A Large-Scale Experimental Evaluation of High-Performing Multi- and Many-Objective Evolutionary Algorithms

Leonardo C. T. Bezerra leobezerra@imd.ufrn.br
IMD, Universidade Federal do Rio Grande do Norte, Natal, RN, Brazil
DCC, CI, Universidade Federal da Paraíba, João Pessoa, PB, Brazil

Manuel López-Ibáñez manuel.lopez-ibanez@manchester.ac.uk
Alliance Manchester Business School, University of Manchester, UK

Thomas Stützle stuetzle@ulb.ac.be
IRIDIA, CoDE, Université Libre de Bruxelles, Belgium

doi:10.1162/EVCO_a_00217

Abstract

Research on multi-objective evolutionary algorithms (MOEAs) has produced over the past decades a large number of algorithms and a rich literature on performance assessment tools to evaluate and compare them. Yet, newly proposed MOEAs are typically compared against very few, often a decade older MOEAs. One reason for this apparent contradiction is the lack of a common baseline for comparison, with each subsequent study often devising its own experimental scenario, slightly different from other studies. As a result, the state of the art in MOEAs is a disputed topic. This article reports a systematic, comprehensive evaluation of a large number of MOEAs that covers a wide range of experimental scenarios. A novelty of this study is the separation between the higher-level algorithmic components related to multi-objective optimization (MO), which characterize each particular MOEA, and the underlying parameters—such as evolutionary operators, population size, etc.—whose configuration may be tuned for each scenario. Instead of relying on a common or "default" parameter configuration that may be low-performing for particular MOEAs or scenarios and unintentionally biased, we tune the parameters of each MOEA for each scenario using automatic algorithm configuration methods. Our results confirm some of the assumed knowledge in the field, while at the same time they provide new insights on the relative performance of MOEAs for many-objective problems. For example, under certain conditions, indicator-based MOEAs are more competitive for such problems than previously assumed. We also analyze problem-specific features affecting performance, the agreement between performance metrics, and the improvement of tuned configurations over the default configurations used in the literature. Finally, the data produced is made publicly available to motivate further analysis and a baseline for future comparisons.

Keywords

Multi-objective optimization, evolutionary algorithms, performance assessment, automatic algorithm configuration.

1 Introduction

Multi-objective evolutionary algorithms (MOEAs) are one of the most widespread approaches for tackling multi-objective optimization problems (MOPs). Over the decades

Manuscript received: 13 January 2017; revised: 16 October 2017; accepted: 26 October 2017.

of MOEA research, a great number of algorithms have been proposed in the literature, as reviewed in various surveys (Coello Coello et al., 2007; Mezura-Montes et al., 2008; Chand and Wagner, 2015; Li et al., 2015). Nowadays, the study of algorithms for *many-objective* problems (MaOPs), those with four or more objectives, is one of the most active fields within MOEA research (Aguirre, 2013). This recent activity on MaOPs is motivated by algorithmic and practical needs. First, as MOEA research on two- and three-objective problems reached a mature stage, it became clear that MOEAs present severe performance limitations when facing MaOPs (Khare et al., 2003; Purshouse and Fleming, 2007; Bezerra et al., 2016). Second, many engineering applications are modeled as MaOPs, where constraints are considered objectives (Fleming et al., 2005), thus creating a need for improving existing MOEAs to tackle these problems. This need has led to many novel MOEAs specifically designed for MaOPs (Chand and Wagner, 2015; Li et al., 2015) and traditional MOEAs being redesigned to account for the particular characteristics of MaOPs (Deb and Jain, 2014).

Besides algorithmic advances, the MOEA community has been a driving force in the advancement of the performance assessment of multi-objective algorithms (Zitzler and Thiele, 1999; Zitzler et al., 2003; Deb et al., 2005; Knowles et al., 2006). Those works shaped the research on MOEAs, defining benchmark problems, performance metrics, and experimental setups to be reused in most of the subsequent MOEA research. However, in the case of continuous MOPs, which is the main application domain of MOEAs, experimental comparisons have mostly focused on demonstrating that a newly proposed MOEA is better, in some sense, than one or two widely-known MOEAs and/or a previous version of the new proposal. No previous work has done a comprehensive comparison and analysis of a large number of algorithms that allows identifying the state of the art for continuous MOPs. As a result, new proposals are repeatedly measured against the most widely-known MOEAs, even if previous proposals were shown to outperform them in given scenarios.

In this article, we carry out a detailed experimental study of MOEAs in order to better understand the state of the art in the field. Compared to recent performance assessments of MOEAs on many-objective problems (Hadka and Reed, 2012; Li et al., 2013; Ishibuchi, Akedo et al., 2015), our study has several novel characteristics. In particular, we study MOEAs considering five factors: (i) a large and representative set of MOEAs; (ii) the parameter settings of the MOEAs; (iii) a diverse and challenging benchmark set; (iv) the computational budget or stopping criteria allocated for solving each benchmark problem; and (v) the metrics used for performance assessment.

Considering the first experimental factor, we select at least two relevant MOEAs from each main MOEA paradigm: dominance-based, indicator-based, and decomposition-based approaches. In total, our analysis encompasses 9 MOEAs from the literature: MOGA (Fonseca and Fleming, 1993), NSGA-II (Deb et al., 2002), SPEA2 (Zitzler et al., 2002), IBEA (Zitzler and Künzli, 2004), SMS (Beume et al., 2007), HypE (Bader and Zitzler, 2011), MOEA/D (Li and Zhang, 2009), MO-CMA-ES (Igel et al., 2007), and NSGA-III (Deb and Jain, 2014). In addition, we make a distinction between the algorithmic components specifically related to multi-objective optimization (MO-components) and other non-MO-specific components, such as whether the underlying evolutionary operators (underlying EA) are based on genetic algorithms (GA), differential evolution (DE), or CMA-ES. This is a crucial distinction because of two reasons. First, some MOEAs proposed in the literature differ only in the underlying EA and are almost identical with respect to the MO-components. If we wish to conclude anything about the MO-components, we need to be able to isolate the influence of the underlying

EA on performance in order to not attribute an effect to the MO-components that is actually caused by a difference in the underlying EA. Second, there is no clear best choice of EA-components, and our recent work has shown that interactions between MO-components and underlying EA play a critical role in performance (Bezerra et al., 2015a,b). Thus, our study focuses on comparing MO-components that characterize particular MOEAs and, whenever possible, the underlying EA is studied as an additional component to be configured.

The second experimental factor concerns the parameter settings of the MOEAs, which strongly influence their behavior. Apart from the choice of the underlying EA discussed above, MOEAs contain other crucial parameters such as population size, mutation probability, and various MOEA-specific parameters. It has become standard to reuse the parameter configuration given in the papers that originally proposed each MOEA, regardless of differences in experimental scenarios. This approach is prone to misguided conclusions, as parameter values are often critical to performance while not being easily transferable between scenarios (Birattari, 2009; Hoos, 2012), for example, between continuous and combinatorial MOPs (Bezerra et al., 2016). Sensitivity analysis and preliminary experiments typically evaluate a very small number of possible parameter configurations without reporting the effort dedicated to each MOEA, or formally defining the specific criteria used to make the final choice. If a different configuration from the default is chosen, this is often a common “good enough” configuration for all MOEAs under study, thus implicitly assuming that there are no interactions between parameters and MO-components. Due to the ad-hoc nature of this method for parameter configuration, the potential of unintended biases influencing the final choice is large. To prevent such biases and to take into account potential interactions, we use an automatic parameter configuration tool with a formally defined setup and tuning budget to find a high-performing parameter configuration of each MOEA for each scenario considered here.

The third experimental factor in our comparison concerns benchmark problems. While the DTLZ benchmark (Deb et al., 2005) was a major contribution in early works, several problem features were not considered in its conceptual design. The WFG benchmark (Huband et al., 2006) was proposed a few years later providing a more diverse set of benchmark functions, but only a few MOEAs have been evaluated on it (Robič and Filipič, 2005; Tušar and Filipič, 2007; Bader and Zitzler, 2011; Bezerra et al., 2016). The difficulty of the individual functions is mainly controlled by the number of decision variables and the number of objectives. The use of a relatively small number of decision variables may produce *floor effects* (Cohen, 1995) that lead to identifying “state-of-the-art” MOEAs that are good at solving only easy problems. However, increasing the number of variables may lead to a *ceiling effect*, where a problem becomes so difficult that all MOEAs produce very poor approximations and their differences are irrelevant. In addition, previous experimental comparisons often aim at identifying an overall best-performing MOEA for multi- (two and three) and many-objective (four or more) problems. Yet, the number of objectives is a discrete factor that is known before solving a problem, and we should not expect that there exists a single MOEA, or configuration thereof, that performs optimally for any number of objectives. Instead, we find the best parameter configuration of each MOEA for a given number of objectives and analyze the impact of the number of objectives on MO-components.[1] Following this

[1]We would not necessarily advocate to configure each MOEA for each possible number of objectives, even if this could be done. An alternative may be, for example, to automatically configure MOEAs

perspective, our benchmark set comprises the DTLZ and WFG benchmarks with different numbers of variables (20 to 60) and we separately analyze scenarios with various number of objectives (2, 3, 5, and 10).

The fourth experimental factor that must be defined is the computational budget available to the MOEAs. In the literature, computational effort is often measured in terms of the number of function evaluations (FEs), under the assumption that the computational overhead of the MOEA itself is negligible in comparison to the evaluation cost of real-world problems. Yet, many MOEAs use computationally expensive algorithmic components that limit the number of FEs that can be executed in practice even for benchmark problems. At the same time, some experimental studies allow for very large number of FEs, hence implicitly assuming that evaluation cost is relatively low, which contradicts the previous assumption. Thus, it is not uncommon that different studies consider computational budgets that differ by an order of magnitude, while never evaluating algorithms under various budgets. In addition, the best-performing parameter settings for a given MOEA depend on the computational budget, thus a comparison between "default settings" hardly makes sense and the conclusions obtained under one computational budget cannot be generalized to another scenario with a budget that is an order of magnitude larger. In order to take into account such effects, we study scenarios with various budgets (2 500, 10 000, and 40 000 FEs), which simulate different levels of computational overhead posed by function evaluations, and we tune the settings of each MOEA for each scenario.

Finally, the fifth factor concerns the specific metrics used for performance assessment. When the algorithms under comparison produce outcomes that are incomparable in terms of Pareto-optimality, these metrics introduce an additional refinement that allows a ranking of the outcomes. This is particularly crucial in MaOPs, where the fraction of mutually nondominated solutions typically grows exponentially with the number of objectives. While the use of Pareto-compliant metrics is generally acknowledged nowadays to be essential, several such metrics exist, each of them introducing different preferences. A diverse set of metrics will measure conflicting preferences (Jiang et al., 2014), and we should not expect a single MOEA to be the best on all metrics (otherwise, we could simply aggregate the multiple metrics into a single one). Another potential pitfall is to generalize conclusions obtained with one metric to scenarios evaluated with respect to a different metric. To account for these effects, our study considers three diverse unary metrics (hypervolume relative deviation, I_H^{rd}, unary additive epsilon metric, $I_{\epsilon+}^1$, and inverse generational distance, *IGD*).

The experimental study of MOEAs presented here is the most comprehensive we are aware of. In addition to being a concrete first step towards identifying the state-of-the-art MOEAs for continuous optimization on two-, three-, five-, and ten-objective problems, we also analyze the effect of underlying EA choices, parameter tuning, computational budget, problem characteristics, and the correlation between performance metrics. Yet, the potential studies that could be conducted from the experimental data collected in this work far outnumber the analyses that we were able to fit within this single article. Moreover, one of our goals is that the results presented here help researchers to correctly identify improvements on the state of the art. For these reasons, we also make all our data publicly available for further analysis, verification, and extension.

for increasingly larger ranges of objectives, for example, for 2, 3, 4, {5, 6}, {7, 8, 9}, {10, 11, .., 15}, etc. objectives. We leave this point for future research.

The remainder of this article is organized as follows. In Section 2, we review MOEAs and highlight the particular ones we use in this study. Next, we detail our performance assessment setup in Sections 3 and 4. In Sections 5 to 7, we present our experimental results, starting by an overall comparison and moving on to further detailed analyses. We conclude and discuss future work in Section 8.

2 MOEAs, Paradigms, and Underlying EAs

The number of different MOEA proposals in the literature is too large to be exhaustively reviewed here (Coello Coello et al., 2007; Mezura-Montes et al., 2008; Chand and Wagner, 2015; Li et al., 2015). In this work, we summarize MOEAs first from a historical perspective and then we detail the most relevant MOEA paradigms, that is, MOEAs that share similar design characteristics. This taxonomy is instrumental for this work as we select the most relevant MOEAs from each paradigm for our performance assessment. Finally, we discuss the conceptual separation between the high-level MO components and the underlying EAs, showing the equivalence between some independently proposed algorithms.

2.1 MOEA Design Evolution

The historical evolution of MOEA design can be split into three main epochs. In the first years, MOEA researchers designed MO components that allowed EAs to find multiple Pareto-optimal solutions in a single run. Algorithms could be seen as a combination of an EA with two MO components: (i) a Pareto dominance-based metric to provide convergence pressure, and (ii) a metric to preserve the search diversity, that is, the spread over the decision and/or objective space. MOGA (Fonseca and Fleming, 1993), NSGA (Srinivas and Deb, 1994), and NPGA (Horn et al., 1994) are probably the most relevant MOEAs from this epoch.

The second epoch in MOEA design focused on the development of effective methods to explore the potential demonstrated by the first MOEAs. Important concepts proposed in this epoch include MO elitism (Zitzler and Thiele, 1999; Knowles and Corne, 2000; Deb et al., 2002; Zitzler et al., 2002), external archives (Knowles and Corne, 2000; López-Ibáñez et al., 2011), ϵ-dominance (Laumanns et al., 2002), the integration of quality indicators into the search (Zitzler and Künzli, 2004; Beume et al., 2007), the decomposition of the original MOP into several single-objective problems (Zhang and Li, 2007; Li and Zhang, 2009), and the use of underlying EAs other than genetic algorithms (Knowles and Corne, 2000; Robič and Filipič, 2005; Kukkonen and Lampinen, 2005; Beume et al., 2007; Tušar and Filipič, 2007; Igel et al., 2007). Also during this epoch, significant advancements were made for a proper performance assessment of multi-objective optimizers (Grunert da Fonseca et al., 2001; Zitzler et al., 2003; Deb et al., 2005; Knowles et al., 2006). The empirical analysis conducted by the authors of the DTLZ benchmark (Deb et al., 2005) using NSGA-II (Deb et al., 2002) and SPEA2 (Zitzler et al., 2002) demonstrated the good performance displayed by both algorithms on MOPs with two and three objectives. Since then, it has become standard in the MOEA community to compare novel algorithms to one (or both) of these MOEAs.

