronts in at least 78% (6%/16%) of all test problems compared to IC-SA-NSGA-II measured with both the hypervolume and Inverted Generational Distance+ performance metric. The empirical cumulative distribution functions confirm this advantage for both algorithm variants that exploit the inexpensive constraints. In addition, the proposed method is compared against state-of-the-art practices on a real-world cargo vessel design problem. On this 17-dimensional two-objective practical problem, the proposed IOC-SAMO-COBRA outperforms SA-NSGA-II as well. From an algorithmic perspective, the comparison identifies specific strengths of both approaches and indicates how they should be hybridized to combine their best components.

Keywords: Multi-Objective Optimization, Constraint Optimization, Surrogate-assisted Optimization, Parallel Computing, Ship Design.

1. Introduction

Real-world problems are often defined through multiple objectives and constraints, combined with the fact that objectives or constraints can be time-consuming ("expensive") to evaluate [1, 2, 3]. In the continuous domain, a constrained multi-objective problem with k objectives and m inequality constraints can be formulated as follows [4]:

$$\begin{aligned} &\text{minimize: } \mathbf{f} : \Omega \to \mathbb{R}^k \ , \ \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))^\top \\ &\text{subject to: } g_i(\mathbf{x}) \leq 0 \ \forall i \in \{1, \ldots, m\} \\ &\qquad \mathbf{x} \in \Omega \subset \mathbb{R}^d. \end{aligned}$$

Equality constraints are not considered, as they can be replaced with two inequality constraints without resorting to any special equality constraint handling method. Typical examples of optimization problems include design optimization problems from industries, such as automotive [5], aviation [6, 7], and maritime engineering [8, 9], in which (commercial licenses of) finite element simulation and computational fluid dynamic tools are used for computing the performance characteristics of a design. These third-party softwares are computationally expensive to run, thereby increasing the overall duration of the optimization process. This leads to a very limited amount of allowed solution evaluations for the optimization algorithms.

Assuming that, at a maximum, a few hundred simulation runs are possible (i.e., solution evaluations of objective and constraint functions), the goal becomes to approximate the true Pareto front of feasible solutions as close as possible with the given limited budget. To decrease the wall-clock-time, solution evaluations can be run in parallel. Provided that p simulator licenses are available, the total elapsed time can in theory be reduced by a factor of p [10]. Consequently, the *research challenge is to develop a parallel algorithm for multi-objective constrained optimization that can yield a high-quality set of feasible and Pareto-optimal solutions by using only a small number of solution evaluations*.

Candidate algorithm classes include multi-objective variants of evolutionary algorithms [11] and of Bayesian optimization [12]. In general, the former offers naturally built-in parallelism while typically requiring more function evaluations and the latter is more efficient in terms of function evaluations while typically not allowing for parallelism. Bayesian optimization can also use different *acquisition functions* (often also called *infill criterion*) on the surrogate model for searching a solution

*Corresponding author at: Leiden Institute of Advanced Computer Science, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, Netherlands
Email address: r.de.winter@liacs.leidenuniv.nl (Roy de Winter)

candidate.

However, researchers have extended evolutionary algorithms by using surrogate models trained on the evaluated search points to allow for a fast prediction of objective and constraint function values for new candidate solutions (infill points), making them more efficient while keeping the benefits of parallelism [13]. A state-of-the-art algorithm from this class is the *Surrogate-assisted Non-dominated Sorting Genetic Algorithm* (SA-NSGA-II) [14], which uses radial basis functions [15] as surrogate models for *all objectives and all constraints*. Its most recent *Inexpensive Constraint* extension (IC-SA-NSGA-II [14]) uses radial basis function surrogates *only for the objectives* and assumes that all constraints are inexpensive to evaluate.

Conversely, Bayesian optimization has been extended with acquisition functions which can be used to propose multiple Pareto-optimal solutions per iteration, which opens the door to parallelism. A state-of-the-art algorithm from this class is the *Self-Adaptive Algorithm for Multi-Objective Constraint Optimization by using Radial Basis Function Approximations* (SAMO-COBRA) [16, 17] in combination with the recently introduced multi-point acquisition function [18]. Like SA-NSGA-II, SAMO-COBRA learns radial basis function approximations for *all objectives and all constraints*. In this sense, the original versions of these two algorithms were designed with the purpose of modeling and optimizing surrogates of all objectives and constraints. However, there is a fundamental difference between the working of these two algorithms. While SA-NSGA-II and IC-SA-NSGA-II use a genetic algorithm's operators to create new candidate solutions, SAMO-COBRA uses a local search-based hypervolume maximization approach for creating new candidate solutions.

While both are addressing the same research challenge stated above, including the use of radial basis function approximations, a thorough comparison of the strengths and weaknesses of these algorithms has not been presented yet. The *first contribution* of this paper is to fill this gap, based on an empirical comparison of these algorithms on a set of well-known multi-objective constrained test functions which are representative of real-world problems.

To facilitate a complete experimental comparison, we also propose, as the *second contribution* of this paper, a SAMO-COBRA variant that is inspired by IC-SA-NSGA-II's approach to differentiate between inexpensive constraints and expensive objectives, but generalizes this approach. The proposed *Inexpensive Objectives and Constraints*-SAMO-COBRA (IOC-SAMO-COBRA) allows the user to identify the expensive objectives and constraints, for which IOC-SAMO-COBRA will then use radial basis function surrogates, while it will use the inexpensive objectives and inexpensive constraints directly. A tabular overview of these four different algorithms and how they deal with expensive and inexpensive objectives and constraints is given in Table 1.

As a *third contribution*, going beyond the use of test problems, SA-NSGA-II and IOC-SAMO-COBRA are also compared on a real-world two-objective constraint ship design optimization problem involving 17 variables, one expensive (simulation-based) objective, one inexpensive objective, one expensive (simulation-based) constraint, and three inexpensive constraints[1].

Table 1: Overview of how the four algorithms deal with the (in)expensiveness of constraints and objectives. "Surrogate" means a surrogate replaces the objective/constraint, direct means that the objective/constraint is used without learning a surrogate for it.

Algorithm	Expensive constraints	Inexpensive constraints	Expensive objectives	Inexpensive objectives
SA-NSGA-II	surrogate	surrogate	surrogate	surrogate
IC-SA-NSGA-II	direct	direct	surrogate	surrogate
SAMO-COBRA	surrogate	surrogate	surrogate	surrogate
IOC-SAMO-COBRA	surrogate	direct	surrogate	direct

The remainder of this paper is organized as follows: First, the preliminaries are given in Section 2. Section 3 discusses related work, and Section 4 describes how the use of inexpensive objectives and constraints is added to the SAMO-COBRA algorithm. Section 5 then describes the experimental approach. Section 6 presents and discusses the results of the experiments, and in Section 7 the algorithms are compared on the ship design optimization problem. Conclusions regarding the performance of the algorithms are presented in Section 8.

2. Preliminaries

In the preliminaries the groundwork is laid for surrogate assisted multi-objective optimization by giving the definition of a Feasible Pareto Efficient Solution, introducing the two most used multi-objective performance metrics, describing well-known design of experiments strategies, and introducing Radial Basis Functions as surrogates.

2.1. Feasible Pareto Efficient Solution

In constrained multi-objective optimization the goal is to find solutions that are feasible according to the constraints and simultaneously Pareto-optimal in the objective space. A mathematical description of a feasible Pareto-optimal solution is given in Def. 1.

Definition 1. (Feasible Pareto-optimal solution [16])*: $\mathbf{x} \in \Omega$ is called* feasible Pareto-optimal *with respect to Ω and $g_i(\mathbf{x}) \leq 0 \ \forall i \in \{1, \ldots, m\}$, if and only if there is no solution $\mathbf{x}'$ for which $\mathbf{v} = f(\mathbf{x}') = (f_1(\mathbf{x}'), \ldots, f_k(\mathbf{x}'))^\top$ dominates $\mathbf{u} = f(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))^\top$ where $g_i(\mathbf{x}) \leq 0$ and $g_i(\mathbf{x}') \leq 0 \ \forall i \in \{1, \ldots, m\}$.*

2.2. Multi-objective Performance Metrics

The two commonly used multi-objective performance metrics considered in this work are the hypervolume (HV), and the Inverted Generational Distance+ (IGD+) metric. The *hypervolume* (also known as the Lebesgue measure) translates the multi-objective problem into a unary performance score that represents the volume of the region in the objective space that is dominated by a given set of solutions [19, 20]. It is the most used performance metric in multi-objective optimization [21]

[1]The problem was provided by C-Job Naval Architects, Netherlands.

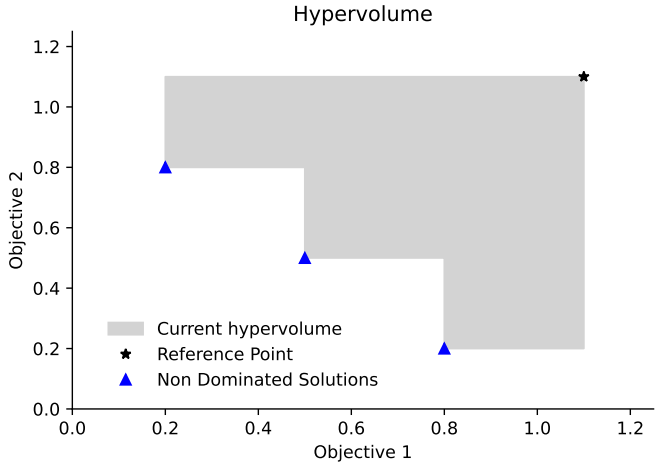


Figure 1: Visual representation of hypervolume. Total hypervolume between the 3 solutions and the reference point is $0.3 \times 0.8 + 0.3 \times 0.5 + 0.3 \times 0.3 = 0.48$.

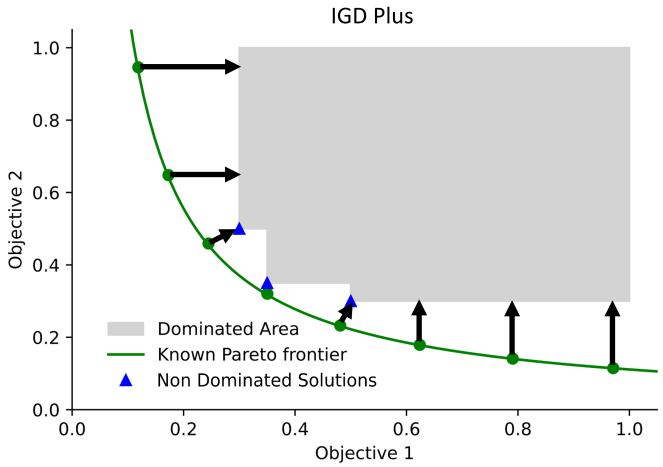


Figure 2: IGD+ score visualized on a two objective problem. IGD+ is the average length of the arrows from the well-spread known solutions to the closest point of the dominated area.

and it measures and captures the overall convergence and diversity of the set of solutions that form the Pareto front. The HV is calculated by determining the volume of the region in the objective space between the solutions on the obtained Pareto front and a pre-defined reference point (also sometimes referred to as the anti-optimal point [22]). The HV is a useful measure for comparing the performance of different optimization algorithms, as well as for comparing different solution sets with each other. Solution sets with higher HV are considered better compared to solutions with lower HV. The HV of three solutions is visualized in Figure 1, where the triangles represent the evaluated non-dominated solutions, and the reference point is indicated by a star. The grey area is the HV of this particular Pareto front.

Another performance metric often used in multi-objective optimization is the *inverted generational distance+* metric (IGD+) [23]. The IGD+ metric evaluates diversity and convergence as follows:

$$IGD^+(A, S) = \frac{1}{|S|}\left(\sum_{i=1}^{|S|}(d_i^+)^2\right)^{\frac{1}{2}} \quad (1)$$

Here S is the known Pareto front, A is the dominated area by a Pareto front obtained by an algorithm, and d_i^+ is the smallest Euclidean distance from a solution on the known Pareto front s_i to the dominated area A. This way, if the obtained dominated area of the solutions found by the algorithm is far away from the known Pareto front, the IGD+ value increases. A smaller IGD+ value is therefore preferred over a larger IGD+ value. The IGD+ metric and the distances of 8 known solutions are visualized in Figure 2.

The IGD+ metric can only be used on test instances where the Pareto front is known. In this work, the Pareto front used for the IGD+ metric is approximated by combining all obtained feasible Pareto efficient points of all experiments, then normalizing the objective scores, and finally selecting a well-spread set of solutions. The true Pareto front of computationally expensive engineering problems are often impossible to determine, therefore this revised metric is only used on test problems.

2.3. Design of Experiments

Typically, in surrogate-assisted optimization, or black box optimization, the algorithm starts with an initial design of experiments (DoE). In this design of experiments, the first set of solutions is generated and evaluated. The location where these first solutions are placed in the input space (also sometimes referred to as initial sampling) is dependent on the DoE strategy. Several possible choices for a DoE exist, including uniform random sampling, full factorial design [24], Latin Hypercube Sampling [25], Halton sampling [26], and Sobol sampling [27]. Each of these methods has its own strengths, however, a large empirical comparison was presented by Bossek et. al. [28]. This empirical study concludes that spending as few evaluations on the DoE as possible is often beneficial because this leaves more room for evaluations proposed by the optimization algorithm [28]. This empirical finding is also shown to work best for most constraint multi-objective problems [17].

A recently proposed new sampling method is the Riesz s-energy based sampling method [29]. The Riesz s-energy-based sampling method iteratively improves and proposes an arbitrary number of well-spaced points in the design space [30]. This method has been modified for constrained search spaces [14] so that it samples solutions only in the feasible area of the design space. This however is only practically applicable if the constraint functions are inexpensive to evaluate.

In industrial settings, the p solutions within the initial DoE can often be evaluated in parallel (depending on the available computational resources and, if applicable, the number of commercial simulator licenses available). In this case, it can be advised to choose a DoE size of $\lceil DoE_{min}/p \rceil \cdot p$, where p is the maximum possible number of simultaneous parallel evaluations and DoE_{min} the smallest DoE size required for training the first surrogate models.

2.4. Surrogate Models

For the algorithm proposed in this research Radial Basis Functions (RBFs) and Kriging (also known as Gaussian process regression) are considered surrogate models. RBF and Kriging surrogates are fundamentally very similar, however, RBFs have many advantages: 1) RBFs require smaller initial sample sizes to fit a surrogate, 2) they are computationally cheaper (also for larger input spaces), 3) have fewer assumptions on the underlying data, 4) deliver in many cases equal or better accuracy, 5) and with a newly developed uncertainty quantification method RBFs can now also be used in infill criteria that require this [31, 32]. Besides these fundamental arguments, an empirical comparison in earlier work showed faster convergence and better results for algorithms with RBF surrogates compared to the algorithm with Kriging surrogates [17]. For these reasons, the algorithm proposed in this work uses RBFs as surrogate models.

Radial Basis Functions (RBFs) are a type of mathematical function used for approximating the relationship between input and output variables [33]. In surrogate-assisted optimization, the input variables are often the decision variables ($\mathbf{x}$), and the output variables are the objective ($\mathbf{f}$) or constraint ($\mathbf{g}$) value of the evaluated solutions. The relationship is learned by fitting a weighted combination of RBFs $\varphi(\|\mathbf{x} - \mathbf{c}\|)$ [34]. The RBFs used in this work are: $\varphi = \{Cubic, Gaussian, Multiquadric, InverseQuadratic, InverseMultiquadric, ThinPlateSpline\}$. The optimal weights $\boldsymbol{\theta}$ for the RBFs make sure that all evaluated function values $\mathbf{f}$ are exactly interpolated. Finding these optimal weights is done by inverting $\boldsymbol{\Phi} \in \mathbb{R}^{n \times n}$ where n is the number of samples and $\boldsymbol{\Phi}_{i,j} = \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|)$:

$$\boldsymbol{\theta} = \boldsymbol{\Phi}^{-1} \times \mathbf{f} \tag{2}$$

To enhance the RBF fit, a linear or polynomial tail can be added [35]. Further improvements can be achieved by scaling and standardization of the input and output space. In some cases, a Plog transformation of the objective or constraint values can be beneficial [34, 17].

$$\text{Plog}(y) = \begin{cases} +\ln(1+y), & \text{if} \quad y \geq 0 \\ -\ln(1-y), & \text{if} \quad y < 0 \end{cases} \tag{3}$$

3. Related Work

There is a growing interest in surrogate-assisted optimization [36, 37], surrogate-assisted constraint optimization [38], surrogate-assisted optimization in combination with parallelism [10], surrogate-assisted multi-objective optimization [39], and problems with heterogeneous evaluation times [40]. Different approaches have been developed for solving constrained multi-objective problems. Several algorithms have been published that can deal with a subset of the above-mentioned characteristics:

- cK-RVEA is a many-objective reference vector-guided evolutionary algorithm that uses Kriging models as surrogates for the objectives and deals with the constraints by only using the feasible solutions for surrogate training [41].
- SBMO, a multi-objective algorithm that uses Kriging models as a surrogate for both the constraints and objectives. Because of the scalarization of the objectives, it can propose multiple solutions per iteration [42].
- HSMEA, a many-objective optimization algorithm that uses Kriging, polynomial response surface modeling, and RBFs as surrogates for the objectives and constraints [43].
- GP-CMOEA, a multi-objective optimization algorithm that uses Gaussian process regression for the objectives while it exploits the inexpensiveness of constraints to find feasible Pareto-optimal solutions [44].
- EGMOCO, a constraint multi-objective optimization algorithm that uses Kriging as a surrogate and exploits four different acquisition functions to propose multiple feasible Pareto-optimal solutions per iteration [7].
- SCCMA, a constraint multi-objective optimization algorithm that uses decomposition of the problem in combination with co-evolution and local search on the surrogates to find Pareto-optimal solutions [45].
- KTS, a kriging-assisted multi-objective evolutionary algorithm that uses a strategy to switch between unconstrained optimization and constraint optimization search modes. Furthermore, it exploits the correlation between constraints and objectives and a special selection strategy for samples that are used in training the Kriging surrogates [46].

However, very little research has been done on surrogate-assisted algorithms that can deal with a mix of both expensive and inexpensive constraints and objective functions. It is therefore that in this work we propose a parallel constraint multi-objective optimization algorithm capable of dealing with mixed expensiveness of objective and constraint functions.

In the following subsections, the closely related relevant methods (SA-NSGA-II, IC-SAMO-COBRA, and SAMO-COBRA with the multi-point acquisition function) that are used as reference algorithms are described in more detail.

3.1. SA-NSGA-II

A variant of NSGA-II [47], called Surrogate-Assisted NSGA-II (SA-NSGA-II)[2], integrates surrogate assistance into the optimization cycle for the optimization of unconstrained and constrained multi-objective optimization problems. The surrogates employed in the SA-NSGA-II algorithm are RBFs with a *Cubic* kernel and a linear tail. One independent RBF surrogate is trained for each objective and constraint. The algorithm executes the NSGA-II optimization algorithm for 20 generations with a population size of 100 only on the surrogate

[2] Availabe on pysamoo as SSA-NSGA-II [48].

models before calling the expensive optimization function. This embedded surrogate-based optimization loop provides a set of candidate solutions, from which a subset is selected. Assuming p solutions shall be evaluated using the expensive simulation in each optimization cycle, the candidates are first separated into p clusters based on their objective space values before determining the selected solution for each cluster by performing a roulette wheel selection based on their crowding distances. After evaluating these p solutions on the expensive function, all surrogate models are updated and the new optimization cycle is started if the solution evaluation budget is not exhausted yet. SA-NSGA-II can also optimize constrained optimization problems by using the parameter-less domination approach [49] used in NSGA-II's selection operators.

3.2. IC-SA-NSGA-II

Later, an extension of SA-NSGA-II has been proposed to address optimization problems where objectives are computationally expensive, but the constraints are not [14]. For such problems, the optimization method shall exploit the asymmetry of expensiveness, or in other words, the fact that one can collect significantly more information regarding the feasibility of a solution before having to run an expensive simulation. The novelty of the proposed method is the constrained sampling for finding feasible designs in the first optimization cycle. The challenge of finding a feasible yet diverse set of solutions is addressed by incorporating a Riesz s-energy [29] based sampling method [30] modified for constrained search spaces. Furthermore, to make IC-SA-NSGA-II more efficient for the optimization of highly constrained problems (still with inexpensive constraints), the embedded surrogate-based optimization loop has been extended by a repair operator applied to each solution after mating [50]. The repair operator ensures that only feasible solutions are evaluated (on the surrogates and on the expensive functions) and has demonstrated to be effective for problems with complex constraints – commonly occurring in engineering problems such as electric machine design. Moreover, a proof-of-principle study investigating the optimization of heterogeneously expensive objective and constraint functions has been proposed [51].

The pseudo-code of the IC-SA-NSGA-II algorithm can be found in Algorithm 1. A more extensive explanation of the IC-SA-NSGA-II algorithm is given in [14].

3.3. SAMO-COBRA with Multi-Point Acquisition Function

SAMO-COBRA [16, 17] is a Bayesian optimization algorithm specifically designed to solve expensive constraint multi-objective problems with continuous decision parameters. As an extension of its single objective predecessor SACOBRA [34], SAMO-COBRA has been developed to find a Pareto front approximation with as few function evaluations as possible. It does so by optimizing the HV between a user-defined reference point and all solutions that form the Pareto front.

SAMO-COBRA starts with an as small as possible [28] initial Halton sample [26] as a DoE. Every solution in the DoE is evaluated with all the constraint and objective functions. When the evaluation of the DoE is finished, the algorithm starts learning from the evaluated solutions. This learning is done by fitting RBFs for the objectives and the constraints. The RBFs employed in SAMO-COBRA are all RBFs φ from Section 2.4 with a polynomial tail. Besides the RBFs kernel options, a PLOG transformation is applied where this transformation is beneficial. This results in $6 \times 2 = 12$ options to choose from: $\Phi = \{Cubic, Gaussian, Multiquadric, InverseQuadratic, InverseMultiquadric, ThinPlateSpline\} \times \{\text{PLOG}, standardized\}$. Choosing the best kernel and transformation is done based on selecting the kernel and transformation which had the smallest approximation error in all previous evaluations. After the optimal RBF strategy is chosen, the algorithm starts to search for solutions that form the Pareto front with adaptive sampling steps [10].

The acquisition function used to obtain multiple solutions per iteration which satisfy all constraints and which simultaneously optimize the HV is the multi-point HV improvement acquisition function for constrained multi-objective problems [18]. This purely exploitative acquisition function is optimized with the Constraint Optimization BY Linear Approximations algorithm (COBYLA) [52]. The candidate solutions that are predicted to jointly contribute the largest HV are evaluated with the expensive objective and constraint functions. After evaluation of the solutions, the RBF approximation error is computed, the best RBF kernel and transformation method are selected, the RBF surrogates are updated, and the acquisition function is optimized again. This process continues until a user-defined maximum number of function evaluations has been reached.

A more extensive explanation of the SAMO-COBRA algorithm with the multi-point acquisition function is given in [17, 18].

4. Proposed Framework

In the original SAMO-COBRA algorithm for every objective and constraint function, an RBF surrogate is used during the search for new candidate solutions. In the IOC-SAMO-COBRA extension, one or more of the RBFs can be replaced with the real inexpensive constraint or objective function. Instead of finding good solutions on the RBFs, in IOC-SAMO-COBRA the inexpensive constraints and objectives are used directly during the search for feasible Pareto efficient solutions that contribute HV to the Pareto front. The direct use of inexpensive functions can be beneficial because the real functions do not make approximation mistakes like RBF surrogates do in unseen regions. This should, especially in the early iterations, lead to better results compared to the use of RBFs since in early iterations the RBF approximation error might still be large. Besides in the early iterations, the inexpensive constraints can also be exploited when finding the Pareto fronts of optimization problems with very few feasible solutions. The pseudocode of the IOC-SAMO-COBRA algorithm is given in Algorithm. 2.

Algorithm 1: IC-SA-NSGA-II. (Adapted from [14].)
Input: Number of decision variables d, expensive objective functions $\mathbf{f}(\mathbf{x})$, constraint function(s) $\mathbf{g}(\mathbf{x})$, maximum number of solution evaluations N_{max}, number of initial samples N_{init}, number of exploration points N_{explr}, number of exploitation points $N_{exploit}$, number of generations for NSGA-II exploitation N_{gen}, offspring multiplier s for exploration.
Output: Evaluated solutions.

1 **Function** IC-SA-NSGA-II(d, $\mathbf{f}$, $\mathbf{g}$, N_{init}, N_{max}, $N_{exploit}$, N_{explr}, N_{gen}, s):
2 $\quad\mathbf{X} \leftarrow$ S-EnergySampling($d, N_{init}, \mathbf{g}$) ▷ initialize feasible solutions using the constraint functions $\mathbf{g}$
3 $\quad\mathbf{F} \leftarrow \mathbf{f}(\mathbf{X})$ ▷ Evaluate objective functions
4 $\quad\mathbf{G} \leftarrow \mathbf{g}(\mathbf{X})$ ▷ Evaluate constraint functions
5 $\quad j \leftarrow N_{init}$ ▷ Initialize expensive evaluation counter
6 $\quad$**while** $j < N_{max}$ **do**
7 $\quad\quad\mathbf{S} \leftarrow \{\text{FitRBF}(\mathbf{X}, f, Cubic, rescaled) \mid \forall\ f \in \mathbf{f}\}$ ▷ Fit *Cubic* RBF for all expensive objectives
8 $\quad\quad\left(\mathbf{X}^{\text{cand}}, \mathbf{F}^{\text{cand}}\right) \leftarrow$ NSGA-II($\mathbf{S}, \mathbf{g}, \mathbf{X}, \mathbf{F}, N_{gen}$) ▷ Run NSGA-II on RBFs and constraints
9 $\quad\quad\left(\mathbf{X}^{\text{cand}}, \mathbf{F}^{\text{cand}}\right) \leftarrow$ EliminateDuplicates($\mathbf{X}, \mathbf{X}^{\text{cand}}$) ▷ Delete duplicates from input candidates
10 $\quad\quad C \leftarrow$ Kmeans($N_{exploit}, \mathbf{F}^{\text{cand}}$) ▷ Make $N_{exploit}$ clusters from candidate solutions with Kmeans
11 $\quad\quad\mathbf{X}^{\text{exploit}} \leftarrow$ RankingSelection($\mathbf{X}^{\text{cand}}, C,$ Crowding($\mathbf{F}^{\text{cand}}$)) ▷ Keep one solution per cluster
12 $\quad\quad\mathbf{X}', \mathbf{F}' \leftarrow$ Survival($\mathbf{X}, \mathbf{F}$) ▷ NSGA-II survival for mating pool
13 $\quad\quad\mathbf{X}^{(\text{mat})} \leftarrow$ Mating($\mathbf{X}', \mathbf{F}', s \times N_{explr}$) ▷ Create offspring population using mating
14 $\quad\quad\mathbf{X}^{\text{explr}} \leftarrow$ FeasAndMaxDistSelection($\mathbf{X}^{(\text{mat})}, \mathbf{X}^{\text{cand}}, X, \mathbf{g}$) ▷ Keep feasible least crowded solutions
15 $\quad\quad\mathbf{x}_1^*, \ldots, \mathbf{x}_p^* \leftarrow \left[\mathbf{X}^{\text{explr}}, \mathbf{X}^{\text{exploit}}\right]$ ▷ Merge exploitation and exploration solutions
16 $\quad\quad j \leftarrow j + N_{exploit} + s \times N_{explr}$ ▷ Increase iteration counter to new matrix sizes
17 $\quad\quad\mathbf{X} \leftarrow \left[\mathbf{X},\ \mathbf{x}_1^*,\ \ldots,\ \mathbf{x}_p^*\right]$ ▷ Add p new solution vectors, $\mathbf{X} \in \mathbb{R}^{d \times j}$
18 $\quad\quad\mathbf{F} \leftarrow \left[\mathbf{F},\ \mathbf{f}(\mathbf{x}_1^*),\ \ldots,\ \mathbf{f}(\mathbf{x}_p^*)\right]$ ▷ Add vectors of evaluated objectives, $\mathbf{F} \in \mathbb{R}^{k \times j}$
19 $\quad\quad\mathbf{G} \leftarrow \left[\mathbf{G},\ \mathbf{g}(\mathbf{x}_1^*),\ \ldots,\ \mathbf{g}(\mathbf{x}_p^*)\right]$ ▷ Add vectors of evaluated constraints, $\mathbf{G} \in \mathbb{R}^{m \times j}$
20 $\quad$**end**
21 **return** ($\mathbf{F}$, $\mathbf{G}$, $\mathbf{X}$)

4.1. Hypervolume Maximization

HV maximization (line 12 in Algorithm 2) is done by maximizing the recently proposed multi-point acquisition function [18]. The multi-point acquisition function computes the joint HV contribution of a set of p solutions. One solution is represented by d continuous decision variables and the RBF surrogates are used to predict the objective and constraint values. If one or more of the constraints or objectives are inexpensive to evaluate, then the constraint and objective functions are directly used to compute the corresponding function value. The constraints g_i or RBF predictions thereof are used to quantify the constraint satisfaction ($g_i(\mathbf{x}) \leq 0$) or violation ($g_i(\mathbf{x}) > 0$) value. Each solution, therefore, has m constraint values. The k objective function values or RBF predictions thereof are used to quantify the quality of solutions in the objective space, where their HV contribution is computed.

To simultaneously maximize the multi-point acquisition function, i.e., the joint HV contribution of p solution candidates at an algorithm iteration, we formulate the task as a $p \cdot d$-dimensional optimization problem. The COBYLA algorithm [52] is then used to find a $p \cdot d$-dimensional solution vector $\mathbf{x}$ that maximizes the HV contribution of these p solutions simultaneously. In addition, COBYLA is also provided with the $p \cdot m$ constraint functions (directly or represented by RBFs) and the acquisition function which computes the HV contributions for the $p \cdot k$ objectives. The acquisition function also uses the the objective functions directly when the objectives are inexpensive, and the RBFs otherwise.

