European Journal of Operational Research 289 (2021) 1209–1222

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Incorporating decision-maker's preferences into the automatic configuration of bi-objective optimisation algorithms

Juan Esteban Diaz[a,*], Manuel López-Ibáñez[b]

[a] USFQ Business School, Universidad San Francisco de Quito, Ecuador
[b] Alliance Manchester Business School, University of Manchester, UK

ARTICLE INFO

Article history:
Received 29 June 2019
Accepted 29 July 2020
Available online 6 August 2020

Keywords:
Metaheuristics
Automatic algorithm design and configuration
Multi-objective optimisation
Decision maker's preferences

ABSTRACT

Automatic configuration (AC) methods are increasingly used to tune and design optimisation algorithms for problems with multiple objectives. Most AC methods use unary quality indicators, which assign a single scalar value to an approximation to the Pareto front, to compare the performance of different optimisers. These quality indicators, however, imply preferences beyond Pareto-optimality that may differ from those of the decision maker (DM). Although it is possible to incorporate DM's preferences into quality indicators, e.g., by means of the weighted hypervolume indicator (HV^w), expressing preferences in terms of weight function is not always intuitive nor an easy task for a DM, in particular, when comparing the stochastic outcomes of several algorithm configurations. A more visual approach to compare such outcomes is the visualisation of their empirical attainment functions (EAFs) differences. This paper proposes using such visualisations as a way of eliciting information about regions of the objective space that are preferred by the DM. We present a method to convert the information about EAF differences into a HV^w that will assign higher quality values to approximation fronts that result in EAF differences preferred by the DM. We show that the resulting HV^w may be used by an AC method to guide the configuration of multi-objective optimisers according to the preferences of the DM. We evaluate the proposed approach on a well-known benchmark problem. Finally, we apply our approach to re-configuring, according to different DM's preferences, a multi-objective optimiser tackling a real-world production planning problem arising in the manufacturing industry.

1. Introduction

Most real-world problems require the simultaneous consideration of multiple and often conflicting criteria and, thus ideally, they should be addressed as multi-objective optimisation problems (MOPs) (Deb, 2001; Miettinen, 1999). In MOPs, there exists a set of solutions known as *Pareto-optimal set* that consists of all solutions that are not dominated by any other solution. The image of the Pareto set in objective space is the so-called Pareto front. Given the Pareto front, or a good approximation of it, the decision maker (DM) may choose their preferred solution(s) from it. The goal of some multi-objective optimisation methods is to generate as good an approximation of the Pareto front as possible.

Meta-heuristics, such as evolutionary and local search algorithms, are attractive optimisation methods in the context of Pareto optimality. For instance, multi-objective evolutionary algorithms (MOEAs) are ideal for tackling multi-objective problems, not only due to their wide applicability, flexibility (Coello Coello, 2006) (e.g., they do not require linearity, differentiability, convexity and other strict assumptions), and ease of use (Coello Coello, 2015), but also because multiple non-dominated solutions can be found in a single iteration due to their population-based nature (Coello Coello, 2006; Deb, 2001). For similar reasons, multi-objective local search methods are widely used in combinatorial optimisation (Dubois-Lacoste, López-Ibáñez, & Stützle, 2011a; 2015; Jaszkiewicz, 2018; Lust & Teghem, 2010; Paquete, Schiavinotto, & Stützle, 2007). Hybridisations of evolutionary algorithms and local search (sometimes called memetic algorithms) are also possible (Jaszkiewicz, Ishibuchi, & Zhang, 2011; Knowles & Corne, 2005).

The performance assessment of these multi-objective optimisers is not a trivial task due to various reasons. First, meta-heuristic algorithms typically produce approximations to the Pareto-optimal front. Second, due to the inherent stochasticity of meta-heuristics and the variability of input problems, these approximation fronts can be seen as stochastic sets, and we must assess their

* Corresponding author.
E-mail addresses: jediaz@usfq.edu.ec (J.E. Diaz), manuel.lopez-ibanez@manchester.ac.uk (M. López-Ibáñez).

https://doi.org/10.1016/j.ejor.2020.07.059
0377-2217/

expectation and variance. Third, although it is possible that the approximation fronts produced by one algorithm completely dominate the ones produced by an alternative algorithm, the most usual case is that approximation fronts are incomparable in terms of Pareto optimality.

To ease the analysis and comparison of multi-objective optimisers, several techniques are employed in the literature. The most prominent among them (Knowles, Thiele, & Zitzler, 2006; Zitzler, Knowles, & Thiele, 2008) are unary quality indicators, such as the hypervolume indicator (*HV*) (Zitzler, Thiele, Laumanns, Fonseca, & Grunert da Fonseca, 2003), and the empirical attainment function (EAF) (Grunert da Fonseca, Fonseca, & Hall, 2001). The EAF is particularly useful for visualising performance differences in the objective space between optimisers (López-Ibáñez, Paquete, & Stützle, 2010).