The third (and current) epoch in MOEA design is characterized by an increased interest on *many-objective optimization* problems (Purshouse and Fleming, 2007). MOPs with more than three objectives pose significant further challenges, as standard MOEAs such as NSGA-II and SPEA2 were shown to have scalability issues w.r.t. convergence, diversity, and/or running time (Khare et al., 2003). Generally speaking, standard MOEA components were unable to scale because of *dominance resistance* (Ikeda et al., 2001;

Purshouse and Fleming, 2007): in the presence of a large number of conflicting objectives, the proportion of the feasible space that is mutually nondominated is large and the entire population of an MOEA becomes mutually nondominated and spread too early, making further progress difficult. To overcome these limitations, two main approaches have been adopted. The first one revises the main components of existing MOEAs to propose novel algorithms. HypE (Bader and Zitzler, 2011) and NSGA-III (Deb and Jain, 2014) are two such examples. The other approach incorporates new high-level components, such as dimensionality reduction (Tenenbaum et al., 2000), space partitioning (Aguirre and Tanaka, 2009), or the use of surrogate models (Aguirre, 2013), to mention a few. In this work, we concentrate on the algorithms proposed using the first approach since they are, in principle, general enough that they can be applied to multi- and many-objective problems. Moreover, components from the second approach may be, in principle, combined with any existing MOEA; thus, a more detailed analysis would be required to isolate their contribution.

2.2 MOEA Paradigms

Traditionally, MOEAs have been categorized in the literature into one of the three following design paradigms. To make our experimental study comprehensive, we select at least two algorithms from each paradigm.

Dominance-based approaches. These MOEAs rely on metrics based on Pareto dominance to direct their search. The best known example is probably **NSGA-II** (Deb et al., 2002), which considers the dominance depth of solutions, that is, the dominance level the solution belongs to. Besides NSGA-II, we consider **MOGA** (Fonseca and Fleming, 1993), and **SPEA2** (Zitzler et al., 2002). MOGA bases its search on dominance rankings. SPEA2 uses a similar idea, but takes into account the number of solutions that each solution dominates. Each algorithm proposes a different approach to promote diversity. Due to dominance resistance, dominance-based algorithms are not expected to be as effective on MaOPs as they are on MOPs with two and three objectives. Moreover, the performance of MOGA is presumed to be poor in comparison to the other algorithms because it lacks elitism, that is, MOGA replaces the whole population with a new one at each generation. We include MOGA in this assessment due to its historical relevance and as a baseline to put other results in perspective.

Indicator-based approaches. After the proposal of Pareto-compliant quality indicators, some authors envisioned using these indicators within MOEAs to direct their search. Indicator-based approaches are expected to outperform dominance-based ones on MaOPs because the convergence pressure provided by quality indicators is not directly influenced by the number of objectives. In addition, some indicators account for diversity, keeping the population well-spread along the objective space. The drawback of indicator-based approaches is the computational complexity of some quality indicators such as the hypervolume, which is exponential in the number of objectives in the worst case (Beume et al., 2009). The most relevant MOEAs from this paradigm are **IBEA** (Zitzler and Künzli, 2004), **SMS** (Beume et al., 2007), and **HypE** (Bader and Zitzler, 2011). IBEA uses binary quality indicators to compute a fitness for each solution. Since pairwise comparisons between solutions using binary indicators are cheap to compute, IBEA has very little computational overhead when compared to other indicator-based MOEAs. SMS and HypE are based on the exclusive hypervolume contribution and the shared hypervolume contribution, respectively. HypE was the first indicator-based algorithm specifically designed for MaOPs; its shared hypervolume contribution requires estimation for problems with more than three objectives, resulting in a less accurate metric, whereas modern implementations of the exclusive hypervolume contribution impose little overhead for SMS (While and Bradstreet, 2012; Nowak et al., 2014).

Decomposition-based approaches. An alternative research direction already during the first epoch of MOEA design was to decompose the original MOP into single-objective subproblems (Hajela and Lin, 1992; Jaszkiewicz, 2002), typically by means of scalarizations. This paradigm, however, remained little explored for many years. More recently, the interest from the MOEA community on this paradigm was stirred by the proposal of the **MOEA/D** algorithms (Zhang and Li, 2007; Li and Zhang, 2009). Here we consider the variant that won the IEEE CEC 2009 competition on unconstrained multi-objective continuous optimization (Li and Zhang, 2009), which uses a *dynamic resource allocation* (DRA) approach. This approach is an online adaptive strategy proposed to reduce the computational budget wasted on non-promising search directions. Besides MOEA/D, we also include **NSGA-III** (Deb and Jain, 2014) as a representative of this paradigm. Although the convergence pressure of its search is provided in an identical fashion to NSGA-II, the diversity mechanism proposed for NSGA-III uses weight-based reference lines to keep the population spread over the objective space. More importantly, this algorithm was designed specifically for MaOPs.

2.3 Underlying EAs

One important feature that is typically overlooked when designing and comparing MOEAs concerns their underlying evolutionary algorithm. As previously discussed, MOEAs can be seen as a combination of high-level MO components, responsible for dealing with the particular aspects of multi-objective optimization such as dominance, convergence, and diversity, coupled with an underlying EA responsible for selection and variation operators. In general, MOEA authors have focused their efforts on devising effective MO components, and reused the same underlying EA choices traditionally adopted in the literature, that is, GAs (or evolution strategies, ESs) typically using SBX crossover and polynomial mutation. As a result, the literature exploring different underlying EAs is limited, and the effect of variation operators of the underlying EAs on performance is still rather poorly understood for both multi- and many-objective optimization. Next we discuss some works that have tried to extend MOEA research in this direction.

2.3.1 Differential Evolution (DE)

During the second epoch of MOEA design, some of the new MOEAs used DE as the underlying EA, often reusing existing approaches for the MO components (Madavan, 2002; Abbass, 2002; Robič and Filipič, 2005; Kukkonen and Lampinen, 2005; Tušar and Filipič, 2007; Tagawa et al., 2011). For instance, many of the first DE-based MOEAs were fairly similar to NSGA-II concerning mechanisms to deal with convergence and diversity, the two most relevant examples being DEMO (Robič and Filipič, 2005) and GDE3 (Kukkonen and Lampinen, 2005). The positive results obtained by these DE-based MOEAs motivated researchers to combine DE with other existing MO components, such as in DEMO^{SP2} and DEMO^{IB} (Tušar and Filipič, 2007), where MO components are reused from SPEA2 and IBEA, respectively. Table 1 shows a set of algorithms that are essentially equal with respect to their MO components, but differ in the underlying EA. Not all of these algorithm pairs had been previously recognized as equivalent,

Table 1: MOEAs that differ only in the underlying EA used.

	Underlying EA	
MO components	GA/ES	DE
dominance depth, crowding distance	NSGA-II	DEMO, GDE3
dominance strength, k-nn density estimator	SPEA2	DEMO^{SP2}
binary indicator	IBEA	DEMO^{IB}
dominance depth, hypervolume contribution	SMS	IBDE

for example, SMS and IBDE (Tagawa et al., 2011), most likely because they were originally described using different terminology.

In this work, we treat MOEAs that differ only in their underlying EA as the same MOEA. More precisely, the choice of underlying EA is treated as a parameter that is set, together with other algorithmic parameters, by means of automatic configuration. By doing so, we ensure that each MOEA uses its best configuration of the underlying EA. For example, henceforth, SPEA2 may denote the use of either GA or DE as underlying EA, depending on the particular experimental scenario considered. A choice of GA implies the use of the SBX crossover and polynomial mutation operators, whereas a choice of DE uses binomial crossover and differential mutation operator. As far as we know, several algorithms, including MOGA, HypE, and NSGA-III, have never been studied with DE as the underlying EA, but such extensions are straightforward and we include them here.

We could argue that our study includes at least 14 different MOEAs (including the special case of MO-CMA-ES, which is discussed next), because the automatic configuration procedure may choose to replace the GA operators of the eight MOEAs mentioned in Section 2.2 by DE operators, thereby instantiating at least the five DE-based MOEAs shown in Table 1. Since the GA-based MOEAs considered here were proposed before their DE-based counterparts, we use in the following the original name given to the GA-based MOEAs independently of the underlying EA.

2.3.2 Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

Given its excellent performance on single-objective problems, researchers have proposed adaptations of the CMA-ES algorithm to solve MOPs. The basic MO-CMA-ES algorithm (Igel et al., 2007) consists of a number of concurrent executions of CMA-ES starting from different initial solutions. Its remaining MO components comprise dominance depth and the exclusive hypervolume contribution, resembling SMS. Other variants of this algorithm were proposed later, allowing different quality indicators, selection for survival, and learning strategies for the underlying CMA-ES algorithm (Voß et al., 2010).

Ideally, we would like to study CMA-ES as yet another possible underlying EA for all MOEAs. However, devising such CMA-ES-based MOEAs is not as straightforward as in the case of DE, since we would need to somehow couple the learning mechanisms from CMA-ES with other MO components, which is beyond the scope of this article. Instead, **MO-CMA-ES** is the single representative MOEA using CMA-ES as the underlying EA, and we study its different variants as parameters to be configured, as explained in Section 3.1.

3 Experimental Factors

Besides the MOEAs described above, our experimental study considers four additional experimental factors, detailed below.

3.1 Parameter Configuration Space

Apart from the choice of the underlying EA, each MOEA has several parameters that need to be set, that is, we need to specify a parameter configuration. Normally, two approaches are adopted in the literature regarding parameter settings in experimental comparisons. Some studies use a given configuration, either the "default" one from the literature or a configuration found after limited, and often unspecified, preliminary experiments. Other studies investigate the effect of various parameter settings, and their interactions, but due to the large number of possibilities, only a very small number of parameter configurations can be studied and conclusions are often not conducive to identifying the highest-performing settings. Ideally, we would like to compare algorithms using their highest-performing parameter configuration for a given problem. Although we may know a priori certain features of the application scenario, such as the number of objectives and the stopping criterion, we cannot assume that the actual problem being tackled, and thus the optimal parameter configuration for each problem, is known.

Automatic algorithm configuration methods find high-performing parameter configurations for a given application scenario given a set of training problems, which are assumed to be representative of the actual, not yet known, problems we aim at tackling in practice. The use of such automatic configuration methods allows us to consider a very large parameter space and still find, for a given application scenario, a parameter configuration that is general enough to be expected to be high-performing over a diverse set of benchmark problems.

In this work, the automatic configuration step considers the following parameter domains:[2]

Population size (μ). For the dominance- and indicator-based algorithms, we consider $\mu \in \{10, 20, \ldots, 100\}$. However, we use $\mu \in \{100, 200, \ldots, 500\}$ for MOEA/D, as its population size determines the number of weights and it therefore cannot be too small; in fact, a small number of weights on a problem with a large number of objectives may miss large regions of the objective space. Finally, for NSGA-III, the population size and the number of weights used are tightly related parameters, so we detail them later.

[2]Following the original proposals of the MOEAs studied here, we do not use any external archive. Nevertheless, we argue in the conclusions that one should apply the methodology proposed here to compare MOEAs with archives and we expect conclusions will be different. Since we do not use an external archive, the population size determines the size of the approximation front. However, each MOEA performs differently depending on the population size and prescribing a fixed population size a priori may favor specific MOEAs while hindering others. Therefore, we let the automatic configuration procedure choose the population size of each MOEA for each scenario that minimizes a performance metric. This may result in very small approximation fronts that are of higher quality (according to the metrics) than much larger ones. If this is a concern in practice, then the metrics used in the literature, which we follow here, are inadequate, but it is trivial to apply our methodology and repeat our analysis with different metrics. Moreover, if an approximation front of a specific size is desired, then it is arguably better to use an external bounded archive (Schütze et al., 2008, 2010; López-Ibáñez et al., 2011), and still tune the population size of each MOEA.

Table 2: Parameter space of variation operators for all MOEAs but MO-CMA-ES.

	GA			DE	
Parameter	t_{size}	p_c, p_m, p_v	η_c, η_m	CR	F
Domain	{2,4,8}	[0, 1]	$\{1, \ldots, 50\}$	[0.01, 1]	[0.1, 2]

Table 3: Parameter space for tuning specific MOEA/D parameters.

Parameter	ρ	δ	ϕ	t_{size}	ν
Domain	[0.1, 1]	[0, 1]	[0.01, 1]	$\{1, 2, \ldots, 20\}$	$\{2, 3, \ldots, 10\}$

Underlying EA. As previously explained, we give algorithms a choice between two different underlying EAs: (i) GA, using the traditional deterministic tournament, SBX crossover (Deb and Agrawal, 1995) and polynomial mutation (Deb and Agrawal, 1999) operators; or (ii) DE, using binomial crossover and differential mutation operators in the DE/rand/1/bin fashion (Robič and Filipič, 2005). Additional parameters associated with each variation operator are given in Table 2, namely tournament size (t_{size}), the crossover and mutation probabilities and distribution indices (p_c, p_m, η_c and η_m, respectively) for the GA variation operators,[3] and crossover rate and scaling factor (CR and F, respectively) for the DE operators. As previously explained, these parameters do not apply to MO-CMA-ES.

A few other MOEAs present parameters additional to the ones we discussed above, which we briefly list below. We refer to the original papers for a better understanding of all parameters.

MOGA. The fitness sharing diversity metric used by MOGA requires the definition of a niche radius, which we configure as $\sigma_{\text{share}} \in [0.01, 1]$.