COBYLA linearly approximates all the constraints and the HV contribution in a small trust region. In this trust region, COBYLA maximizes the HV subject to the constraints by maximizing the following function:

$$\Psi(\mathbf{x}) = \hat{F}(\mathbf{x}) + \mu\left(-\max(c_i(\mathbf{x}) : i = i, \ldots, p \cdot m)\right)_+, \quad \mathbf{x} \in \mathbb{R}^{p \cdot d} \tag{4}$$

Here, $\mathbf{x}$ is the $p \cdot d$-dimensional solution vector, $\hat{F}$ is in our case the linear approximation of the HV contribution, c_i is the i-th linear approximation from the $p \cdot m$ constraint functions, the subscript + means that the expression in the brackets becomes 0 if none of the constraints are violated, and μ is a self-adaptive penalty parameter that makes sure that the approximation of a new solution $\hat{\Psi}(\mathbf{x}^*)$ with a smaller constraint violation and better hypervolume score is preferred over the starting solution approximation $\Psi(\mathbf{x}^0)$. After the best solution in the trust region on the linear approximations is found, it is evaluated on the trained RBFs for the expensive functions and the real functions for the inexpensive constraints and objectives. When the objective values are obtained, the HV contribution of the p solutions is computed. When the linear approximation of COBYLA in the trust region underestimates the HV approximation, the trust region increases in size, while if it overestimates, the trust region becomes smaller. This way, when nearing the solutions that have the highest HV contribution, the trust region becomes smaller and smaller until it falls below an $\varepsilon > 0$ value and COBYLA terminates. It should be noted that their overlapping HV is only counted once when the HV contribution is computed for all p solutions. The acquisition function therefore automatically prefers a set of diverse solutions over a set of solutions close to each other, as the latter would mostly contribute overlapping HV.

Since COBYLA is a local optimizer, it can get stuck in a local optimum [53]. To overcome this problem, 16 instances

Algorithm 2: IOC-SAMO-COBRA.

Input: Number of decision variables d, objective functions $\mathbf{f}(\mathbf{x})$, split where required into expensive objective function(s) $\mathbf{f}_e(\mathbf{x})$, computationally inexpensive objective function(s) $\mathbf{f}_c(\mathbf{x})$, constraint function(s) $\mathbf{g}(\mathbf{x})$, split where required into expensive constraint function(s) $\mathbf{g}_e(\mathbf{x})$, computationally inexpensive constraint function(s) $\mathbf{g}_c(\mathbf{x})$, decision parameters' lower and upper bounds $[\mathbf{lb}, \mathbf{ub}] \subset \mathbb{R}^d$, reference point $\mathbf{ref} \in \mathbb{R}^k$, number of initial samples N_{init}, maximum evaluation budget N_{max}, RBF strategy domain $\Phi = \{Cubic, Gaussian, Multiquadric, InverseQuadratic, InverseMultiquadric, ThinPlateS pline\} \times \{\text{PLOG}, standardized\}$, acquisition function HV.

Output: Evaluated solutions.

1 **Function** IOC-SAMO-COBRA(d, $\mathbf{f}$, $\mathbf{g}$, $[\mathbf{lb}, \mathbf{ub}]$, $\mathbf{ref}$, N_{init}, N_{max}, $RBF_{kernels}$, HV):
2 $\quad \mathbf{X} \leftarrow Halton(N_{init}, d, \mathbf{lb}, \mathbf{ub})$ ▷ Generate initial design of experiments, $\mathbf{X} \in \mathbb{R}^{d \times N}$
3 $\quad \mathbf{F} \leftarrow \mathbf{f}(\mathbf{X})$ ▷ Evaluate objective functions, $\mathbf{F} \in \mathbb{R}^{k \times N}$
4 $\quad \mathbf{G} \leftarrow \mathbf{g}(\mathbf{X})$ ▷ Evaluate constraint functions, $\mathbf{G} \in \mathbb{R}^{m \times N}$
5 $\quad \mathbf{h} \leftarrow \{\mathbf{f}_e \cup \mathbf{g}_e\}$ ▷ Union of expensive objective and constraint functions
6 $\quad \varphi^* \leftarrow \{(Cubic, standardized) \mid \forall\, h \in \mathbf{h}\}$ ▷ initialize best RBF, $\varphi^* \in \Phi$
7 $\quad \mathbf{E} \leftarrow \{0 \mid \forall\, h \in \{\mathbf{h} \times \Phi\}\}$ ▷ initialize RBF approximation errors for every function for every configuration $\varphi^* \in \Phi$
8 $\quad j \leftarrow N$ ▷ Initialize expensive evaluation counter
9 $\quad$ **while** $j < N_{max}$ **do**
10 $\quad\quad \mathbf{S}^{\Phi} \leftarrow \{\text{FITRBF}(\mathbf{X}, h, \Phi, \mathbf{lb}, \mathbf{ub}) \mid \forall\, h \in \mathbf{h}\}$ ▷ Fit RBF with all Φ strategies for all $\mathbf{h}$
11 $\quad\quad \mathbf{S}^{\varphi^*} \leftarrow \left\{\mathbf{S}^{\varphi^*} \mid \forall\, h \in \mathbf{h}\right\}$ ▷ Select best RBF surrogate φ^* based on line 6 or 17
12 $\quad\quad \mathbf{x}_1^*, \ldots, \mathbf{x}_p^* \leftarrow \text{MAX}(\text{HV}, p, \mathbf{ref}, \mathbf{S}^{\varphi^*}, \mathbf{f}_c, \mathbf{g}_c)$ ▷ Get p new solutions based on maximized HV infill criteria
13 $\quad\quad j \leftarrow j + p$ ▷ Increase iteration counter to new matrix sizes
14 $\quad\quad \mathbf{X} \leftarrow \left[\mathbf{X},\ \mathbf{x}_1^*,\ \ldots,\ \mathbf{x}_p^*\right]$ ▷ Add p new solution vectors, $\mathbf{X} \in \mathbb{R}^{d \times j}$
15 $\quad\quad \mathbf{F} \leftarrow \left[\mathbf{F},\ \mathbf{f}(\mathbf{x}_1^*),\ \ldots,\ \mathbf{f}(\mathbf{x}_p^*)\right]$ ▷ Add vectors of evaluated objectives, $\mathbf{F} \in \mathbb{R}^{k \times j}$
16 $\quad\quad \mathbf{G} \leftarrow \left[\mathbf{G},\ \mathbf{g}(\mathbf{x}_1^*),\ \ldots,\ \mathbf{g}(\mathbf{x}_p^*)\right]$ ▷ Add vectors of evaluated constraints, $\mathbf{G} \in \mathbb{R}^{m \times j}$
17 $\quad\quad \text{HV}, \varphi^*, \mathbf{E} \leftarrow \text{SELECTBESTSTRATEGY}(\mathbf{E}, S^{\Phi}, \mathbf{X}, \mathbf{F}, \mathbf{G})$ ▷ Get new HV, new RBF approximation errors $\mathbf{E}$ and best φ^* based on $\mathbf{E}$
18 $\quad$ **end**
19 **return** ($\mathbf{F}$, $\mathbf{G}$, $\mathbf{X}$)

of COBYLA are run in parallel starting from 16 random starting points. Start searching for optimal solutions in multiple locations simultaneously is also a well-known strategy in the Island model for parallel optimization [54], in parallel simulated annealing [55], and in ant colony optimization [56]. After all COBYLA instances have converged, all solutions found are 10000 times randomly combined in groups of size p. Since there are $\binom{16 \cdot p \cdot d}{p}$ such groups, for small values of p and d less combinations are sufficient. However, due to the negligible computational effort, we decided to fix this number to 10000. The set of p solutions which together contribute the most HV are selected for evaluation on the expensive objective and constraint functions.

After the parallel evaluation of the solutions on the real functions (Line 14, 15, 16 of algorithm 2), the RBF approximation errors ($\mathbf{E}$) are stored for each RBF modeling strategy (Line 17 of algorithm 2), the RBFs are updated (Line 10 of algorithm 2), the best RBF modeling strategy is selected based on the historic approximation errors (Line 11 of algorithm 2) and COBYLA is used again to find the next set of optimal solutions (Line 12 of algorithm 2). This optimization process continues until the expensive evaluation budget is exhausted (Line 9 of algorithm 2).

4.2. Acquisition Function Switching

IOC-SAMO-COBRA maximizes the predicted HV contribution every iteration, meaning that by default it does not use any uncertainty quantification of the RBF models for the objectives. Just like the RBF functions, by default, the inexpensive objectives also do not have an uncertainty quantification method. Other Bayesian optimization algorithms, however, often use Kriging or Gaussian process regression models, which provide an uncertainty quantification method for the objectives to encourage exploration [57, 58, 59]. Earlier experiments, however, showed that the use of uncertainty quantification is in many cases redundant because by maximizing the HV, the algorithm is naturally forced to explore the objective space [17]. If, however, IOC-SAMO-COBRA gets stuck and does not find any HV improvement for three consecutive iterations, an uncertainty quantification method for RBFs [31] is enabled to help with exploration (this is part of line 17 of algorithm 2, but for space reasons not explicitly formulated in the pseudocode). By enabling the uncertainty quantification method, the acquisition function changes to an RBF variant of the S-Metric selection criterion [58]. Note that the inexpensive objectives still do not have an uncertainty quantification and therefore, only for the objectives modeled with RBFs the uncertainty is calculated.

5. Experimental Setup

The four algorithms (SA-NSGA-II, IC-SA-NSGA-II, SAMO-COBRA, IOC-SAMO-COBRA) are compared on a diverse set of test functions. The surrogate-assisted algorithm and the Inexpensive function exploiting counterparts are compared to confirm our hypothesis that exploiting inexpensive functions in the optimization process directly is beneficial. The metrics used to compare the algorithms performance are the HV and the IGD+ performance metric which are described in more detail in Section 2.2.

5.1. Test Functions

The test functions used to assess the performance of the algorithms are given in Table 2. In this table, the reference point, the approximated nadir point, the number of objectives k, the number of dimensions d, the number of constraints m, and the feasibility ratio (P%) after one million random samples are given. The following functions are artificially created test functions: BNH [60], CEXP [61], SRN [62], TNK [62], CTP1 [61], C3DTLZ4 [63], OSY [60, 62], NBP [64], BICOP1 [65], BICOP2 [65], TRICOP [65], MW1 [66], MW2 [66], MW3 [66], MW11 [66]. The following functions are real-world-like problems: Two-bar truss design (TBTD) [67], disk brake design (DBD) [67], ship parametric design (SPD) [68], car-side impact (CSI) [69], speed reducer design (SRD) [70], welded beam (WB) [67], water resource mangement problem (WP) [69].

The test functions are selected because they are diverse, well known, and because some mimic industrial problems. The Pareto frontiers of the functions vary between 2 and 5 dimensions and the shapes can be classified as concave, convex, connected, disconnected, or even mixes of these characteristics. The constraints of the selected test problems are also diverse since for some problems they are very strict, while for other problems (almost) the entire search space is feasible. Next to the feasibility of the problems, on some Pareto frontiers, the constraints are active, while on other problems they are not, or partially active. Next to the artificially created test functions, we consider it very important to select a set of real-world-inspired problems to assess how well the algorithms operate in situations resembling industrial optimization scenarios.

Table 2: Test function name, reference point used by the SAMO-COBRA algorithm, Nadir point approximation based on all results, number of objectives k, number of decision variables d, number of constraints m, percentage of feasible solutions $P(\%)$ after one million random samples.

Function	Reference point	Nadir point	k	d	m	$P(\%)$
BNH	(140, 50)	(136.00, 50.00)	2	2	2	96.92
CEXP	(1, 9)	(1.00, 9.00)	2	2	2	57.14
SRN	(301, 72)	(222.99, 2.62)	2	2	2	16.18
TNK	(2, 2)	(1.04, 1.04)	2	2	2	5.05
CTP1	(1, 2)	(1.00, 1.00)	2	2	2	92.67
C3DTLZ4	(3, 3)	(2.00, 2.00)	2	6	2	22.22
OSY	(0, 386)	(-41.81, 76.00)	2	6	6	2.78
TBTD	(0.1, 50000)	(0.1, 10000)	2	3	2	19.46
NBP	(11150, 12500)	(12500, 114.09)	2	2	5	41.34
DBD	(5,50)	(2.79, 16.86)	2	4	5	28.55
SRD	(7000, 1700)	(5879.98, 1696.46)	2	7	11	96.92
WB	(350, 0.1)	(35.31, 0.0145)	2	4	5	35.28
BICOP1	(9, 9)	(1.00, 1.00)	2	10	1	100
BICOP2	(70, 70)	(1.10, 1.11)	2	10	2	10.55
MW1	(1,7)	(1.00, 1.00)	2	8	1	0.007
MW2	(1,7)	(1.00, 1.00)	2	6	1	0.55
MW3	(1,7)	(1.00, 1.00)	2	6	2	1.32
MW11	(30,30)	(2.06, 2.04)	2	6	4	1.38
TRIPCOP	(34, -4, 90)	(7.67, -11.77, 25.91)	3	2	3	15.85
SPD	(16, 19000, -260000)	(11.16, 12435.27, -259148.04)	3	6	9	3.27
CSI	(42, 4.5, 13)	(42.77, 4.00, 12.52)	3	7	10	18.17
WP	(83000, 1350, 2.85, 15989825, 25000)	(74573, 1350, 2.85, 7874925, 25000)	5	3	7	92.06

5.2. Experiment Algorithm Settings

The allowed number of function evaluations for the different algorithms is set to $40 \cdot d$ for all functions experimented with. The performances of the algorithms are checked with a different number of candidate solutions per iteration (In the SA-NSGA-II variants also referred to as population sizes) $p \in \{1, 2, 3, 4, 5, 6, 10, 20\}$. To get statistically significant results on all test functions, each test function is optimized 10 times per algorithm configuration.

All test functions in Table 2 are inexpensive to evaluate. However, SA-NSGA-II and SAMO-COBRA are developed to optimize computationally expensive problems. To test this functionality, in the experiments done with SA-NSGA-II and SAMO-COBRA all constraints and objectives are considered to be expensive and are therefore modeled with the RBF surrogates. To test the functionality where inexpensive functions are directly used instead of a surrogate with IC-SA-NSGA-II and IOC-SAMO-COBRA, a decision needs to be made concerning the expensiveness of the objective and constraint functions. To be able to compare IOC-SAMO-COBRA to IC-SA-NSGA-II as fairly as possible, the assumption from IC-SA-NSGA-II that the constraints are inexpensive and the objectives are expensive to evaluate is also adopted in the experiments with IC-SA-NSGA-II and IOC-SAMO-COBRA. A description and implementation of the test functions, the obtained Pareto frontiers for the IGD+ performance metric, all raw experiment results, and implementation of the IOC-SAMO-COBRA algorithm can be found on a dedicated Github page [71].

6. Results

The results obtained from the four algorithm variants that optimized the test functions are presented in tables, empirical cumulative distribution function plots, and empirical attainment function difference plots. Special attention is given to the problems with a very small feasibility ratio since these test problems benefit the most from using the inexpensive constraint functions directly in the optimization algorithms.

6.1. Performance Metrics Results

The two performance metrics used to assess and compare the performance of the different algorithms are the IGD+ metric and the HV metric. Table 3 and Table 4, respectively, report the mean and standard deviation of the HV and the IGD+ performance metric after $40 \cdot d$ function evaluations. The HV is computed between the Nadir point and the obtained Pareto fronts, the IGD+ metric is computed between a well-spread Pareto front approximation and the obtained Pareto fronts by the different algorithms. The performance metrics for the SA-NSGA-II, IC-SA-NSGA-II, and SAMO-COBRA are statistically compared with a Wilcoxon rank sum test to the results of IOC-SAMO-COBRA at a 5% confidence level. A ($-$) in the tables indicates significantly worse results, ($\approx$) indicates indifference between the results, and ($+$) indicates significantly better results of the given algorithm, compared to IOC-SAMO-COBRA. In the second last row of Table 3 and Table 4 a summary is given of the results of the significance test. Inspection of this summary shows that IOC-SAMO-COBRA in most cases achieves the best or statistically indistinguishable results after the number of function evaluations is exhausted for both

the HV and IGD+ metric. On 14 out of 22 test problems, IOC-SAMO-COBRA outperforms the other algorithms if we evaluate their performance based on data for all values of p that were tested. On 4 out of 22 test problems, SAMO-COBRA achieves a larger HV compared to IOC-SAMO-COBRA, however, these results are often not significant and differences are too small to be captured in the table with only two numbers after the decimal point. On the remaining 4 out of 22 test problems, the IC-SA-NSGA-II algorithm performs better compared to IOC-SAMO-COBRA, especially on BICOP1 and MW2. The mean Friedman rank test confirmed (with $p = 1 \cdot 10^{-16}$) the alternative hypothesis which states that there is a significant difference in the mean ranks of the algorithms. In the last rows of Table 3 and Table 4, respectively, the mean ranks of the algorithms are reported (a low rank indicates a better rank for both performance metrics).

6.2. Empirical Cumulative Distribution Function

Table 3 and Table 4 do not tell us anything about the convergence rate or how fast the different algorithms are able to find Pareto efficient solutions. Empirical Cumulative Distribution Functions (ECDF) (see e.g. [72, 73] for a formal definition) visualize the convergence of the different algorithms. The aggregated results of the HV and IGD+ metric of the four different algorithm variants are visualized in Figure 3 and figure 4 by means of their ECDF, based on a fixed-target perspective. The ECDF plot is based on 40 target values which are linearly distributed between zero and the maximum achievable performance score per test function, and shows the proportion of the target values attained by the algorithms, depending on the number of function evaluations up to the maximum number of 400 (which results from the 10-dimensional BICOP problems). For each algorithm, the corresponding ECDF curve is aggregated over all functions and the number of candidate solutions per iteration. The four curves illustrate the advantage of the *Inexpensive Constrained* approach, independently of the base algorithm. This finding highlights the importance of using as accurate as possible models (by IOC-SAMO-COBRA's approach to evaluate and compare all RBF configurations in the configuration space Φ) and shows the relevance of using the constraint and objective functions directly if they are computationally inexpensive and available.

6.3. Visual Comparison

The Pareto fronts obtained by the IC-SA-NSGA-II, and the IOC-SAMO-COBRA algorithm can be visually compared with the Empirical Attainment Difference Functions [74]. In Figure 5 the EAF difference plot on the TBTD test function is given as an example, with all results per algorithm aggregated. The dark areas mark where the two algorithms obtained different results. As can be seen, the IOC-SAMO-COBRA algorithm manages to find the minimum values of objective 2 on the Pareto frontier, while IC-SA-NSGA-II found smaller values for objective 1. The EAF plots of other two objective test functions can be found in Appendix Appendix A.

6.4. Discussion

Table 3 and Table 4 show that IOC-SAMO-COBRA performs better in most cases compared to the other algorithms. The ECDF plot (Figure 3) also shows that the algorithm on average also finds good solutions faster since it is able to reach a higher portion of the run target pairs. The EAF different plots from appendix Appendix A also show in most cases that IOC-SAMO-COBRA finds solutions closer to the Pareto frontier compared to IC-SA-NSGA-II algorithm. However, two things become apparent when all results are analyzed in more detail. (1) IC-SA-NSGA-II significantly outperforms the IOC-SAMO-COBRA algorithm on the BICOP1 and MW2 test problems. BICOP1 and MW2 do not have any active constraints on the Pareto front and the difference between the performances of the algorithms becomes even larger when the number of candidate solutions per iteration increases. This indicates that IC-SA-NSGA-II has more difficulty finding feasible solutions on the Pareto fronts with active constraints and IOC-SAMO-COBRA is directed too much towards the constraint boundaries and has more difficulty finding the Pareto front if the Pareto front is unconstrained. (2) For test problems with a very low feasibility ratio (MW1, MW2, MW3, and MW11) the IC-SA-NSGA-II and IOC-SAMO-COBRA significantly outperform their original counterparts where the constraint functions are not directly used in the algorithm. In a few algorithm runs on the MW test functions not a single feasible solution was found. This indicates that the more strict the constraints are, the more beneficial it is to directly use the constraints instead of attempting to learn them with surrogates.

7. Cargo Vessel Design Application

As a real-world application example, the mid-ship section of a single-hold general cargo vessel design[3] as presented in Figure 6 is optimized for two conflicting objectives: stability (↑ max) after potential damage (survivability), and cargo hold capacity (↑ max). Besides the conflicting objectives, the problem has three volumetric constraints and one regulatory constraint:

- Volumetric: The two fuel tanks should be of sufficient size so that enough fuel can be stored and the technical space should be large enough to host the equipment.
- Regulatory: The attained damage stability index (survivability) score should be larger compared to the required damage stability index.

The objectives and constraints depend on 17 geometric parameters, which influence the longitudinal and transversal positioning of the bulkheads and the heights of openings. The bulkheads split the different compartments and tanks together with the height of decks and openings in the vicinity of the cargo hold. The evaluation of the damage stability (survivability)

[3]Figure courtesy of C-Job Naval Architects, Hoofddorp, Netherlands.

Table 3: Hypervolume score ± standard deviation of HV, Wilcoxon rank sum test with probability value = 0.05 (reference algorithm: IOC-SAMO-COBRA), per test function and candidate solutions size p. The highest HV per row is reported in **bold**, best scoring algorithm per test function is highlighted.