In this paper, we first describe a procedure for articulating DM's preferences by visualising EAF differences between two alternative algorithm configurations or designs, and asking the DM to state which alternative is their preferred one. The EAF differences are then translated into a weight function over the objective space that biases the computation of the *HV* in favour of the differences that are preferred by the DM. Our proposed procedure is useful on its own for articulating DM's preferences in performance assessment of multi-objective optimisers. However, our main motivation is actually the automatic configuration (AC) and design of such optimisers.

AC methods are increasingly being adopted for tuning the large number of parameters and design choices present in modern optimisation algorithms (Birattari, 2009; Hoos, 2012). The application of AC methods to multi-objective optimisers relies in the use of unary quality indicators. Unfortunately, these quality indicators introduce unstated preferences (Auger, Bader, Brockhoff, & Zitzler, 2012) that may not correspond to the preferences of the DM, who in the AC context is a human designing a new optimiser or tuning an existing one for a class of problems of their interest.

To enable a DM to state their preferences during the AC process, we combine our proposed procedure with irace (López-Ibáñez, Dubois-Lacoste, Pérez Cáceres, Stützle, & Birattari, 2016), a widely used AC method. At some point of the AC process, the DM is shown the EAF differences between two configurations exhibiting the largest differences, and asked to choose which side of the differences they prefer. After the interaction, irace continues the AC procedure guided by the stated preferences.

We demonstrate the proposed procedure in two scenarios. First, we consider a simple tuning task involving a single parameter. The expected algorithms' behaviour according to the value of this parameter is well-known, and we show that the interaction with the DM guides irace towards the expected parameter values, depending on the particular DM's choice. This simple scenario serves to validate and illustrate the proposed approach.

Our second scenario arises in the context of a real-world multi-objective production planning problem under uncertainty for a chemical manufacturer (Diaz, Handl, & Xu, 2017). Here we show how diverse DM's preferences expressed through our proposed method result in different configurations of a multi-objective optimiser and different solutions to a real-world problem.

The remainder of this paper is structured as follows. The following section continues with important background information. Section 3 describes our proposed method for eliciting preferences from visualisations of the EAF differences. Section 4 explains how to select the pair of algorithm configurations to be shown to the DM during the AC process. The experimental setup and results are discussed in Section 5. Finally, conclusions derived from this study, limitations and future research directions are given in Section 6.

2. Background

2.1. Multi-objective optimisation

In a MOP, m conflicting objective functions need to be optimised simultaneously, and thus each solution in the solution space $\mathcal{X}$ maps to an m-dimensional vector in the objective space $\mathbb{R}^m$, that is, $\vec{f} : \mathcal{X} \to \mathbb{R}^m$, and the objective vector $\vec{z} \in \mathbb{R}^m$ corresponding to a solution $x \in \mathcal{X}$ is given by $\vec{z} = \vec{f}(x) = (z_1 = f_1(x), \ldots, z_m = f_m(x))$.

A partial order can be defined among all vectors in the objective space, called weak Pareto dominance relation. Let us assume, without loss of generality, that all objectives must be minimised. We say that vector $\vec{p}$ in the objective space *weakly dominates* $\vec{q}$, $\vec{p} \preceq \vec{q}$, iff $p_i \leq q_i \, \forall i$. In the case that $\vec{p} \preceq \vec{q}$ and $\vec{p} \neq \vec{q}$, we say that $\vec{p}$ *dominates* $\vec{q}$ ($\vec{p} \prec \vec{q}$) in terms of Pareto optimality. When $p_i < q_i$ $\forall i$, we say that $\vec{p}$ *strongly dominates* $\vec{q}$. Solutions whose objective vectors are not dominated by any other feasible solution form the *Pareto set* and the image of this set in objective space is the *Pareto front*.

Any set of mutually nondominated objective vectors ($A \subseteq \mathbb{R}^m$, $\forall \vec{a} \in A, \nexists \vec{a}' \in A \setminus \{\vec{a}\} : \vec{a}' \preceq \vec{a}$) is called an *approximation front* (Zitzler et al., 2003). Multi-objective optimisers generate approximations of the true Pareto front and to assess their performance, their resulting approximation fronts need to be compared. The weak Pareto dominance relation can be generalised to approximation fronts as $A \preceq B$ iff $\forall \vec{b} \in B, \exists \vec{a} \in A : \vec{a} \preceq \vec{b}$. If $A \preceq B$ and $A \neq B$, then we say that A is *better in terms of Pareto optimality* than B. However, this relation only defines a partial order among approximation fronts, and it is often the case that they are *incomparable* in terms of Pareto optimality, that is, $A \npreceq B \wedge B \npreceq A$.