SPEA2. When used for mating selection, the k-th nearest neighbor density estimation strategy adopted by SPEA2 computes k using a predefined formula (we refer to this option as $k_{\text{method}} = \mathit{auto}$). Alternatively, k may be set directly, with $k \in \{1, 2, \ldots, 9\}$.

IBEA. Although any binary indicator could be used by IBEA, we consider the ones originally proposed, that is, the binary ϵ-indicator (I_ϵ) and the binary hypervolume indicator (I_H^-).

MOEA/D. The DRA strategy (Li and Zhang, 2009) used by MOEA/D requires a number of additional parameters, whose domains are given in Table 3. Furthermore, the decomposition part of the algorithm can be selected from three options: weighted sum aggregation (WS), Tchebycheff aggregation (TA), or penalty-based boundary intersection (PBI). When PBI decomposition is used, an additional penalty parameter $\xi \in \{1, 2, \ldots, 10\}$ needs to be set.

[3]Following Bezerra et al. (2016), we use two probability parameters for the mutation: p_m decides if an individual will undergo mutation, whereas p_v decides whether to mutate a particular decision variable from the selected individual. By default, p_v is set to $1/n_{\text{var}}$.

Table 4: Parameter space for MO-CMA-ES learning parameters.

Parameter	σ_0	d	p_{target}	p_{thresh}	c_p	c_c	c'_{cov}	u'_{cov}
Domain	$[0, 1]$	$\{1, 2, \ldots, 50\}$	$[0, 0.5[$	$[0, 0.5[$	$[0, 1]$	$]0, 1]$	$[0, 1[$	$[0, 1[$

MO-CMA-ES. In the original proposal, the values of learning-related parameters are set as a function of the number of decision variables. In this work, for each of the parameters listed, the parameter may either use the default formula or a specific value within the ranges given in Table 4.[4] Two parameters constitute an exception to this pattern: (i) the initial step size σ_0, for which no default formula is available, and (ii) the covariance matrix unlearning rate u'_{cov}, which is not a parameter of the original MO-CMA-ES, yet it is part of the most widely adopted MO-CMA-ES implementation in Shark (Igel et al., 2008). For the latter, we allow three alternates: (i) the original approach, i.e., $u'_{cov} = c'_{cov}$; (ii) the formula used in Shark (Igel et al., 2008); or (iii) a specific value according to the range given in Table 4. Concerning the multi-objective components of MO-CMA-ES, we consider three parameters to be set: (i) the quality indicator, which can be *epsilon* or *hypervolume*; (ii) using steady-state replacement or not; and (iii) the notion of success, which can be *population* or *individual*-based (Voß et al., 2010).

NSGA-III. It adopts the traditional approach of uniform weight generation, but also proposes a two-layer approach for MaOPs. In the two-layer approach, the number of weights generated for the outer and the inner layer are a function of two parameters H_1 and H_2, respectively. In the uniform approach, the number of weights is a function of the number of objectives M and a numerical parameter, p in the original paper (Deb and Jain, 2014), that we also call H_1 for simplicity. In our experiments, we use uniform generation for $M \in \{2, 3, 5\}$ and we consider the domains:

$$H_1 \in \begin{cases} \{2, \ldots, 100\} & \text{if } M = 2, \\ \{2, \ldots, 30\} & \text{if } M = 3, \\ \{2, \ldots, 10\} & \text{if } M = 5, \end{cases} \tag{1}$$

and H_2 does not need to be defined. When $M = 10$, we use the two-layer approach and the domain of both H_1 and H_2 becomes $\{1, \ldots, 3\}$. The population size is then calculated as $\mu = \mu' \cdot W$, where W is the number of weights generated and μ' is a numerical parameter to be configured, $\mu' \in [0.5, 1.5]$.

3.2 Benchmark Problems

Although multi-objective continuous optimization is ultimately a practical field rich in real-world problem examples, MOEA research has traditionally used artificial benchmark problems. The main advantages of artificial benchmarks are that (i) one has better control of the problem characteristics being analyzed, and (ii) these problems are designed to be scalable as to the number of variables and objectives. However, we observed that scaling the number of variables of DTLZ1 and DTLZ3 leads to ceiling effects, and therefore we do not include them in our analysis. In this work, we consider 14 box-constrained functions, namely the remaining 5 functions that comprise the DTLZ benchmark (Deb et al., 2005) and the 9 functions that comprise the WFG benchmark (Huband

[4]We use surrogate parameters to tune the covariance matrix learning (c_{cov}) and unlearning (u_{cov}) rates and ensure they are always smaller than c_c. Concretely, $c_{cov} = c'_{cov} \cdot c_c$ and $u_{cov} = u'_{cov} \cdot c_c$.

Table 5: Summarized description of the benchmark problems used in this work. DTLZ2–6 are considered partially separable (indicated by *), since optimizing single variables leads to some, but not all globally optimal solutions. U: unimodal; M: multimodal; D: deceptive. DTLZ5-6 are degenerate for $M = 3$, but it is not possible to determine their geometry for $M > 3$. Further details are provided by Huband et al. (2006).

MOP	Separability	Local fronts	Modality	Bias	Geometry
DTLZ2	$\surd^*$	–	U	–	concave
DTLZ4	$\surd^*$	–	U	polynomial	concave
DTLZ5	$\surd^*$	–	U	–	degenerate ($M = 3$)
DTLZ6	$\surd^*$	$\surd$	U	–	degenerate ($M = 3$)
DTLZ7	$\surd$	–	U,M	–	disconnected
WFG1	$\surd$	–	U	polynomial	convex, mixed
WFG2	–	–	U,M	–	convex, disconnected
WFG3	–	–	U	–	linear, degenerate
WFG4	$\surd$	–	M	–	concave
WFG5	$\surd$	–	D	–	concave
WFG6	–	–	U	–	concave
WFG7	$\surd$	–	U	parameter dependent	concave
WFG8	–	–	U	parameter dependent	concave
WFG9	–	–	M,D	parameter dependent	concave

et al., 2006). We do not include the CEC 2009 competition benchmark (Zhang and Suganthan, 2009) as one cannot scale the number of objectives for each problem. Table 5 summarizes the characteristics of these benchmark problems.

Number of objectives (M). We consider four values for the number of objectives, namely $M \in \{2, 3, 5, 10\}$. This allows us to conduct in-depth analysis for each of the values considered as well as to observe trends that arise as the number of objectives increases. We choose to represent MaOPs using five and ten objectives to analyze the effect of this large increase in the number of conflicting objectives. Our intuition is that MOEAs will have considerable difficulty dealing with the large effect of dominance resistance present in MaOPs with ten objectives. By contrast, previous experiments (Bezerra et al., 2015b, 2016) indicate that some MOEAs not specifically designed for MaOPs are still able to cope with the dominance resistance of five-objective problems if properly tuned.

Problem sizes (n_{var}). An experimental factor often ignored in the MOEA literature is problem size, that is, the number of variables present in the benchmark problems. For instance, studies with the DTLZ benchmark consider a number of decision variables, e.g., $n_{\text{var}} = 12$ for the three-objective DTLZ2 (Deb et al., 2005), that is relatively small compared to the usual single-objective benchmarks. In this work, we assess MOEAs in terms of their scalability with respect to problem size by studying the range $n_{\text{var}} \in \{20, 21, \ldots, 60\}$.[5]

[5]The WFG benchmark presents a configurable ratio between the number of distance and position variables, as well as constraints on problem sizes. For further details on how we handle these constraints and ratio, we refer to the supplementary material (Bezerra et al., 2017b).

3.3 Computational Cost and Stopping Criteria

Traditionally, MOEAs have been designed and tested using as stopping criterion a maximum number of function evaluations (FEs), rather than actual computation time. The rationale is that FEs in real-world problems are computationally and/or financially expensive, and much more expensive than the time required by the MOEA itself. We have observed three pitfalls when adopting this approach. First, many MOEAs contain algorithmic components that are computationally expensive, and it is assumed that the overhead posed by such components would be compensated by the overhead presented by the FEs. In the literature, however, many studies use a very large number of FEs, contradicting the assumption that FEs are expensive (Deb et al., 2005; Huband et al., 2006; Zhang and Suganthan, 2009). Second, other MOEAs present a very aggressive intensification behavior at early evolution stages under the assumption that not many FEs would be available. However, the optimization literature is rich on examples of drawbacks caused by such behavior (Hoos and Stützle, 2004), in particular, the propensity of getting stuck in locally optimal solutions (in this case, local Pareto fronts). Finally, MOEAs perform differently depending on the stopping criterion (Bezerra et al., 2016).

In this work, we simulate different levels of computational cost by means of different maximum numbers of FEs given to each run ($FE_{\max}$). Concretely, we consider $FE_{\max} \in \{2500, 10000, 40000\}$, representing high, medium, and low computational cost per FE, respectively. In addition, we also account for computationally expensive algorithmic components by setting a maximum cut-off CPU-time ($t_{\max}$); where $t_{\max}$ is set to 1 hour for $FE_{\max} = 2500$, and to 10 minutes for $FE_{\max} \in \{10000, 40000\}$. Although the same benchmark functions are used in all experiments, scenarios where MOEAs are given a low number of function evaluations ($FE_{\max} = 2\,500$ and $t_{\max} = 1$ hour) simulate real-world problems that are expensive to evaluate, hence, not many evaluations can be used and the algorithm can spend significant computational time. By contrast, scenarios where $FE_{\max}$ is high[6] ($FE_{\max} = 40\,000$ and $t_{\max} = 10$ minutes) simulate real-world problems for which function evaluations are relatively cheap to compute and the MOEA should also execute relatively fast.

3.4 Performance Metrics

Once a single run of a MOEA reaches the stopping criterion, the quality of the obtained approximation of the Pareto-optimal front is evaluated according to a set of performance metrics, where each metric tends to favor different characteristics of the approximation front. The general desirable characteristics of an approximation front are the following (Jiang et al., 2014).

Convergence refers to the closeness to optimality. A front is said to have converged if all of its solutions are Pareto-optimal.

Spread refers to the extent of the front, more specifically to the distance between the extreme solutions of a front.

[6]High in comparison to the other scenarios we consider in our setup. In practice, $FE_{\max} = 40\,000$ may still be considered a low value depending on the application, but we adopt it here because $FE_{\max} = 10\,000$ is a standard value used in the literature (Bader and Zitzler, 2011), and $FE_{\max} \in \{2\,500, 40\,000\}$ preserve a ratio between $FE_{\max}$ values across different scenarios (a factor of 4). Other works have used a factor of 10 (Deb et al., 2005; Huband et al., 2006), for example, $FE_{\max} \in \{2\,500, 25\,000, 250\,000\}$, but our experiments indicate that a value such as $FE_{\max} = 250\,000$ leads to floor effects for many of the benchmark functions.

Distribution refers to the evenness of the front, more specifically to the uniformity of the distances between pairs of adjacent solutions. Gaps in the approximation that do not exist in the Pareto-optimal front are, in principle, undesirable.

The performance metrics adopted here and the characteristics they measure are described next (all of them should be minimized).

The hypervolume relative deviation (I_H^{rd}). The hypervolume metric $I_H(A)$ computes the volume of the objective space dominated by an approximation front A and bounded by above (in the case of minimization) by a reference point $\mathbf{r}$. Instead of directly evaluating $I_H(A)$, we compute the relative deviation with respect to a reference front R:

$$I_H^{\text{rd}}(A) = \frac{I_H(R) - I_H(A)}{I_H(R)}, \tag{2}$$

where R is a very good approximation of the Pareto-optimal front for a given problem. The range of the hypervolume metric, and I_H^{rd} by extension, becomes considerably small with increasing number of objectives, making it difficult to assess the actual closeness to the Pareto-optimal front. Moreover, negative I_H^{rd} values are possible in practice, because R is often a subset of the true Pareto-optimal front, and it may present gaps or not be spread enough.

The unary additive epsilon metric ($I_{\epsilon+}^1$). This metric measures the minimum ϵ-translation in all objectives required such that a given approximation front A dominates the reference front R. A result of $I_{\epsilon+}^1 \leq \upsilon$ means that the MOEA has successfully approximated the actual front with υ tolerance. However, a result such as $I_{\epsilon+}^1 > \upsilon$ may represent a lack of convergence or spread. Moreover, an approximation front may have the same $I_{\epsilon+}^1$ value as another because of one point being far away from the Pareto-optimal front, despite the former completely dominating the latter for all the other points. Thus, this metric can be seen as a worst-case measure.

Inverted generational distance (*IGD*) measures convergence, distribution, and spread like I_H^{rd}, but it is distance-based like $I_{\epsilon+}^1$. Concretely, *IGD* computes the average Euclidean distance between the points in the reference front and the approximation front considered. Despite the well-known issues with the classical *IGD*, such as lack of Pareto-compliance and its dependence on the quality and size of the reference fronts (Ishibuchi, Masuda et al., 2015; Rudolph et al., 2016; Bezerra et al., 2017a), it is still commonly used to assess the state of the art (e.g., Deb and Jain (2014)). Therefore, we use it here for the sake of comparison with those studies.

The use of multiple metrics is motivated by the fact that, when evaluating fronts that are incomparable in terms of Pareto-optimality, each metric favors different features of the approximation fronts. The number of objectives (M) and shape of the fronts also influence the interpretation of the various metrics. For convex fronts, as M grows, $I_{\epsilon+}^1$ favors spread over distribution, while the other two metrics do the opposite (Jiang et al., 2014). Thus, convex fronts with smaller (better) values for $I_{\epsilon+}^1$ will show increasingly larger (worse) values for both I_H^{rd} and *IGD*, as M grows. By contrast, on concave fronts, *IGD* prefers distribution over spread and decreasing *IGD* values result in increasing values for the other two metrics for any number of objectives.

We only consider unary metrics because the main benefit of binary metrics is being able to identify more precisely cases where one approximation front is better than another in terms of Pareto-optimality (Zitzler et al., 2003). However, this advantage is not very useful in our context, where most approximation fronts are Pareto-incomparable, and the binary metrics produce no better conclusions than their unary counterparts.