Function	p	SA-NSGA-II	IC-SA-NSGA-II	SAMO-COBRA	IOC-SAMO-COBRA
BNH	1	$4.89 \cdot 10^{3} \pm 3.06 \cdot 10^{1}$ (−)	$4.85 \cdot 10^{3} \pm 3.85 \cdot 10^{1}$ (−)	**$5.07 \cdot 10^{3} \pm 4.10 \cdot 10^{-2}$ (≈)**	$5.07 \cdot 10^{3} \pm 2.44 \cdot 10^{-2}$
	2	$4.86 \cdot 10^{3} \pm 1.54 \cdot 10^{1}$ (−)	$4.83 \cdot 10^{3} \pm 3.36 \cdot 10^{1}$ (−)	$5.07 \cdot 10^{3} \pm 3.47 \cdot 10^{-2}$ (≈)	**$5.07 \cdot 10^{3} \pm 3.85 \cdot 10^{-2}$**
	3	$4.88 \cdot 10^{3} \pm 3.12 \cdot 10^{1}$ (−)	$4.85 \cdot 10^{3} \pm 3.04 \cdot 10^{1}$ (−)	**$5.07 \cdot 10^{3} \pm 5.44 \cdot 10^{-2}$ (≈)**	$5.07 \cdot 10^{3} \pm 2.86 \cdot 10^{-2}$
	4	$4.89 \cdot 10^{3} \pm 1.99 \cdot 10^{1}$ (−)	$4.84 \cdot 10^{3} \pm 2.61 \cdot 10^{1}$ (−)	**$5.07 \cdot 10^{3} \pm 1.57 \cdot 10^{-1}$ (+)**	$5.07 \cdot 10^{3} \pm 9.29 \cdot 10^{-2}$
	5	$4.86 \cdot 10^{3} \pm 2.52 \cdot 10^{1}$ (−)	$4.85 \cdot 10^{3} \pm 2.46 \cdot 10^{1}$ (−)	$5.07 \cdot 10^{3} \pm 1.83 \cdot 10^{-1}$ (≈)	**$5.07 \cdot 10^{3} \pm 2.24 \cdot 10^{-1}$**
	6	$4.88 \cdot 10^{3} \pm 1.37 \cdot 10^{1}$ (−)	$4.88 \cdot 10^{3} \pm 3.05 \cdot 10^{1}$ (−)	$5.07 \cdot 10^{3} \pm 1.25 \cdot 10^{-1}$ (≈)	**$5.07 \cdot 10^{3} \pm 1.94 \cdot 10^{-1}$**
	10	$4.87 \cdot 10^{3} \pm 1.92 \cdot 10^{1}$ (−)	$4.86 \cdot 10^{3} \pm 2.68 \cdot 10^{1}$ (−)	**$5.07 \cdot 10^{3} \pm 2.71 \cdot 10^{-1}$ (≈)**	$5.07 \cdot 10^{3} \pm 1.53 \cdot 10^{-1}$
	20	$4.90 \cdot 10^{3} \pm 3.03 \cdot 10^{1}$ (−)	$4.87 \cdot 10^{3} \pm 2.54 \cdot 10^{1}$ (−)	**$5.06 \cdot 10^{3} \pm 4.25 \cdot 10^{-1}$ (≈)**	$5.06 \cdot 10^{3} \pm 3.94 \cdot 10^{-1}$
CEXP	1	$3.65 \cdot 10^{0} \pm 2.23 \cdot 10^{-2}$ (−)	$3.64 \cdot 10^{0} \pm 6.21 \cdot 10^{-2}$ (−)	$3.80 \cdot 10^{0} \pm 4.36 \cdot 10^{-4}$ (−)	**$3.80 \cdot 10^{0} \pm 8.37 \cdot 10^{-5}$**
	2	$3.58 \cdot 10^{0} \pm 4.48 \cdot 10^{-2}$ (−)	$3.57 \cdot 10^{0} \pm 5.72 \cdot 10^{-2}$ (−)	$3.80 \cdot 10^{0} \pm 1.16 \cdot 10^{-3}$ (−)	**$3.80 \cdot 10^{0} \pm 3.89 \cdot 10^{-4}$**
	3	$3.58 \cdot 10^{0} \pm 3.02 \cdot 10^{-2}$ (−)	$3.57 \cdot 10^{0} \pm 3.76 \cdot 10^{-2}$ (−)	$3.80 \cdot 10^{0} \pm 1.73 \cdot 10^{-4}$ (≈)	**$3.80 \cdot 10^{0} \pm 5.62 \cdot 10^{-5}$**
	4	$3.57 \cdot 10^{0} \pm 3.56 \cdot 10^{-2}$ (−)	$3.56 \cdot 10^{0} \pm 3.62 \cdot 10^{-2}$ (−)	$3.80 \cdot 10^{0} \pm 3.68 \cdot 10^{-5}$ (−)	**$3.80 \cdot 10^{0} \pm 2.65 \cdot 10^{-4}$**
	5	$3.58 \cdot 10^{0} \pm 3.18 \cdot 10^{-2}$ (−)	$3.56 \cdot 10^{0} \pm 4.78 \cdot 10^{-2}$ (−)	**$3.80 \cdot 10^{0} \pm 2.35 \cdot 10^{-4}$ (≈)**	$3.80 \cdot 10^{0} \pm 1.09 \cdot 10^{-4}$
	6	$3.58 \cdot 10^{0} \pm 2.40 \cdot 10^{-2}$ (−)	$3.58 \cdot 10^{0} \pm 3.35 \cdot 10^{-2}$ (−)	$3.80 \cdot 10^{0} \pm 2.67 \cdot 10^{-4}$ (≈)	**$3.80 \cdot 10^{0} \pm 1.54 \cdot 10^{-4}$**
	10	$3.56 \cdot 10^{0} \pm 4.87 \cdot 10^{-2}$ (−)	$3.60 \cdot 10^{0} \pm 2.43 \cdot 10^{-2}$ (−)	$3.79 \cdot 10^{0} \pm 6.69 \cdot 10^{-4}$ (≈)	**$3.79 \cdot 10^{0} \pm 7.53 \cdot 10^{-4}$**
	20	$3.55 \cdot 10^{0} \pm 2.89 \cdot 10^{-2}$ (−)	$3.59 \cdot 10^{0} \pm 3.41 \cdot 10^{-2}$ (−)	**$3.77 \cdot 10^{0} \pm 3.43 \cdot 10^{-3}$ (≈)**	$3.77 \cdot 10^{0} \pm 4.06 \cdot 10^{-3}$
SRN	1	$2.38 \cdot 10^{4} \pm 1.14 \cdot 10^{2}$ (−)	$2.40 \cdot 10^{4} \pm 1.53 \cdot 10^{2}$ (−)	$2.50 \cdot 10^{4} \pm 3.86 \cdot 10^{0}$ (≈)	**$2.50 \cdot 10^{4} \pm 2.28 \cdot 10^{0}$**
	2	$2.29 \cdot 10^{4} \pm 2.99 \cdot 10^{2}$ (−)	$2.34 \cdot 10^{4} \pm 1.97 \cdot 10^{2}$ (−)	$2.50 \cdot 10^{4} \pm 1.13 \cdot 10^{1}$ (−)	**$2.50 \cdot 10^{4} \pm 3.96 \cdot 10^{0}$**
	3	$2.34 \cdot 10^{4} \pm 2.55 \cdot 10^{2}$ (−)	$2.35 \cdot 10^{4} \pm 2.62 \cdot 10^{2}$ (−)	$2.50 \cdot 10^{4} \pm 6.39 \cdot 10^{0}$ (−)	**$2.50 \cdot 10^{4} \pm 2.56 \cdot 10^{0}$**
	4	$2.30 \cdot 10^{4} \pm 2.84 \cdot 10^{2}$ (−)	$2.33 \cdot 10^{4} \pm 2.10 \cdot 10^{2}$ (−)	**$2.50 \cdot 10^{4} \pm 2.80 \cdot 10^{0}$ (+)**	$2.50 \cdot 10^{4} \pm 2.14 \cdot 10^{0}$
	5	$2.33 \cdot 10^{4} \pm 2.32 \cdot 10^{2}$ (−)	$2.35 \cdot 10^{4} \pm 2.89 \cdot 10^{2}$ (−)	$2.50 \cdot 10^{4} \pm 2.62 \cdot 10^{0}$ (≈)	**$2.50 \cdot 10^{4} \pm 2.78 \cdot 10^{0}$**
	6	$2.33 \cdot 10^{4} \pm 1.64 \cdot 10^{2}$ (−)	$2.36 \cdot 10^{4} \pm 1.43 \cdot 10^{2}$ (−)	$2.50 \cdot 10^{4} \pm 7.42 \cdot 10^{0}$ (≈)	**$2.50 \cdot 10^{4} \pm 3.92 \cdot 10^{0}$**
	10	$2.33 \cdot 10^{4} \pm 2.03 \cdot 10^{2}$ (−)	$2.37 \cdot 10^{4} \pm 2.46 \cdot 10^{2}$ (−)	**$2.49 \cdot 10^{4} \pm 3.07 \cdot 10^{1}$ (≈)**	$2.49 \cdot 10^{4} \pm 2.99 \cdot 10^{1}$
	20	$2.31 \cdot 10^{4} \pm 3.95 \cdot 10^{2}$ (−)	$2.37 \cdot 10^{4} \pm 1.39 \cdot 10^{2}$ (−)	**$2.48 \cdot 10^{4} \pm 1.18 \cdot 10^{1}$ (≈)**	$2.48 \cdot 10^{4} \pm 2.05 \cdot 10^{1}$
TNK	1	$2.05 \cdot 10^{-1} \pm 1.38 \cdot 10^{-2}$ (−)	$2.87 \cdot 10^{-1} \pm 3.37 \cdot 10^{-3}$ (−)	$2.96 \cdot 10^{-1} \pm 1.65 \cdot 10^{-3}$ (−)	**$3.03 \cdot 10^{-1} \pm 5.49 \cdot 10^{-4}$**
	2	$2.31 \cdot 10^{-1} \pm 1.75 \cdot 10^{-2}$ (−)	$2.75 \cdot 10^{-1} \pm 5.24 \cdot 10^{-3}$ (−)	$2.96 \cdot 10^{-1} \pm 1.99 \cdot 10^{-3}$ (−)	**$3.05 \cdot 10^{-1} \pm 4.94 \cdot 10^{-4}$**
	3	$2.49 \cdot 10^{-1} \pm 1.70 \cdot 10^{-2}$ (−)	$2.84 \cdot 10^{-1} \pm 3.80 \cdot 10^{-3}$ (−)	$2.95 \cdot 10^{-1} \pm 3.09 \cdot 10^{-3}$ (−)	**$3.06 \cdot 10^{-1} \pm 2.40 \cdot 10^{-4}$**
	4	$2.47 \cdot 10^{-1} \pm 1.34 \cdot 10^{-2}$ (−)	$2.71 \cdot 10^{-1} \pm 8.31 \cdot 10^{-3}$ (−)	$2.97 \cdot 10^{-1} \pm 1.80 \cdot 10^{-3}$ (−)	**$3.06 \cdot 10^{-1} \pm 2.68 \cdot 10^{-4}$**
	5	$2.48 \cdot 10^{-1} \pm 1.13 \cdot 10^{-2}$ (−)	$2.77 \cdot 10^{-1} \pm 7.15 \cdot 10^{-3}$ (−)	$2.95 \cdot 10^{-1} \pm 2.26 \cdot 10^{-3}$ (−)	**$3.06 \cdot 10^{-1} \pm 1.34 \cdot 10^{-4}$**
	6	$2.48 \cdot 10^{-1} \pm 1.47 \cdot 10^{-2}$ (−)	$2.81 \cdot 10^{-1} \pm 2.97 \cdot 10^{-3}$ (−)	$2.94 \cdot 10^{-1} \pm 1.25 \cdot 10^{-3}$ (−)	**$3.06 \cdot 10^{-1} \pm 2.19 \cdot 10^{-4}$**
	10	$2.35 \cdot 10^{-1} \pm 1.21 \cdot 10^{-2}$ (−)	$2.73 \cdot 10^{-1} \pm 5.91 \cdot 10^{-3}$ (−)	$2.93 \cdot 10^{-1} \pm 2.56 \cdot 10^{-3}$ (−)	**$3.06 \cdot 10^{-1} \pm 1.49 \cdot 10^{-4}$**
	20	$2.16 \cdot 10^{-1} \pm 1.11 \cdot 10^{-2}$ (−)	$2.72 \cdot 10^{-1} \pm 7.76 \cdot 10^{-3}$ (−)	$2.83 \cdot 10^{-1} \pm 3.23 \cdot 10^{-3}$ (−)	**$3.01 \cdot 10^{-1} \pm 8.46 \cdot 10^{-4}$**
CTP1	1	$2.86 \cdot 10^{-1} \pm 4.12 \cdot 10^{-3}$ (−)	$2.89 \cdot 10^{-1} \pm 2.64 \cdot 10^{-3}$ (−)	**$3.02 \cdot 10^{-1} \pm 1.29 \cdot 10^{-4}$ (≈)**	$3.02 \cdot 10^{-1} \pm 2.34 \cdot 10^{-4}$
	2	$2.76 \cdot 10^{-1} \pm 2.99 \cdot 10^{-3}$ (−)	$2.74 \cdot 10^{-1} \pm 7.15 \cdot 10^{-3}$ (−)	$3.00 \cdot 10^{-1} \pm 1.63 \cdot 10^{-3}$ (≈)	**$3.01 \cdot 10^{-1} \pm 1.45 \cdot 10^{-3}$**
	3	$2.78 \cdot 10^{-1} \pm 5.88 \cdot 10^{-3}$ (−)	$2.81 \cdot 10^{-1} \pm 6.28 \cdot 10^{-3}$ (−)	$3.02 \cdot 10^{-1} \pm 4.18 \cdot 10^{-4}$ (≈)	**$3.02 \cdot 10^{-1} \pm 8.68 \cdot 10^{-4}$**
	4	$2.80 \cdot 10^{-1} \pm 2.48 \cdot 10^{-3}$ (−)	$2.77 \cdot 10^{-1} \pm 4.06 \cdot 10^{-3}$ (−)	$3.02 \cdot 10^{-1} \pm 4.10 \cdot 10^{-4}$ (≈)	**$3.02 \cdot 10^{-1} \pm 3.57 \cdot 10^{-4}$**
	5	$2.74 \cdot 10^{-1} \pm 6.52 \cdot 10^{-3}$ (−)	$2.76 \cdot 10^{-1} \pm 4.81 \cdot 10^{-3}$ (−)	$3.02 \cdot 10^{-1} \pm 3.58 \cdot 10^{-4}$ (≈)	**$3.02 \cdot 10^{-1} \pm 3.58 \cdot 10^{-4}$**
	6	$2.78 \cdot 10^{-1} \pm 4.94 \cdot 10^{-3}$ (−)	$2.79 \cdot 10^{-1} \pm 2.59 \cdot 10^{-3}$ (−)	**$3.02 \cdot 10^{-1} \pm 3.14 \cdot 10^{-4}$ (≈)**	$3.02 \cdot 10^{-1} \pm 3.14 \cdot 10^{-4}$
	10	$2.76 \cdot 10^{-1} \pm 5.45 \cdot 10^{-3}$ (−)	$2.81 \cdot 10^{-1} \pm 3.46 \cdot 10^{-3}$ (−)	**$3.01 \cdot 10^{-1} \pm 2.53 \cdot 10^{-4}$ (≈)**	$3.01 \cdot 10^{-1} \pm 2.82 \cdot 10^{-4}$
	20	$2.74 \cdot 10^{-1} \pm 4.44 \cdot 10^{-3}$ (−)	$2.81 \cdot 10^{-1} \pm 4.28 \cdot 10^{-3}$ (−)	$2.99 \cdot 10^{-1} \pm 8.59 \cdot 10^{-4}$ (≈)	**$2.99 \cdot 10^{-1} \pm 1.05 \cdot 10^{-3}$**
C3DTLZ4	1	$1.54 \cdot 10^{0} \pm 9.41 \cdot 10^{-2}$ (−)	$1.23 \cdot 10^{0} \pm 2.01 \cdot 10^{-1}$ (−)	$1.44 \cdot 10^{0} \pm 5.31 \cdot 10^{-2}$ (−)	**$1.74 \cdot 10^{0} \pm 5.19 \cdot 10^{-3}$**
	2	$1.54 \cdot 10^{0} \pm 9.22 \cdot 10^{-2}$ (−)	$1.54 \cdot 10^{0} \pm 1.07 \cdot 10^{-1}$ (−)	$1.27 \cdot 10^{0} \pm 5.91 \cdot 10^{-2}$ (−)	**$1.75 \cdot 10^{0} \pm 8.81 \cdot 10^{-3}$**
	3	$1.64 \cdot 10^{0} \pm 2.19 \cdot 10^{-2}$ (−)	$1.65 \cdot 10^{0} \pm 3.30 \cdot 10^{-2}$ (−)	$1.40 \cdot 10^{0} \pm 5.46 \cdot 10^{-2}$ (−)	**$1.76 \cdot 10^{0} \pm 1.01 \cdot 10^{-3}$**
	4	$1.66 \cdot 10^{0} \pm 1.50 \cdot 10^{-2}$ (−)	$1.69 \cdot 10^{0} \pm 1.12 \cdot 10^{-2}$ (−)	$1.39 \cdot 10^{0} \pm 3.50 \cdot 10^{-2}$ (−)	**$1.77 \cdot 10^{0} \pm 1.37 \cdot 10^{-3}$**
	5	$1.66 \cdot 10^{0} \pm 2.01 \cdot 10^{-2}$ (−)	$1.69 \cdot 10^{0} \pm 1.20 \cdot 10^{-2}$ (−)	$1.43 \cdot 10^{0} \pm 4.12 \cdot 10^{-2}$ (−)	**$1.77 \cdot 10^{0} \pm 8.47 \cdot 10^{-4}$**
	6	$1.67 \cdot 10^{0} \pm 1.57 \cdot 10^{-2}$ (−)	$1.71 \cdot 10^{0} \pm 7.88 \cdot 10^{-3}$ (−)	$1.44 \cdot 10^{0} \pm 5.84 \cdot 10^{-2}$ (−)	**$1.77 \cdot 10^{0} \pm 5.92 \cdot 10^{-4}$**
	10	$1.66 \cdot 10^{0} \pm 1.84 \cdot 10^{-2}$ (−)	$1.72 \cdot 10^{0} \pm 4.84 \cdot 10^{-3}$ (−)	$1.46 \cdot 10^{0} \pm 6.98 \cdot 10^{-2}$ (−)	**$1.77 \cdot 10^{0} \pm 1.42 \cdot 10^{-3}$**
	20	$1.64 \cdot 10^{0} \pm 2.17 \cdot 10^{-2}$ (−)	$1.72 \cdot 10^{0} \pm 3.89 \cdot 10^{-3}$ (−)	$1.52 \cdot 10^{0} \pm 3.39 \cdot 10^{-2}$ (−)	**$1.76 \cdot 10^{0} \pm 1.46 \cdot 10^{-3}$**
OSY	1	$9.62 \cdot 10^{3} \pm 1.98 \cdot 10^{3}$ (−)	$1.13 \cdot 10^{4} \pm 4.75 \cdot 10^{2}$ (−)	**$1.26 \cdot 10^{4} \pm 4.21 \cdot 10^{0}$ (≈)**	$1.26 \cdot 10^{4} \pm 2.78 \cdot 10^{0}$
	2	$1.18 \cdot 10^{4} \pm 3.45 \cdot 10^{2}$ (−)	$1.18 \cdot 10^{4} \pm 2.70 \cdot 10^{2}$ (−)	$1.26 \cdot 10^{4} \pm 3.34 \cdot 10^{0}$ (−)	**$1.26 \cdot 10^{4} \pm 3.66 \cdot 10^{0}$**
	3	$1.21 \cdot 10^{4} \pm 2.35 \cdot 10^{2}$ (−)	$1.23 \cdot 10^{4} \pm 6.57 \cdot 10^{1}$ (−)	$1.26 \cdot 10^{4} \pm 3.02 \cdot 10^{0}$ (≈)	**$1.26 \cdot 10^{4} \pm 2.63 \cdot 10^{0}$**
	4	$1.22 \cdot 10^{4} \pm 1.36 \cdot 10^{2}$ (−)	$1.23 \cdot 10^{4} \pm 8.03 \cdot 10^{1}$ (−)	**$1.26 \cdot 10^{4} \pm 2.79 \cdot 10^{0}$ (≈)**	$1.26 \cdot 10^{4} \pm 5.11 \cdot 10^{0}$
	5	$1.23 \cdot 10^{4} \pm 6.76 \cdot 10^{1}$ (−)	$1.23 \cdot 10^{4} \pm 7.50 \cdot 10^{1}$ (−)	**$1.26 \cdot 10^{4} \pm 4.56 \cdot 10^{0}$ (≈)**	$1.26 \cdot 10^{4} \pm 3.79 \cdot 10^{0}$
	6	$1.23 \cdot 10^{4} \pm 4.06 \cdot 10^{1}$ (−)	$1.24 \cdot 10^{4} \pm 4.16 \cdot 10^{1}$ (−)	$1.26 \cdot 10^{4} \pm 6.01 \cdot 10^{0}$ (≈)	**$1.26 \cdot 10^{4} \pm 6.76 \cdot 10^{0}$**
	10	$1.24 \cdot 10^{4} \pm 1.06 \cdot 10^{2}$ (−)	$1.24 \cdot 10^{4} \pm 5.34 \cdot 10^{1}$ (−)	$1.24 \cdot 10^{4} \pm 3.24 \cdot 10^{1}$ (≈)	**$1.24 \cdot 10^{4} \pm 2.87 \cdot 10^{1}$**
	20	$1.23 \cdot 10^{4} \pm 1.40 \cdot 10^{2}$ (+)	**$1.23 \cdot 10^{4} \pm 1.92 \cdot 10^{2}$ (+)**	$1.13 \cdot 10^{4} \pm 3.43 \cdot 10^{2}$ (−)	$1.16 \cdot 10^{4} \pm 1.67 \cdot 10^{2}$
TBTD	1	$3.46 \cdot 10^{2} \pm 9.91 \cdot 10^{1}$ (−)	$3.92 \cdot 10^{2} \pm 4.59 \cdot 10^{1}$ (−)	$4.95 \cdot 10^{2} \pm 3.40 \cdot 10^{0}$ (≈)	**$4.96 \cdot 10^{2} \pm 9.50 \cdot 10^{0}$**
	2	$4.00 \cdot 10^{2} \pm 3.40 \cdot 10^{1}$ (−)	$4.37 \cdot 10^{2} \pm 1.41 \cdot 10^{1}$ (−)	$4.88 \cdot 10^{2} \pm 6.06 \cdot 10^{0}$ (≈)	**$4.89 \cdot 10^{2} \pm 8.70 \cdot 10^{0}$**
	3	$4.18 \cdot 10^{2} \pm 1.40 \cdot 10^{1}$ (−)	$4.44 \cdot 10^{2} \pm 1.56 \cdot 10^{1}$ (−)	$4.73 \cdot 10^{2} \pm 9.80 \cdot 10^{0}$ (−)	**$4.90 \cdot 10^{2} \pm 6.64 \cdot 10^{0}$**
	4	$4.17 \cdot 10^{2} \pm 1.91 \cdot 10^{1}$ (−)	$4.42 \cdot 10^{2} \pm 2.26 \cdot 10^{1}$ (−)	$4.70 \cdot 10^{2} \pm 9.65 \cdot 10^{0}$ (−)	**$4.86 \cdot 10^{2} \pm 8.77 \cdot 10^{0}$**
	5	$4.16 \cdot 10^{2} \pm 1.47 \cdot 10^{1}$ (−)	$4.38 \cdot 10^{2} \pm 1.54 \cdot 10^{1}$ (−)	$4.77 \cdot 10^{2} \pm 7.71 \cdot 10^{0}$ (−)	**$4.86 \cdot 10^{2} \pm 5.72 \cdot 10^{0}$**
	6	$4.25 \cdot 10^{2} \pm 1.80 \cdot 10^{1}$ (−)	$4.43 \cdot 10^{2} \pm 1.44 \cdot 10^{1}$ (−)	$4.72 \cdot 10^{2} \pm 1.09 \cdot 10^{1}$ (≈)	**$4.76 \cdot 10^{2} \pm 1.03 \cdot 10^{1}$**
	10	$4.15 \cdot 10^{2} \pm 2.92 \cdot 10^{1}$ (−)	$4.46 \cdot 10^{2} \pm 1.34 \cdot 10^{1}$ (−)	$4.71 \cdot 10^{2} \pm 6.75 \cdot 10^{0}$ (≈)	**$4.76 \cdot 10^{2} \pm 5.01 \cdot 10^{0}$**
	20	$4.26 \cdot 10^{2} \pm 1.70 \cdot 10^{1}$ (−)	$4.50 \cdot 10^{2} \pm 1.19 \cdot 10^{1}$ (≈)	**$4.68 \cdot 10^{2} \pm 4.84 \cdot 10^{0}$ (≈)**	$4.61 \cdot 10^{2} \pm 9.27 \cdot 10^{0}$
NBP	1	$7.71 \cdot 10^{5} \pm 4.45 \cdot 10^{3}$ (−)	$7.72 \cdot 10^{5} \pm 8.82 \cdot 10^{3}$ (−)	$7.98 \cdot 10^{5} \pm 4.53 \cdot 10^{2}$ (−)	**$8.01 \cdot 10^{5} \pm 8.88 \cdot 10^{0}$**
	2	$7.62 \cdot 10^{5} \pm 7.06 \cdot 10^{3}$ (−)	$7.63 \cdot 10^{5} \pm 5.29 \cdot 10^{3}$ (−)	$7.99 \cdot 10^{5} \pm 8.82 \cdot 10^{2}$ (−)	**$8.01 \cdot 10^{5} \pm 6.72 \cdot 10^{1}$**
	3	$7.67 \cdot 10^{5} \pm 6.99 \cdot 10^{3}$ (−)	$7.69 \cdot 10^{5} \pm 3.09 \cdot 10^{3}$ (−)	$7.99 \cdot 10^{5} \pm 3.30 \cdot 10^{2}$ (−)	**$8.01 \cdot 10^{5} \pm 1.03 \cdot 10^{1}$**
	4	$7.56 \cdot 10^{5} \pm 7.57 \cdot 10^{3}$ (−)	$7.65 \cdot 10^{5} \pm 7.17 \cdot 10^{3}$ (−)	$7.98 \cdot 10^{5} \pm 5.00 \cdot 10^{2}$ (−)	**$8.01 \cdot 10^{5} \pm 3.08 \cdot 10^{1}$**
	5	$7.63 \cdot 10^{5} \pm 5.21 \cdot 10^{3}$ (−)	$7.69 \cdot 10^{5} \pm 3.60 \cdot 10^{3}$ (−)	$7.97 \cdot 10^{5} \pm 8.66 \cdot 10^{2}$ (−)	**$8.00 \cdot 10^{5} \pm 1.36 \cdot 10^{2}$**
	6	$7.66 \cdot 10^{5} \pm 4.83 \cdot 10^{3}$ (−)	$7.68 \cdot 10^{5} \pm 6.03 \cdot 10^{3}$ (−)	$7.98 \cdot 10^{5} \pm 6.20 \cdot 10^{2}$ (−)	**$8.00 \cdot 10^{5} \pm 1.48 \cdot 10^{2}$**
	10	$7.66 \cdot 10^{5} \pm 5.68 \cdot 10^{3}$ (−)	$7.72 \cdot 10^{5} \pm 3.84 \cdot 10^{3}$ (−)	$7.96 \cdot 10^{5} \pm 1.01 \cdot 10^{3}$ (−)	**$7.99 \cdot 10^{5} \pm 5.26 \cdot 10^{2}$**
	20	$7.61 \cdot 10^{5} \pm 6.10 \cdot 10^{3}$ (−)	$7.68 \cdot 10^{5} \pm 4.32 \cdot 10^{3}$ (−)	$7.78 \cdot 10^{5} \pm 6.92 \cdot 10^{3}$ (−)	**$7.95 \cdot 10^{5} \pm 5.81 \cdot 10^{2}$**
DBD	1	$3.38 \cdot 10^{1} \pm 2.84 \cdot 10^{-1}$ (−)	$3.29 \cdot 10^{1} \pm 1.12 \cdot 10^{0}$ (−)	**$3.46 \cdot 10^{1} \pm 2.66 \cdot 10^{-2}$ (+)**	$3.46 \cdot 10^{1} \pm 1.06 \cdot 10^{-1}$
	2	$3.37 \cdot 10^{1} \pm 2.18 \cdot 10^{-1}$ (−)	$3.32 \cdot 10^{1} \pm 3.25 \cdot 10^{-1}$ (−)	**$3.46 \cdot 10^{1} \pm 2.00 \cdot 10^{-2}$ (+)**	$3.44 \cdot 10^{1} \pm 1.15 \cdot 10^{-1}$
	3	$3.40 \cdot 10^{1} \pm 1.14 \cdot 10^{-1}$ (−)	$3.32 \cdot 10^{1} \pm 4.90 \cdot 10^{-1}$ (−)	$3.46 \cdot 10^{1} \pm 2.35 \cdot 10^{-2}$ (−)	**$3.47 \cdot 10^{1} \pm 2.46 \cdot 10^{-3}$**
	4	$3.37 \cdot 10^{1} \pm 2.15 \cdot 10^{-1}$ (−)	$3.34 \cdot 10^{1} \pm 2.14 \cdot 10^{-1}$ (−)	$3.45 \cdot 10^{1} \pm 1.17 \cdot 10^{-1}$ (−)	**$3.47 \cdot 10^{1} \pm 4.22 \cdot 10^{-3}$**
	5	$3.37 \cdot 10^{1} \pm 3.16 \cdot 10^{-1}$ (−)	$3.34 \cdot 10^{1} \pm 2.82 \cdot 10^{-1}$ (−)	$3.46 \cdot 10^{1} \pm 8.13 \cdot 10^{-2}$ (≈)	**$3.46 \cdot 10^{1} \pm 5.64 \cdot 10^{-3}$**
	6	$3.39 \cdot 10^{1} \pm 9.60 \cdot 10^{-2}$ (−)	$3.31 \cdot 10^{1} \pm 6.13 \cdot 10^{-1}$ (−)	**$3.46 \cdot 10^{1} \pm 5.80 \cdot 10^{-2}$ (≈)**	$3.46 \cdot 10^{1} \pm 5.86 \cdot 10^{-2}$
	10	$3.38 \cdot 10^{1} \pm 1.96 \cdot 10^{-1}$ (−)	$3.34 \cdot 10^{1} \pm 3.22 \cdot 10^{-1}$ (−)	$3.45 \cdot 10^{1} \pm 4.46 \cdot 10^{-2}$ (≈)	**$3.46 \cdot 10^{1} \pm 1.62 \cdot 10^{-2}$**
	20	$3.37 \cdot 10^{1} \pm 2.51 \cdot 10^{-1}$ (−)	$3.31 \cdot 10^{1} \pm 3.73 \cdot 10^{-1}$ (−)	**$3.45 \cdot 10^{1} \pm 1.72 \cdot 10^{-2}$ (+)**	$3.44 \cdot 10^{1} \pm 2.75 \cdot 10^{-2}$
SRD	1	$3.04 \cdot 10^{6} \pm 1.91 \cdot 10^{4}$ (≈)	$2.96 \cdot 10^{6} \pm 5.49 \cdot 10^{4}$ (−)	**$3.07 \cdot 10^{6} \pm 5.91 \cdot 10^{2}$ (+)**	$3.06 \cdot 10^{6} \pm 6.25 \cdot 10^{3}$
	2	$3.06 \cdot 10^{6} \pm 8.30 \cdot 10^{3}$ (≈)	$3.06 \cdot 10^{6} \pm 4.81 \cdot 10^{3}$ (−)	**$3.06 \cdot 10^{6} \pm 7.83 \cdot 10^{2}$ (≈)**	$3.06 \cdot 10^{6} \pm 1.02 \cdot 10^{3}$
	3	$3.06 \cdot 10^{6} \pm 2.73 \cdot 10^{3}$ (+)	**$3.07 \cdot 10^{6} \pm 1.45 \cdot 10^{3}$ (+)**	$3.06 \cdot 10^{6} \pm 8.53 \cdot 10^{2}$ (−)	$3.06 \cdot 10^{6} \pm 7.05 \cdot 10^{2}$
	4	$3.06 \cdot 10^{6} \pm 9.27 \cdot 10^{2}$ (+)	**$3.06 \cdot 10^{6} \pm 9.80 \cdot 10^{2}$ (+)**	$3.06 \cdot 10^{6} \pm 1.90 \cdot 10^{3}$ (−)	$3.06 \cdot 10^{6} \pm 8.32 \cdot 10^{2}$
	5	$3.06 \cdot 10^{6} \pm 9.63 \cdot 10^{2}$ (+)	**$3.07 \cdot 10^{6} \pm 5.04 \cdot 10^{2}$ (+)**	$3.06 \cdot 10^{6} \pm 2.08 \cdot 10^{3}$ (−)	$3.06 \cdot 10^{6} \pm 6.44 \cdot 10^{2}$
	6	$3.06 \cdot 10^{6} \pm 1.18 \cdot 10^{3}$ (+)	**$3.07 \cdot 10^{6} \pm 6.02 \cdot 10^{2}$ (+)**	$3.06 \cdot 10^{6} \pm 1.90 \cdot 10^{3}$ (−)	$3.06 \cdot 10^{6} \pm 7.64 \cdot 10^{2}$
	10	$3.06 \cdot 10^{6} \pm 8.15 \cdot 10^{2}$ (+)	**$3.07 \cdot 10^{6} \pm 5.43 \cdot 10^{2}$ (+)**	$3.05 \cdot 10^{6} \pm 1.71 \cdot 10^{3}$ (−)	$3.06 \cdot 10^{6} \pm 1.25 \cdot 10^{3}$
	20	$3.06 \cdot 10^{6} \pm 9.27 \cdot 10^{2}$ (+)	**$3.06 \cdot 10^{6} \pm 7.54 \cdot 10^{2}$ (+)**	$3.04 \cdot 10^{6} \pm 2.83 \cdot 10^{3}$ (−)	$3.05 \cdot 10^{6} \pm 1.61 \cdot 10^{3}$

Table continues on next page.

Continuation of Table 3.

Function	p	SA-NSGA-II	IC-SA-NSGA-II	SAMO-COBRA	IOC-SAMO-COBRA
WB	1	$2.46 \cdot 10^{-1} \pm 5.47 \cdot 10^{-2}$ (−)	**$4.19 \cdot 10^{-1} \pm 2.11 \cdot 10^{-2}$ (+)**	$3.77 \cdot 10^{-1} \pm 1.01 \cdot 10^{-2}$ (−)	$4.15 \cdot 10^{-1} \pm 1.43 \cdot 10^{-3}$
	2	$3.46 \cdot 10^{-1} \pm 4.48 \cdot 10^{-2}$ (−)	**$4.20 \cdot 10^{-1} \pm 3.02 \cdot 10^{-3}$ (≈)**	$3.87 \cdot 10^{-1} \pm 1.70 \cdot 10^{-2}$ (−)	$4.15 \cdot 10^{-1} \pm 8.41 \cdot 10^{-3}$
	3	$3.73 \cdot 10^{-1} \pm 3.96 \cdot 10^{-2}$ (−)	**$4.23 \cdot 10^{-1} \pm 3.73 \cdot 10^{-3}$ (+)**	$4.06 \cdot 10^{-1} \pm 1.32 \cdot 10^{-2}$ (≈)	$4.14 \cdot 10^{-1} \pm 5.39 \cdot 10^{-3}$
	4	$3.96 \cdot 10^{-1} \pm 1.95 \cdot 10^{-2}$ (−)	**$4.23 \cdot 10^{-1} \pm 1.86 \cdot 10^{-3}$ (+)**	$3.86 \cdot 10^{-1} \pm 1.39 \cdot 10^{-2}$ (−)	$4.11 \cdot 10^{-1} \pm 7.66 \cdot 10^{-3}$
	5	$3.72 \cdot 10^{-1} \pm 6.19 \cdot 10^{-2}$ (−)	**$4.22 \cdot 10^{-1} \pm 2.55 \cdot 10^{-3}$ (+)**	$3.84 \cdot 10^{-1} \pm 2.23 \cdot 10^{-2}$ (−)	$4.14 \cdot 10^{-1} \pm 1.11 \cdot 10^{-2}$
	6	$3.83 \cdot 10^{-1} \pm 3.70 \cdot 10^{-2}$ (≈)	**$4.24 \cdot 10^{-1} \pm 1.84 \cdot 10^{-3}$ (+)**	$3.79 \cdot 10^{-1} \pm 1.73 \cdot 10^{-2}$ (−)	$4.02 \cdot 10^{-1} \pm 1.64 \cdot 10^{-2}$
	10	$3.92 \cdot 10^{-1} \pm 9.35 \cdot 10^{-3}$ (≈)	**$4.25 \cdot 10^{-1} \pm 2.64 \cdot 10^{-3}$ (+)**	$3.76 \cdot 10^{-1} \pm 1.69 \cdot 10^{-2}$ (−)	$3.96 \cdot 10^{-1} \pm 5.10 \cdot 10^{-3}$
	20	$3.67 \cdot 10^{-1} \pm 7.21 \cdot 10^{-2}$ (≈)	**$4.24 \cdot 10^{-1} \pm 2.30 \cdot 10^{-3}$ (+)**	$3.72 \cdot 10^{-1} \pm 1.20 \cdot 10^{-2}$ (≈)	$3.78 \cdot 10^{-1} \pm 1.24 \cdot 10^{-2}$
BICOP1	1	$6.38 \cdot 10^{-2} \pm 9.98 \cdot 10^{-2}$ (≈)	$9.60 \cdot 10^{-2} \pm 1.05 \cdot 10^{-1}$ (≈)	**$1.23 \cdot 10^{-1} \pm 1.62 \cdot 10^{-1}$ (≈)**	$7.91 \cdot 10^{-2} \pm 1.16 \cdot 10^{-1}$
	2	$5.98 \cdot 10^{-1} \pm 1.92 \cdot 10^{-2}$ (+)	**$6.07 \cdot 10^{-1} \pm 1.34 \cdot 10^{-2}$ (+)**	$3.17 \cdot 10^{-1} \pm 2.60 \cdot 10^{-1}$ (≈)	$4.16 \cdot 10^{-1} \pm 2.11 \cdot 10^{-1}$
	3	$6.29 \cdot 10^{-1} \pm 1.03 \cdot 10^{-2}$ (≈)	**$6.36 \cdot 10^{-1} \pm 4.84 \cdot 10^{-3}$ (≈)**	$5.06 \cdot 10^{-1} \pm 2.54 \cdot 10^{-1}$ (≈)	$5.79 \cdot 10^{-1} \pm 8.34 \cdot 10^{-2}$
	4	$6.41 \cdot 10^{-1} \pm 6.41 \cdot 10^{-3}$ (≈)	**$6.43 \cdot 10^{-1} \pm 6.09 \cdot 10^{-3}$ (≈)**	$6.34 \cdot 10^{-1} \pm 1.06 \cdot 10^{-2}$ (≈)	$6.09 \cdot 10^{-1} \pm 7.65 \cdot 10^{-2}$
	5	$6.49 \cdot 10^{-1} \pm 4.38 \cdot 10^{-3}$ (+)	**$6.50 \cdot 10^{-1} \pm 5.78 \cdot 10^{-3}$ (+)**	$6.25 \cdot 10^{-1} \pm 1.39 \cdot 10^{-2}$ (≈)	$6.20 \cdot 10^{-1} \pm 1.31 \cdot 10^{-2}$
	6	$6.53 \cdot 10^{-1} \pm 4.50 \cdot 10^{-3}$ (+)	**$6.53 \cdot 10^{-1} \pm 3.46 \cdot 10^{-3}$ (+)**	$5.89 \cdot 10^{-1} \pm 1.88 \cdot 10^{-2}$ (≈)	$5.99 \cdot 10^{-1} \pm 1.37 \cdot 10^{-2}$
	10	**$6.60 \cdot 10^{-1} \pm 1.08 \cdot 10^{-3}$ (+)**	$6.59 \cdot 10^{-1} \pm 1.88 \cdot 10^{-3}$ (+)	$4.91 \cdot 10^{-1} \pm 4.87 \cdot 10^{-2}$ (≈)	$5.08 \cdot 10^{-1} \pm 3.28 \cdot 10^{-2}$
	20	$6.60 \cdot 10^{-1} \pm 8.35 \cdot 10^{-4}$ (+)	**$6.60 \cdot 10^{-1} \pm 7.77 \cdot 10^{-4}$ (+)**	$2.98 \cdot 10^{-1} \pm 9.13 \cdot 10^{-2}$ (≈)	$2.68 \cdot 10^{-1} \pm 7.79 \cdot 10^{-2}$
BICOP2	1	$1.04 \cdot 10^{-1} \pm 2.31 \cdot 10^{-2}$ (−)	$1.17 \cdot 10^{-1} \pm 2.88 \cdot 10^{-2}$ (−)	$2.16 \cdot 10^{-1} \pm 4.01 \cdot 10^{-2}$ (−)	**$2.82 \cdot 10^{-1} \pm 1.79 \cdot 10^{-2}$**
	2	$1.06 \cdot 10^{-1} \pm 3.53 \cdot 10^{-2}$ (−)	$1.76 \cdot 10^{-1} \pm 3.44 \cdot 10^{-2}$ (−)	$2.15 \cdot 10^{-1} \pm 4.28 \cdot 10^{-2}$ (−)	**$3.11 \cdot 10^{-1} \pm 3.03 \cdot 10^{-2}$**
	3	$1.22 \cdot 10^{-1} \pm 3.01 \cdot 10^{-2}$ (−)	$1.53 \cdot 10^{-1} \pm 4.97 \cdot 10^{-2}$ (−)	$2.23 \cdot 10^{-1} \pm 4.93 \cdot 10^{-2}$ (−)	**$3.01 \cdot 10^{-1} \pm 5.25 \cdot 10^{-2}$**
	4	$1.21 \cdot 10^{-1} \pm 3.67 \cdot 10^{-2}$ (−)	$1.67 \cdot 10^{-1} \pm 5.22 \cdot 10^{-2}$ (≈)	$2.34 \cdot 10^{-1} \pm 5.56 \cdot 10^{-2}$ (≈)	**$2.50 \cdot 10^{-1} \pm 7.37 \cdot 10^{-2}$**
	5	$1.27 \cdot 10^{-1} \pm 4.19 \cdot 10^{-2}$ (−)	$1.77 \cdot 10^{-1} \pm 4.21 \cdot 10^{-2}$ (≈)	**$2.53 \cdot 10^{-1} \pm 3.15 \cdot 10^{-2}$ (≈)**	$2.24 \cdot 10^{-1} \pm 6.76 \cdot 10^{-2}$
	6	$1.26 \cdot 10^{-1} \pm 3.79 \cdot 10^{-2}$ (−)	$1.55 \cdot 10^{-1} \pm 4.65 \cdot 10^{-2}$ (≈)	**$2.65 \cdot 10^{-1} \pm 1.68 \cdot 10^{-2}$ (≈)**	$2.13 \cdot 10^{-1} \pm 6.50 \cdot 10^{-2}$
	10	$1.53 \cdot 10^{-1} \pm 3.98 \cdot 10^{-2}$ (−)	$1.45 \cdot 10^{-1} \pm 3.91 \cdot 10^{-2}$ (−)	$2.38 \cdot 10^{-1} \pm 2.77 \cdot 10^{-2}$ (≈)	**$2.44 \cdot 10^{-1} \pm 4.47 \cdot 10^{-2}$**
	20	$1.54 \cdot 10^{-1} \pm 4.41 \cdot 10^{-2}$ (−)	$1.50 \cdot 10^{-1} \pm 4.22 \cdot 10^{-2}$ (−)	$2.25 \cdot 10^{-1} \pm 2.07 \cdot 10^{-2}$ (−)	**$2.72 \cdot 10^{-1} \pm 1.60 \cdot 10^{-2}$**
MW1	1	$0.00 \cdot 10^{0} \pm 0.00 \cdot 10^{0}$ (−)	$2.73 \cdot 10^{-1} \pm 4.54 \cdot 10^{-2}$ (−)	$1.66 \cdot 10^{-2} \pm 3.27 \cdot 10^{-2}$ (−)	**$3.99 \cdot 10^{-1} \pm 5.85 \cdot 10^{-5}$**
	2	$2.40 \cdot 10^{-1} \pm 6.35 \cdot 10^{-2}$ (−)	$3.37 \cdot 10^{-1} \pm 5.49 \cdot 10^{-3}$ (−)	$1.92 \cdot 10^{-1} \pm 1.13 \cdot 10^{-1}$ (−)	**$3.98 \cdot 10^{-1} \pm 8.02 \cdot 10^{-5}$**
	3	$2.82 \cdot 10^{-1} \pm 4.11 \cdot 10^{-2}$ (−)	$3.40 \cdot 10^{-1} \pm 1.00 \cdot 10^{-2}$ (−)	$3.10 \cdot 10^{-1} \pm 7.11 \cdot 10^{-2}$ (−)	**$3.98 \cdot 10^{-1} \pm 1.55 \cdot 10^{-4}$**
	4	$3.24 \cdot 10^{-1} \pm 3.86 \cdot 10^{-2}$ (−)	$3.51 \cdot 10^{-1} \pm 1.15 \cdot 10^{-2}$ (−)	$2.20 \cdot 10^{-1} \pm 1.61 \cdot 10^{-1}$ (−)	**$3.98 \cdot 10^{-1} \pm 1.56 \cdot 10^{-4}$**
	5	$3.49 \cdot 10^{-1} \pm 1.34 \cdot 10^{-2}$ (−)	$3.66 \cdot 10^{-1} \pm 6.28 \cdot 10^{-3}$ (−)	$2.09 \cdot 10^{-1} \pm 1.73 \cdot 10^{-1}$ (−)	**$3.98 \cdot 10^{-1} \pm 1.70 \cdot 10^{-4}$**
	6	$3.56 \cdot 10^{-1} \pm 1.61 \cdot 10^{-2}$ (−)	$3.80 \cdot 10^{-1} \pm 5.43 \cdot 10^{-3}$ (−)	$2.55 \cdot 10^{-1} \pm 1.25 \cdot 10^{-1}$ (−)	**$3.98 \cdot 10^{-1} \pm 5.34 \cdot 10^{-4}$**
	10	$3.56 \cdot 10^{-1} \pm 2.92 \cdot 10^{-2}$ (−)	$3.90 \cdot 10^{-1} \pm 1.81 \cdot 10^{-3}$ (−)	$1.63 \cdot 10^{-1} \pm 1.23 \cdot 10^{-1}$ (−)	**$3.97 \cdot 10^{-1} \pm 1.10 \cdot 10^{-3}$**
	20	$3.72 \cdot 10^{-1} \pm 1.15 \cdot 10^{-2}$ (≈)	**$3.93 \cdot 10^{-1} \pm 1.16 \cdot 10^{-3}$ (+)**	$2.09 \cdot 10^{-1} \pm 1.15 \cdot 10^{-1}$ (≈)	$2.73 \cdot 10^{-1} \pm 1.46 \cdot 10^{-1}$
MW2	1	$2.86 \cdot 10^{-2} \pm 5.73 \cdot 10^{-2}$ (−)	**$4.24 \cdot 10^{-1} \pm 1.51 \cdot 10^{-2}$ (+)**	$1.60 \cdot 10^{-1} \pm 6.38 \cdot 10^{-2}$ (−)	$3.85 \cdot 10^{-1} \pm 2.32 \cdot 10^{-2}$
	2	$2.63 \cdot 10^{-1} \pm 6.16 \cdot 10^{-2}$ (−)	**$4.33 \cdot 10^{-1} \pm 6.25 \cdot 10^{-3}$ (≈)**	$1.82 \cdot 10^{-1} \pm 1.22 \cdot 10^{-1}$ (−)	$4.19 \cdot 10^{-1} \pm 2.06 \cdot 10^{-2}$
	3	$2.93 \cdot 10^{-1} \pm 8.71 \cdot 10^{-2}$ (−)	**$4.41 \cdot 10^{-1} \pm 8.60 \cdot 10^{-3}$ (≈)**	$1.98 \cdot 10^{-1} \pm 1.12 \cdot 10^{-1}$ (−)	$4.00 \cdot 10^{-1} \pm 6.68 \cdot 10^{-2}$
	4	$3.42 \cdot 10^{-1} \pm 8.05 \cdot 10^{-2}$ (≈)	**$4.40 \cdot 10^{-1} \pm 8.97 \cdot 10^{-3}$ (+)**	$1.66 \cdot 10^{-1} \pm 1.03 \cdot 10^{-1}$ (−)	$3.47 \cdot 10^{-1} \pm 7.45 \cdot 10^{-2}$
	5	$3.38 \cdot 10^{-1} \pm 7.90 \cdot 10^{-2}$ (−)	**$4.42 \cdot 10^{-1} \pm 8.72 \cdot 10^{-3}$ (+)**	$1.35 \cdot 10^{-1} \pm 7.24 \cdot 10^{-2}$ (−)	$3.96 \cdot 10^{-1} \pm 5.05 \cdot 10^{-2}$
	6	$3.40 \cdot 10^{-1} \pm 7.84 \cdot 10^{-2}$ (−)	**$4.42 \cdot 10^{-1} \pm 7.95 \cdot 10^{-3}$ (+)**	$1.43 \cdot 10^{-1} \pm 9.95 \cdot 10^{-2}$ (−)	$4.11 \cdot 10^{-1} \pm 2.91 \cdot 10^{-2}$
	10	$3.20 \cdot 10^{-1} \pm 1.07 \cdot 10^{-1}$ (≈)	**$4.45 \cdot 10^{-1} \pm 1.08 \cdot 10^{-2}$ (+)**	$1.04 \cdot 10^{-1} \pm 8.28 \cdot 10^{-2}$ (−)	$3.78 \cdot 10^{-1} \pm 3.47 \cdot 10^{-2}$
	20	$3.33 \cdot 10^{-1} \pm 9.66 \cdot 10^{-2}$ (≈)	**$4.49 \cdot 10^{-1} \pm 9.99 \cdot 10^{-3}$ (+)**	$1.31 \cdot 10^{-1} \pm 1.09 \cdot 10^{-1}$ (−)	$3.10 \cdot 10^{-1} \pm 4.45 \cdot 10^{-2}$
MW3	1	$1.04 \cdot 10^{-1} \pm 1.48 \cdot 10^{-1}$ (−)	$4.10 \cdot 10^{-1} \pm 9.41 \cdot 10^{-3}$ (−)	$3.72 \cdot 10^{-1} \pm 2.46 \cdot 10^{-2}$ (−)	**$4.50 \cdot 10^{-1} \pm 9.80 \cdot 10^{-4}$**
	2	$4.06 \cdot 10^{-1} \pm 1.30 \cdot 10^{-2}$ (−)	$4.22 \cdot 10^{-1} \pm 3.38 \cdot 10^{-3}$ (−)	$4.07 \cdot 10^{-1} \pm 8.56 \cdot 10^{-3}$ (−)	**$4.51 \cdot 10^{-1} \pm 1.70 \cdot 10^{-3}$**
	3	$4.22 \cdot 10^{-1} \pm 6.32 \cdot 10^{-3}$ (−)	$4.29 \cdot 10^{-1} \pm 3.08 \cdot 10^{-3}$ (−)	$4.29 \cdot 10^{-1} \pm 1.08 \cdot 10^{-2}$ (−)	**$4.52 \cdot 10^{-1} \pm 4.92 \cdot 10^{-4}$**
	4	$4.22 \cdot 10^{-1} \pm 2.48 \cdot 10^{-3}$ (−)	$4.32 \cdot 10^{-1} \pm 3.15 \cdot 10^{-3}$ (−)	$4.43 \cdot 10^{-1} \pm 5.59 \cdot 10^{-3}$ (−)	**$4.52 \cdot 10^{-1} \pm 2.32 \cdot 10^{-4}$**
	5	$4.26 \cdot 10^{-1} \pm 4.97 \cdot 10^{-3}$ (−)	$4.36 \cdot 10^{-1} \pm 2.39 \cdot 10^{-3}$ (−)	$4.43 \cdot 10^{-1} \pm 3.76 \cdot 10^{-3}$ (−)	**$4.51 \cdot 10^{-1} \pm 1.01 \cdot 10^{-3}$**
	6	$4.25 \cdot 10^{-1} \pm 2.38 \cdot 10^{-3}$ (−)	$4.37 \cdot 10^{-1} \pm 3.88 \cdot 10^{-3}$ (−)	$4.42 \cdot 10^{-1} \pm 3.55 \cdot 10^{-3}$ (−)	**$4.50 \cdot 10^{-1} \pm 7.45 \cdot 10^{-4}$**
	10	$4.29 \cdot 10^{-1} \pm 3.28 \cdot 10^{-3}$ (−)	$4.41 \cdot 10^{-1} \pm 1.21 \cdot 10^{-3}$ (−)	$4.36 \cdot 10^{-1} \pm 1.97 \cdot 10^{-3}$ (−)	**$4.48 \cdot 10^{-1} \pm 6.46 \cdot 10^{-4}$**
	20	$4.28 \cdot 10^{-1} \pm 4.92 \cdot 10^{-3}$ (−)	$4.40 \cdot 10^{-1} \pm 1.41 \cdot 10^{-3}$ (−)	$4.29 \cdot 10^{-1} \pm 2.55 \cdot 10^{-3}$ (−)	**$4.44 \cdot 10^{-1} \pm 7.06 \cdot 10^{-4}$**
MW11	1	$6.65 \cdot 10^{-1} \pm 2.63 \cdot 10^{-1}$ (−)	**$1.36 \cdot 10^{0} \pm 4.41 \cdot 10^{-2}$ (+)**	$9.80 \cdot 10^{-1} \pm 3.80 \cdot 10^{-1}$ (≈)	$1.10 \cdot 10^{0} \pm 1.99 \cdot 10^{-1}$
	2	$1.17 \cdot 10^{0} \pm 1.75 \cdot 10^{-1}$ (≈)	**$1.42 \cdot 10^{0} \pm 2.43 \cdot 10^{-2}$ (+)**	$9.82 \cdot 10^{-1} \pm 1.74 \cdot 10^{-1}$ (−)	$1.17 \cdot 10^{0} \pm 1.55 \cdot 10^{-1}$
	3	$1.09 \cdot 10^{0} \pm 2.30 \cdot 10^{-1}$ (−)	$1.44 \cdot 10^{0} \pm 1.83 \cdot 10^{-2}$ (−)	$9.92 \cdot 10^{-1} \pm 1.97 \cdot 10^{-1}$ (−)	**$1.49 \cdot 10^{0} \pm 4.23 \cdot 10^{-2}$**
	4	$1.03 \cdot 10^{0} \pm 2.44 \cdot 10^{-1}$ (−)	$1.46 \cdot 10^{0} \pm 1.81 \cdot 10^{-2}$ (−)	$9.99 \cdot 10^{-1} \pm 1.23 \cdot 10^{-1}$ (−)	**$1.51 \cdot 10^{0} \pm 1.40 \cdot 10^{-2}$**
	5	$1.04 \cdot 10^{0} \pm 2.41 \cdot 10^{-1}$ (−)	$1.46 \cdot 10^{0} \pm 9.42 \cdot 10^{-3}$ (−)	$1.06 \cdot 10^{0} \pm 1.80 \cdot 10^{-1}$ (−)	**$1.52 \cdot 10^{0} \pm 8.07 \cdot 10^{-3}$**
	6	$9.08 \cdot 10^{-1} \pm 1.57 \cdot 10^{-1}$ (−)	$1.48 \cdot 10^{0} \pm 8.50 \cdot 10^{-3}$ (−)	$9.75 \cdot 10^{-1} \pm 2.81 \cdot 10^{-1}$ (−)	**$1.52 \cdot 10^{0} \pm 1.26 \cdot 10^{-2}$**
	10	$9.52 \cdot 10^{-1} \pm 2.21 \cdot 10^{-1}$ (−)	$1.49 \cdot 10^{0} \pm 8.09 \cdot 10^{-3}$ (−)	$8.27 \cdot 10^{-1} \pm 1.73 \cdot 10^{-1}$ (−)	**$1.52 \cdot 10^{0} \pm 5.60 \cdot 10^{-3}$**
	20	$8.02 \cdot 10^{-1} \pm 1.80 \cdot 10^{-1}$ (−)	$1.49 \cdot 10^{0} \pm 1.53 \cdot 10^{-2}$ (−)	$8.78 \cdot 10^{-1} \pm 5.96 \cdot 10^{-2}$ (−)	**$1.50 \cdot 10^{0} \pm 6.86 \cdot 10^{-3}$**
TRICOP	1	$4.47 \cdot 10^{1} \pm 2.03 \cdot 10^{0}$ (−)	$4.57 \cdot 10^{1} \pm 1.19 \cdot 10^{0}$ (−)	**$4.97 \cdot 10^{1} \pm 6.30 \cdot 10^{-3}$ (≈)**	$4.97 \cdot 10^{1} \pm 3.81 \cdot 10^{-2}$
	2	$4.19 \cdot 10^{1} \pm 1.56 \cdot 10^{0}$ (−)	$4.55 \cdot 10^{1} \pm 7.37 \cdot 10^{-1}$ (−)	$4.96 \cdot 10^{1} \pm 2.76 \cdot 10^{-2}$ (≈)	**$4.97 \cdot 10^{1} \pm 3.93 \cdot 10^{-2}$**
	3	$4.31 \cdot 10^{1} \pm 1.75 \cdot 10^{0}$ (−)	$4.63 \cdot 10^{1} \pm 5.46 \cdot 10^{-1}$ (−)	**$4.97 \cdot 10^{1} \pm 1.93 \cdot 10^{-2}$ (+)**	$4.96 \cdot 10^{1} \pm 3.41 \cdot 10^{-2}$
	4	$4.31 \cdot 10^{1} \pm 1.50 \cdot 10^{0}$ (−)	$4.63 \cdot 10^{1} \pm 6.92 \cdot 10^{-1}$ (−)	**$4.96 \cdot 10^{1} \pm 4.30 \cdot 10^{-2}$ (≈)**	$4.96 \cdot 10^{1} \pm 3.22 \cdot 10^{-2}$
	5	$4.26 \cdot 10^{1} \pm 1.45 \cdot 10^{0}$ (−)	$4.66 \cdot 10^{1} \pm 3.57 \cdot 10^{-1}$ (−)	**$4.97 \cdot 10^{1} \pm 2.46 \cdot 10^{-2}$ (≈)**	$4.97 \cdot 10^{1} \pm 2.45 \cdot 10^{-2}$
	6	$4.36 \cdot 10^{1} \pm 1.48 \cdot 10^{0}$ (−)	$4.63 \cdot 10^{1} \pm 5.04 \cdot 10^{-1}$ (−)	**$4.97 \cdot 10^{1} \pm 3.34 \cdot 10^{-2}$ (≈)**	$4.97 \cdot 10^{1} \pm 5.05 \cdot 10^{-2}$
	10	$4.40 \cdot 10^{1} \pm 1.43 \cdot 10^{0}$ (−)	$4.71 \cdot 10^{1} \pm 3.92 \cdot 10^{-1}$ (−)	**$4.97 \cdot 10^{1} \pm 3.00 \cdot 10^{-2}$ (≈)**	$4.97 \cdot 10^{1} \pm 1.95 \cdot 10^{-2}$
	20	$4.55 \cdot 10^{1} \pm 8.60 \cdot 10^{-1}$ (−)	$4.76 \cdot 10^{1} \pm 3.49 \cdot 10^{-1}$ (−)	$4.95 \cdot 10^{1} \pm 4.39 \cdot 10^{-2}$ (≈)	**$4.95 \cdot 10^{1} \pm 2.77 \cdot 10^{-2}$**
SPD	1	$5.04 \cdot 10^{9} \pm 8.11 \cdot 10^{7}$ (−)	$4.95 \cdot 10^{9} \pm 1.07 \cdot 10^{8}$ (−)	$5.87 \cdot 10^{9} \pm 1.68 \cdot 10^{7}$ (−)	**$6.01 \cdot 10^{9} \pm 1.95 \cdot 10^{6}$**
	2	$4.88 \cdot 10^{9} \pm 1.58 \cdot 10^{8}$ (−)	$5.05 \cdot 10^{9} \pm 6.73 \cdot 10^{7}$ (−)	$5.91 \cdot 10^{9} \pm 2.36 \cdot 10^{7}$ (−)	**$6.02 \cdot 10^{9} \pm 3.06 \cdot 10^{6}$**
	3	$5.02 \cdot 10^{9} \pm 7.48 \cdot 10^{7}$ (−)	$5.08 \cdot 10^{9} \pm 8.10 \cdot 10^{7}$ (−)	$5.93 \cdot 10^{9} \pm 9.35 \cdot 10^{6}$ (−)	**$6.01 \cdot 10^{9} \pm 2.47 \cdot 10^{6}$**
	4	$4.97 \cdot 10^{9} \pm 1.19 \cdot 10^{8}$ (−)	$5.02 \cdot 10^{9} \pm 7.63 \cdot 10^{7}$ (−)	$5.93 \cdot 10^{9} \pm 6.67 \cdot 10^{6}$ (−)	**$6.01 \cdot 10^{9} \pm 3.71 \cdot 10^{6}$**
	5	$5.06 \cdot 10^{9} \pm 8.50 \cdot 10^{7}$ (−)	$5.03 \cdot 10^{9} \pm 1.44 \cdot 10^{8}$ (−)	$5.91 \cdot 10^{9} \pm 1.68 \cdot 10^{7}$ (−)	**$6.01 \cdot 10^{9} \pm 3.20 \cdot 10^{6}$**
	6	$5.07 \cdot 10^{9} \pm 5.48 \cdot 10^{7}$ (−)	$5.05 \cdot 10^{9} \pm 5.88 \cdot 10^{7}$ (−)	$5.91 \cdot 10^{9} \pm 1.17 \cdot 10^{7}$ (−)	**$6.00 \cdot 10^{9} \pm 5.75 \cdot 10^{6}$**
	10	$5.08 \cdot 10^{9} \pm 6.05 \cdot 10^{7}$ (−)	$5.06 \cdot 10^{9} \pm 9.73 \cdot 10^{7}$ (−)	$5.86 \cdot 10^{9} \pm 2.25 \cdot 10^{7}$ (−)	**$5.99 \cdot 10^{9} \pm 3.85 \cdot 10^{6}$**
	20	$5.08 \cdot 10^{9} \pm 1.15 \cdot 10^{8}$ (−)	$5.04 \cdot 10^{9} \pm 8.77 \cdot 10^{7}$ (−)	$5.80 \cdot 10^{9} \pm 2.77 \cdot 10^{7}$ (−)	**$5.93 \cdot 10^{9} \pm 1.02 \cdot 10^{7}$**
CSI	1	$7.31 \cdot 10^{0} \pm 9.10 \cdot 10^{-2}$ (−)	$6.06 \cdot 10^{0} \pm 3.59 \cdot 10^{-1}$ (−)	$8.33 \cdot 10^{0} \pm 6.84 \cdot 10^{-2}$ (≈)	**$8.35 \cdot 10^{0} \pm 9.34 \cdot 10^{-2}$**
	2	$7.11 \cdot 10^{0} \pm 1.34 \cdot 10^{-1}$ (−)	$7.04 \cdot 10^{0} \pm 7.90 \cdot 10^{-2}$ (−)	$8.45 \cdot 10^{0} \pm 1.70 \cdot 10^{-2}$ (≈)	**$8.46 \cdot 10^{0} \pm 2.16 \cdot 10^{-2}$**
	3	$7.20 \cdot 10^{0} \pm 1.01 \cdot 10^{-1}$ (−)	$7.12 \cdot 10^{0} \pm 1.45 \cdot 10^{-1}$ (−)	$8.47 \cdot 10^{0} \pm 1.32 \cdot 10^{-2}$ (−)	**$8.49 \cdot 10^{0} \pm 5.44 \cdot 10^{-3}$**
	4	$7.13 \cdot 10^{0} \pm 6.49 \cdot 10^{-2}$ (−)	$7.14 \cdot 10^{0} \pm 7.68 \cdot 10^{-2}$ (−)	$8.46 \cdot 10^{0} \pm 1.36 \cdot 10^{-2}$ (−)	**$8.48 \cdot 10^{0} \pm 9.85 \cdot 10^{-3}$**
	5	$7.09 \cdot 10^{0} \pm 6.13 \cdot 10^{-2}$ (−)	$7.10 \cdot 10^{0} \pm 1.75 \cdot 10^{-1}$ (−)	$8.46 \cdot 10^{0} \pm 1.11 \cdot 10^{-2}$ (−)	**$8.48 \cdot 10^{0} \pm 7.05 \cdot 10^{-3}$**
	6	$7.14 \cdot 10^{0} \pm 1.04 \cdot 10^{-1}$ (−)	$7.17 \cdot 10^{0} \pm 9.63 \cdot 10^{-2}$ (−)	$8.45 \cdot 10^{0} \pm 1.00 \cdot 10^{-2}$ (≈)	**$8.46 \cdot 10^{0} \pm 1.21 \cdot 10^{-2}$**
	10	$7.21 \cdot 10^{0} \pm 1.05 \cdot 10^{-1}$ (−)	$7.19 \cdot 10^{0} \pm 6.00 \cdot 10^{-2}$ (−)	$8.33 \cdot 10^{0} \pm 1.89 \cdot 10^{-2}$ (−)	**$8.35 \cdot 10^{0} \pm 1.28 \cdot 10^{-2}$**
	20	$7.10 \cdot 10^{0} \pm 1.68 \cdot 10^{-1}$ (−)	$7.22 \cdot 10^{0} \pm 1.10 \cdot 10^{-1}$ (−)	**$7.90 \cdot 10^{0} \pm 8.04 \cdot 10^{-2}$ (≈)**	$7.86 \cdot 10^{0} \pm 4.02 \cdot 10^{-2}$
WP	1	$1.03 \cdot 10^{18} \pm 3.31 \cdot 10^{17}$ (−)	$1.00 \cdot 10^{18} \pm 2.70 \cdot 10^{17}$ (−)	$3.42 \cdot 10^{18} \pm 8.57 \cdot 10^{15}$ (≈)	**$3.42 \cdot 10^{18} \pm 7.12 \cdot 10^{15}$**
	2	$1.47 \cdot 10^{18} \pm 2.17 \cdot 10^{17}$ (−)	$1.36 \cdot 10^{18} \pm 4.05 \cdot 10^{17}$ (−)	**$3.44 \cdot 10^{18} \pm 1.62 \cdot 10^{15}$ (+)**	$3.42 \cdot 10^{18} \pm 5.17 \cdot 10^{15}$
	3	$1.61 \cdot 10^{18} \pm 2.10 \cdot 10^{17}$ (−)	$1.47 \cdot 10^{18} \pm 2.37 \cdot 10^{17}$ (−)	$3.42 \cdot 10^{18} \pm 1.04 \cdot 10^{16}$ (≈)	**$3.42 \cdot 10^{18} \pm 8.90 \cdot 10^{15}$**
	4	$1.70 \cdot 10^{18} \pm 1.89 \cdot 10^{17}$ (−)	$1.53 \cdot 10^{18} \pm 2.55 \cdot 10^{17}$ (−)	**$3.42 \cdot 10^{18} \pm 7.99 \cdot 10^{15}$ (≈)**	$3.42 \cdot 10^{18} \pm 5.15 \cdot 10^{15}$
	5	$1.69 \cdot 10^{18} \pm 2.27 \cdot 10^{17}$ (−)	$1.57 \cdot 10^{18} \pm 3.08 \cdot 10^{17}$ (−)	**$3.42 \cdot 10^{18} \pm 6.03 \cdot 10^{15}$ (+)**	$3.41 \cdot 10^{18} \pm 1.46 \cdot 10^{16}$
	6	$1.86 \cdot 10^{18} \pm 1.69 \cdot 10^{17}$ (−)	$1.58 \cdot 10^{18} \pm 3.50 \cdot 10^{17}$ (−)	**$3.42 \cdot 10^{18} \pm 5.69 \cdot 10^{15}$ (+)**	$3.40 \cdot 10^{18} \pm 1.47 \cdot 10^{16}$
	10	$1.97 \cdot 10^{18} \pm 2.78 \cdot 10^{17}$ (−)	$1.70 \cdot 10^{18} \pm 2.86 \cdot 10^{17}$ (−)	$3.37 \cdot 10^{18} \pm 1.51 \cdot 10^{16}$ (−)	**$3.40 \cdot 10^{18} \pm 6.14 \cdot 10^{15}$**
	20	$1.90 \cdot 10^{18} \pm 2.51 \cdot 10^{17}$ (−)	$1.79 \cdot 10^{18} \pm 2.84 \cdot 10^{17}$ (−)	**$3.37 \cdot 10^{18} \pm 7.01 \cdot 10^{15}$ (≈)**	$3.37 \cdot 10^{18} \pm 1.09 \cdot 10^{16}$
Wilcoxon test summary		151−, 13 ≈, 12+	138−, 10 ≈, 28+	95−, 71 ≈, 10+	reference algorithm (total 176 comparisons)
mean Friedman rank		3.35	2.79	2.30	**1.56**

Table 4: IGD+ score ± standard deviation of IGD+, Wilcoxon rank sum test with probability value = 0.05 (reference algorithm: IOC-SAMO-COBRA), per test function and candidate solutions size p. The lowest IGD+ per row is reported in **bold**, best scoring algorithm per test function is highlighted.