4 Experimental Setup

Given the experimental factors discussed above, our experimental setup proceeds as follows. First, we define one application scenario for each combination of number of objectives and stopping criterion, that is, 12 scenarios in total. For each scenario, we use irace (López-Ibáñez et al., 2016), an automatic algorithm configuration method, to find a high-performing parameter configuration for each MOEA, within the parameter space previously defined. Finally, the best parameter configuration of each MOEA for each scenario is evaluated by executing them on a set of benchmark problems that is slightly different from the one used for configuring parameters, and computing the various performance metrics discussed above.

In the context of automatic algorithm configuration, the use of different benchmark problems for the configuration (tuning or training) phase and the evaluation (testing) phase prevents the overfitting of parameter settings to particular problems. However, given that the benchmark functions used in continuous optimization are considerably heterogeneous, any disjoint partition in terms of functions would result in a training set for tuning that is not representative of the evaluation set. Instead we partition the benchmark set with respect to the number of decision variables (Liao et al., 2014; Bezerra et al., 2016); that is, we use $n_{\text{var}} \in \{20, \ldots, 60\} \setminus n_{\text{testing}}$ for the tuning phase, where $n_{\text{testing}} \in \{30, 40, 50\}$ are the sizes we reserve for testing. We follow this approach as an alternative to the leave-one-problem-out cross-validation since that would incur a computational cost 14 times higher than the current cost, which would make this study close to infeasible.

All our experiments were run on a single core of Intel Xeon E5410 CPUs, running at 2.33 GHz with 6 MB cache size under Cluster Rocks Linux version 6.2/CentOS 6.2. In addition, MOEAs were implemented in `C++` and compiled using GCC version 4.7.2 and the `-O3` optimization flag. We verified our implementations by comparing our results against those obtained by the publically available reference implementations. When these were not available, we verified our implementation against results from the original papers proposing them.

The performance evaluation of a given algorithm (or candidate configuration) and the setup of the tuning and testing phases are described in the following.

4.1 Performance Evaluation

We compute reference fronts for each benchmark problem by randomly generating 1 000 Pareto-optimal points using the methodology described in the original DTLZ (Deb et al., 2005) and WFG (Huband et al., 2006) papers. We noticed, however, that these randomly generated fronts were not evenly distributed and large gaps could be identified for increasing number of objectives. Thus, we followed a series of steps to further improve the generated reference fronts. First, we used the randomly generated reference sets to automatically tune a subset of MOEAs for scenarios with $FE_{\text{max}} = 10\,000$, obtaining high-quality parameter settings. We then ran these MOEAs with $FE_{\text{max}} = 150\,000$ on each benchmark problem using the parameter settings obtained in the previous step. Finally, for each benchmark problem, we merged all the approximation fronts obtained in the previous step with the randomly generated reference sets, and filtered out the dominated solutions.

Before computing any performance metric, we took precautions to avoid skewed results in the presence of strong outliers. Concretely, we have filtered out solutions that were dominated by an upper bound point $\mathbf{u}$, determined based on the extreme solutions found in the improved reference sets:

$$\mathbf{u} = \begin{cases} [10]^M, & \text{if } M \in \{2, 3\} \\ [15]^5, & \text{if } M = 5 \\ [25]^{10}, & \text{if } M = 10 \end{cases} . \tag{3}$$

This upper bound also determines the reference point $\mathbf{r}$ used by the I_H^{rd} metric, with $\mathbf{r} = \alpha \cdot \mathbf{u}$, where α is a factor that ensures extreme solutions influence the hypervolume; here we use $\alpha = 1.1$.

4.2 Parameter Tuning Setup

We use an offline automatic algorithm configuration tool, irace (López-Ibáñez et al., 2016), to determine high-performing parameter settings of each MOEA for each scenario defined as a combination of number of objectives and maximum number of FEs. Our reasoning is that the number of objectives and the maximum number of FEs are usually known in advance before executing the algorithm, and that they may require very different parameter settings.[7] For instance, MOEA designers have traditionally tried to devise effective algorithms regardless of the number of objectives presented by the problem. Based on the current insights available in the literature, however, we believe that such a goal is counter-productive, leading to generalist parameter settings that perform worse than parameter settings tuned for each number of objectives. The same reasoning goes for the stopping criterion, as previously discussed.

For fairness, we use a common tuning setup for all algorithms and scenarios. Each run of irace is given a maximum budget of 20 000 experiments, that is, executions of the MOEA being tuned. Each MOEA execution is stopped according to the stopping criteria of the given scenario. All runs of irace use, as training benchmark problems, the union of the DTLZ, except for DTLZ1 and DTLZ3, and WFG benchmarks with a fixed number of objectives given by the scenario and with problem sizes $n_{\text{var}} \in \{20, \ldots, 60\} \setminus n_{\text{testing}}$ for tuning, where $n_{\text{testing}} \in \{30, 40, 50\}$ are the sizes we reserve for testing.

The irace method was originally designed for tuning single-objective optimizers, but it may use unary performance metrics to evaluate the quality of the approximation fronts returned by each MOEA execution (López-Ibáñez and Stützle, 2012). In particular, for scenarios with $M \in \{2, 3, 5\}$, we use the hypervolume relative deviation (I_H^{rd}) metric, whereas for scenarios with $M = 10$, we use the unary additive epsilon ($I_{\epsilon+}^1$) metric because I_H^{rd} becomes exceedingly time-intensive given the large number of experiments executed during tuning. Metrics are computed as described in Section 3.4.

4.3 Evaluation (Testing) Setup

Once all MOEAs have been properly tuned for each scenario, we end up with one high-performing configuration of each MOEA for each scenario. Next, we run each of them 25 times on the test benchmarks. For each scenario, the test benchmarks are also the union of the DTLZ, except for DTLZ1 and DTLZ3, and WFG benchmark sets for a given number of objectives M with problem sizes $n_{\text{testing}} \in \{30, 40, 50\}$. For each scenario, the

[7]Even if the precise setting for a scenario is only available a short time before actually executing the algorithm, one can easily imagine that MOEAs may have been previously tuned for many different scenarios and tuned settings are available from a database.

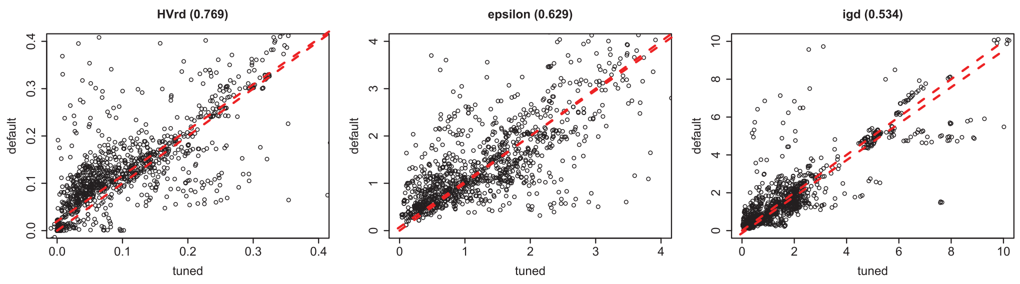

Figure 1: Scatter plots comparing the performance of tuned (x-axis) and default (y-axis) parameter configurations of MOEAs according to the I_H^{rd} (left), $I_{\epsilon+}^1$ (center), and *IGD* (right) metrics. Each point is the mean result of 25 runs of the same MOEA on a particular benchmark problem with the same stopping criterion.

stopping criterion for all runs is the same used for the tuning (see Section 3.3). The approximation sets returned by each run are evaluated according to performance metrics discussed earlier (Section 3.4): I_H^{rd}, $I_{\epsilon+}^1$ and *IGD*.

The following sections analyze the results of these experiments and compare the various MOEAs according to each scenario and performance metric.

5 Preliminary Analysis

This section conducts a preliminary analysis to assess our experimental setup. In particular, we verify the effectiveness of parameter tuning, the effect of the underlying EA, the importance of the stopping criterion, and the correlations between performance metrics.

5.1 Effectiveness of Parameter Tuning

As previously discussed, we used irace as an offline automatic algorithm configurator to find high-performing parameter configurations of the MOEAs for each experimental scenario. All tuned configurations for the various MOEAs and scenarios are available from the supplementary material (Bezerra et al., 2017b). Since both irace and the MOEAs are stochastic optimizers and the benchmark functions are very heterogeneous, we cannot expect that the configurations found are better than the default configurations from the literature in every run; yet we expect that they are better overall.

We compare the results obtained by MOEAs using tuned parameter settings versus default ones in the scatter plots shown in Figure 1. In particular, the points depicted in the plots are pairs (t, d), where t and d are the mean results, according to the corresponding metric, over 25 runs of the tuned and the default version of a MOEA, respectively. Each point corresponds to one MOEA on a specific MOP from the testing benchmark set and a specific stopping criterion. In total, the number of samples of each plot comprises 4032 pairs (8 MOEAs, 14 MOPs, 3 values of n_{testing}, 4 values of M, and 3 stopping criteria). The only algorithm left out of this analysis was MOGA, since the original work does not provide default parameter settings. Points above the diagonal denote runs where the tuned settings perform better than default ones. In parenthesis at the top of each plot, we provide the ratio of pairs for which $t < d$; for example, the value 0.769 means that the tuned MOEAs are better in terms of the I_H^{rd} metric than the default settings in 76.9% of the runs. In terms of the $I_{\epsilon+}^1$ and *IGD*, the values are 62.9% and 53.5%, respectively. A consequence of using I_H^{rd} as the metric guiding irace in most scenarios is that

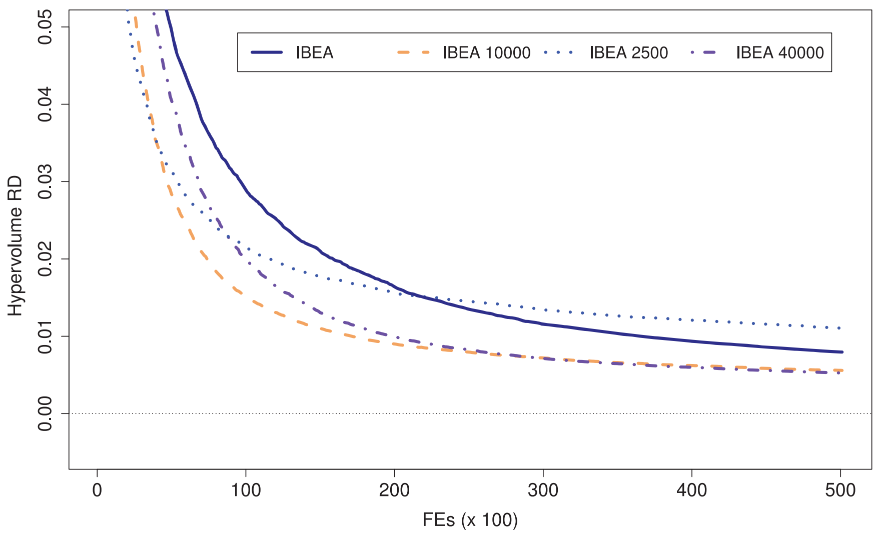

Figure 2: I_H^{rd} performance of IBEA using DE as underlying EA with different numerical parameter settings on WFG8 ($M = 3$, $n_{\text{var}} = 30$).

the benefits of the tuning are most significant for I_H^{rd}. Moreover, since optimizing I_H^{rd} does not necessarily optimize *IGD*, the tuned parameter settings may not be better than the default ones with respect to *IGD*. This analysis shows that it is possible to find much better parameter configurations than the defaults suggested in the original proposals or in benchmark assessments (Deb et al., 2005; Huband et al., 2006); yet, the setup utilized to find these configurations must be thoroughly described in order to understand the results produced.

Even more noteworthy is the effect of parameter tuning for a given stopping criterion. The example in Figure 2 shows the development of I_H^{rd} over the number of function evaluations (averaged across 25 runs) of IBEA with DE as underlying EA on problem WFG8 ($M = 3$, $n_{\text{var}} = 30$), using four different parameter configurations. The first configuration, labelled *IBEA*, uses the parameter settings proposed in the original DEMO$^{\text{IB}}$ paper (Tušar and Filipič, 2007). The remaining three curves were obtained by parameter configurations selected by irace for different $FE_{\max}$ values, and are labelled *IBEA* $FE_{\max}$. To generate the plot, all configurations were run up to $FE_{\max} = 50\,000$. The plot shows that *IBEA 2500* performs well only for small values of FEs and performs poorly for larger values of FEs. Interestingly, the default configuration is only better than *IBEA 2500* when the number of FEs is large, and always worse than the rest. A similar pattern is observed in other curves: the best performance of a configuration with regards to other configurations is typically obtained around the number of FEs for which the algorithm was tuned.

5.2 Underlying EA Choices

The classification tree given in Figure 3, generated by the rpart R library, analyzes which underlying EA was chosen by irace for each high-performing algorithm configuration. In the tree, the left subtree of a node represents the subset of scenarios that satisfy the condition stated by the node. For instance, the root of the classification tree shows that the most important factor affecting the choice between the different underlying EAs

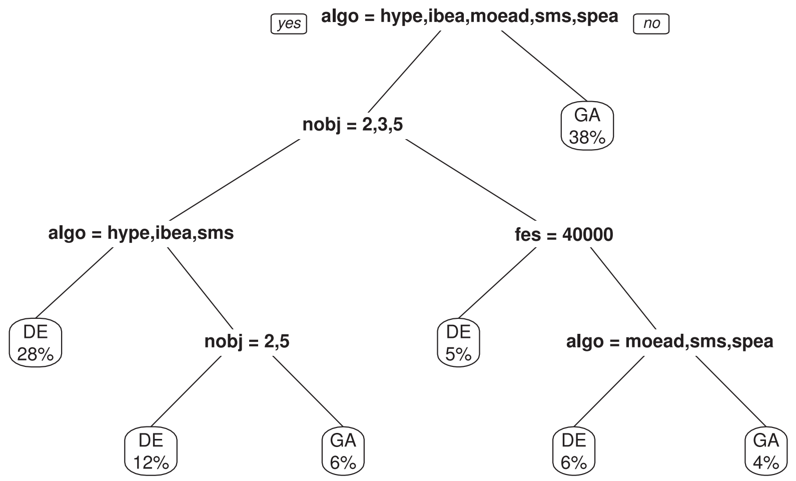

Figure 3: Classification tree of the underlying EA selected by irace for the 96 high-performing MOEA configurations (excluding MO-CMA-ES) found in the tuning phase. Cluster cardinalities are given as percentages in the leaf nodes.

is the MOEA considered. The left subtree refers to HypE, IBEA, MOEA/D, SMS, and SPEA2, while the right subtree refers to MOGA, NSGA-II, and NSGA-III. The first insight from this analysis is that, for the three latter MOEAs, high-performing MOEA configurations tend to use GA, as traditionally adopted in the literature, regardless of other experimental factors.