Function	p	SA-NSGA-II	IC-SA-NSGA-II	SAMO-COBRA	IOC-SAMO-COBRA
BNH	1	$1.77 \cdot 10^{-2} \pm 2.89 \cdot 10^{-3}$ (−)	$2.15 \cdot 10^{-2} \pm 3.60 \cdot 10^{-3}$ (−)	$2.06 \cdot 10^{-3} \pm 1.44 \cdot 10^{-5}$ (≈)	**$2.06 \cdot 10^{-3} \pm 1.08 \cdot 10^{-5}$**
	2	$1.95 \cdot 10^{-2} \pm 1.39 \cdot 10^{-3}$ (−)	$2.21 \cdot 10^{-2} \pm 2.89 \cdot 10^{-3}$ (−)	**$2.12 \cdot 10^{-3} \pm 1.14 \cdot 10^{-5}$** (≈)	$2.12 \cdot 10^{-3} \pm 1.52 \cdot 10^{-5}$
	3	$1.81 \cdot 10^{-2} \pm 2.50 \cdot 10^{-3}$ (−)	$2.03 \cdot 10^{-2} \pm 2.92 \cdot 10^{-3}$ (−)	$2.13 \cdot 10^{-3} \pm 3.93 \cdot 10^{-5}$ (≈)	**$2.12 \cdot 10^{-3} \pm 2.83 \cdot 10^{-5}$**
	4	$1.75 \cdot 10^{-2} \pm 1.61 \cdot 10^{-3}$ (−)	$2.11 \cdot 10^{-2} \pm 1.95 \cdot 10^{-3}$ (−)	**$2.12 \cdot 10^{-3} \pm 4.18 \cdot 10^{-5}$** (≈)	$2.15 \cdot 10^{-3} \pm 4.39 \cdot 10^{-5}$
	5	$1.94 \cdot 10^{-2} \pm 2.28 \cdot 10^{-3}$ (−)	$2.05 \cdot 10^{-2} \pm 2.19 \cdot 10^{-3}$ (−)	$2.14 \cdot 10^{-3} \pm 2.59 \cdot 10^{-5}$ (≈)	**$2.13 \cdot 10^{-3} \pm 3.13 \cdot 10^{-5}$**
	6	$1.84 \cdot 10^{-2} \pm 1.03 \cdot 10^{-3}$ (−)	$1.79 \cdot 10^{-2} \pm 2.52 \cdot 10^{-3}$ (−)	$2.09 \cdot 10^{-3} \pm 5.02 \cdot 10^{-5}$ (≈)	**$2.09 \cdot 10^{-3} \pm 3.86 \cdot 10^{-5}$**
	10	$1.85 \cdot 10^{-2} \pm 1.72 \cdot 10^{-3}$ (−)	$1.97 \cdot 10^{-2} \pm 2.28 \cdot 10^{-3}$ (−)	**$2.40 \cdot 10^{-3} \pm 5.77 \cdot 10^{-5}$** (≈)	$2.42 \cdot 10^{-3} \pm 5.79 \cdot 10^{-5}$
	20	$1.67 \cdot 10^{-2} \pm 2.40 \cdot 10^{-3}$ (−)	$1.89 \cdot 10^{-2} \pm 2.32 \cdot 10^{-3}$ (−)	**$3.03 \cdot 10^{-3} \pm 5.58 \cdot 10^{-5}$** (≈)	$3.06 \cdot 10^{-3} \pm 6.72 \cdot 10^{-5}$
CEXP	1	$1.79 \cdot 10^{-2} \pm 2.35 \cdot 10^{-3}$ (−)	$1.83 \cdot 10^{-2} \pm 6.60 \cdot 10^{-3}$ (−)	$2.54 \cdot 10^{-3} \pm 4.97 \cdot 10^{-5}$ (−)	**$2.17 \cdot 10^{-3} \pm 1.17 \cdot 10^{-5}$**
	2	$2.50 \cdot 10^{-2} \pm 4.41 \cdot 10^{-3}$ (−)	$2.60 \cdot 10^{-2} \pm 6.07 \cdot 10^{-3}$ (−)	$2.43 \cdot 10^{-3} \pm 1.16 \cdot 10^{-4}$ (−)	**$2.35 \cdot 10^{-3} \pm 4.84 \cdot 10^{-5}$**
	3	$2.50 \cdot 10^{-2} \pm 3.26 \cdot 10^{-3}$ (−)	$2.66 \cdot 10^{-2} \pm 3.94 \cdot 10^{-3}$ (−)	$2.17 \cdot 10^{-3} \pm 5.72 \cdot 10^{-5}$ (≈)	**$2.15 \cdot 10^{-3} \pm 8.76 \cdot 10^{-6}$**
	4	$2.57 \cdot 10^{-2} \pm 3.90 \cdot 10^{-3}$ (−)	$2.69 \cdot 10^{-2} \pm 3.73 \cdot 10^{-3}$ (−)	**$2.36 \cdot 10^{-3} \pm 1.49 \cdot 10^{-5}$** (≈)	$2.38 \cdot 10^{-3} \pm 4.91 \cdot 10^{-5}$
	5	$2.51 \cdot 10^{-2} \pm 3.15 \cdot 10^{-3}$ (−)	$2.73 \cdot 10^{-2} \pm 5.23 \cdot 10^{-3}$ (−)	**$2.45 \cdot 10^{-3} \pm 3.29 \cdot 10^{-5}$** (≈)	$2.46 \cdot 10^{-3} \pm 3.09 \cdot 10^{-5}$
	6	$2.52 \cdot 10^{-2} \pm 2.47 \cdot 10^{-3}$ (−)	$2.46 \cdot 10^{-2} \pm 3.33 \cdot 10^{-3}$ (−)	$2.34 \cdot 10^{-3} \pm 5.45 \cdot 10^{-5}$ (≈)	**$2.33 \cdot 10^{-3} \pm 4.39 \cdot 10^{-5}$**
	10	$2.67 \cdot 10^{-2} \pm 5.05 \cdot 10^{-3}$ (−)	$2.31 \cdot 10^{-2} \pm 2.66 \cdot 10^{-3}$ (−)	$2.88 \cdot 10^{-3} \pm 8.27 \cdot 10^{-5}$ (≈)	**$2.84 \cdot 10^{-3} \pm 6.72 \cdot 10^{-5}$**
	20	$2.81 \cdot 10^{-2} \pm 3.01 \cdot 10^{-3}$ (−)	$2.39 \cdot 10^{-2} \pm 3.53 \cdot 10^{-3}$ (−)	$5.00 \cdot 10^{-3} \pm 3.61 \cdot 10^{-4}$ (≈)	**$4.99 \cdot 10^{-3} \pm 4.43 \cdot 10^{-4}$**
SRN	1	$1.89 \cdot 10^{-2} \pm 1.70 \cdot 10^{-3}$ (−)	$1.54 \cdot 10^{-2} \pm 1.73 \cdot 10^{-3}$ (−)	$3.47 \cdot 10^{-3} \pm 4.37 \cdot 10^{-5}$ (−)	**$3.39 \cdot 10^{-3} \pm 3.39 \cdot 10^{-5}$**
	2	$3.06 \cdot 10^{-2} \pm 6.16 \cdot 10^{-3}$ (−)	$2.23 \cdot 10^{-2} \pm 4.60 \cdot 10^{-3}$ (−)	$3.66 \cdot 10^{-3} \pm 1.52 \cdot 10^{-4}$ (−)	**$3.32 \cdot 10^{-3} \pm 5.62 \cdot 10^{-5}$**
	3	$2.09 \cdot 10^{-2} \pm 2.54 \cdot 10^{-3}$ (−)	$2.02 \cdot 10^{-2} \pm 2.50 \cdot 10^{-3}$ (−)	$3.56 \cdot 10^{-3} \pm 5.90 \cdot 10^{-5}$ (−)	**$3.31 \cdot 10^{-3} \pm 3.23 \cdot 10^{-5}$**
	4	$2.45 \cdot 10^{-2} \pm 3.00 \cdot 10^{-3}$ (−)	$2.25 \cdot 10^{-2} \pm 2.36 \cdot 10^{-3}$ (−)	**$3.82 \cdot 10^{-3} \pm 5.09 \cdot 10^{-5}$** (+)	$4.02 \cdot 10^{-3} \pm 3.30 \cdot 10^{-5}$
	5	$2.24 \cdot 10^{-2} \pm 2.40 \cdot 10^{-3}$ (−)	$1.98 \cdot 10^{-2} \pm 3.02 \cdot 10^{-3}$ (−)	$3.25 \cdot 10^{-3} \pm 3.31 \cdot 10^{-5}$ (≈)	**$3.23 \cdot 10^{-3} \pm 4.57 \cdot 10^{-5}$**
	6	$2.17 \cdot 10^{-2} \pm 1.66 \cdot 10^{-3}$ (−)	$1.89 \cdot 10^{-2} \pm 1.60 \cdot 10^{-3}$ (−)	$3.25 \cdot 10^{-3} \pm 9.15 \cdot 10^{-5}$ (≈)	**$3.20 \cdot 10^{-3} \pm 7.84 \cdot 10^{-5}$**
	10	$2.20 \cdot 10^{-2} \pm 2.10 \cdot 10^{-3}$ (−)	$1.82 \cdot 10^{-2} \pm 2.36 \cdot 10^{-3}$ (−)	**$4.97 \cdot 10^{-3} \pm 3.33 \cdot 10^{-4}$** (≈)	$5.01 \cdot 10^{-3} \pm 2.88 \cdot 10^{-4}$
	20	$2.37 \cdot 10^{-2} \pm 4.11 \cdot 10^{-3}$ (−)	$1.75 \cdot 10^{-2} \pm 1.39 \cdot 10^{-3}$ (−)	**$5.58 \cdot 10^{-3} \pm 1.63 \cdot 10^{-4}$** (≈)	$5.68 \cdot 10^{-3} \pm 2.12 \cdot 10^{-4}$
TNK	1	$1.01 \cdot 10^{-1} \pm 2.12 \cdot 10^{-2}$ (−)	$1.42 \cdot 10^{-2} \pm 2.16 \cdot 10^{-3}$ (−)	$9.36 \cdot 10^{-3} \pm 1.10 \cdot 10^{-3}$ (−)	**$3.81 \cdot 10^{-3} \pm 3.15 \cdot 10^{-4}$**
	2	$6.63 \cdot 10^{-2} \pm 2.19 \cdot 10^{-2}$ (−)	$2.13 \cdot 10^{-2} \pm 3.89 \cdot 10^{-3}$ (−)	$9.14 \cdot 10^{-3} \pm 1.52 \cdot 10^{-3}$ (−)	**$2.68 \cdot 10^{-3} \pm 2.64 \cdot 10^{-4}$**
	3	$4.75 \cdot 10^{-2} \pm 1.65 \cdot 10^{-2}$ (−)	$1.97 \cdot 10^{-2} \pm 4.01 \cdot 10^{-3}$ (−)	$1.03 \cdot 10^{-2} \pm 2.01 \cdot 10^{-3}$ (−)	**$2.34 \cdot 10^{-3} \pm 1.24 \cdot 10^{-4}$**
	4	$4.29 \cdot 10^{-2} \pm 8.54 \cdot 10^{-3}$ (−)	$2.12 \cdot 10^{-2} \pm 3.24 \cdot 10^{-3}$ (−)	$8.88 \cdot 10^{-3} \pm 1.46 \cdot 10^{-3}$ (−)	**$2.26 \cdot 10^{-3} \pm 1.64 \cdot 10^{-4}$**
	5	$3.56 \cdot 10^{-2} \pm 6.84 \cdot 10^{-3}$ (−)	$1.99 \cdot 10^{-2} \pm 4.01 \cdot 10^{-3}$ (−)	$1.04 \cdot 10^{-2} \pm 1.77 \cdot 10^{-3}$ (−)	**$2.31 \cdot 10^{-3} \pm 7.63 \cdot 10^{-5}$**
	6	$3.27 \cdot 10^{-2} \pm 7.32 \cdot 10^{-3}$ (−)	$1.72 \cdot 10^{-2} \pm 2.30 \cdot 10^{-3}$ (−)	$1.14 \cdot 10^{-2} \pm 1.07 \cdot 10^{-3}$ (−)	**$2.33 \cdot 10^{-3} \pm 1.28 \cdot 10^{-4}$**
	10	$3.84 \cdot 10^{-2} \pm 5.80 \cdot 10^{-3}$ (−)	$2.14 \cdot 10^{-2} \pm 3.90 \cdot 10^{-3}$ (−)	$1.11 \cdot 10^{-2} \pm 1.54 \cdot 10^{-3}$ (−)	**$2.84 \cdot 10^{-3} \pm 1.19 \cdot 10^{-4}$**
	20	$4.53 \cdot 10^{-2} \pm 6.25 \cdot 10^{-3}$ (−)	$2.08 \cdot 10^{-2} \pm 3.36 \cdot 10^{-3}$ (−)	$1.85 \cdot 10^{-2} \pm 1.75 \cdot 10^{-3}$ (−)	**$5.97 \cdot 10^{-3} \pm 5.56 \cdot 10^{-4}$**
CTP1	1	$2.29 \cdot 10^{-2} \pm 4.86 \cdot 10^{-3}$ (−)	$1.87 \cdot 10^{-2} \pm 2.91 \cdot 10^{-3}$ (−)	**$4.39 \cdot 10^{-3} \pm 1.56 \cdot 10^{-4}$** (≈)	$4.48 \cdot 10^{-3} \pm 2.87 \cdot 10^{-4}$
	2	$3.43 \cdot 10^{-2} \pm 3.52 \cdot 10^{-3}$ (−)	$3.62 \cdot 10^{-2} \pm 8.82 \cdot 10^{-3}$ (−)	$6.82 \cdot 10^{-3} \pm 1.72 \cdot 10^{-3}$ (≈)	**$6.38 \cdot 10^{-3} \pm 1.41 \cdot 10^{-3}$**
	3	$3.13 \cdot 10^{-2} \pm 6.48 \cdot 10^{-3}$ (−)	$2.75 \cdot 10^{-2} \pm 6.81 \cdot 10^{-3}$ (−)	**$4.93 \cdot 10^{-3} \pm 4.39 \cdot 10^{-4}$** (≈)	$5.00 \cdot 10^{-3} \pm 8.95 \cdot 10^{-4}$
	4	$2.91 \cdot 10^{-2} \pm 2.52 \cdot 10^{-3}$ (−)	$3.26 \cdot 10^{-2} \pm 4.12 \cdot 10^{-3}$ (−)	$5.12 \cdot 10^{-3} \pm 4.82 \cdot 10^{-4}$ (≈)	**$5.06 \cdot 10^{-3} \pm 4.51 \cdot 10^{-4}$**
	5	$3.56 \cdot 10^{-2} \pm 6.89 \cdot 10^{-3}$ (−)	$3.38 \cdot 10^{-2} \pm 5.08 \cdot 10^{-3}$ (−)	**$5.24 \cdot 10^{-3} \pm 4.87 \cdot 10^{-4}$** (≈)	$5.24 \cdot 10^{-3} \pm 4.87 \cdot 10^{-4}$
	6	$3.17 \cdot 10^{-2} \pm 5.38 \cdot 10^{-3}$ (−)	$2.98 \cdot 10^{-2} \pm 2.93 \cdot 10^{-3}$ (−)	**$4.64 \cdot 10^{-3} \pm 2.99 \cdot 10^{-4}$** (≈)	$4.64 \cdot 10^{-3} \pm 2.99 \cdot 10^{-4}$
	10	$3.43 \cdot 10^{-2} \pm 6.21 \cdot 10^{-3}$ (−)	$2.85 \cdot 10^{-2} \pm 3.84 \cdot 10^{-3}$ (−)	**$5.59 \cdot 10^{-3} \pm 3.13 \cdot 10^{-4}$** (≈)	$5.71 \cdot 10^{-3} \pm 3.34 \cdot 10^{-4}$
	20	$3.47 \cdot 10^{-2} \pm 4.60 \cdot 10^{-3}$ (−)	$2.91 \cdot 10^{-2} \pm 5.41 \cdot 10^{-3}$ (−)	$8.78 \cdot 10^{-3} \pm 1.04 \cdot 10^{-3}$ (≈)	**$8.26 \cdot 10^{-3} \pm 1.26 \cdot 10^{-3}$**
C3DTLZ4	1	$3.69 \cdot 10^{-2} \pm 1.18 \cdot 10^{-2}$ (−)	$7.80 \cdot 10^{-2} \pm 3.13 \cdot 10^{-2}$ (−)	$4.38 \cdot 10^{-2} \pm 6.68 \cdot 10^{-3}$ (−)	**$5.71 \cdot 10^{-3} \pm 6.34 \cdot 10^{-4}$**
	2	$4.23 \cdot 10^{-2} \pm 1.83 \cdot 10^{-2}$ (−)	$3.22 \cdot 10^{-2} \pm 1.46 \cdot 10^{-2}$ (−)	$6.59 \cdot 10^{-2} \pm 7.44 \cdot 10^{-3}$ (−)	**$4.63 \cdot 10^{-3} \pm 1.08 \cdot 10^{-3}$**
	3	$2.12 \cdot 10^{-2} \pm 3.48 \cdot 10^{-3}$ (−)	$1.77 \cdot 10^{-2} \pm 4.18 \cdot 10^{-3}$ (−)	$4.79 \cdot 10^{-2} \pm 6.48 \cdot 10^{-3}$ (−)	**$2.48 \cdot 10^{-3} \pm 1.45 \cdot 10^{-4}$**
	4	$1.98 \cdot 10^{-2} \pm 1.49 \cdot 10^{-3}$ (−)	$1.33 \cdot 10^{-2} \pm 1.59 \cdot 10^{-3}$ (−)	$5.18 \cdot 10^{-2} \pm 4.42 \cdot 10^{-3}$ (−)	**$2.42 \cdot 10^{-3} \pm 1.64 \cdot 10^{-4}$**
	5	$1.86 \cdot 10^{-2} \pm 1.81 \cdot 10^{-3}$ (−)	$1.20 \cdot 10^{-2} \pm 1.58 \cdot 10^{-3}$ (−)	$4.75 \cdot 10^{-2} \pm 5.26 \cdot 10^{-3}$ (−)	**$2.23 \cdot 10^{-3} \pm 1.31 \cdot 10^{-4}$**
	6	$1.68 \cdot 10^{-2} \pm 1.94 \cdot 10^{-3}$ (−)	$1.00 \cdot 10^{-2} \pm 1.02 \cdot 10^{-3}$ (−)	$4.51 \cdot 10^{-2} \pm 8.05 \cdot 10^{-3}$ (−)	**$2.15 \cdot 10^{-3} \pm 7.18 \cdot 10^{-5}$**
	10	$1.66 \cdot 10^{-2} \pm 2.44 \cdot 10^{-3}$ (−)	$8.14 \cdot 10^{-3} \pm 7.08 \cdot 10^{-4}$ (−)	$5.22 \cdot 10^{-2} \pm 1.80 \cdot 10^{-2}$ (−)	**$2.33 \cdot 10^{-3} \pm 1.67 \cdot 10^{-4}$**
	20	$1.91 \cdot 10^{-2} \pm 2.98 \cdot 10^{-3}$ (−)	$8.04 \cdot 10^{-3} \pm 5.96 \cdot 10^{-4}$ (−)	$6.22 \cdot 10^{-2} \pm 1.54 \cdot 10^{-2}$ (−)	**$2.71 \cdot 10^{-3} \pm 1.66 \cdot 10^{-4}$**
OSY	1	$1.08 \cdot 10^{-1} \pm 7.19 \cdot 10^{-2}$ (−)	$4.87 \cdot 10^{-2} \pm 1.38 \cdot 10^{-2}$ (−)	**$9.78 \cdot 10^{-4} \pm 1.23 \cdot 10^{-4}$** (+)	$1.07 \cdot 10^{-3} \pm 4.00 \cdot 10^{-5}$
	2	$3.11 \cdot 10^{-2} \pm 1.13 \cdot 10^{-2}$ (−)	$3.23 \cdot 10^{-2} \pm 9.08 \cdot 10^{-3}$ (−)	$9.60 \cdot 10^{-4} \pm 4.80 \cdot 10^{-5}$ (−)	**$8.35 \cdot 10^{-4} \pm 8.30 \cdot 10^{-5}$**
	3	$2.05 \cdot 10^{-2} \pm 6.24 \cdot 10^{-3}$ (−)	$1.61 \cdot 10^{-2} \pm 3.05 \cdot 10^{-3}$ (−)	$9.91 \cdot 10^{-4} \pm 6.88 \cdot 10^{-5}$ (−)	**$9.38 \cdot 10^{-4} \pm 4.70 \cdot 10^{-5}$**
	4	$1.69 \cdot 10^{-2} \pm 4.00 \cdot 10^{-3}$ (−)	$1.54 \cdot 10^{-2} \pm 4.47 \cdot 10^{-3}$ (−)	**$1.24 \cdot 10^{-3} \pm 7.74 \cdot 10^{-5}$** (≈)	$1.26 \cdot 10^{-3} \pm 1.30 \cdot 10^{-4}$
	5	$1.35 \cdot 10^{-2} \pm 3.60 \cdot 10^{-3}$ (−)	$1.23 \cdot 10^{-2} \pm 2.71 \cdot 10^{-3}$ (−)	$1.54 \cdot 10^{-3} \pm 8.39 \cdot 10^{-5}$ (≈)	**$1.54 \cdot 10^{-3} \pm 1.02 \cdot 10^{-4}$**
	6	$1.21 \cdot 10^{-2} \pm 2.18 \cdot 10^{-3}$ (−)	$1.17 \cdot 10^{-2} \pm 2.25 \cdot 10^{-3}$ (−)	$2.14 \cdot 10^{-3} \pm 1.76 \cdot 10^{-4}$ (≈)	**$2.01 \cdot 10^{-3} \pm 1.61 \cdot 10^{-4}$**
	10	$1.14 \cdot 10^{-2} \pm 3.75 \cdot 10^{-3}$ (−)	$1.20 \cdot 10^{-2} \pm 3.15 \cdot 10^{-3}$ (−)	$7.60 \cdot 10^{-3} \pm 1.10 \cdot 10^{-3}$ (≈)	**$7.26 \cdot 10^{-3} \pm 8.61 \cdot 10^{-4}$**
	20	**$1.31 \cdot 10^{-2} \pm 7.56 \cdot 10^{-3}$** (+)	$1.31 \cdot 10^{-2} \pm 5.45 \cdot 10^{-3}$ (+)	$4.22 \cdot 10^{-2} \pm 1.06 \cdot 10^{-2}$ (≈)	$3.67 \cdot 10^{-2} \pm 5.01 \cdot 10^{-3}$
TBTD	1	$4.43 \cdot 10^{-2} \pm 3.30 \cdot 10^{-2}$ (−)	$2.20 \cdot 10^{-2} \pm 8.82 \cdot 10^{-3}$ (−)	$6.43 \cdot 10^{-3} \pm 9.72 \cdot 10^{-4}$ (−)	**$4.27 \cdot 10^{-3} \pm 2.41 \cdot 10^{-3}$**
	2	$2.87 \cdot 10^{-2} \pm 6.48 \cdot 10^{-3}$ (−)	$1.46 \cdot 10^{-2} \pm 5.57 \cdot 10^{-3}$ (−)	$1.10 \cdot 10^{-2} \pm 2.68 \cdot 10^{-3}$ (−)	**$5.94 \cdot 10^{-3} \pm 2.08 \cdot 10^{-3}$**
	3	$2.17 \cdot 10^{-2} \pm 3.41 \cdot 10^{-3}$ (−)	$1.40 \cdot 10^{-2} \pm 4.45 \cdot 10^{-3}$ (−)	$1.37 \cdot 10^{-2} \pm 5.00 \cdot 10^{-3}$ (−)	**$5.24 \cdot 10^{-3} \pm 1.22 \cdot 10^{-3}$**
	4	$1.56 \cdot 10^{-2} \pm 3.89 \cdot 10^{-3}$ (−)	$1.14 \cdot 10^{-2} \pm 2.67 \cdot 10^{-3}$ (−)	$1.46 \cdot 10^{-2} \pm 3.13 \cdot 10^{-3}$ (−)	**$6.47 \cdot 10^{-3} \pm 1.29 \cdot 10^{-3}$**
	5	$1.72 \cdot 10^{-2} \pm 3.94 \cdot 10^{-3}$ (−)	$1.20 \cdot 10^{-2} \pm 2.80 \cdot 10^{-3}$ (−)	$1.19 \cdot 10^{-2} \pm 2.65 \cdot 10^{-3}$ (−)	**$6.60 \cdot 10^{-3} \pm 1.12 \cdot 10^{-3}$**
	6	$1.38 \cdot 10^{-2} \pm 4.03 \cdot 10^{-3}$ (−)	$1.12 \cdot 10^{-2} \pm 2.78 \cdot 10^{-3}$ (≈)	$1.53 \cdot 10^{-2} \pm 4.81 \cdot 10^{-3}$ (−)	**$8.85 \cdot 10^{-3} \pm 2.62 \cdot 10^{-3}$**
	10	$1.20 \cdot 10^{-2} \pm 3.09 \cdot 10^{-3}$ (≈)	$1.07 \cdot 10^{-2} \pm 3.27 \cdot 10^{-3}$ (≈)	$1.56 \cdot 10^{-2} \pm 6.24 \cdot 10^{-3}$ (−)	**$1.02 \cdot 10^{-2} \pm 1.53 \cdot 10^{-3}$**
	20	$1.14 \cdot 10^{-2} \pm 1.54 \cdot 10^{-3}$ (≈)	**$9.82 \cdot 10^{-3} \pm 2.00 \cdot 10^{-3}$** (+)	$1.55 \cdot 10^{-2} \pm 3.37 \cdot 10^{-3}$ (≈)	$1.36 \cdot 10^{-2} \pm 3.73 \cdot 10^{-3}$
NBP	1	$1.83 \cdot 10^{-2} \pm 2.50 \cdot 10^{-3}$ (−)	$1.82 \cdot 10^{-2} \pm 4.90 \cdot 10^{-3}$ (−)	$3.76 \cdot 10^{-3} \pm 1.78 \cdot 10^{-4}$ (−)	**$2.33 \cdot 10^{-3} \pm 3.94 \cdot 10^{-5}$**
	2	$2.32 \cdot 10^{-2} \pm 4.35 \cdot 10^{-3}$ (−)	$2.27 \cdot 10^{-2} \pm 2.88 \cdot 10^{-3}$ (−)	$3.64 \cdot 10^{-3} \pm 4.00 \cdot 10^{-4}$ (−)	**$2.32 \cdot 10^{-3} \pm 5.64 \cdot 10^{-5}$**
	3	$2.00 \cdot 10^{-2} \pm 3.63 \cdot 10^{-3}$ (−)	$1.91 \cdot 10^{-2} \pm 1.80 \cdot 10^{-3}$ (−)	$3.64 \cdot 10^{-3} \pm 1.72 \cdot 10^{-4}$ (−)	**$2.46 \cdot 10^{-3} \pm 1.46 \cdot 10^{-5}$**
	4	$2.60 \cdot 10^{-2} \pm 4.15 \cdot 10^{-3}$ (−)	$2.13 \cdot 10^{-2} \pm 3.90 \cdot 10^{-3}$ (−)	$3.87 \cdot 10^{-3} \pm 2.37 \cdot 10^{-4}$ (−)	**$2.45 \cdot 10^{-3} \pm 2.87 \cdot 10^{-5}$**
	5	$2.23 \cdot 10^{-2} \pm 2.68 \cdot 10^{-3}$ (−)	$1.88 \cdot 10^{-2} \pm 2.01 \cdot 10^{-3}$ (−)	$4.41 \cdot 10^{-3} \pm 4.01 \cdot 10^{-4}$ (−)	**$2.88 \cdot 10^{-3} \pm 9.64 \cdot 10^{-5}$**
	6	$2.09 \cdot 10^{-2} \pm 2.49 \cdot 10^{-3}$ (−)	$1.98 \cdot 10^{-2} \pm 3.48 \cdot 10^{-3}$ (−)	$4.03 \cdot 10^{-3} \pm 3.02 \cdot 10^{-4}$ (−)	**$3.03 \cdot 10^{-3} \pm 7.37 \cdot 10^{-5}$**
	10	$2.11 \cdot 10^{-2} \pm 2.92 \cdot 10^{-3}$ (−)	$1.77 \cdot 10^{-2} \pm 2.28 \cdot 10^{-3}$ (−)	$5.38 \cdot 10^{-3} \pm 4.40 \cdot 10^{-4}$ (−)	**$3.58 \cdot 10^{-3} \pm 2.04 \cdot 10^{-4}$**
	20	$2.29 \cdot 10^{-2} \pm 3.26 \cdot 10^{-3}$ (−)	$1.94 \cdot 10^{-2} \pm 2.35 \cdot 10^{-3}$ (−)	$1.44 \cdot 10^{-2} \pm 3.86 \cdot 10^{-3}$ (−)	**$6.09 \cdot 10^{-3} \pm 3.19 \cdot 10^{-4}$**
DBD	1	$1.17 \cdot 10^{-2} \pm 3.63 \cdot 10^{-3}$ (−)	$2.43 \cdot 10^{-2} \pm 1.51 \cdot 10^{-2}$ (−)	**$1.35 \cdot 10^{-3} \pm 3.36 \cdot 10^{-4}$** (+)	$2.26 \cdot 10^{-3} \pm 1.40 \cdot 10^{-3}$
	2	$1.41 \cdot 10^{-2} \pm 2.84 \cdot 10^{-3}$ (−)	$2.06 \cdot 10^{-2} \pm 4.21 \cdot 10^{-3}$ (−)	**$1.24 \cdot 10^{-3} \pm 2.49 \cdot 10^{-4}$** (+)	$4.29 \cdot 10^{-3} \pm 1.55 \cdot 10^{-3}$
	3	$1.01 \cdot 10^{-2} \pm 1.49 \cdot 10^{-3}$ (−)	$2.01 \cdot 10^{-2} \pm 6.65 \cdot 10^{-3}$ (−)	$1.32 \cdot 10^{-3} \pm 3.12 \cdot 10^{-4}$ (−)	**$9.60 \cdot 10^{-4} \pm 5.64 \cdot 10^{-5}$**
	4	$1.39 \cdot 10^{-2} \pm 2.75 \cdot 10^{-3}$ (−)	$1.69 \cdot 10^{-2} \pm 2.64 \cdot 10^{-3}$ (−)	$2.89 \cdot 10^{-3} \pm 1.54 \cdot 10^{-3}$ (−)	**$1.18 \cdot 10^{-3} \pm 4.70 \cdot 10^{-5}$**
	5	$1.42 \cdot 10^{-2} \pm 4.50 \cdot 10^{-3}$ (−)	$1.75 \cdot 10^{-2} \pm 3.70 \cdot 10^{-3}$ (−)	$1.84 \cdot 10^{-3} \pm 1.04 \cdot 10^{-3}$ (≈)	**$1.31 \cdot 10^{-3} \pm 8.45 \cdot 10^{-5}$**
	6	$1.07 \cdot 10^{-2} \pm 1.31 \cdot 10^{-3}$ (−)	$2.10 \cdot 10^{-2} \pm 8.34 \cdot 10^{-3}$ (−)	**$2.08 \cdot 10^{-3} \pm 7.46 \cdot 10^{-4}$** (≈)	$2.21 \cdot 10^{-3} \pm 7.33 \cdot 10^{-4}$
	10	$1.22 \cdot 10^{-2} \pm 2.49 \cdot 10^{-3}$ (−)	$1.72 \cdot 10^{-2} \pm 4.16 \cdot 10^{-3}$ (−)	$2.62 \cdot 10^{-3} \pm 5.78 \cdot 10^{-4}$ (≈)	**$2.33 \cdot 10^{-3} \pm 2.25 \cdot 10^{-4}$**
	20	$1.30 \cdot 10^{-2} \pm 3.30 \cdot 10^{-3}$ (−)	$2.06 \cdot 10^{-2} \pm 5.11 \cdot 10^{-3}$ (−)	**$3.74 \cdot 10^{-3} \pm 2.29 \cdot 10^{-4}$** (+)	$4.09 \cdot 10^{-3} \pm 3.45 \cdot 10^{-4}$
SRD	1	$4.42 \cdot 10^{-3} \pm 3.15 \cdot 10^{-3}$ (≈)	$1.86 \cdot 10^{-2} \pm 9.63 \cdot 10^{-3}$ (−)	**$6.51 \cdot 10^{-4} \pm 9.70 \cdot 10^{-5}$** (+)	$2.45 \cdot 10^{-3} \pm 9.97 \cdot 10^{-4}$
	2	$2.26 \cdot 10^{-3} \pm 1.37 \cdot 10^{-3}$ (≈)	$2.09 \cdot 10^{-3} \pm 7.84 \cdot 10^{-4}$ (−)	**$1.24 \cdot 10^{-3} \pm 1.31 \cdot 10^{-4}$** (+)	$1.40 \cdot 10^{-3} \pm 1.77 \cdot 10^{-4}$
	3	$1.22 \cdot 10^{-3} \pm 4.50 \cdot 10^{-4}$ (≈)	**$1.03 \cdot 10^{-3} \pm 2.35 \cdot 10^{-4}$** (+)	$1.91 \cdot 10^{-3} \pm 1.38 \cdot 10^{-4}$ (−)	$1.50 \cdot 10^{-3} \pm 1.16 \cdot 10^{-4}$
	4	$1.15 \cdot 10^{-3} \pm 1.29 \cdot 10^{-4}$ (+)	**$1.09 \cdot 10^{-3} \pm 1.64 \cdot 10^{-4}$** (+)	$2.11 \cdot 10^{-3} \pm 3.10 \cdot 10^{-4}$ (−)	$1.58 \cdot 10^{-3} \pm 1.41 \cdot 10^{-4}$
	5	$1.09 \cdot 10^{-3} \pm 1.41 \cdot 10^{-4}$ (+)	**$9.05 \cdot 10^{-4} \pm 8.43 \cdot 10^{-5}$** (+)	$2.20 \cdot 10^{-3} \pm 3.38 \cdot 10^{-4}$ (−)	$1.56 \cdot 10^{-3} \pm 9.89 \cdot 10^{-5}$
	6	$1.13 \cdot 10^{-3} \pm 1.61 \cdot 10^{-4}$ (+)	**$9.45 \cdot 10^{-4} \pm 1.19 \cdot 10^{-4}$** (+)	$2.27 \cdot 10^{-3} \pm 3.09 \cdot 10^{-4}$ (−)	$1.70 \cdot 10^{-3} \pm 1.29 \cdot 10^{-4}$
	10	$1.19 \cdot 10^{-3} \pm 1.44 \cdot 10^{-4}$ (+)	**$9.06 \cdot 10^{-4} \pm 9.72 \cdot 10^{-5}$** (+)	$2.95 \cdot 10^{-3} \pm 2.89 \cdot 10^{-4}$ (−)	$2.26 \cdot 10^{-3} \pm 2.06 \cdot 10^{-4}$
	20	$1.25 \cdot 10^{-3} \pm 1.61 \cdot 10^{-4}$ (+)	**$1.09 \cdot 10^{-3} \pm 1.24 \cdot 10^{-4}$** (+)	$4.92 \cdot 10^{-3} \pm 4.88 \cdot 10^{-4}$ (−)	$3.20 \cdot 10^{-3} \pm 3.17 \cdot 10^{-4}$

Table continues on next page.

Continuation of Table 4.

Function	p	SA-NSGA-II	IC-SA-NSGA-II	SAMO-COBRA	IOC-SAMO-COBRA
WB	1	$2.57 \cdot 10^{-1} \pm 8.72 \cdot 10^{-2}$ (−)	**$2.10 \cdot 10^{-2} \pm 2.89 \cdot 10^{-2}$ (+)**	$7.51 \cdot 10^{-2} \pm 1.35 \cdot 10^{-2}$ (−)	$2.63 \cdot 10^{-2} \pm 2.92 \cdot 10^{-3}$
	2	$1.10 \cdot 10^{-1} \pm 6.20 \cdot 10^{-2}$ (−)	**$1.99 \cdot 10^{-2} \pm 4.07 \cdot 10^{-3}$ (≈)**	$6.06 \cdot 10^{-2} \pm 2.23 \cdot 10^{-2}$ (−)	$2.65 \cdot 10^{-2} \pm 1.23 \cdot 10^{-2}$
	3	$7.69 \cdot 10^{-2} \pm 5.10 \cdot 10^{-2}$ (−)	**$1.47 \cdot 10^{-2} \pm 4.74 \cdot 10^{-3}$ (+)**	$3.65 \cdot 10^{-2} \pm 1.71 \cdot 10^{-2}$ (≈)	$2.68 \cdot 10^{-2} \pm 7.70 \cdot 10^{-3}$
	4	$4.66 \cdot 10^{-2} \pm 2.34 \cdot 10^{-2}$ (≈)	**$1.48 \cdot 10^{-2} \pm 2.35 \cdot 10^{-3}$ (+)**	$6.29 \cdot 10^{-2} \pm 2.13 \cdot 10^{-2}$ (−)	$2.92 \cdot 10^{-2} \pm 1.05 \cdot 10^{-2}$
	5	$8.12 \cdot 10^{-2} \pm 8.78 \cdot 10^{-2}$ (−)	**$1.59 \cdot 10^{-2} \pm 3.51 \cdot 10^{-3}$ (+)**	$6.44 \cdot 10^{-2} \pm 3.21 \cdot 10^{-2}$ (−)	$2.74 \cdot 10^{-2} \pm 1.57 \cdot 10^{-2}$
	6	$6.67 \cdot 10^{-2} \pm 4.83 \cdot 10^{-2}$ (≈)	**$1.41 \cdot 10^{-2} \pm 2.51 \cdot 10^{-3}$ (+)**	$7.32 \cdot 10^{-2} \pm 2.59 \cdot 10^{-2}$ (−)	$4.19 \cdot 10^{-2} \pm 2.31 \cdot 10^{-2}$
	10	$5.06 \cdot 10^{-2} \pm 1.16 \cdot 10^{-2}$ (≈)	**$1.33 \cdot 10^{-2} \pm 3.94 \cdot 10^{-3}$ (+)**	$7.73 \cdot 10^{-2} \pm 2.58 \cdot 10^{-2}$ (−)	$4.89 \cdot 10^{-2} \pm 8.56 \cdot 10^{-3}$
	20	$8.73 \cdot 10^{-2} \pm 1.03 \cdot 10^{-1}$ (≈)	**$1.38 \cdot 10^{-2} \pm 2.70 \cdot 10^{-3}$ (+)**	$7.29 \cdot 10^{-2} \pm 1.18 \cdot 10^{-2}$ (≈)	$6.47 \cdot 10^{-2} \pm 1.55 \cdot 10^{-2}$
BICOP1	1	$6.41 \cdot 10^{-1} \pm 2.12 \cdot 10^{-1}$ (−)	$6.35 \cdot 10^{-1} \pm 3.17 \cdot 10^{-1}$ (−)	**$2.89 \cdot 10^{-1} \pm 9.21 \cdot 10^{-2}$ (≈)**	$3.48 \cdot 10^{-1} \pm 7.81 \cdot 10^{-2}$
	2	$3.58 \cdot 10^{-2} \pm 1.03 \cdot 10^{-2}$ (+)	**$3.10 \cdot 10^{-2} \pm 6.98 \cdot 10^{-3}$ (+)**	$2.45 \cdot 10^{-1} \pm 2.17 \cdot 10^{-1}$ (≈)	$1.23 \cdot 10^{-1} \pm 1.04 \cdot 10^{-1}$
	3	$1.88 \cdot 10^{-2} \pm 5.26 \cdot 10^{-3}$ (≈)	**$1.56 \cdot 10^{-2} \pm 2.56 \cdot 10^{-3}$ (≈)**	$8.36 \cdot 10^{-2} \pm 1.34 \cdot 10^{-1}$ (≈)	$4.29 \cdot 10^{-2} \pm 4.04 \cdot 10^{-2}$
	4	$1.26 \cdot 10^{-2} \pm 3.18 \cdot 10^{-3}$ (≈)	**$1.15 \cdot 10^{-2} \pm 2.99 \cdot 10^{-3}$ (+)**	$1.67 \cdot 10^{-2} \pm 5.16 \cdot 10^{-3}$ (≈)	$2.87 \cdot 10^{-2} \pm 3.66 \cdot 10^{-2}$
	5	$8.76 \cdot 10^{-3} \pm 2.13 \cdot 10^{-3}$ (+)	**$8.44 \cdot 10^{-3} \pm 2.83 \cdot 10^{-3}$ (+)**	$2.13 \cdot 10^{-2} \pm 6.69 \cdot 10^{-3}$ (≈)	$2.44 \cdot 10^{-2} \pm 7.39 \cdot 10^{-3}$
	6	$6.67 \cdot 10^{-3} \pm 2.16 \cdot 10^{-3}$ (+)	**$6.62 \cdot 10^{-3} \pm 1.66 \cdot 10^{-3}$ (+)**	$4.21 \cdot 10^{-2} \pm 1.06 \cdot 10^{-2}$ (≈)	$3.69 \cdot 10^{-2} \pm 8.02 \cdot 10^{-3}$
	10	**$3.38 \cdot 10^{-3} \pm 5.54 \cdot 10^{-4}$ (+)**	$3.78 \cdot 10^{-3} \pm 9.19 \cdot 10^{-4}$ (+)	$1.03 \cdot 10^{-1} \pm 3.26 \cdot 10^{-2}$ (≈)	$9.08 \cdot 10^{-2} \pm 1.96 \cdot 10^{-2}$
	20	$3.41 \cdot 10^{-3} \pm 4.23 \cdot 10^{-4}$ (+)	**$3.36 \cdot 10^{-3} \pm 4.14 \cdot 10^{-4}$ (+)**	$2.42 \cdot 10^{-1} \pm 6.91 \cdot 10^{-2}$ (≈)	$2.70 \cdot 10^{-1} \pm 5.84 \cdot 10^{-2}$
BICOP2	1	$1.83 \cdot 10^{-1} \pm 1.21 \cdot 10^{-2}$ (−)	$1.59 \cdot 10^{-1} \pm 2.57 \cdot 10^{-2}$ (−)	$7.70 \cdot 10^{-2} \pm 2.97 \cdot 10^{-2}$ (−)	**$2.93 \cdot 10^{-2} \pm 9.30 \cdot 10^{-3}$**
	2	$1.73 \cdot 10^{-1} \pm 3.16 \cdot 10^{-2}$ (−)	$1.07 \cdot 10^{-1} \pm 2.61 \cdot 10^{-2}$ (−)	$7.41 \cdot 10^{-2} \pm 3.19 \cdot 10^{-2}$ (−)	**$1.60 \cdot 10^{-2} \pm 1.77 \cdot 10^{-2}$**
	3	$1.58 \cdot 10^{-1} \pm 2.53 \cdot 10^{-2}$ (−)	$1.28 \cdot 10^{-1} \pm 4.18 \cdot 10^{-2}$ (−)	$7.19 \cdot 10^{-2} \pm 3.52 \cdot 10^{-2}$ (−)	**$2.31 \cdot 10^{-2} \pm 3.25 \cdot 10^{-2}$**
	4	$1.63 \cdot 10^{-1} \pm 2.98 \cdot 10^{-2}$ (−)	$1.17 \cdot 10^{-1} \pm 4.08 \cdot 10^{-2}$ (−)	$7.00 \cdot 10^{-2} \pm 4.01 \cdot 10^{-2}$ (≈)	**$5.61 \cdot 10^{-2} \pm 4.71 \cdot 10^{-2}$**
	5	$1.61 \cdot 10^{-1} \pm 2.69 \cdot 10^{-2}$ (−)	$1.08 \cdot 10^{-1} \pm 3.34 \cdot 10^{-2}$ (≈)	**$4.78 \cdot 10^{-2} \pm 1.21 \cdot 10^{-2}$ (≈)**	$7.25 \cdot 10^{-2} \pm 4.46 \cdot 10^{-2}$
	6	$1.59 \cdot 10^{-1} \pm 2.69 \cdot 10^{-2}$ (−)	$1.26 \cdot 10^{-1} \pm 3.67 \cdot 10^{-2}$ (−)	**$4.46 \cdot 10^{-2} \pm 8.25 \cdot 10^{-3}$ (≈)**	$7.96 \cdot 10^{-2} \pm 4.29 \cdot 10^{-2}$
	10	$1.30 \cdot 10^{-1} \pm 3.10 \cdot 10^{-2}$ (−)	$1.35 \cdot 10^{-1} \pm 3.17 \cdot 10^{-2}$ (−)	$5.90 \cdot 10^{-2} \pm 1.55 \cdot 10^{-2}$ (≈)	**$5.68 \cdot 10^{-2} \pm 2.92 \cdot 10^{-2}$**
	20	$1.35 \cdot 10^{-1} \pm 3.16 \cdot 10^{-2}$ (−)	$1.29 \cdot 10^{-1} \pm 3.42 \cdot 10^{-2}$ (−)	$7.75 \cdot 10^{-2} \pm 1.54 \cdot 10^{-2}$ (−)	**$3.76 \cdot 10^{-2} \pm 1.00 \cdot 10^{-2}$**
MW1	1	$1.00 \cdot 10^{+0} \pm 0.00 \cdot 10^{+0}$ (−)	$1.05 \cdot 10^{-1} \pm 3.91 \cdot 10^{-2}$ (−)	$7.18 \cdot 10^{-1} \pm 3.04 \cdot 10^{-1}$ (−)	**$6.09 \cdot 10^{-4} \pm 7.62 \cdot 10^{-5}$**
	2	$1.46 \cdot 10^{-1} \pm 7.65 \cdot 10^{-2}$ (−)	$4.35 \cdot 10^{-2} \pm 3.54 \cdot 10^{-3}$ (−)	$2.11 \cdot 10^{-1} \pm 1.19 \cdot 10^{-1}$ (−)	**$6.40 \cdot 10^{-4} \pm 1.02 \cdot 10^{-4}$**
	3	$9.49 \cdot 10^{-2} \pm 4.32 \cdot 10^{-2}$ (−)	$4.30 \cdot 10^{-2} \pm 7.40 \cdot 10^{-3}$ (−)	$8.58 \cdot 10^{-2} \pm 7.18 \cdot 10^{-2}$ (−)	**$7.57 \cdot 10^{-4} \pm 1.99 \cdot 10^{-4}$**
	4	$5.70 \cdot 10^{-2} \pm 3.73 \cdot 10^{-2}$ (−)	$3.46 \cdot 10^{-2} \pm 7.84 \cdot 10^{-3}$ (−)	$2.32 \cdot 10^{-1} \pm 2.49 \cdot 10^{-1}$ (−)	**$9.87 \cdot 10^{-4} \pm 1.55 \cdot 10^{-4}$**
	5	$3.47 \cdot 10^{-2} \pm 1.34 \cdot 10^{-2}$ (−)	$2.42 \cdot 10^{-2} \pm 4.97 \cdot 10^{-3}$ (−)	$2.38 \cdot 10^{-1} \pm 2.47 \cdot 10^{-1}$ (−)	**$1.07 \cdot 10^{-3} \pm 1.54 \cdot 10^{-4}$**
	6	$2.97 \cdot 10^{-2} \pm 1.20 \cdot 10^{-2}$ (−)	$1.47 \cdot 10^{-2} \pm 2.61 \cdot 10^{-3}$ (−)	$1.45 \cdot 10^{-1} \pm 1.31 \cdot 10^{-1}$ (−)	**$1.42 \cdot 10^{-3} \pm 3.54 \cdot 10^{-4}$**
	10	$3.23 \cdot 10^{-2} \pm 2.70 \cdot 10^{-2}$ (−)	$9.65 \cdot 10^{-3} \pm 1.91 \cdot 10^{-3}$ (−)	$2.63 \cdot 10^{-1} \pm 1.70 \cdot 10^{-1}$ (−)	**$2.31 \cdot 10^{-3} \pm 8.80 \cdot 10^{-4}$**
	20	$1.74 \cdot 10^{-2} \pm 1.23 \cdot 10^{-2}$ (≈)	**$8.06 \cdot 10^{-3} \pm 2.14 \cdot 10^{-3}$ (≈)**	$1.91 \cdot 10^{-1} \pm 1.24 \cdot 10^{-1}$ (≈)	$1.81 \cdot 10^{-1} \pm 2.22 \cdot 10^{-1}$
MW2	1	$7.92 \cdot 10^{-1} \pm 2.55 \cdot 10^{-1}$ (−)	**$3.84 \cdot 10^{-2} \pm 9.30 \cdot 10^{-3}$ (+)**	$3.14 \cdot 10^{-1} \pm 9.67 \cdot 10^{-2}$ (−)	$6.63 \cdot 10^{-2} \pm 1.40 \cdot 10^{-2}$
	2	$1.87 \cdot 10^{-1} \pm 7.99 \cdot 10^{-2}$ (−)	**$3.03 \cdot 10^{-2} \pm 3.62 \cdot 10^{-3}$ (+)**	$3.06 \cdot 10^{-1} \pm 1.95 \cdot 10^{-1}$ (−)	$4.20 \cdot 10^{-2} \pm 1.15 \cdot 10^{-2}$
	3	$1.39 \cdot 10^{-1} \pm 7.28 \cdot 10^{-2}$ (−)	**$2.63 \cdot 10^{-2} \pm 5.49 \cdot 10^{-3}$ (+)**	$2.74 \cdot 10^{-1} \pm 1.28 \cdot 10^{-1}$ (−)	$5.89 \cdot 10^{-2} \pm 4.97 \cdot 10^{-2}$
	4	$9.77 \cdot 10^{-2} \pm 6.52 \cdot 10^{-2}$ (≈)	**$2.59 \cdot 10^{-2} \pm 5.51 \cdot 10^{-3}$ (+)**	$3.02 \cdot 10^{-1} \pm 1.31 \cdot 10^{-1}$ (−)	$1.00 \cdot 10^{-1} \pm 5.85 \cdot 10^{-2}$
	5	$1.12 \cdot 10^{-1} \pm 7.36 \cdot 10^{-2}$ (≈)	**$2.45 \cdot 10^{-2} \pm 5.14 \cdot 10^{-3}$ (+)**	$3.25 \cdot 10^{-1} \pm 1.44 \cdot 10^{-1}$ (−)	$6.08 \cdot 10^{-2} \pm 3.79 \cdot 10^{-2}$
	6	$1.08 \cdot 10^{-1} \pm 7.13 \cdot 10^{-2}$ (−)	**$2.48 \cdot 10^{-2} \pm 5.52 \cdot 10^{-3}$ (+)**	$3.52 \cdot 10^{-1} \pm 1.57 \cdot 10^{-1}$ (−)	$5.54 \cdot 10^{-2} \pm 3.19 \cdot 10^{-2}$
	10	$1.21 \cdot 10^{-1} \pm 8.27 \cdot 10^{-2}$ (≈)	**$2.34 \cdot 10^{-2} \pm 6.84 \cdot 10^{-3}$ (+)**	$4.05 \cdot 10^{-1} \pm 1.08 \cdot 10^{-1}$ (−)	$7.55 \cdot 10^{-2} \pm 2.99 \cdot 10^{-2}$
	20	$1.21 \cdot 10^{-1} \pm 7.84 \cdot 10^{-2}$ (≈)	**$2.15 \cdot 10^{-2} \pm 5.98 \cdot 10^{-3}$ (+)**	$3.82 \cdot 10^{-1} \pm 1.54 \cdot 10^{-1}$ (−)	$1.04 \cdot 10^{-1} \pm 3.26 \cdot 10^{-2}$
MW3	1	$6.07 \cdot 10^{-1} \pm 3.85 \cdot 10^{-1}$ (−)	$2.33 \cdot 10^{-2} \pm 4.57 \cdot 10^{-3}$ (−)	$4.14 \cdot 10^{-2} \pm 1.26 \cdot 10^{-2}$ (−)	**$2.53 \cdot 10^{-3} \pm 5.11 \cdot 10^{-4}$**
	2	$2.70 \cdot 10^{-2} \pm 8.21 \cdot 10^{-3}$ (−)	$1.70 \cdot 10^{-2} \pm 1.75 \cdot 10^{-3}$ (−)	$2.40 \cdot 10^{-2} \pm 4.48 \cdot 10^{-3}$ (−)	**$1.72 \cdot 10^{-3} \pm 8.36 \cdot 10^{-4}$**
	3	$1.78 \cdot 10^{-2} \pm 3.25 \cdot 10^{-3}$ (−)	$1.32 \cdot 10^{-2} \pm 1.40 \cdot 10^{-3}$ (−)	$1.32 \cdot 10^{-2} \pm 5.51 \cdot 10^{-3}$ (−)	**$1.32 \cdot 10^{-3} \pm 2.29 \cdot 10^{-4}$**
	4	$1.73 \cdot 10^{-2} \pm 1.18 \cdot 10^{-3}$ (−)	$1.12 \cdot 10^{-2} \pm 1.58 \cdot 10^{-3}$ (−)	$6.28 \cdot 10^{-3} \pm 2.94 \cdot 10^{-3}$ (−)	**$1.40 \cdot 10^{-3} \pm 1.12 \cdot 10^{-4}$**
	5	$1.55 \cdot 10^{-2} \pm 2.82 \cdot 10^{-3}$ (−)	$9.62 \cdot 10^{-3} \pm 1.17 \cdot 10^{-3}$ (−)	$6.39 \cdot 10^{-3} \pm 1.92 \cdot 10^{-3}$ (−)	**$1.93 \cdot 10^{-3} \pm 4.53 \cdot 10^{-4}$**
	6	$1.67 \cdot 10^{-2} \pm 1.46 \cdot 10^{-3}$ (−)	$8.82 \cdot 10^{-3} \pm 1.96 \cdot 10^{-3}$ (−)	$7.27 \cdot 10^{-3} \pm 2.02 \cdot 10^{-3}$ (−)	**$1.97 \cdot 10^{-3} \pm 3.23 \cdot 10^{-4}$**
	10	$1.38 \cdot 10^{-2} \pm 1.98 \cdot 10^{-3}$ (−)	$7.04 \cdot 10^{-3} \pm 5.85 \cdot 10^{-4}$ (−)	$1.03 \cdot 10^{-2} \pm 1.12 \cdot 10^{-3}$ (−)	**$2.91 \cdot 10^{-3} \pm 3.10 \cdot 10^{-4}$**
	20	$1.41 \cdot 10^{-2} \pm 2.59 \cdot 10^{-3}$ (−)	$7.15 \cdot 10^{-3} \pm 5.79 \cdot 10^{-4}$ (−)	$1.43 \cdot 10^{-2} \pm 1.63 \cdot 10^{-3}$ (−)	**$5.19 \cdot 10^{-3} \pm 3.65 \cdot 10^{-4}$**
MW11	1	$3.79 \cdot 10^{-1} \pm 2.23 \cdot 10^{-1}$ (−)	**$3.50 \cdot 10^{-2} \pm 8.38 \cdot 10^{-3}$ (+)**	$1.75 \cdot 10^{-1} \pm 2.77 \cdot 10^{-1}$ (≈)	$7.91 \cdot 10^{-2} \pm 3.33 \cdot 10^{-2}$
	2	$1.02 \cdot 10^{-1} \pm 9.49 \cdot 10^{-2}$ (≈)	**$2.15 \cdot 10^{-2} \pm 4.27 \cdot 10^{-3}$ (+)**	$1.65 \cdot 10^{-1} \pm 1.00 \cdot 10^{-1}$ (−)	$6.79 \cdot 10^{-2} \pm 2.65 \cdot 10^{-2}$
	3	$1.59 \cdot 10^{-1} \pm 1.21 \cdot 10^{-1}$ (−)	$1.88 \cdot 10^{-2} \pm 3.75 \cdot 10^{-3}$ (−)	$1.30 \cdot 10^{-1} \pm 7.44 \cdot 10^{-2}$ (−)	**$6.96 \cdot 10^{-3} \pm 4.95 \cdot 10^{-3}$**
	4	$1.99 \cdot 10^{-1} \pm 1.33 \cdot 10^{-1}$ (−)	$1.50 \cdot 10^{-2} \pm 2.65 \cdot 10^{-3}$ (−)	$1.44 \cdot 10^{-1} \pm 8.01 \cdot 10^{-2}$ (−)	**$4.78 \cdot 10^{-3} \pm 1.31 \cdot 10^{-3}$**
	5	$1.93 \cdot 10^{-1} \pm 1.31 \cdot 10^{-1}$ (−)	$1.49 \cdot 10^{-2} \pm 2.39 \cdot 10^{-3}$ (−)	$1.33 \cdot 10^{-1} \pm 8.63 \cdot 10^{-2}$ (−)	**$4.34 \cdot 10^{-3} \pm 6.51 \cdot 10^{-4}$**
	6	$2.62 \cdot 10^{-1} \pm 9.96 \cdot 10^{-2}$ (−)	$1.26 \cdot 10^{-2} \pm 2.12 \cdot 10^{-3}$ (−)	$1.53 \cdot 10^{-1} \pm 1.09 \cdot 10^{-1}$ (−)	**$4.29 \cdot 10^{-3} \pm 1.14 \cdot 10^{-3}$**
	10	$2.31 \cdot 10^{-1} \pm 1.23 \cdot 10^{-1}$ (−)	$9.83 \cdot 10^{-3} \pm 1.40 \cdot 10^{-3}$ (−)	$2.50 \cdot 10^{-1} \pm 8.82 \cdot 10^{-2}$ (−)	**$4.12 \cdot 10^{-3} \pm 1.00 \cdot 10^{-3}$**
	20	$3.02 \cdot 10^{-1} \pm 8.69 \cdot 10^{-2}$ (−)	$9.72 \cdot 10^{-3} \pm 1.20 \cdot 10^{-3}$ (−)	$2.28 \cdot 10^{-1} \pm 1.66 \cdot 10^{-2}$ (−)	**$8.07 \cdot 10^{-3} \pm 1.31 \cdot 10^{-3}$**
TRICOP	1	$6.21 \cdot 10^{-2} \pm 2.49 \cdot 10^{-2}$ (−)	$4.96 \cdot 10^{-2} \pm 1.36 \cdot 10^{-2}$ (−)	$1.04 \cdot 10^{-2} \pm 5.92 \cdot 10^{-5}$ (≈)	**$1.03 \cdot 10^{-2} \pm 5.14 \cdot 10^{-4}$**
	2	$9.52 \cdot 10^{-2} \pm 1.90 \cdot 10^{-2}$ (−)	$4.90 \cdot 10^{-2} \pm 9.31 \cdot 10^{-3}$ (−)	$1.08 \cdot 10^{-2} \pm 2.23 \cdot 10^{-4}$ (≈)	**$1.05 \cdot 10^{-2} \pm 5.52 \cdot 10^{-4}$**
	3	$8.17 \cdot 10^{-2} \pm 2.51 \cdot 10^{-2}$ (−)	$3.97 \cdot 10^{-2} \pm 5.04 \cdot 10^{-3}$ (−)	**$9.88 \cdot 10^{-3} \pm 1.41 \cdot 10^{-4}$ (+)**	$1.06 \cdot 10^{-2} \pm 4.73 \cdot 10^{-4}$
	4	$7.72 \cdot 10^{-2} \pm 2.22 \cdot 10^{-2}$ (−)	$4.02 \cdot 10^{-2} \pm 6.24 \cdot 10^{-3}$ (−)	**$1.02 \cdot 10^{-2} \pm 3.85 \cdot 10^{-4}$ (+)**	$1.07 \cdot 10^{-2} \pm 4.01 \cdot 10^{-4}$
	5	$8.22 \cdot 10^{-2} \pm 1.90 \cdot 10^{-2}$ (−)	$3.68 \cdot 10^{-2} \pm 2.98 \cdot 10^{-3}$ (−)	**$9.75 \cdot 10^{-3} \pm 3.34 \cdot 10^{-4}$ (≈)**	$9.99 \cdot 10^{-3} \pm 4.32 \cdot 10^{-4}$
	6	$7.11 \cdot 10^{-2} \pm 2.22 \cdot 10^{-2}$ (−)	$3.76 \cdot 10^{-2} \pm 4.30 \cdot 10^{-3}$ (−)	**$1.02 \cdot 10^{-2} \pm 4.04 \cdot 10^{-4}$ (≈)**	$1.03 \cdot 10^{-2} \pm 6.96 \cdot 10^{-4}$
	10	$6.58 \cdot 10^{-2} \pm 2.09 \cdot 10^{-2}$ (−)	$3.18 \cdot 10^{-2} \pm 4.24 \cdot 10^{-3}$ (−)	**$9.87 \cdot 10^{-3} \pm 3.53 \cdot 10^{-4}$ (≈)**	$9.98 \cdot 10^{-3} \pm 3.69 \cdot 10^{-4}$
	20	$4.62 \cdot 10^{-2} \pm 1.10 \cdot 10^{-2}$ (−)	$2.63 \cdot 10^{-2} \pm 3.19 \cdot 10^{-3}$ (−)	$1.21 \cdot 10^{-2} \pm 6.52 \cdot 10^{-4}$ (≈)	**$1.20 \cdot 10^{-2} \pm 5.02 \cdot 10^{-4}$**
SPD	1	$5.95 \cdot 10^{-2} \pm 3.31 \cdot 10^{-3}$ (−)	$6.46 \cdot 10^{-2} \pm 5.16 \cdot 10^{-3}$ (−)	$1.55 \cdot 10^{-2} \pm 9.35 \cdot 10^{-4}$ (−)	**$8.78 \cdot 10^{-3} \pm 1.68 \cdot 10^{-4}$**
	2	$6.75 \cdot 10^{-2} \pm 9.27 \cdot 10^{-3}$ (−)	$5.78 \cdot 10^{-2} \pm 3.81 \cdot 10^{-3}$ (−)	$1.33 \cdot 10^{-2} \pm 1.15 \cdot 10^{-3}$ (−)	**$8.62 \cdot 10^{-3} \pm 2.00 \cdot 10^{-4}$**
	3	$6.02 \cdot 10^{-2} \pm 4.04 \cdot 10^{-3}$ (−)	$5.73 \cdot 10^{-2} \pm 5.38 \cdot 10^{-3}$ (−)	$1.22 \cdot 10^{-2} \pm 4.52 \cdot 10^{-4}$ (−)	**$8.61 \cdot 10^{-3} \pm 2.55 \cdot 10^{-4}$**
	4	$6.37 \cdot 10^{-2} \pm 5.26 \cdot 10^{-3}$ (−)	$6.03 \cdot 10^{-2} \pm 4.45 \cdot 10^{-3}$ (−)	$1.21 \cdot 10^{-2} \pm 3.87 \cdot 10^{-4}$ (−)	**$8.84 \cdot 10^{-3} \pm 1.78 \cdot 10^{-4}$**
	5	$5.96 \cdot 10^{-2} \pm 5.43 \cdot 10^{-3}$ (−)	$5.95 \cdot 10^{-2} \pm 7.26 \cdot 10^{-3}$ (−)	$1.29 \cdot 10^{-2} \pm 5.93 \cdot 10^{-4}$ (−)	**$8.77 \cdot 10^{-3} \pm 1.54 \cdot 10^{-4}$**
	6	$5.78 \cdot 10^{-2} \pm 2.68 \cdot 10^{-3}$ (−)	$5.87 \cdot 10^{-2} \pm 3.80 \cdot 10^{-3}$ (−)	$1.28 \cdot 10^{-2} \pm 5.15 \cdot 10^{-4}$ (−)	**$9.19 \cdot 10^{-3} \pm 1.91 \cdot 10^{-4}$**
	10	$5.90 \cdot 10^{-2} \pm 3.95 \cdot 10^{-3}$ (−)	$5.93 \cdot 10^{-2} \pm 5.02 \cdot 10^{-3}$ (−)	$1.51 \cdot 10^{-2} \pm 8.73 \cdot 10^{-4}$ (−)	**$9.68 \cdot 10^{-3} \pm 2.19 \cdot 10^{-4}$**
	20	$5.89 \cdot 10^{-2} \pm 6.15 \cdot 10^{-3}$ (−)	$6.10 \cdot 10^{-2} \pm 4.56 \cdot 10^{-3}$ (−)	$1.84 \cdot 10^{-2} \pm 1.02 \cdot 10^{-3}$ (−)	**$1.30 \cdot 10^{-2} \pm 3.34 \cdot 10^{-4}$**
CSI	1	$6.04 \cdot 10^{-2} \pm 3.48 \cdot 10^{-3}$ (−)	$1.10 \cdot 10^{-1} \pm 1.53 \cdot 10^{-2}$ (−)	$1.91 \cdot 10^{-2} \pm 2.07 \cdot 10^{-3}$ (≈)	**$1.82 \cdot 10^{-2} \pm 2.69 \cdot 10^{-3}$**
	2	$6.82 \cdot 10^{-2} \pm 5.25 \cdot 10^{-3}$ (−)	$7.00 \cdot 10^{-2} \pm 3.47 \cdot 10^{-3}$ (−)	$1.60 \cdot 10^{-2} \pm 5.08 \cdot 10^{-4}$ (−)	**$1.55 \cdot 10^{-2} \pm 4.95 \cdot 10^{-4}$**
	3	$6.37 \cdot 10^{-2} \pm 4.53 \cdot 10^{-3}$ (−)	$6.82 \cdot 10^{-2} \pm 6.44 \cdot 10^{-3}$ (−)	$1.54 \cdot 10^{-2} \pm 5.45 \cdot 10^{-4}$ (−)	**$1.48 \cdot 10^{-2} \pm 2.45 \cdot 10^{-4}$**
	4	$6.78 \cdot 10^{-2} \pm 3.57 \cdot 10^{-3}$ (−)	$6.84 \cdot 10^{-2} \pm 3.28 \cdot 10^{-3}$ (−)	$1.61 \cdot 10^{-2} \pm 4.98 \cdot 10^{-4}$ (−)	**$1.54 \cdot 10^{-2} \pm 3.11 \cdot 10^{-4}$**
	5	$6.93 \cdot 10^{-2} \pm 3.22 \cdot 10^{-3}$ (−)	$6.81 \cdot 10^{-2} \pm 7.55 \cdot 10^{-3}$ (−)	$1.60 \cdot 10^{-2} \pm 4.18 \cdot 10^{-4}$ (−)	**$1.55 \cdot 10^{-2} \pm 3.41 \cdot 10^{-4}$**
	6	$6.75 \cdot 10^{-2} \pm 3.50 \cdot 10^{-3}$ (−)	$6.59 \cdot 10^{-2} \pm 4.40 \cdot 10^{-3}$ (−)	$1.64 \cdot 10^{-2} \pm 3.98 \cdot 10^{-4}$ (≈)	**$1.61 \cdot 10^{-2} \pm 5.18 \cdot 10^{-4}$**
	10	$6.45 \cdot 10^{-2} \pm 5.78 \cdot 10^{-3}$ (−)	$6.60 \cdot 10^{-2} \pm 2.35 \cdot 10^{-3}$ (−)	$2.03 \cdot 10^{-2} \pm 7.23 \cdot 10^{-4}$ (≈)	**$2.00 \cdot 10^{-2} \pm 5.22 \cdot 10^{-4}$**
	20	$7.03 \cdot 10^{-2} \pm 7.85 \cdot 10^{-3}$ (−)	$6.45 \cdot 10^{-2} \pm 5.63 \cdot 10^{-3}$ (−)	**$3.78 \cdot 10^{-2} \pm 3.68 \cdot 10^{-3}$ (≈)**	$3.94 \cdot 10^{-2} \pm 1.83 \cdot 10^{-3}$
WP	1	$2.77 \cdot 10^{-1} \pm 6.16 \cdot 10^{-2}$ (−)	$2.82 \cdot 10^{-1} \pm 4.32 \cdot 10^{-2}$ (−)	**$2.92 \cdot 10^{-2} \pm 9.66 \cdot 10^{-4}$ (≈)**	$2.96 \cdot 10^{-2} \pm 1.52 \cdot 10^{-3}$
	2	$2.11 \cdot 10^{-1} \pm 2.80 \cdot 10^{-2}$ (−)	$2.19 \cdot 10^{-1} \pm 5.69 \cdot 10^{-2}$ (−)	**$2.85 \cdot 10^{-2} \pm 4.62 \cdot 10^{-4}$ (+)**	$3.00 \cdot 10^{-2} \pm 9.23 \cdot 10^{-4}$
	3	$1.92 \cdot 10^{-1} \pm 2.40 \cdot 10^{-2}$ (−)	$2.02 \cdot 10^{-1} \pm 3.50 \cdot 10^{-2}$ (−)	**$3.03 \cdot 10^{-2} \pm 9.67 \cdot 10^{-4}$ (≈)**	$3.07 \cdot 10^{-2} \pm 1.21 \cdot 10^{-3}$
	4	$1.79 \cdot 10^{-1} \pm 2.15 \cdot 10^{-2}$ (−)	$1.99 \cdot 10^{-1} \pm 2.96 \cdot 10^{-2}$ (−)	$2.97 \cdot 10^{-2} \pm 1.43 \cdot 10^{-3}$ (≈)	**$2.94 \cdot 10^{-2} \pm 9.56 \cdot 10^{-4}$**
	5	$1.82 \cdot 10^{-1} \pm 2.65 \cdot 10^{-2}$ (−)	$1.93 \cdot 10^{-1} \pm 3.50 \cdot 10^{-2}$ (−)	**$2.97 \cdot 10^{-2} \pm 7.08 \cdot 10^{-4}$ (≈)**	$3.08 \cdot 10^{-2} \pm 1.36 \cdot 10^{-3}$
	6	$1.61 \cdot 10^{-1} \pm 1.77 \cdot 10^{-2}$ (−)	$1.94 \cdot 10^{-1} \pm 4.62 \cdot 10^{-2}$ (−)	**$2.95 \cdot 10^{-2} \pm 4.75 \cdot 10^{-4}$ (+)**	$3.15 \cdot 10^{-2} \pm 1.15 \cdot 10^{-3}$
	10	$1.50 \cdot 10^{-1} \pm 3.08 \cdot 10^{-2}$ (−)	$1.76 \cdot 10^{-1} \pm 3.03 \cdot 10^{-2}$ (−)	$3.45 \cdot 10^{-2} \pm 1.80 \cdot 10^{-3}$ (−)	**$3.09 \cdot 10^{-2} \pm 1.14 \cdot 10^{-3}$**
	20	$1.57 \cdot 10^{-1} \pm 2.86 \cdot 10^{-2}$ (−)	$1.70 \cdot 10^{-1} \pm 3.11 \cdot 10^{-2}$ (−)	$3.38 \cdot 10^{-2} \pm 9.03 \cdot 10^{-4}$ (≈)	**$3.35 \cdot 10^{-2} \pm 1.01 \cdot 10^{-3}$**
Wilcoxon test summary		148 − 17 ≈ 11+	139 − 6 ≈ 31+	99 − 66 ≈ 11+	reference algorithm (176 comparisons)
mean Friedman rank		3.33	2.77	2.33	**1.56**