Moving along the left subtree, we see that for the remaining MOEAs, DE is chosen more often for most of the scenarios considered. There are some exceptions, though. For example, by following the right subtree from the second node, we can observe that the chosen EA was often GA when solving ten-objective problems using HypE or IBEA given a low or moderate number of FEs. Finally, by summing up the percentages of the different EA choices (given in the leaf nodes), we see that DE was selected in 52% of the configurations generated, with GA being selected in the remaining 48%. Although we remark that the differences in performance between using DE or GA vary significantly according to the MOEA considered, this result is strong evidence that the "best" underlying EA depends not only on the scenario, but also on the particular MOEA (Tušar and Filipič, 2007; Bezerra et al., 2015b).

5.3 The Importance of $FE_{\max}$

The effect of generalizing results obtained with a given stopping criterion to a different one is illustrated in Figure 4, which shows the development of mean I_H^{rd} (25 independent runs) over the number of function evaluations for various MOEAs that were tuned with a stopping criterion of $FE_{\max} = 10\,000$ and then run with $FE_{\max} = 50\,000$. The plot shows the curves crossing at several points of the plot, meaning that different stopping criteria would favor different algorithms.

A more general observation is obtained by measuring the correlation between the performance of the MOEAs for two different $FE_{\max}$ values. More precisely, we rank all results for each performance metric and each experimental scenario. Then, we compute the sum of the ranks for each MOEA according to each metric and each scenario, and

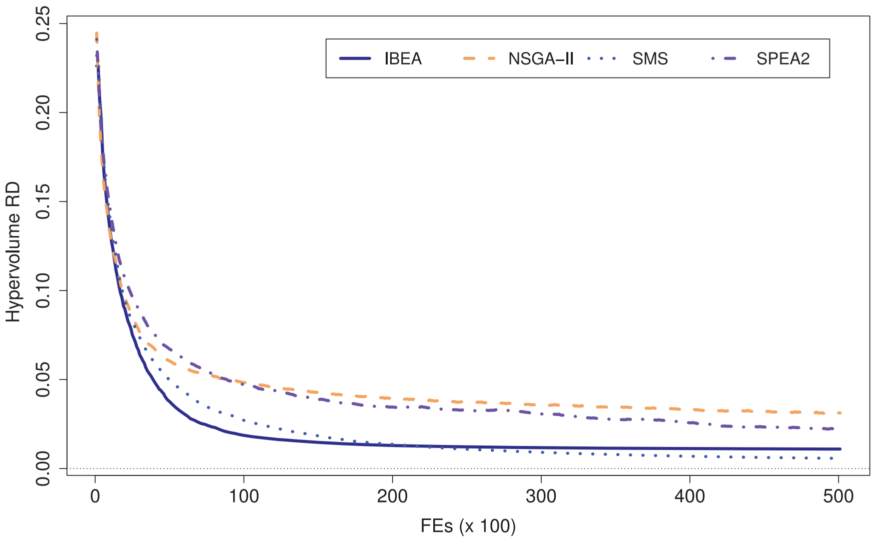

Figure 4: Mean I_H^{rd} values of different MOEAs using parameter settings tuned for a common stopping criterion ($FE_{\max} = 10\,000$) on WFG3 ($M = 3$, $n_{\text{var}} = 50$).

Table 6: Pearson's correlation coefficient between MOEA rankings for different $FE_{\max}$.

	tuned	*default*
corr(2 500, 10 000)	**0.86**	0.77
corr(2 500, 40 000)	**0.81**	0.68
corr(10 000, 40 000)	0.94	**0.96**

split the data according to $FE_{\max}$ value. Finally, we compute the correlation between rank-sum values for different $FE_{\max}$ values, paired by MOEA, metric and experimental factor. Table 6 gives the Pearson's correlation coefficients, considering both results from tuned and from default configurations of the MOEAs.

The correlation coefficients show a significant positive correlation, which suggests that the overall ranking between MOEAs does not strongly depend on the particular stopping criterion. However, the correlations between results obtained with $FE_{\max} = 2500$ and with larger values of $FE_{\max}$ are much lower than the correlation between $FE_{\max} = 10000$ and $FE_{\max} = 40000$, which indicates that generalizing conclusions obtained with very low $FE_{\max}$ to large $FE_{\max}$ or vice versa is not advisable. A more detailed analysis is obtained by studying scatter plots, but for brevity, this analysis is provided as supplementary material (Bezerra et al., 2017b).

5.4 Performance Metric Correlations

Each performance metric prefers different characteristics of approximation fronts. For this reason, we examine the correlations between these metrics on selected scenarios. First, it is common to see different rankings for the same MOEAs depending on the metric chosen, independently of the scenario considered. An example of this is shown by boxplots of the I_H^{rd} and IGD for SMS and IBEA on the WFG8 function with $M = 2$ and

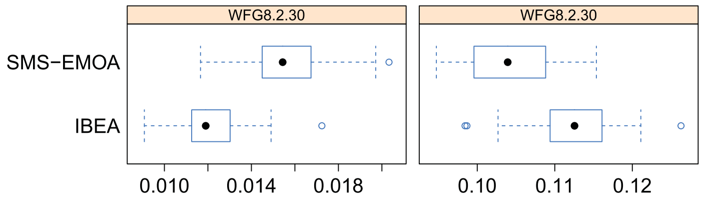

Figure 5: SMS and IBEA performance according to the I_H^{rd} (left) and IGD (right) on function WFG8 ($M = 2$, $n_{\mathrm{var}} = 30$, $FE_{\mathrm{max}} = 10\,000$).

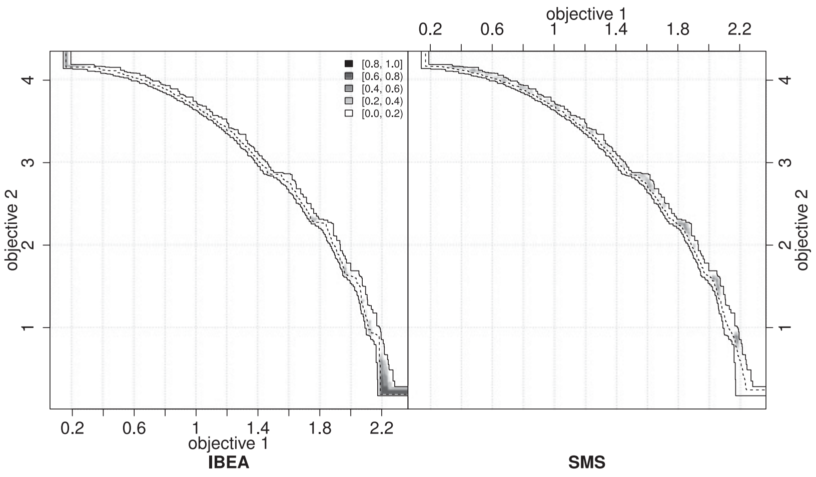

Figure 6: EAF difference plot of the fronts produced by IBEA (left) and SMS (right) on function WFG8 ($M = 2$, $n_{\mathrm{var}} = 30$, $FE_{\mathrm{max}} = 10\,000$).

$n_{\mathrm{var}} = 30$ given in Figure 5, where the I_H^{rd} values indicate better performance for IBEA while *IGD* indicates the opposite. As previously mentioned, on concave problems the I_H^{rd} prefers spread over distribution, while the *IGD* does the opposite. This is confirmed by the empirical attainment function (EAF) difference plot (López-Ibáñez et al., 2010) given in Figure 6, where IBEA finds more solutions on the extremes of the objective space, whereas SMS is better at filling the gaps throughout the front.

Although the ranking of the MOEAs depends on the metric considered, the best and worst performing MOEAs are consistent on most scenarios when $M \in \{2, 3, 5\}$. It is only when $M = 10$ that one sees major variations in the best and worst MOEAs according to the metric chosen. Table 7 shows Pearson's correlation coefficients between the rankings assigned by the performance metrics to MOEAs on different scenarios grouped by M. The first column, for instance, gives correlation coefficients between different metrics on scenarios with $M = 2$. The correlation is high for $M \in \{2, 3, 5\}$ and much lower when $M = 10$ for all metric pairs assessed. Concerning I_H^{rd} and *IGD*, these metrics have recently been shown to consistently disagree on concave problems (Jiang et al., 2014) and, as we will discuss in Section 7, this is the type of problem that becomes considerably harder with increasing M. Correlations involving $I_{\epsilon+}^1$ are considerably smaller because of the large disparity of the ranks obtained by MOEA/D for $M = 10$ among the various metrics, as shown later in Table 8.

Table 7: Pearson's correlation coefficient between MOEA rankings for different performance metrics.

M	2	3	5	10
corr(I_H^{rd}, $I_{\epsilon+}^1$)	0.97	0.94	0.95	**0.50**
corr(I_H^{rd}, IGD)	0.89	0.86	0.85	**0.78**
corr($I_{\epsilon+}^1$, IGD)	0.95	0.88	0.90	**0.35**

6 Comparison of MOEAs

We now move on to discuss the performance of the MOEAs across all scenarios considered. We group our discussion for scenarios where $M \in \{2, 3, 5\}$ since the previously discussed agreement between performance metrics allows us to clearly identify the best-performing MOEAs on these scenarios, and we discuss results for $M = 10$ separately. Moreover, we show only results for $FE_{\max} = 10\,000$ for brevity, although we remark that our conclusions are general enough to account for all scenarios. The complete set of results as well as a more detailed analysis concerning each specific scenario are provided as supplementary material (Bezerra et al., 2017b).

6.1 Performance Patterns for $M \in \{2, 3, 5\}$

For each metric and each scenario, we partition the results according to individual benchmark instances and the random seeds used to repeat each experiment 25 times. Within each partition, we rank the MOEAs from 1 to 9, according to the performance value obtained by their run. Then, we sum up the ranks to obtain an overall ranking, where lower rank-sum values indicate better performance. Table 8 summarizes this rank-sum analysis, from which the following performance patterns can be observed.

From the rank-sum analysis, we can cluster algorithms into four different groups according to their performance on scenarios with $M \in \{2, 3, 5\}$. In general, SMS and IBEA are the algorithms that present the best and most robust performance, regardless of the experimental factors considered. The second group is formed exclusively by MOGA, which consistently, as expected, presents the worst performance among all MOEAs. The remaining two groups comprise algorithms whose performance is sensitive to specific problem characteristics, the stopping criterion and the number of objectives. The results of NSGA-II, SPEA2, and HypE worsen with increasing number of objectives. By contrast, MOEA/D, NSGA-III, and MO-CMA-ES rank clearly worse than IBEA and SMS for low number of objectives, but rank comparably better for $M = 10$ for at least one of the metrics we considered.

The performance patterns observed lead to some interesting insights. First and most important, IBEA and SMS prove that it is possible to design MOEAs that can be applied to a wide range of objectives, as long as their parameter settings are properly configured for each scenario. By contrast, algorithms that were designed specifically for MaOPs, such as HypE and NSGA-III, are not the best MOEAs for a low number of objectives. This is also the case for MOEA/D and MO-CMA-ES, which are often assumed to be better than older MOEAs (such as IBEA, SPEA2, or NSGA-II) for problems with $M = 2$ objectives. As our results show, this is not necessarily the case when high-performing configurations are used for these older MOEAs. The second insight we observe is that algorithms are nearly perfectly clustered according to their underlying design paradigm, that is, the performance of indicator-based MOEAs is

Table 8: Ranking of MOEAs for various metrics and values of M ($FE_{\max} = 10\,000$). Numbers in parentheses give the rank-sum difference to the best ranked (left-most) MOEA. MOEAs in boldface present rank sums statistically significantly better than the rest according to Friedman's non-parametrical test.

					$M = 2$				
I_H^{rd}	**IBEA**	**SMS (113)**	SPEA2 (1149)	NSGA-II (1846)	MOEA/D (2819)	HypE (3593)	CMA (3731)	NSGA-III (4017)	MOGA (6682)
I_ϵ	**SMS**	IBEA (425)	SPEA2 (894)	NSGA-II (1993)	MOEA/D (3091)	HypE (3401)	CMA (3749)	NSGA-III (4231)	MOGA (6740)
IGD	**SMS**	SPEA2 (711)	IBEA (1387)	HypE (2382)	NSGA-II (2774)	MOEA/D (4237)	NSGA-III (5068)	CMA (5240)	MOGA (7297)
					$M = 3$				
I_H^{rd}	**SMS**	IBEA (556)	MOEA/D (1805)	HypE (2290)	SPEA2 (2302)	CMA (3616)	NSGA-II (4378)	NSGA-III (4627)	MOGA (7029)
I_ϵ	**SMS**	IBEA (494)	SPEA2 (2516)	CMA (2968)	HypE (3132)	MOEA/D (3552)	NSGA-III (4253)	NSGA-II (4885)	MOGA (7323)
IGD	**IBEA**	SMS (654)	SPEA2 (1041)	MOEA/D (1622)	HypE (2960)	NSGA-II (3640)	CMA (4240)	NSGA-III (4264)	MOGA (6886)
					$M = 5$				
I_H^{rd}	**SMS**	MOEA/D (1471)	IBEA (1535)	SPEA2 (3295)	CMA (3374)	NSGA-III (3897)	NSGA-II (4130)	HypE (5811)	MOGA (7562)
I_ϵ	**SMS**	IBEA (1713)	MOEA/D (2569)	CMA (2588)	NSGA-II (4086)	NSGA-III (4124)	SPEA2 (4701)	HypE (5907)	MOGA (7664)
IGD	**SMS**	IBEA (1898)	MOEA/D (2119)	NSGA-II (2329)	CMA (2515)	SPEA2 (3579)	HypE (5040)	NSGA-III (5225)	MOGA (7398)
					$M = 10$				
I_H^{rd}	**IBEA**	SMS (222)	CMA (1116)	NSGA-III (2326)	SPEA2 (2532)	NSGA-II (3241)	HypE (4846)	MOEA/D (5114)	MOGA (6919)
I_ϵ	**MOEA/D**	IBEA (1258)	SMS (2794)	CMA (3250)	NSGA-III (3347)	NSGA-II (4045)	SPEA2 (4562)	HypE (5214)	MOGA (6922)
IGD	**NSGA-III**	**IBEA** (36)	SPEA2 (646)	NSGA-II (1776)	SMS (1828)	CMA (1987)	HypE (2557)	MOEA/D (4424)	MOGA (5151)

similar, with the exception of HypE; the same can be said, to a lesser degree, of decomposition-based MOEAs and, independently, of dominance-based MOEAs with the exception of MOGA. We now summarize the most important conclusions concerning each paradigm below.

Indicator-based. The performance displayed by SMS and IBEA is remarkable with both algorithms being particularly robust to all the experimental factors considered in this work. More importantly, these MOEAs rank best not only for the indicator that is used to drive their search but for essentially all indicators. By contrast, the performance of HypE depends on the scenario. For $M \leq 3$, HypE appears in the middle of the ranking among the other MOEAs, whereas for $M > 3$, HypE is among the worst-ranked ones. The explanation for the low performance of HypE in $M > 3$ scenarios is not straightforward. A possible explanation is that the estimation (10^4 samples) of the shared hypervolume contribution (I_H^h) done for $M > 3$ is not sufficiently accurate. To test this hypothesis, we carried out additional experiments with different numbers of samples (10^5 and 10^6) to increase the estimation accuracy, stopping after $FE_{\max} = 10\,000$ or $t_{\max} = 10$ minutes. The changes in the number of samples produced no significant differences in the rank sums. We also tested the exact computation of I_H^h instead of the estimation with a stopping criteria of $FE_{\max} = 50\,000$ or $t_{\max} = 3$ hours. The exact computation did improve the performance of HypE by a large margin on most problems, however, it was still not enough to match the results of IBEA and SMS. Nevertheless, this result suggests that the estimation accuracy in HypE is crucial and that a number of samples of around 10^6 is not yet enough to ensure high enough accuracy.

Dominance-based. The performance trends of these MOEAs are highly dependent on M. When $M \leq 3$, the performance differences between dominance- and indicator-based algorithms are relatively small. In fact, SPEA2 is sometimes able to outrank IBEA as a consequence of the particularly good performance of SPEA2 on the WFG benchmark, which has been previously reported (Bradstreet et al., 2007). When $M = 5$, their performance becomes worse relatively to IBEA and SMS. Some instances of particularly poor ranking of SPEA2 and NSGA-II are related to specific problem characteristics that we will discuss later. Finally, as already discussed above, MOGA is consistently ranked in the last place. This is only in part explained by its lack of elitism. Additional experiments show that an elitist variant of MOGA improves over its non-elitist counterpart, yet it is unable to match the performance of NSGA-II.

Decomposition-based. MOEA/D and NSGA-III are outranked by most other MOEAs for $M = 2$. For $M \in \{3, 5\}$, the rankings of MOEA/D improve considerably, and it is ranked as the best MOEA on a few scenarios not shown in Table 8, but available as supplementary material (Bezerra et al., 2017b). By contrast, it is common that NSGA-III and NSGA-II appear close to each other in the rankings. In fact, we conducted a set of additional experiments to compare these two algorithms, including their default configurations. Table 9 shows the ranking of the results according to I_H^{rd}. For $M = 2$, both default and tuned NSGA-II outrank their NSGA-III counterparts. With $M = 5$, the tuned configurations obtain similar results and are significantly better than the default configurations, while the default configuration of NSGA-III outranks the default configuration of NSGA-II. This result reinforces the importance of proper parameter tuning before comparing algorithms. These conclusions are consistent with recent findings about the effectiveness of NSGA-III in comparison with NSGA-II for a low number of objectives (Seada and Deb, 2015).

Table 9: Ranking of tuned and default configurations of NSGA-II and NSGA-III. Numbers in parentheses give the rank-sum difference to the best ranked (left-most).

M	Ranking according to I_H^{rd} ($FE_{\max} = 10\,000$)			
2	**NSGA-II^t**	NSGA-II^d (1270)	NSGA-III^t (1424)	NSGA-III^d (2257)
5	**NSGA-II^t**	**NSGA-III^t (11)**	NSGA-III^d (901)	NSGA-II^d (1212)

MO-CMA-ES. We observe two main characteristics of the performance of MO-CMA-ES. First, on the $FE_{\max} = 2\,500$ scenarios (tables and results available as supplementary material (Bezerra et al., 2017b)), it performs much worse than the other MOEAs. For larger stopping criteria, the performance of MO-CMA-ES improves and becomes even better for larger M. For the $M = 5$ and $FE_{\max} = 40\,000$ scenario, its performance is often similar to that of SMS and IBEA, although it performs consistently worse on WFG1.

6.2 Performance Patterns for Ten Objectives ($M = 10$)

The increase in the number of objectives to ten leads to changes in the patterns reported above. As a first step, we regenerated the reference fronts because MOEAs were able to find up to 30 000 new solutions not dominated by the reference fronts generated in Section 4.1. This would lead to I_H^{rd} values in the range $[-2, 0]$. Thus, we decided to regenerate the reference fronts as the union of the previous reference fronts together with the approximation fronts produced by all MOEAs, and used these to compute the various performance metrics.

One salient observation in the case of $M = 10$ is that the best ranked MOEA strongly depends on the performance metric used, as shown in Table 8. Although this is expected, due to the different preferences implied by each metric (an example is the I_H^{rd} and *IGD* on the concave WFG problems (Jiang et al., 2014)), the effect is stronger for ten objectives. The relative ranking of MOEA/D also changes strongly between I_H^{rd} and $I_{\epsilon+}^1$. A first conclusion is that designing a single MOEA for MaOPs that optimizes simultaneously a diverse set of performance metrics is a challenging task. Moreover, conclusions obtained from a single performance metric may change when using a different metric. This is specially worrying when comparing MOEAs that were configured for metrics different from those used for the final evaluation and when relying exclusively on metrics that are not Pareto-compliant, such as *IGD*.

When considering all scenarios, and not only those shown in Table 8, a few observations stand out. First, the only algorithm to be well-ranked according to all metrics is IBEA, being the best according to I_H^{rd} and statistically indifferent to the top-ranking algorithm for *IGD*. By contrast, MOEA/D and NSGA-III also rank first for particular metrics, but much worse according to others. Finally, SMS and MO-CMA-ES display very good performance according to I_H^{rd} and $I_{\epsilon+}^1$ in many problems, but rank worse according to *IGD*.

We now analyze each algorithm in detail based on the data provided as supplementary material.

IBEA. The only exceptions to the good overall performance of IBEA are the concave WFG problems and the DTLZ6 and DTLZ7 problems. However, most MOEAs perform poorly on these problems. In addition, the performance of IBEA is consistently good across the different stopping criteria considered, but its top performance is observed when $FE_{\max} = 40\,000$, where it ranks the best according to all metrics.

NSGA-III. The performance of NSGA-III varies greatly depending on the given metric. It is the best ranked algorithm according to *IGD*, largely due to its results on the concave WFG functions. Moreover, NSGA-III is always ranked better than NSGA-II, demonstrating that the diversity measure of the former is better suited for scenarios with many objectives than the crowding distance used by NSGA-II.

MOEA/D. It is among the best ranked MOEAs according to $I^1_{\epsilon+}$, whereas it becomes the second worst MOEA according to I^{rd}_H and *IGD*. Given the characteristics of the metrics, these results indicate that MOEA/D does not produce a good distribution of nondominated solutions with respect to the reference sets. One possible explanation is that some implementations of MOEA/D combine DE-operators with polynomial mutation (Li and Zhang, 2009; Zhang et al., 2009), which is a combination never used by other DE-based MOEAs and not included in the parameter space described in Section 3.1 for fairness. However, we evaluated the effects of this choice by tuning a version of MOEA/D that combines DE and polynomial mutation. Experiments on the $M = 10$ and $FE_{\max} = 40\,000$ scenario show that this alternative MOEA/D further improves $I^1_{\epsilon+}$ but at the cost of even worse I^{rd}_H and *IGD*, thus producing an even poorer distribution of nondominated solutions.

The second hypothesis we tested is whether MOEA/D would still suffer from poor distribution when tuned using a different metric than $I^1_{\epsilon+}$. Hence, we tuned the parameters of MOEA/D using the *IGD* metric for $M = 10$ and $FE_{\max} = 40\,000$ scenario. Results show that tuning for *IGD* improves the distribution of the approximation fronts generated, and MOEA/D becomes the fourth best-ranked MOEA according to *IGD*. However, the ranking does not improve for I^{rd}_H and worsens considerably for $I^1_{\epsilon+}$.

SMS. For scenarios with ten objectives, SMS is no longer among the top two MOEAs, except for the scenario with $FE_{\max} = 2\,500$ and $t_{\max} = 1$ hour and according to I^{rd}_H. SMS becomes very computationally expensive with ten objectives, thus the most logical explanation is that the cut-off time of $t_{\max} = 10$ minutes of other scenarios stops SMS from producing better results. To corroborate this hypothesis, we compute the average number of function evaluations (FEs) used by SMS across different M scenarios. While for all scenarios with 2, 3, and 5 objectives, SMS is able to reach $FE_{\max}$, this is not the case for scenarios with 10 objectives, where it uses on average 2478.82, 8858.32, and 32723.9 FEs for scenarios with $FE_{\max}$ equal to 2500, 10000, and 40000, respectively.

Furthermore, the population size selected by irace also decreases on the scenarios where the cut-off time has an effect on the FEs, suggesting that it is better to save time by reducing the complexity of the hypervolume computation in order to perform more FEs. Yet, SMS cannot outrank other MOEAs for high number of objectives if runtime is limited.

MO-CMA-ES. The above analysis of SMS also applies to MO-CMA-ES, since both algorithms use the exclusive hypervolume contribution to guide their search. Moreover, the performance of MO-CMA-ES on $M = 10$ is very similar to that of SMS. Comparing both algorithms on individual benchmark functions reveals performance differences on particular problems. Table 10 summarizes the results of applying the Friedman test only to MO-CMA-ES and SMS on specific benchmark functions. The best ranked algorithm is shown if the test suggests that the difference in ranks is significant, and an equality sign is shown otherwise. The first observation is that metrics agree or not depending on the particular function, confirming similar observations made above. On

Table 10: Comparison between MO-CMA-ES and SMS ($M = 10$, $FE_{\max} = 10\,000$) on specific functions. D stands for DTLZ and W stands for WFG. Each entry shows the best ranked of the two, according to the sum of ranks computed by taking all other MOEAs into account. = denotes no statistically significant difference between the rank-sums.

	D5	D6	W1	W2	W4	W6	W7	W8
I_H^{rd}	SMS	SMS	CMA	=	SMS	CMA	=	SMS
$I_{\epsilon+}^1$	=	SMS	=	SMS	SMS	CMA	=	CMA
IGD	=	CMA	CMA	=	CMA	CMA	CMA	=

convex problems (WFG1–2), $I_{\epsilon+}^1$ disagrees with the other metrics, whereas on concave problems (WFG4–8), the *IGD* tends to disagree with the others.

Another observation is that, according to *IGD*, MO-CMA-ES often ranks better or at least equivalently to SMS on most WFG problems and on DTLZ5–6. In general, we observe that MO-CMA-ES produces consistently good results in terms of *IGD* for different number of variables. By contrast, MO-CMA-ES performs much worse than SMS on WFG3 and on the deceptive WFG problems (WFG5 and WFG9).

Finally, the relatively good ranking of NSGA-II and SPEA2 in most scenarios with ten objectives may seem somewhat surprising. However, we must remark that both algorithms improve significantly after proper parameter tuning. Moreover, the tuned configurations of SPEA2 use DE as the underlying EA for scenarios with $FE_{\max}$ equal to 10 000 and 40 000, highlighting the need of proper tuning and choice of underlying EA before comparison. In fact, when we compute Pearson's correlation coefficient considering only the rankings obtained by these two algorithms before and after tuning, we get only a low correlation (0.31), confirming the positive effects of the automatic tuning coupled with the flexibility of choosing different underlying EAs.

7 Problem-Specific Analysis

As previously discussed, dominance resistance describes the fact that the proportion of locally mutually nondominated solutions increases rapidly for higher number of objectives, and MOEAs cannot rely on dominance to guide the search towards the Pareto front. As an indication of dominance resistance in our experiments, we take the performance of the algorithms as given by the values of the metrics that we measure; thus, we interpret here an increase in metric values for increasing number of objectives as indirect evidence of the fact that an algorithm suffered from increased dominance resistance. From our results, we can make the following two observations. First, the overall trend shown by the results confirms that performance typically decreases as the number of objectives increases. As a result, the performance of MOEAs on ten-objective problems is poor for many problems, showing that MOEAs still need major improvements to deal with dominance resistance.