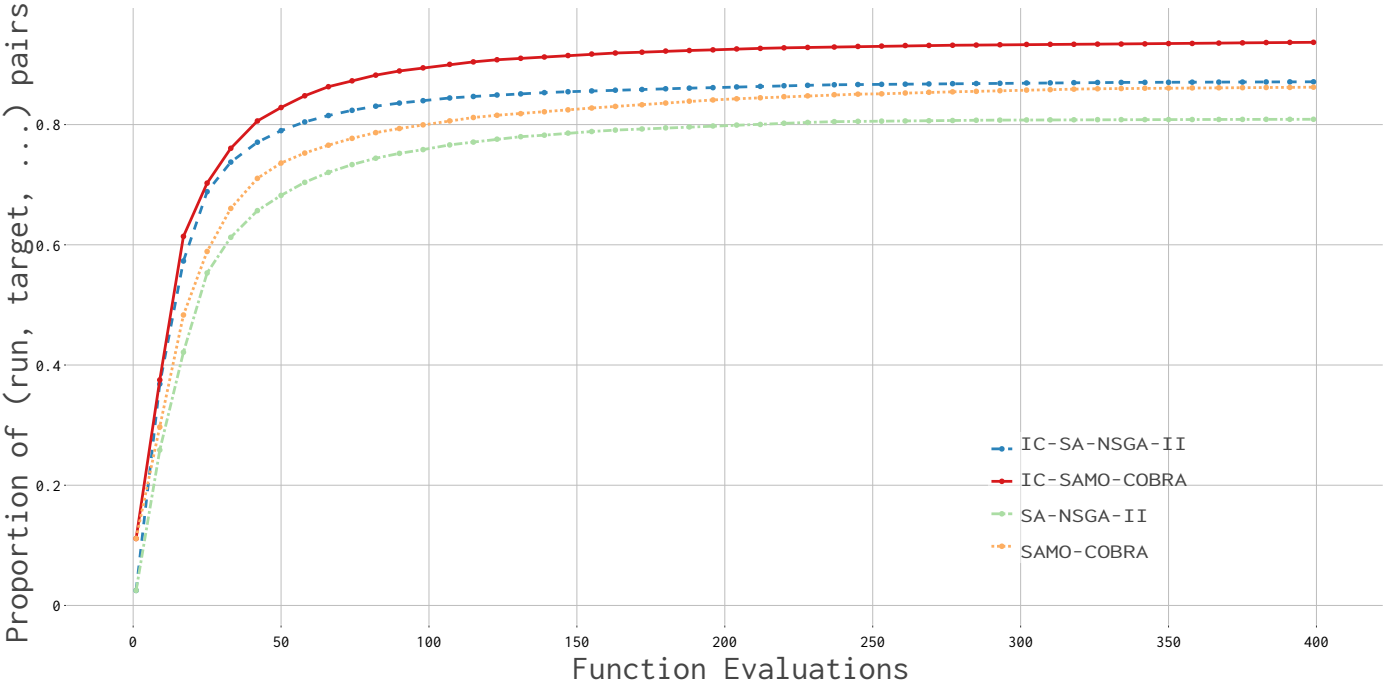


Figure 3: Empirical Cumulative Distribution Functions of hypervolume performance metric for SA-NSGA-II, IC-SA-NSGA-II, SAMO-COBRA and IC-SAMO-COBRA. All experiments with different numbers of candidate solutions per iteration and on different test functions are aggregated.

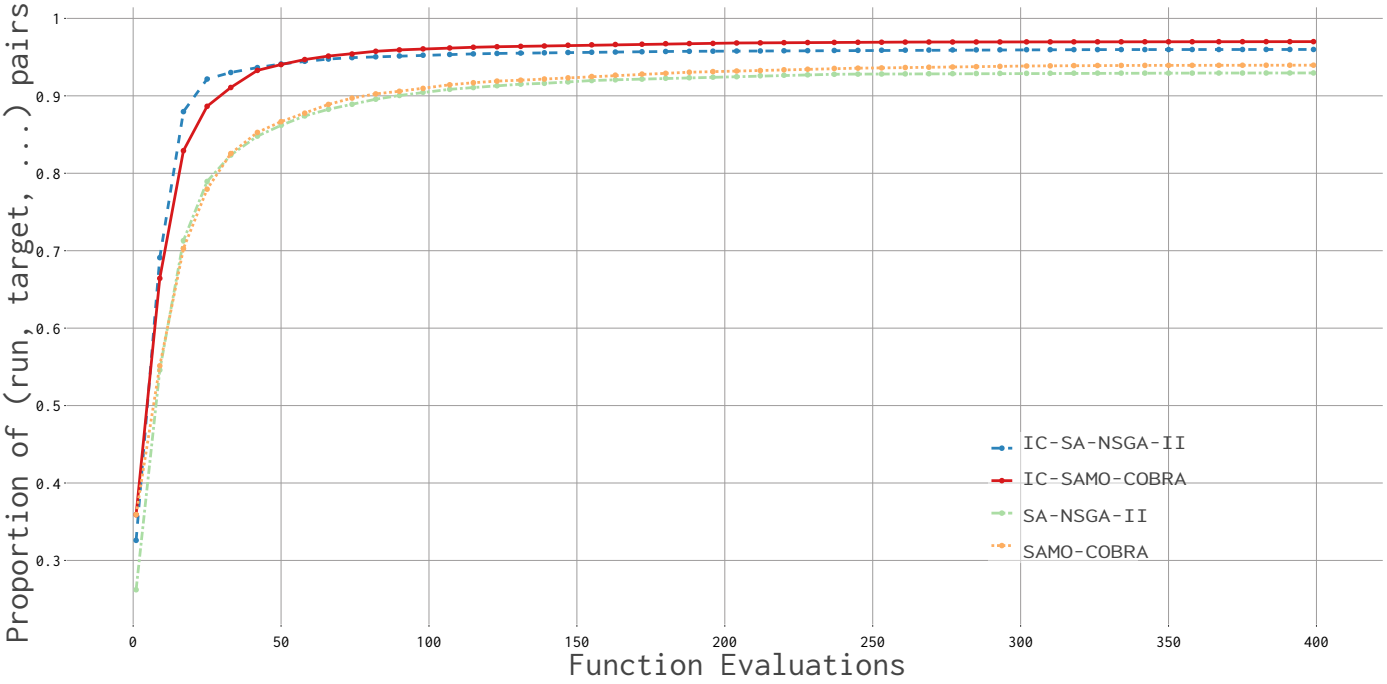


Figure 4: Empirical Cumulative Distribution Functions of IGD+ performance metric for SA-NSGA-II, IC-SA-NSGA-II, SAMO-COBRA and IC-SAMO-COBRA. All experiments with different numbers of candidate solutions per iteration and on different test functions are aggregated.

objective and the corresponding comparison between the required damage stability constraint is computationally expensive. Evaluation of the damage stability index requires a run of the commercial maritime simulator Delftship pro[4]. The volumetric objective and the three volumetric constraints are in-

[4]Version 14.20.343; see Delftship: Visual hull modeling and stability analysis. `https://www.delftship.net/`

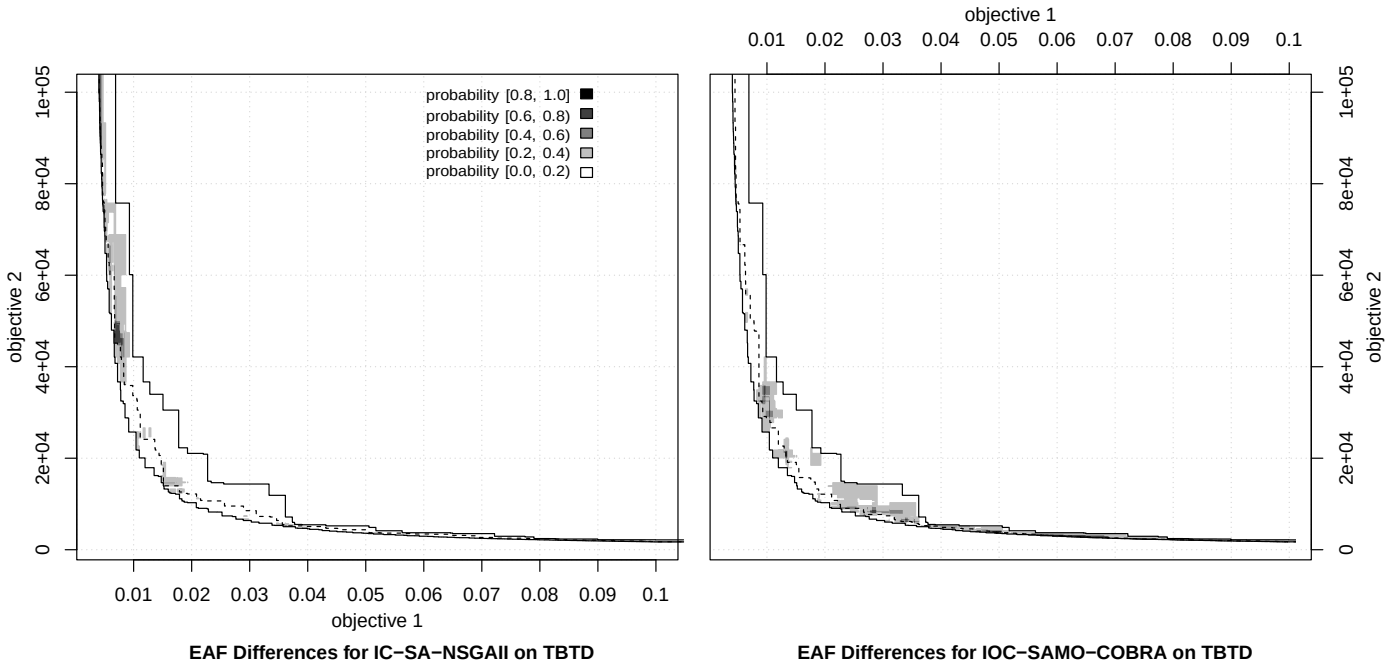


Figure 5: Visualization of the Empirical Attainment Function differences between IC-SA-NSGA-II and the IOC-SAMO-COBRA algorithm on the TBTD problem. The solid, dashed and solid lines from left to right represent the best, median and worst found Pareto frontier of both algorithms combined. The grey level in the plots encodes the probability that the corresponding algorithm outperforms the other algorithm in that region. In objective 1, IC-SA-NSGA-II finds smaller values while in objective 2, IOC-SAMO-COBRA finds smaller values.

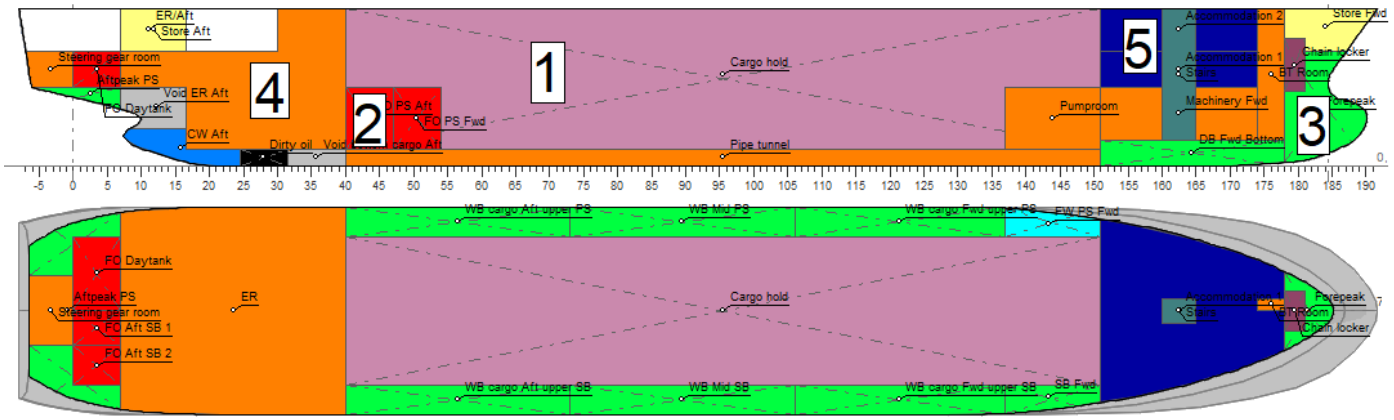


Figure 6: Longitudinal section and top view of cargo vessel design optimized for damage stability (survivability) and cargo hold capacity (both to be maximized). The pink compartments (1) annotates the cargo hold, the red compartments (2) are fuel tanks, the green compartments (3) are water ballast tanks, the orange compartments (4) are the technical spaces, and the blue compartments (5) are part of the crew accommodation.

expensive to evaluate and can be computed much faster compared to fitting and interpolating an RBF surrogate. The inexpensive constraints and objective are therefore directly used in the IOC-SAMO-COBRA algorithm while for the expensive objective and constraint, RBF surrogates are updated and selected every iteration. More details about the ship design problem are given in [75, 76].

On this real-world problem, we perform the following three experiments:

1. IOC-SAMO-COBRA with number of candidate solutions per iteration $p = 1$ and 300 function evaluations.

2. IOC-SAMO-COBRA with number of candidate solutions per iteration $p = 3$ and 501 function evaluations.

3. SA-NSGA-II with number of candidate solutions per iteration $p = 3$ and 501 function evaluations.

The experiments with $p = 3$ aim at investigating the potential benefit of parallelism in terms of wall-clock time provided that the corresponding number of simulator licenses is available. The IGD+ metric can not be computed for this design problem since the Pareto front is unknown, therefore the HV metric is used to compare the performance of SA-NSGA-II and IOC-SAMO-COBRA. IC-SA-NSGA-II is not experimented with since one of the constraints is expensive to evaluate, and IC-SA-NSGA-II does not have a build in option to use RBFs for that constraint.

7.1. Results on Cargo Vessel Design

For the expensive objective (damage stability), requiring a Delftship pro simulator run, the median evaluation time was 248 seconds. In experiment 1 (IOC-SAMO-COBRA, $p = 1$, 300 evaluations), a HV of 9115 with respect to the reference point $(0, 0)$ was obtained. In experiment 2 (IOC-SAMO-COBRA, $p = 3$, 501 evaluations), the same HV was obtained in the 129^{th} iteration (after 385 function evaluations), saving a total wall-clock time of 682 minutes compared to experiment 1.

A comparison of the Pareto fronts resulting from experiments 2 and 3 (i.e., a direct comparison between IOC-SAMO-COBRA and SA-NSGA-II) is shown in Figure 7, where cargo hold capacity (↑ max) is shown on the y-axis and the attained damage stability index (↑ max) on the x-axis. The Pareto front obtained by the SA-NSGA-II algorithm is dominated by the obtained Pareto front obtained by IOC-SAMO-COBRA, and the latter algorithm also finds more extreme solutions (especially for damage stability).

Figure 8 illustrates the convergence of the algorithms by showing the HV (measured between the Pareto fronts obtained by the two algorithms and the approximated Nadir point) over the number of function evaluations. The difference in the two Pareto fronts (Figure 7) is also clearly visible in this illustration. The different behavior in the first few evaluations can be explained by the difference in the initial sampling strategies (Latin Hypercube Sampling [25] for SA-NSGA-II vs. Halton Sampling [26] for IOC-SAMO-COBRA).

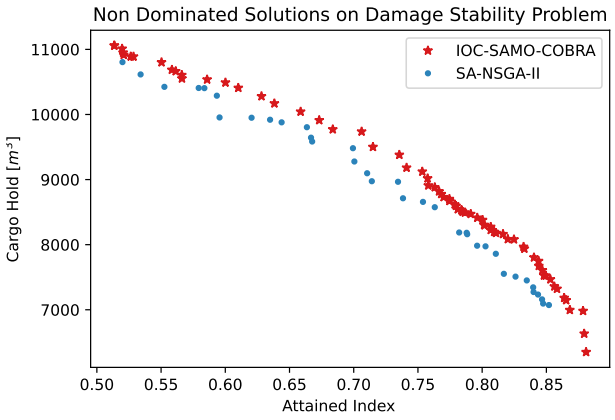


Figure 7: Comparison of Pareto fronts obtained by IOC-SAMO-COBRA and SA-NSGA-II ($p = 3$, each) on the ship design problem. x-axis: Survivability index; y-axis: Cargo hold (both ↑ max).

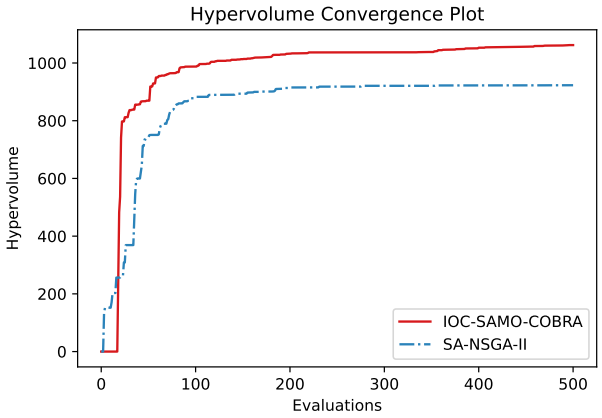


Figure 8: Comparison of the hypervolume maximization progress between IOC-SAMO-COBRA and SA-NSGA-II ($p = 3$, each) on the ship design problem. x-axis: Number of function evaluations; y-axis: Hypervolume.

7.2. Analysis of the Findings for Cargo Vessel Design

The IOC-SAMO-COBRA results were further analyzed by naval architects to understand and interpret them in the light of vessel design expertise, resulting in the following observations:

- For every point on the Pareto front, the parameter that defines the tanktop height has converged to the minimum value. The algorithm learned that extra height in the double bottom of the vessel does not improve the damage stability index. The compartments above the tanktop benefited from this in terms of their size. Interestingly, this finding could be confirmed since it is also prescribed in the International Convention for the Safety of Life at Sea (SOLAS chapter II-1 part B-2 regulation 9) [77].

- The algorithm also found that a large space between the hull and cargo hold is beneficial for the damage stability criterion. This result can be explained well by the fact that a small distance between the hull and cargo hold makes it less likely for the design to survive in case of damage

(flooding of the cargo hold will always lead to the loss of a single cargo hold vessel).

8. Conclusions and Future Work

In this paper, the multi-objective algorithm SAMO-COBRA (designed for expensive objective functions and constraints) has been extended so that it can deal with both expensive and inexpensive objectives and constraints. The resulting new IOC-SAMO-COBRA algorithm has been compared to a state-of-the-art surrogate-assisted genetic algorithm, IC-SA-NSGA-II. By testing on a diverse set of test functions, it has been shown that exploiting the inexpensiveness of constraint functions is beneficial since this will, in the majority of cases, lead to better Pareto front approximations. Besides comparing the algorithm on test functions, both IOC-SAMO-COBRA and SA-NSGA-II have also been applied to a real-world ship design damage stability optimization problem. Due to the combination of build-in parallelism with the multi-point acquisition function and the exploitation of the inexpensive constraints and objectives, IOC-SAMO-COBRA is able to save a significant amount of wall-clock-time. Moreover, IOC-SAMO-COBRA converges fast and is able to find well-spread solutions along the Pareto front. According to domain experts, the solutions found by the algorithm are sensible when the goal is to optimize damage stability and cargo hold volume.

Measured in terms of HV and IGD+, IOC-SAMO-COBRA outperforms IC-SA-NSGA-II in 78% of the test cases and on the real-world vessel design optimization problem. The key algorithmic components we have considered to be responsible for this advantage include (i) computing 12 RBF configurations for each expensive objective/constraint and picking the best as a surrogate model for the respective objective/constraint and (ii) using COBYLA repeatedly to simultaneously find p solution candidates that maximize their joint HV contribution. For two test functions (BICOP1 and MW2) that do not have any active constraints on the Pareto front, IC-SA-NSGA-II has outperformed IOC-SAMO-COBRA. Further research is required to improve the IOC-SAMO-COBRA algorithm to be able to quickly find Pareto fronts not subject to active constraints. These results also indicate that further research efforts should be put into combining the strengths of IOC-SAMO-COBRA and IC-SA-NSGA-II and go beyond their current capabilities, for example as follows:

- Investigate an extension of IOC-SAMO-COBRA by incorporating the crowding distance mechanism of IC-SA-NSGA-II, to improve IOC-SAMO-COBRA for those problems where it currently is not competitive with IC-SA-NSGA-II.
- Investigate how the local search strategy of IOC-SAMO-COBRA, COBYLA, can be used in IC-SA-NSGA-II to speed up the convergence of the genetic algorithm.

A final open issue is handling mixed-integer decision parameters, as the extension to such design spaces is crucial for many real-world applications. For extending the IC-SA-NSGA-II with this functionality, the surrogate modeling strategy needs to be adjusted since the NSGA-II algorithm back-end optimizer can already deal with mixed-integer parameter spaces. For extending IOC-SAMO-COBRA accordingly, there is a need to replace COBYLA, which is designed purely for continuous parameter spaces, with an evolutionary algorithm for handling the mixed-integer parameter space.

CRediT authorship contribution statement

Roy de Winter: Conceptualization, Formal analysis, Investigation, Methodology, Visualization, Writing - original Draft. **Bas Milatz:** Software, Validation, Writing - review & editing. **Julian Blank** Writing - review & editing. **Niki van Stein** Supervision, Writing -review & editing. **Thomas Bäck** Supervision, Validation, Writing - review & editing. **Kalyanmoy Deb** Supervision, Validation, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data Availability

The algorithm code and raw results from the experiments are available and can be found on a dedicated Github page [71].