The second observation concerns the interaction between the number of objectives (M) and particular problem characteristics on the difficulty of benchmark problems. In particular, an increase in M affects the performance of the algorithms in very different ways depending on the particular problem, as we illustrate in Figure 7. Boxplots of I_H^{rd} in scenarios with $FE_{\max} = 10\,000$ are shown grouped by the number of objectives M and benchmark function. For brevity, plots showing other metrics are provided as supplementary material (Bezerra et al., 2017b). Some problems, such as DTLZ7, appear

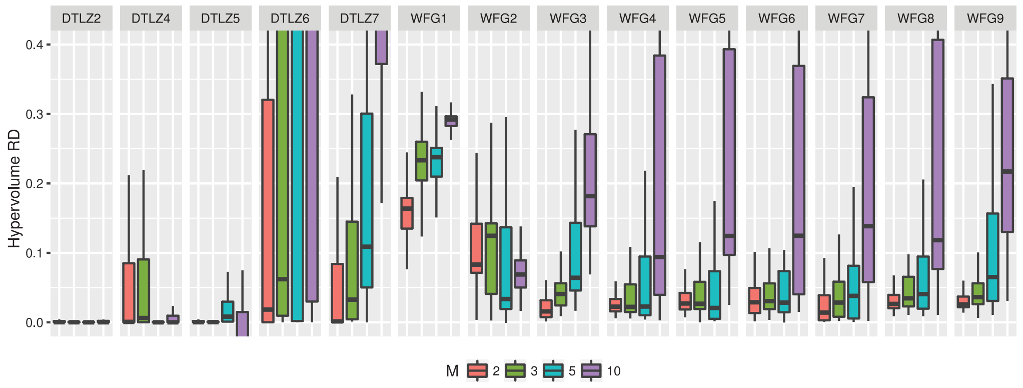

Figure 7: Boxplots of all I_H^{rd} values for $FE_{\max} = 10000$, grouped by M and benchmark function, depicting how dominance resistance affects problems in different ways.

to show more dominance resistance than others. Conversely, for particular problems and metrics one can see the opposite effect. For instance, results according to the I_H^{rd} metric on DTLZ4 get better as M increases, and the same happens for results on WFG2 according to both I_H^{rd} and $I_{\epsilon+}^1$ metrics.

Next, we identify problem-specific features that affect MOEA performance.

7.1 Geometry of Pareto-Optimal Fronts

On problems with concave Pareto-optimal fronts, scenarios with $M \in \{2, 3\}$ show similar metric values, yet metric values become considerably larger (worse) as the number of objectives is increased, regardless of the other characteristics they present. By contrast, on problems with convex Pareto-optimal fronts such as WFG1–3, most MOEAs using GA instead of DE perform worse already for $M \in \{2, 3\}$, but the metric values do not significantly increase by the addition of more objectives. As a result, when $M \in \{5, 10\}$, MOEAs achieve smaller metric values on convex problems than on the concave ones. We illustrate this with the boxplots of IGD shown in Figure 8. The IGD values of MOEAs given $FE_{\max} = 40\,000$ on the (convex) WFG1 problem with $n_{\text{var}} = 40$ and increasing M are shown on the top, and the IGD values on the (concave) WFG4 problem with the same experimental factors are given at the bottom. IGD values are relatively small and homogeneous for all MOEAs on the concave problem with 2 and 3 objectives (bottom), yet the values increase vary rapidly with higher number of objectives. By contrast, on the convex problem (top) the values are more heterogeneous for various MOEAs, but they increase much more slowly with the number of objectives. These results are consistent with the findings of Schütze et al. (2011), who report that the increase in dominance resistance is only clearly observable for concave problems.

7.2 Local Pareto-Optimal Fronts

MOEAs face considerable difficulties on problems that present local Pareto-optimal fronts, such as DTLZ6. We illustrate the effect on performance by running NSGA-II on several modified versions of the DTLZ6 problem, which we parametrize to regulate the number of local Pareto-optimal fronts. In the original DTLZ6 formulation, an auxiliary function $g(x) = \sum_{x \in x_M} x^{\rho}$ with $\rho = 0.1$ induces local Pareto-optimal fronts, where smaller ρ values lead to more local fronts. The plot given in Figure 9 shows the development of the I_H^{rd} value, averaged over 25 runs, obtained by NSGA-II (tuned

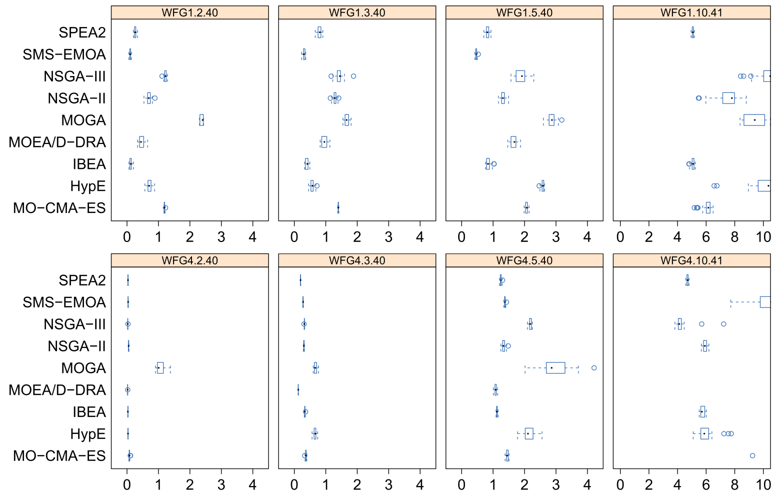

Figure 8: *IGD* performance of MOEAs given 10 000 FEs on selected problems with 40 variables and $M = 2$ to $M = 10$ objectives. Top: WFG1 having convex Pareto-optimal fronts. Bottom: WFG4 having concave Pareto-optimal fronts.

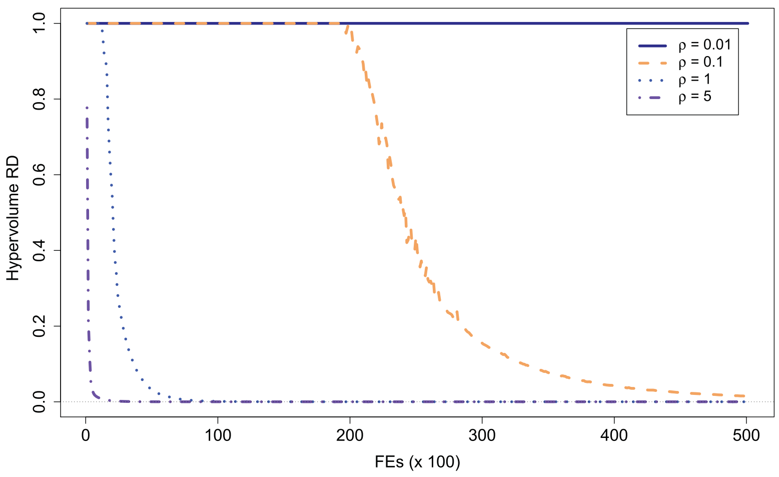

Figure 9: I_H^{rd} performance of NSGA-II (tuned for $M = 3$, $FE_{\max} = 10\,000$) on different versions of the DTLZ6 problem ($M = 3$, $n_{\mathrm{var}} = 50$).

for $M = 3$ and $FE_{\max} = 10\,000$) across 50 000 FEs on the DTLZ6 problem ($n_{\mathrm{var}} = 50$, $\rho \in \{0.01, 0.1, 1.0, 5.0\}$). The shift in the curves towards smaller (better) values follows the increase in ρ, confirming that for $M = 3$ the difficulty of this problem is strongly dependent on the number of local Pareto-optimal fronts.

7.3 Density Bias

Two of the benchmark problems considered (DTLZ4 and WFG1) present bias, that is, a shift in the density of the objective space meant to test whether MOEAs can obtain a well-distributed approximation front. In DTLZ4, for example, the degree of bias is controlled by a parameter α, set by default to 100, where larger α values lead to a stronger

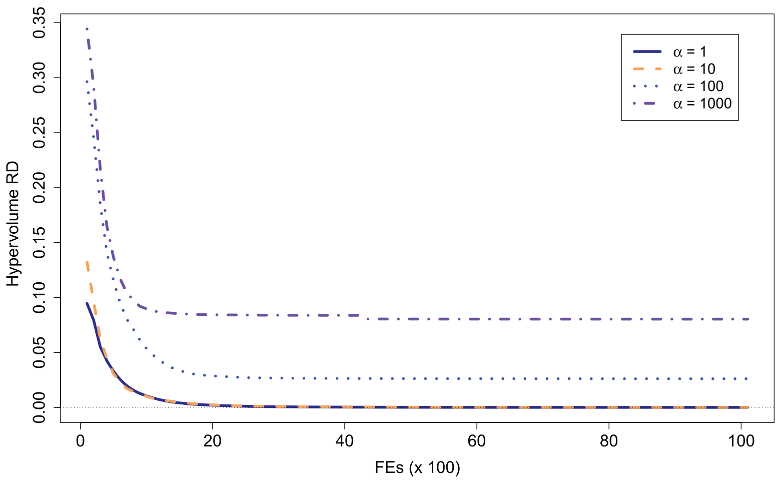

Figure 10: I_H^{rd} performance of SPEA2 (tuned for $M = 3, FE_{\max} = 10\,000$) on different versions of the DTLZ4 problem ($M = 3$, $n_{\mathrm{var}} = 50$).

bias and, presumably, more difficult problems. Figure 10 illustrates this by showing the development of the I_H^{rd} value, averaged over 25 runs, obtained by SPEA2 (tuned for $M = 3$ and $FE_{\max} = 10\,000$) on the DTLZ4 problem with $M = 3$, $n_{\mathrm{var}} = 50$, and several values of $\alpha \in \{1, 10, 100, 1000\}$. The plot shows that large bias leads to much worse approximation fronts (up to 10% from the optimal hypervolume). In general, when $M \in \{2, 3\}$, the performance of some MOEAs, such as SPEA2, is greatly affected by the bias of problems such as DTLZ4 and WFG1, relative to their performance on similar problems, DTLZ2 and WFG2, respectively, without such bias. However, with the addition of more objectives, results become more similar between these problems that mostly differ in the bias; thus we believe that the difficulty introduced by bias does not increase with M. In fact, performance metrics values on DTLZ4 become increasingly better, in general, with larger M, as we had previously discussed.

8 Conclusion

In this work, we have conducted a comprehensive performance assessment of multi-objective evolutionary algorithms (MOEAs) both on multi- and many-objective optimization problems (MOPs and MaOPs, respectively). Our study considered a large number of experimental factors, namely: (i) number of MOEAs, (ii) a formally defined, large parameter space, (iii) benchmark problems with various numbers of objectives, (iv) different stopping criteria, and (v) several performance metrics. We designed different scenarios for each combination of number of objectives and stopping criteria, and, before comparing MOEAs, we identified high-performing parameter configurations for each scenario by means of an automatic algorithm configuration method. Finally, we also established a clear separation between high-level multi-objective (MO) algorithmic components and the underlying evolutionary algorithm (EA) with which they are coupled, thus encompassing a large number of MOEA variants that only differ on the latter, while at the same time using the best underlying EA for each MOEA and experimental scenario.

Given the large amount of data produced in this work, we have focused the discussion on the most important insights. First, we have shown how the use of high-performing (tuned) parameter configurations, instead of the default ones from the literature, can significantly affect the comparison. In particular, the choice of underlying EA, while often overlooked in the literature as a minor detail, depends strongly on the particular scenario and has a significant impact on the performance of MOEAs. Moreover, the choice of performance metric, number of objectives and, to a lesser extent, stopping criterion, all affect the ranking of MOEAs. All this indicates that one should avoid over-generalizing conclusions from one choice to a different one. Nevertheless, we can still identify overall good MOEAs for the scenarios studied here. In particular, SMS shows the best and most robust performance across a number of objectives $M \in \{2, 3, 5\}$, and IBEA is a close competitor to SMS or sometimes also surpasses it. In the case of ten objectives, the best MOEA strongly depends on the particular performance metric used for ranking: IBEA is the best for I_H^{rd}, MOEA/D for $I_{\epsilon+}^1$ and NSGA-III for IGD. Overall, IBEA shows the most consistent performance across all metrics.

Although our analysis confirms and extends previous observations about the influence of problem-specific features, such as convexity, local Pareto fronts, bias, and dominance resistance, the results presented here also cast into doubt some of the accepted intuitions and "common knowledge" in MOEA research. We attribute this to the fact that our experimental setup aims at avoiding over-generalizations and unintended biases caused by floor effects, the choice of parameter settings, limited number of MOEAs and scenarios under study, and overlooked interactions between MO-components and underlying parameters. The characteristics of our study suggest that the conclusions of previous, more restricted studies may have been premature due to under-performing parameter settings and specific scenario characteristics.

Nonetheless, even a study as large as the one presented here is limited in comparison to the number of MOEAs and problems available in the literature. However, experiments with additional algorithms can be easily incorporated into the analysis following the experimental principles discussed above. Moreover, the use of an automatic configuration method for deciding the parameter settings of each MOEA means that the process can be fully automated given a benchmark set and a performance metric, making such extensions straightforward. In fact, an important extension of this investigation concerns benchmark problems. Specifically, we have briefly analyzed problem features and their role on dominance resistance, and we show that designing MOEAs and tuning their parameters according to specific problem features improves their performance. However, the precise problem features are often not known in advance for real-world problems. Further work in this direction would benefit from the creation of a set of benchmark problems parametrized by various features, and from the identification of relevant features that can be cheaply computed in advance or while solving a problem.

Another direction in which this investigation should evolve is to consider the use of external archives, as discussed in Section 3. In particular, many works have addressed the importance of external archives for convergence (López-Ibáñez et al., 2011), and our previous work on the topic has demonstrated how MOEAs can benefit from the complementary roles of populations and archives in search (Bezerra et al., 2016). Existing experimental comparisons (Tanabe et al., 2017) rely on default configurations even when studying the anytime behavior of MOEAs, for which we know that significant different results may obtained after parameter tuning (Radulescu et al., 2013). We expect that an analysis similar to the one conducted in this work but considering external archives could lead to different, yet complementary insights that would be of particular importance in scenarios where large population sizes are prohibitive for some MOEAs (such as the $M = 10$ scenarios studied in this work).

A final contribution of our study is making publicly available (Bezerra et al., 2017b) all the data produced by our experiments to serve as a baseline for future studies and to carry out further analysis. The data provided may also be used to understand correlations between the algorithmic components of MOEAs, problem features and performance metrics, in order to understand what is the best MOEA design to tackle specific real-world problems.

Acknowledgments

This research has received funding from the COMEX project (P7/36) within the Interuniversity Attraction Poles Programme of the Belgian Science Policy Office. T. Stüzle acknowledges support from the Belgian F.R.S.-FNRS of which he is a research director. Part of the research by L.C.T. Bezerra was supported by the Belgian F.R.S.-FNRS through a FRIA fellowship.

References

Abbass, H. A. (2002). The self-adaptive Pareto differential evolution algorithm. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pp. 831–836.