Appendix A. Empirical Attainment Difference Functions

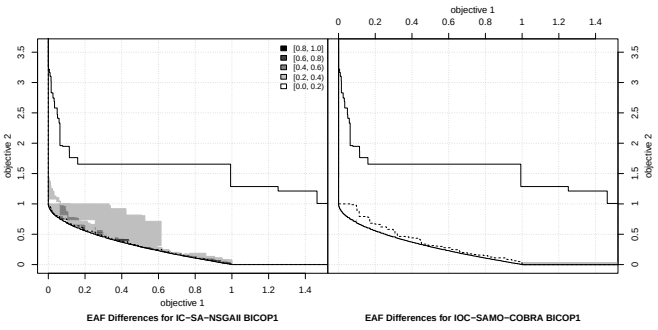


Figure A.9: EAF difference plot BIOCP1

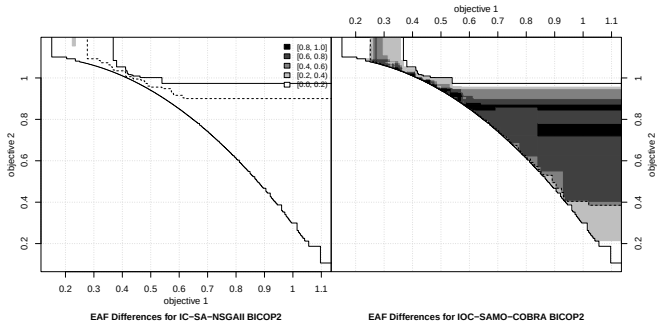


Figure A.10: EAF difference plot BIOCP2

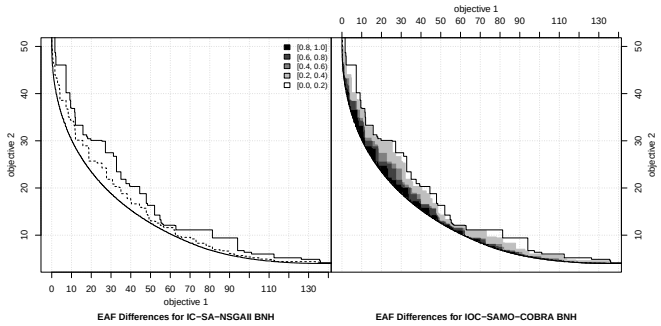


Figure A.11: EAF difference plot BNH

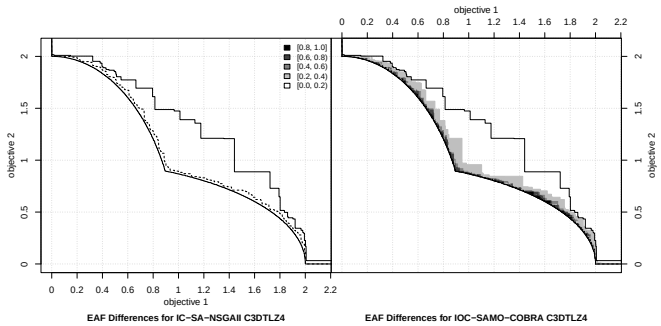


Figure A.12: EAF difference plot C3DTLZ4

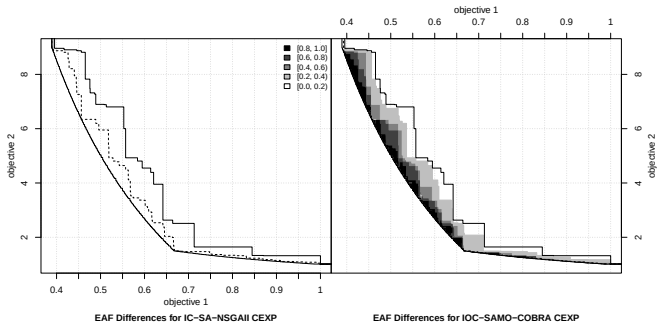


Figure A.13: EAF difference plot CEXP

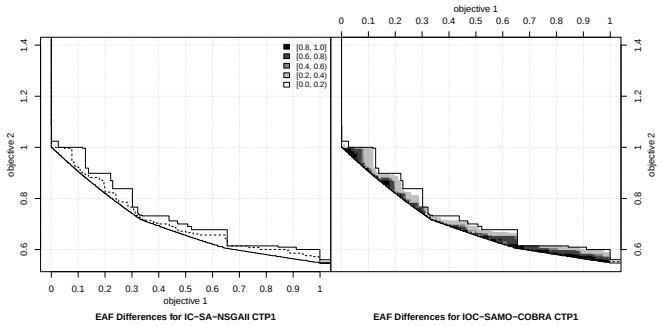


Figure A.14: EAF difference plot CTP1

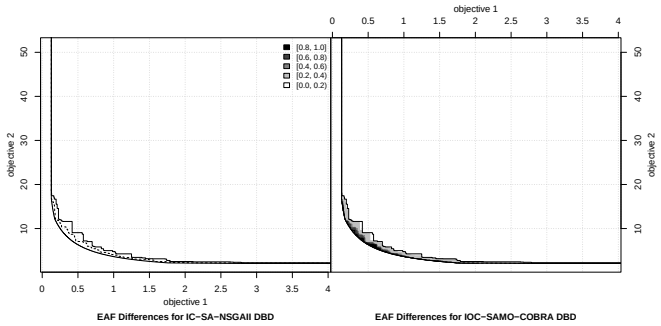


Figure A.15: EAF difference plot DBD

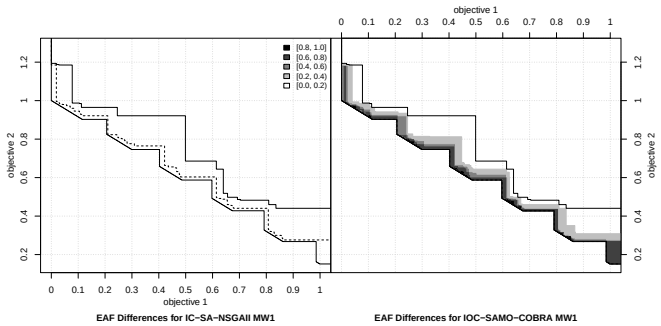


Figure A.16: EAF difference plot MW1

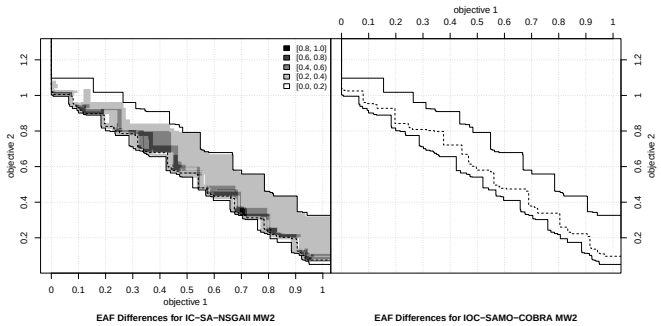


Figure A.17: EAF difference plot MW2

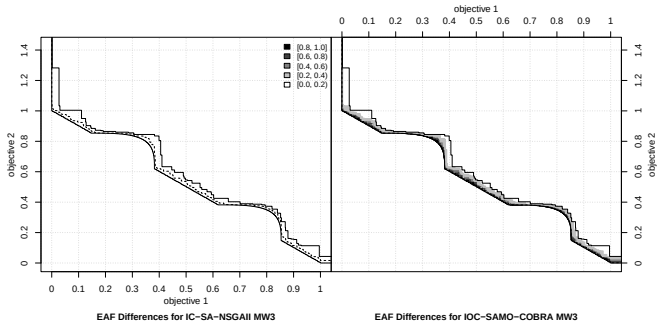


Figure A.18: EAF difference plot MW3

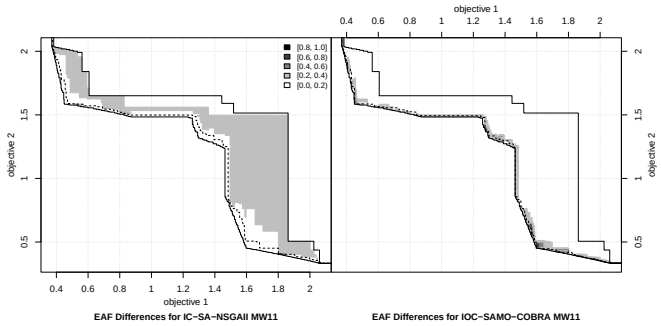


Figure A.19: EAF difference plot MW11

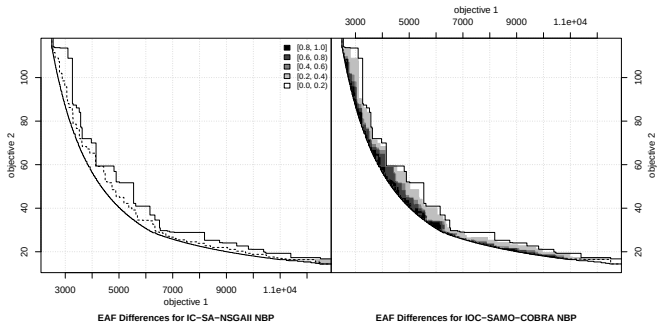


Figure A.20: EAF difference plot NBP

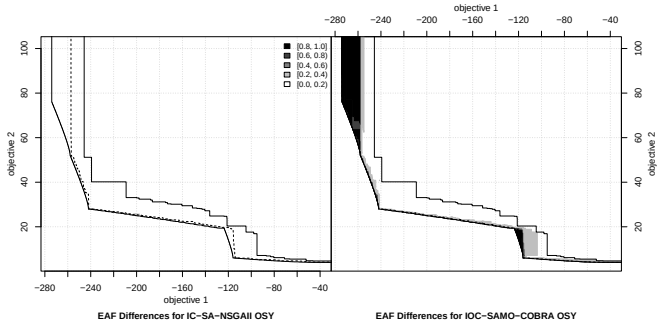


Figure A.21: EAF difference plot OSY

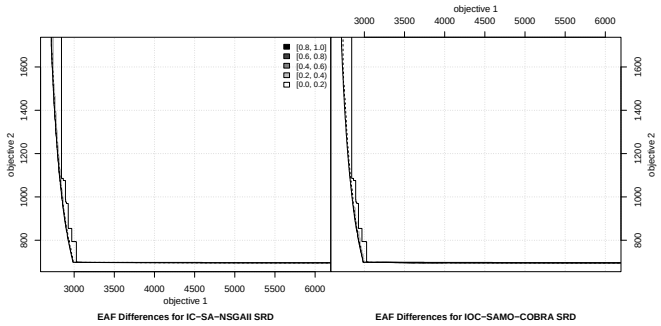


Figure A.22: EAF difference plot SRD

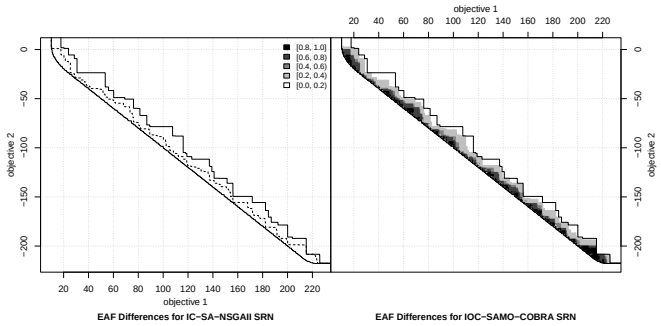


Figure A.23: EAF difference plot SRN

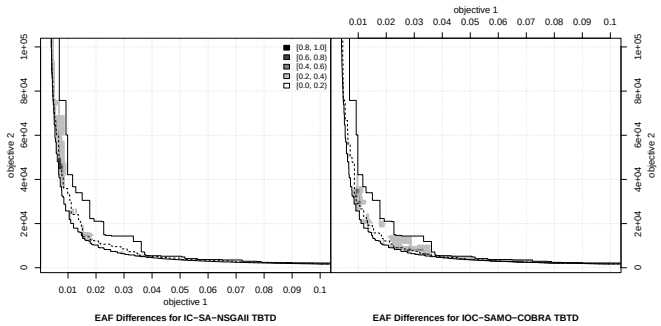


Figure A.24: EAF difference plot TBTD

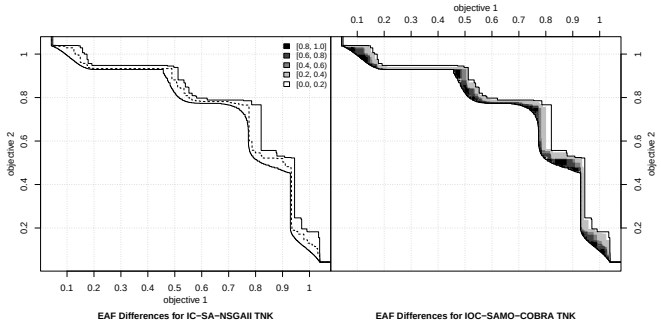


Figure A.25: EAF difference plot TNK

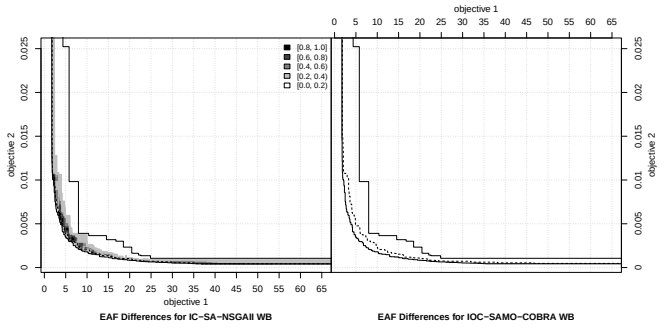


Figure A.26: EAF difference plot WB

References

[1] K. van der Blom, T. M. Deist, V. Volz, M. Marchi, Y. Nojima, B. Naujoks, A. Oyama, T. Tušar, Identifying properties of real-world optimisation problems through a questionnaire, in: Many-Criteria Optimization and Decision Analysis: State-of-the-Art, Present Challenges, and Future Perspectives, Springer, 2023, pp. 59–80.

[2] S. Bandaru, A. H. Ng, K. Deb, Data mining methods for knowledge discovery in multi-objective optimization: Part A-Survey, Expert Systems with Applications 70 (2017) 139–159.

[3] Y. Yang, J. Liu, S. Tan, A multi-objective evolutionary algorithm for steady-state constrained multi-objective optimization problems, Applied Soft Computing 101 (107042) (2021).

[4] K. Deb, Multi-objective optimisation using evolutionary algorithms: an introduction, in: L. Wang, A. H. C. Ng, K. Deb (Eds.), Multi-objective evolutionary optimisation for product design and manufacturing, Springer, 2011, pp. 3–34.

[5] F. X. Long, B. van Stein, M. Frenzel, P. Krause, M. Gitterle, T. Bäck, Learning the characteristics of engineering optimization problems with applications in automotive crash, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2022, pp. 1227–1236.

[6] T. Kumano, S. Jeong, S. Obayashi, Y. Ito, K. Hatanaka, H. Morino, Multidisciplinary design optimization of wing shape for a small jet aircraft using Kriging model, in: Collection of Technical Papers - 44th AIAA Aerospace Sciences Meeting, Collection of Technical Papers - 44th AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics Inc., 2006, pp. 11158–11170. doi:10.2514/6.2006-932.

[7] Y. Yao, X. Yang, Efficient global multi-objective aerodynamic optimization using combined multi-point infilling strategy and surrogate models, in: IEEE Congress on Evolutionary Computation (CEC), IEEE, 2021, pp. 1537–1542.

[8] A. Papanikolaou, G. Zaraphonitis, M. Jokinen, A. Aubert, S. Harries, J. Marzi, G. Mermiris, R. Gunawan, Holistic ship design for green shipping, in: SNAME Maritime Convention, OnePetro, 2022. doi:10.5957/SMC-2022-020.

[9] R. de Winter, J. Furustam, T. Bäck, T. Muller, Optimizing ships using the holistic accelerated concept design methodology, in: T. Okada, K. Suzuki, Y. Kawamura (Eds.), Practical Design of Ships and Other Floating Structures, Springer, Singapore, 2021, pp. 38–50.

[10] R. T. Haftka, D. Villanueva, A. Chaudhuri, Parallel surrogate-assisted global optimization with expensive functions - a survey, Structural and Multidisciplinary Optimization 54 (1) (2016) 3–13. doi:10.1007/s00158-016-1432-3.

[11] A. E. Eiben, J. E. Smith, Introduction to evolutionary computing, Natural Computing Series, Springer, Berlin, 2003. doi:https://doi.org/10.1007/978-3-662-44874-8.

[12] J. Močkus, On Bayesian methods for seeking the extremum, in: J. Stoer (Ed.), Optimization techniques IFIP technical conference, Springer, 1975, pp. 400–404.

[13] Y. S. Ong, P. Nair, A. Keane, K. Wong, Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems, in: Y. Jin (Ed.), Knowledge Incorporation in Evolutionary Computation, Springer, 2005, pp. 307–331.

[14] J. Blank, K. Deb, Constrained bi-objective surrogate-assisted optimization of problems with heterogeneous evaluation times: Expensive objectives and inexpensive constraints, in: H. Ishibuchi, Q. Zhang, R. Cheng, K. Li, H. Li, H. Wang, A. Zhou (Eds.), International Conference on Evolutionary Multi-Criterion Optimization (EMO), Springer, 2021, pp. 257–269.

[15] M. J. Powell, The theory of radial basis function approximation in 1990, Advances in numerical analysis (1992) 105–210.

[16] R. d. Winter, B. v. Stein, T. Bäck, SAMO-COBRA: A fast surrogate assisted constrained multi-objective optimization algorithm, in: H. Ishibuchi, Q. Zhang, R. Cheng, K. Li, H. Li, H. Wang, A. Zhou (Eds.), International Conference on Evolutionary Multi-Criterion Optimization (EMO), Springer, 2021, pp. 270–282.

[17] R. de Winter, P. Bronkhorst, B. van Stein, T. Bäck, Constrained multi-objective optimization with a limited budget of function evaluations, Memetic Computing 14 (2022) 151–164. doi:10.1007/s12293-022-00363-y.

[18] R. de Winter, B. van Stein, T. Bäck, Multi-point acquisition function for constraint parallel efficient multi-objective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2022, pp. 511–519.

[19] E. Zitzler, L. Thiele, Multiobjective optimization using evolutionary algorithms—a comparative case study, in: A. E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (Eds.), International conference on parallel problem solving from nature (PPSN), Springer, 1998, pp. 292–301.

[20] N. Beume, B. Naujoks, M. Emmerich, SMS-EMOA: Multiobjective selection based on dominated hypervolume, European Journal of Operational Research 181 (3) (2007) 1653–1669. doi:https://doi.org/10.1016/j.ejor.2006.08.008.

[21] N. Riquelme, C. Von Lücken, B. Baran, Performance metrics in multi-objective optimization, in: 2015 Latin American computing conference (CLEI), IEEE, 2015, pp. 1–11.

[22] Y. Tian, R. Cheng, X. Zhang, M. Li, Y. Jin, Diversity assessment of multi-objective evolutionary algorithms: Performance metric and benchmark problems, IEEE Computational Intelligence Magazine 14 (3) (2019) 61–74.

[23] H. Ishibuchi, H. Masuda, Y. Tanigaki, Y. Nojima, Modified distance calculation in generational distance and inverted generational distance, in: Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part II 8, Springer, 2015, pp. 110–125.

[24] D. Berengut, Statistics for experimenters: Design, innovation, and discovery, The American Statistician 60 (2006) 341–342.

[25] M. Stein, Large sample properties of simulations using latin hypercube sampling, Technometrics 29 (2) (1987) 143–151.

[26] J. H. Halton, On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals, Numerische Mathematik 2 (1) (1960) 84–90.

[27] I. Sobol, On the distribution of points in a cube and the approximate evaluation of integrals, USSR Computational Mathematics and Mathematical Physics 7 (4) (1967) 86–112. doi:https://doi.org/10.1016/0041-5553(67)90144-9.

[28] J. Bossek, C. Doerr, P. Kerschke, Initial design strategies and their effects on sequential model-based optimization: an exploratory case study based on BBOB, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2020, pp. 778–786.

[29] D. Hardin, E. Saff, Minimal riesz energy point configurations for rectifiable d-dimensional manifolds, Advances in Mathematics 193 (1) (2005) 174–204.

[30] J. Blank, K. Deb, Y. Dhebar, S. Bandaru, H. Seada, Generating well-spaced points on a unit simplex for evolutionary many-objective optimization, IEEE Transactions on Evolutionary Computation 25 (1) (2021) 48–60. doi:10.1109/TEVC.2020.2992387.

[31] S. Bagheri, W. Konen, T. Bäck, Comparing Kriging and Radial Basis Function surrogates, in: F. Hoffmann, E. Hüllermeier, R. Mikut (Eds.), Proc. 27. Workshop Computational Intelligence, Universitätsverlag Karlsruhe, 2017, pp. 243–259.

[32] K. Elsayed, D. Vucinic, R. Dippolito, C. Lacor, Comparison between RBF and Kriging surrogates in design optimization of high dimensional problems, in: 3rd International Conference on Engineering Optimization, 2012.

[33] M. D. Buhmann, Radial basis functions: theory and implementations, Cambridge University Press, 2003. doi:https://doi.org/10.1017/CBO9780511543241.

[34] S. Bagheri, W. Konen, M. Emmerich, T. Bäck, Self-adjusting parameter control for surrogate-assisted constrained optimization under limited budgets, Applied Soft Computing 61 (2017) 377–393. doi:10.1016/j.asoc.2017.07.060.

[35] C. A. Micchelli, Interpolation of scattered data: distance matrices and conditionally positive definite functions, Constructive approximation 2 (1) (1986) 11–22.

[36] J.-Y. Li, Z.-H. Zhan, J. Zhang, Evolutionary computation for expensive optimization: A survey, International Journal of Automation and Computing 18 (2021) 1–21. doi:https://doi.org/10.1007/s11633-022-1317-4.

[37] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm and Evolutionary Computation 1 (2) (2011) 61–70.

[38] R. G. Regis, A survey of surrogate approaches for expensive constrained black-box optimization, in: H. A. L. Thi, H. M. Le, T. P. Dinh (Eds.), World Congress on Global Optimization, Springer, 2019, pp. 37–47.

[39] T. Chugh, K. Sindhya, J. Hakanen, K. Miettinen, A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms, Soft Computing 23 (9) (2019) 3137–3166.

[40] R. Allmendinger, J. Knowles, Heterogeneous objectives: state-of-the-art and future research, arXiv preprint arXiv:2103.15546 (2021).

[41] T. Chugh, K. Sindhya, K. Miettinen, J. Hakanen, Y. Jin, On constraint handling in surrogate-assisted evolutionary many-objective optimization, in: Parallel Problem Solving from Nature–PPSN XIV: 14th International Conference, Edinburgh, UK, September 17-21, 2016, Proceedings 14, Springer, 2016, pp. 214–224.

[42] Z. Han, F. Liu, C. Xu, K. Zhang, Q. Zhang, Efficient multi-objective evolutionary algorithm for constrained global optimization of expensive functions, in: IEEE Congress on Evolutionary Computation (CEC), IEEE, 2019, pp. 2026–2033.

[43] A. Habib, H. K. Singh, T. Chugh, T. Ray, K. Miettinen, A multiple surrogate assisted decomposition-based evolutionary algorithm for expensive multi/many-objective optimization, IEEE Transactions on Evolutionary Computation 23 (6) (2019) 1000–1014.

[44] H. Wu, J. Ding, Q. Chen, Gaussian process-assisted evolutionary algorithm for constrained expensive multi-objective optimization, in: Asian Control Conference (ASCC), IEEE, 2022, pp. 1027–1032. doi:10.23919/ASCC56756.2022.9828304.

[45] C. K. Goh, D. Lim, L. Ma, Y.-S. Ong, P. S. Dutta, A surrogate-assisted memetic co-evolutionary algorithm for expensive constrained optimization problems, in: 2011 IEEE Congress of Evolutionary Computation (CEC), IEEE, 2011, pp. 744–749.

[46] Z. Song, H. Wang, B. Xue, M. Zhang, Y. Jin, Balancing objective optimization and constraint satisfaction in expensive constrained evolutionary multi-objective optimization, IEEE Transactions on Evolutionary Computation (2023).

[47] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[48] J. Blank, K. Deb, pysamoo: Surrogate-assisted multi-objective optimization in python (2022). arXiv:2204.05855.

[49] K. Deb, S. Agrawal, A niched-penalty approach for constraint handling in genetic algorithms, in: A. Dobnikar, N. C. Steele, D. W. Pearson, R. F. Albrecht (Eds.), Artificial neural nets and genetic algorithms, Springer Vienna, 1999, pp. 235–243.

[50] B. Khoshoo, J. Blank, T. Q. Pham, K. Deb, S. N. Foster, Optimal design of electric machine with efficient handling of constraints and surrogate assistance, Engineering Optimization (2023) 1–19arXiv:https://doi.org/10.1080/0305215X.2022.2152805, doi:10.1080/0305215X.2022.2152805.

[51] J. Blank, K. Deb, Handling constrained multi-objective optimization problems with heterogeneous evaluation times: proof-of-principle results, Memetic Computing 14 (2022) 1–16. doi:10.1007/s12293-022-00362-z.

[52] M. J. D. Powell, A direct search optimization method that models the objective and constraint functions by linear interpolation, in: S. Gomez, J.-P. Hennart (Eds.), Advances in Optimization and Numerical Analysis, Springer Netherlands, 1994, pp. 51–67. doi:10.1007/978-94-015-8330-5_4.

[53] T. Wortmann, C. Waibel, G. Nannicini, R. Evins, T. Schroepfer, J. Carmeliet, Are genetic algorithms really the best choice for building energy optimization?, in: M. Turrin, B. Peters, W. O'Brien, R. Stouffs, T. Dogan (Eds.), Proceedings of the Symposium on Simulation for Architecture and Urban Design, 2017, pp. 51–58.

[54] S. Gustafson, E. K. Burke, The speciating island model: An alternative parallel evolutionary algorithm, Journal of Parallel and Distributed Computing 66 (8) (2006) 1025–1036.

[55] S. Lee, S. B. Kim, Parallel simulated annealing with a greedy algorithm for bayesian network structure learning, IEEE Transactions on Knowledge and Data Engineering 32 (6) (2019) 1157–1166.

[56] A. Delévacq, P. Delisle, M. Gravel, M. Krajecki, Parallel ant colony optimization on graphics processing units, Journal of Parallel and Distributed Computing 73 (1) (2013) 52–61.

[57] R. d. Winter, B. v. Stein, M. Dijkman, T. Bäck, Designing ships using constrained multi-objective efficient global optimization, in: G. Nicosia, P. Pardalos, G. Giuffrida, R. Umeton, V. Sciacca (Eds.), International Conference on Machine Learning, Optimization, and Data Science, Springer, 2018, pp. 191–203.

[58] W. Ponweiser, T. Wagner, D. Biermann, M. Vincze, Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$-metric selection, in: G. Rudolph, T. Jansen, N. Beume, S. Lucas, C. Poloni (Eds.), International Conference on Parallel Problem Solving from Nature (PPSN), Springer, 2008, pp. 784–794. doi:10.1007/978-3-540-87700-4_78.

[59] D. R. Jones, M. Schonlau, W. J. Welch, Efficient global optimization of expensive black-box functions, Journal of Global optimization 13 (4) (1998) 455–492.

[60] C. A. C. Coello, G. B. Lamont, D. A. Van Veldhuizen, et al., Evolutionary algorithms for solving multi-objective problems, Springer, 2007. doi:https://doi.org/10.1007/978-0-387-36797-2.

[61] K. Deb, Multi-objective optimization using evolutionary algorithms, John Wiley & Sons, 2001.

[62] K. Deb, A. Pratap, T. Meyarivan, Constrained test problems for multi-objective evolutionary optimization, in: E. Zitzler, L. Thiele, K. Deb, C. A. C. Coello, D. Corne (Eds.), International conference on evolutionary multi-criterion optimization (EMO), Springer, 2001, pp. 284–298. doi:10.1007/3-540-44719-9_20.

[63] R. Tanabe, A. Oyama, A note on constrained multi-objective optimization benchmark problems, in: IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017, pp. 1127–1134.

[64] A. Forrester, A. Sobester, A. Keane, Engineering design via surrogate modelling: a practical guide, John Wiley & Sons, 2008. doi:10.2514/4.479557.

[65] R. Datta, R. G. Regis, A surrogate-assisted evolution strategy for constrained multi-objective optimization, Expert Systems with Applications 57 (2016) 270–284. doi:10.1016/j.eswa.2016.03.044.

[66] Z. Ma, Y. Wang, Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons, IEEE Transactions on Evolutionary Computation 23 (6) (2019) 972–986.

[67] W. Gong, Z. Cai, L. Zhu, An efficient multiobjective differential evolution algorithm for engineering design, Structural and Multidisciplinary Optimization 38 (2) (2009) 137–157. doi:10.1007/s00158-008-0269-9.

[68] M. G. Parsons, R. L. Scott, Formulation of multicriterion design optimization problems for solution with scalar numerical optimization methods, Journal of Ship Research 48 (1) (2004) 61–76. doi:10.1007/s10489-016-0825-8.

[69] H. Jain, K. Deb, An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach., IEEE Transactions on Evolutionary Computation 18 (4) (2014) 602–622. doi:10.1109/tevc.2013.2281534.

[70] S. Mirjalili, P. Jangir, S. Saremi, Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems, Applied Intelligence 46 (1) (2017) 79–95. doi:10.1007/s10489-016-0825-8.

[71] R. de Winter, IOC-SAMO-COBRA: Release 1.1.1 for Swarm and Evolutionary Computation (Jul. 2023). doi:10.5281/zenodo.8112883.
URL `https://doi.org/10.5281/zenodo.8112883`

[72] H. Wang, D. Vermetten, F. Ye, C. Doerr, T. Bäck, IOHanalyzer: Detailed performance analyses for iterative optimization heuristics, ACM Transactions on Evolutionary Learning and Optimization 2 (1) (apr 2022).

[73] N. Hansen, A. Auger, R. Ros, O. Mersmann, T. Tušar, D. Brockhoff, Coco: A platform for comparing continuous optimizers in a black-box setting, Optimization Methods and Software 36 (1) (2021) 114–144.

[74] M. López-Ibáñez, L. Paquete, T. Stützle, Exploratory analysis of stochastic local search algorithms in biobjective optimization, in: homas Bartz-Beielstein, M. Chiarandini, L. Paquete, M. Preuss (Eds.), Experimental methods for the analysis of optimization algorithms, Springer, 2010, pp. 209–222.

[75] B. Milatz, R. de Winter, J. D. van de Ridder, M. van Engeland, F. Mauro, A. A. Kana, Parameter space exploration for the probabilistic damage stability method for dry cargo ships, International Journal of Naval Architecture and Ocean Engineering (2023) 100549.

[76] B. Milatz, Multi-level optimisation and global sensitivity analysis of the probabilistic damage stability method for single hold ships, Master's thesis, Delft University of Technology (2022).
URL `http://resolver.tudelft.nl/uuid:24b94160-8804-4d9a-887f-24e3c68ae89c`

[77] IMO, Chapter II-1 - Construction - Structure, subdivision and stability,

machinery and electrical installations, Part B - Subdivision and stability (2020).