Aguirre, H. E. (2013). Advances on many-objective evolutionary optimization. In C. Blum and E. Alba (Eds.), *GECCO (Companion)*, pp. 641–666, New York: ACM Press.

Aguirre, H. E., and Tanaka, K. (2009). Many-objective optimization by space partitioning and adaptive ϵ-ranking on MNK-landscapes. In M. Ehrgott, C. M. Fonseca, X. Gandibleux, J.-K. Hao, and M. Sevaux (Eds.), *Evolutionary multi-criterion optimization, EMO 2009*, pp. 407–422. Lecture Notes in Computer Science, vol. 5467.

Bader, J., and Zitzler, E. (2011). HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76.

Beume, N., Fonseca, C. M., López-Ibáñez, M., Paquete, L., and Vahrenhold, J. (2009). On the complexity of computing the hypervolume indicator. *IEEE Transactions on Evolutionary Computation*, 13(5):1075–1082.

Beume, N., Naujoks, B., and Emmerich, M. T. M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669.

Bezerra, L. C. T., López-Ibáñez, M., and Stützle, T. (2015a). Comparing decomposition-based and automatically component-wise designed multi-objective evolutionary algorithms. In A. Gaspar-Cunha, C. H. Antunes, and C. A. Coello Coello (Eds.), *Evolutionary multi-criterion optimization, EMO 2015 Part I*, pp. 396–410. Lecture Notes in Computer Science, vol. 9018.

Bezerra, L. C. T., López-Ibáñez, M., and Stützle, T. (2015b). To DE or not to DE? Multi-objective differential evolution revisited from a component-wise perspective. In A. Gaspar-Cunha, C. H. Antunes, and C. A. Coello Coello (Eds.), *Evolutionary multi-criterion optimization, EMO 2015 Part I*, pp. 48–63. Lecture Notes in Computer Science, vol. 9018.

Bezerra, L. C. T., López-Ibáñez, M., and Stützle, T. (2016). Automatic component-wise design of multi-objective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 20(3):403–417.

Bezerra, L. C. T., López-Ibáñez, M., and Stützle, T. (2017a). An empirical assessment of the properties of inverted generational distance indicators on multi- and many-objective optimization. In H. Trautmann, G. Rudolph, K. Klamroth, O. Schütze, M. M. Wiecek, Y. Jin, and C. Grimme (Eds.), *Evolutionary multi-criterion optimization, EMO 2017*, pp. 31–45. Lecture Notes in Computer Science.

Bezerra, L. C. T., López-Ibáñez, M., and Stützle, T. (2017b). A large-scale experimental evaluation of high-performing multi- and many-objective evolutionary algorithms. Retrieved from http://iridia.ulb.ac.be/supp/IridiaSupp2015-007/

Birattari, M. (2009). *Tuning metaheuristics: A machine learning perspective. Studies in computational intelligence*, vol. 197. Berlin/Heidelberg: Springer.

Bradstreet, L., Barone, L., While, L., Huband, S., and Hingston, P. (2007). Use of the WFG toolkit and PISA for comparison of MOEAs. In *IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making*, pp. 382–389.

Chand, S., and Wagner, M. (2015). Evolutionary many-objective optimization: A quick-start guide. *Surveys in Operations Research and Management Science*, 20(2):35–42.

Coello Coello, C. A., Lamont, G. B., and Van Veldhuizen, D. A. (2007). *Evolutionary algorithms for solving multi-objective problems*. New York: Springer.

Cohen, P. R. (1995). *Empirical methods for artificial intelligence*. Cambridge, MA: MIT Press.

Deb, K., and Agrawal, R. B. (1995). Simulated binary crossover for continuous search spaces. *Complex Systems*, 9(2):115–148.

Deb, K., and Agrawal, S. (1999). A niched-penalty approach for constraint handling in genetic algorithms. In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms*, pp. 235–243.

Deb, K., and Jain, S. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601.

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

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2005). Scalable test problems for evolutionary multiobjective optimization. In A. Abraham, L. Jain, and R. Goldberg (Eds.), *Evolutionary multiobjective optimization, Advanced Information and Knowledge Processing*, pp. 105–145. London: Springer.

Fleming, P. J., Purshouse, R. C., and Lygoe, R. J. (2005). Many-objective optimization: An engineering design perspective. In C. A. Coello Coello, A. H. Aguirre, and E. Zitzler (Eds.), *Evolutionary multi-criterion optimization, EMO 2005*, pp. 14–32. Lecture Notes in Computer Science, vol. 3410.

Fonseca, C. M., and Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest (Ed.), *ICGA*, pp. 416–423. New York: Morgan Kaufmann.

Grunert da Fonseca, V., Fonseca, C. M., and Hall, A. O. (2001). Inferential performance assessment of stochastic optimisers and the attainment function. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne (Eds.), *Evolutionary multi-criterion optimization, EMO 2001*, pp. 213–225. Lecture Notes in Computer Science, vol. 1993.

Hadka, D., and Reed, P. (2012). Diagnostic assessment of search controls and failure modes in many-objective evolutionary optimization. *Evolutionary Computation*, 20(3):423–452.

Hajela, P., and Lin, C.-Y. (1992). Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4(2):99–107.

Hoos, H. H. (2012). Automated algorithm configuration and parameter tuning. In Y. Hamadi, E. Monfroy, and F. Saubion (Eds.), *Autonomous search*, pp. 37–71. Berlin: Springer.

Hoos, H. H., and Stützle, T. (2004). *Stochastic local search: Foundations and applications*. Amsterdam: Elsevier.

Horn, J., Nafpliotis, N., and Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multiobjective optimization. In *Proceedings of the 1994 World Congress on Computational Intelligence*, pp. 82–87.

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

Igel, C., Hansen, N., and Roth, S. (2007). Covariance matrix adaptation for multi-objective optimization. *Evolutionary Computation*, 15(1):1–28.

Igel, C., Heidrich-Meisner, V., and Glasmachers, T. (2008). Shark. *Journal of Machine Learning Research*, 9:993–996.

Ikeda, K., Kita, H., and Kobayashi, S. (2001). Failure of Pareto-based MOEAs: Does non-dominated really mean near to optimal? In *Proceedings of the 2001 Congress on Evolutionary Computation*, pp. 957–962.

Ishibuchi, H., Akedo, N., and Nojima, Y. (2015). Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Transactions on Evolutionary Computation*, 19(2):264–283.

Ishibuchi, H., Masuda, H., Tanigaki, Y., and Nojima, Y. (2015). Modified distance calculation in generational distance and inverted generational distance. In A. Gaspar-Cunha, C. H. Antunes, and C. A. Coello Coello (Eds.), *Evolutionary multi-criterion optimization, EMO 2015 Part I*, pp. 110–125. Lecture Notes in Computer Science, vol. 9018.

Jaszkiewicz, A. (2002). Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137(1):50–71.

Jiang, S., Ong, Y. S., Zhang, J., and Feng, L. (2014). Consistencies and contradictions of performance metrics in multiobjective optimization. *IEEE Transactions on Cybernetics*, 44(12):2391–2404.

Khare, V., Yao, X., and Deb, K. (2003). Performance scaling of multi-objective evolutionary algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele (Eds.), *Evolutionary multi-criterion optimization, EMO 2003*, pp. 376–390. Lecture Notes in Computer Science, vol. 2632.

Knowles, J. D., and Corne, D. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172.

Knowles, J. D., Thiele, L., and Zitzler, E. (2006). A tutorial on the performance assessment of stochastic multiobjective optimizers. TIK-Report 214, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zürich, Switzerland. Revised version.

Kukkonen, S., and Lampinen, J. (2005). GDE3: The third evolution step of generalized differential evolution. In *Proceedings of the 2005 Congress on Evolutionary Computation*, pp. 443–450.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282.

Li, B., Li, J., Tang, K., and Yao, X. (2015). Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys*, 48(1):1–35.

Li, H., and Zhang, Q. (2009). Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13(2):284–302.

Li, M., Yang, S., Liu, X., and Shen, R. (2013). A comparative study on evolutionary algorithms for many-objective optimization. In R. C. Purshouse, P. J. Fleming, C. M. Fonseca, S. Greco, and

J. Shaw (Eds.), *Evolutionary multi-criterion optimization*, pp. 261–275. Lecture Notes in Computer Science, vol. 2013.

Liao, T., Stützle, T., Montes de Oca, M. A., and Dorigo, M. (2014). A unified ant colony optimization algorithm for continuous optimization. *European Journal of Operational Research*, 234(3):597–609.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Stützle, T., and Birattari, M. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspective*, 3:43–58.

López-Ibáñez, M., Knowles, J. D., and Laumanns, M. (2011). On sequential online archiving of objective vectors. In R. H. C. Takahashi et al. (Eds.), *Evolutionary multi-criterion optimization, EMO 2011*, pp. 46–60. Lecture Notes in Computer Science, vol. 6576.

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

López-Ibáñez, M., and Stützle, T. (2012). The automatic design of multi-objective ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 16(6):861–875.

Madavan, N. K. (2002). Multiobjective optimization using a Pareto differential evolution approach. In *Proceedings of the 2002 World Congress on Computational Intelligence*, pp. 1145–1150.

Mezura-Montes, E., Reyes-Sierra, M., and Coello Coello, C. A. (2008). Multi-objective optimization using differential evolution: A survey of the state-of-the-art. In U. K. Chakraborty (Ed.), *Advances in differential evolution*, pp. 173–196. Heidelberg: Springer.

Nowak, K., Märtens, M., and Izzo, D. (2014). Empirical performance of the approximation of the least hypervolume contributor. In T. Bartz-Beielstein, J. Branke, B. Filipič, and J. Smith (Eds.), *Parallel Problem Solving*, pp. 662–671. Lecture Notes in Computer Science, vol. 8672.

Purshouse, R. C., and Fleming, P. J. (2007). On the evolutionary optimization of many conflicting objectives. *IEEE Transactions on Evolutionary Computation*, 11(6):770–784.

Radulescu, A., López-Ibáñez, M., and Stützle, T. (2013). Automatically improving the anytime behaviour of multiobjective evolutionary algorithms. In R. C. Purshouse, P. J. Fleming, C. M. Fonseca, S. Greco, and J. Shaw (Eds.), *Evolutionary multi-criterion optimization, EMO 2013*, pp. 825–840. Lecture Notes in Computer Science, vol. 7811.

Robič, T., and Filipič, B. (2005). DEMO: Differential evolution for multiobjective optimization. In C. A. Coello Coello, A. H. Aguirre, and E. Zitzler (Eds.), *Evolutionary multi-criterion optimization, EMO 2005*, pp. 520–533. Lecture Notes in Computer Science, vol. 3410.

Rudolph, G., Schütze, O., Grimme, C., Domínguez-Medina, C., and Trautmann, H. (2016). Optimal averaged Hausdorff archives for bi-objective problems: Theoretical and numerical results. *Computational Optimization and Applications*, 64(2):589–618.

Schütze, O., Lara, A., and Coello Coello, C. A. (2011). On the influence of the number of objectives on the hardness of a multiobjective optimization problem. *IEEE Transactions on Evolutionary Computation*, 15(4):444–455.

Schütze, O., Laumanns, M., Coello Coello, C. A., Dellnitz, M., and Talbi, E.-G. (2008). Convergence of stochastic search algorithms to finite size Pareto set approximations. *Journal of Global Optimization*, 41(4):559–577.

Schütze, O., Laumanns, M., Tantar, E., Coello Coello, C. A., and Talbi, E.-G. (2010). Computing gap free Pareto front approximations with stochastic search algorithms. *Evolutionary Computation*, 18(1):65–96.

Seada, H., and Deb, K. (2015). U-NSGA-III: A unified evolutionary optimization procedure for single, multiple, and many objectives: Proof-of-principle results. In A. Gaspar-Cunha, C. H. Antunes, and C. A. Coello Coello (Eds.), *Evolutionary multi-criterion optimization, EMO 2015 Part I*, pp. 34–49. Lecture Notes in Computer Science, vol. 9018.

Srinivas, N., and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248.

Tagawa, K., Shimizu, H., and Nakamura, H. (2011). Indicator-based differential evolution using exclusive hypervolume approximation and parallelization for multi-core processors. In N. Krasnogor, and P. L. Lanzi (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2011)*, pp. 657–664.

Tanabe, R., Ishibuchi, H., and Oyama, A. (2017). Benchmarking multi- and many-objective evolutionary algorithms under two optimization scenarios. *IEEE Access*, 5:19597–19619.

Tenenbaum, J. B., Silva, V. D., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

Tušar, T., and Filipič, B. (2007). Differential evolution versus genetic algorithms in multiobjective optimization. In S. Obayashi, et al. (Eds.), *Evolutionary multi-criterion optimization, EMO 2007*, pp. 257–271. Lecture Notes in Computer Science, vol. 4403.

Voß, T., Hansen, N., and Igel, C. (2010). Improved step size adaptation for the MO-CMA-ES. In M. Pelikan, and J. Branke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010)*, pp. 487–494.

While, L., and Bradstreet, L. (2012). Applying the WFG algorithm to calculate incremental hypervolumes. In *Proceedings of the 2012 Congress on Evolutionary Computation*, pp. 1–8.

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

Zhang, Q., Liu, W., and Li, H. (2009). The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. In *Proceedings of the 2009 Congress on Evolutionary Computation*, pp. 203–208.

Zhang, Q., and Suganthan, P. N. (2009). Special session on performance assessment of multiobjective optimization algorithms/CEC'09 MOEA competition. Retrieved from http://dces.essex.ac.uk/staff/qzhang/moeacompetition09.htm

Zitzler, E., and Künzli, S. (2004). Indicator-based selection in multiobjective search. In *Proceedings of PPSN-VIII, Eighth International Conference on Parallel Problem Solving from Nature*, pp. 832–842. Lecture Notes in Computer Science, vol. 3242.

Zitzler, E., Laumanns, M., and Thiele, L. (2002). SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In K. C. Giannakoglou, D. T. Tsahalis, J. Periaux, K. D. Papaliliou, and T. Fogarty (Eds.), *Evolutionary methods for design, optimisation and control*, pp. 95–100. Barcelona: CIMNE.

Zitzler, E., and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto evolutionary algorithm. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271.

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