2.2. (Weighted) Hypervolume Indicator

Quality indicators that map approximation fronts onto scalar values have been proposed to ease such a comparison. A desirable feature of quality indicators is Pareto compliance (Zitzler et al., 2003), i.e., they must not contradict the partial order given by the relation among fronts that states when an approximation front is better than another in terms of Pareto optimality, as defined above. Perhaps the most prominent Pareto-compliant indicator is the *HV*, which measures the volume of the objective space weakly dominated by an approximation front up to a given reference point (Auger et al., 2012; Zitzler & Thiele, 1999). More formally, given an approximation front A and a reference point $\vec{r}$, the *HV* can be defined as:

$$HV(A) = \int^{\vec{r}} \alpha_A(\vec{z}) d\vec{z} \tag{1}$$

where $\alpha_A(\vec{z})$ is the *attainment function* $\alpha_A : \mathbb{R}^m \to \{0, 1\}$ specifying which points in the objective space are weakly dominated by A, that is,

$$\alpha_A(\vec{z}) = \begin{cases} 1 & \text{if } A \preceq \{\vec{z}\} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

According to the definition of the *HV* given above, two different approximation fronts that dominate different regions of the objective space, but with the same *HV* value, would be indistinguishable in terms of quality. In reality, however, certain regions of the objective space may be preferred by a DM and, by using the *HV* as a quality indicator, those preferences are completely ignored.

The weighted hypervolume indicator (HV^w) (Auger, Bader, Brockhoff, & Zitzler, 2009; Zitzler, Brockhoff, & Thiele, 2007) encodes DM's preferences by assigning a weight function to the objective space, so that two solutions with the same *HV* may differ in terms of their HV^w, depending upon the objective space regions those solutions dominate.

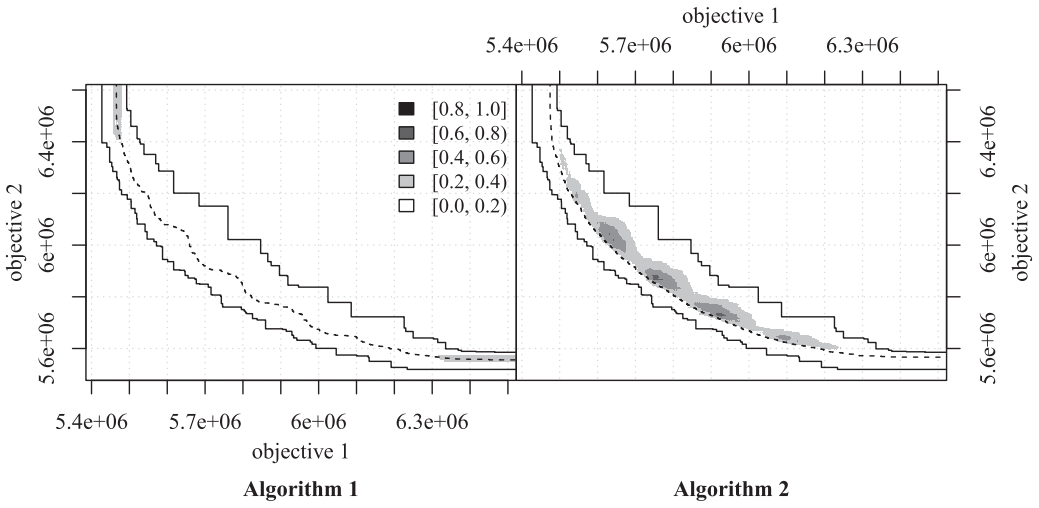


Fig. 1. EAF differences between two different algorithms.

The HV^w assigns a positive weight to each point in the objective space, $w : \mathbb{R}^m \to \mathbb{R}_{>0}$, and modifies the definition of HV as follows:

$$HV^w(A) = \int^{\vec{r}} w(\vec{z})\alpha_A(\vec{z})d\vec{z} \tag{3}$$

In principle, the DM could directly define weight functions that induce different approximation front shapes (Auger et al., 2009) as well as convert other forms of preference (e.g., Tchebycheff, ϵ-constraint and desirability functions) to weight functions (Brockhoff, Bader, Thiele, & Zitzler, 2013). However, in the case of expressing a preference between two competing algorithms (or configurations thereof), we believe that the visualisation of differences between EAFs provides a more intuitive interface.

2.3. Empirical Attainment Function (EAF)

The concept of attainment function can be generalised to describe the frequency that a specific objective vector was attained by a multiset (a set that allows repeated elements) of approximation fronts. If the approximation fronts are generated by an optimiser, this EAF (Fonseca & Fleming, 1996; Fonseca, Grunert da Fonseca, & Paquete, 2005; Fonseca, Guerreiro, López-Ibáñez, & Paquete, 2011; Grunert da Fonseca & Fonseca, 2010; Grunert da Fonseca et al., 2001) approximates the probability of the algorithm finding at least one solution whose objective vector weakly dominates a specific objective vector $\vec{z}$. More formally, let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be a multiset of n approximation fronts, the EAF is the function $\alpha_{\mathcal{A}} : \mathbb{R}^m \to [0, 1]$ such that:

$$\alpha_{\mathcal{A}}(\vec{z}) = \frac{1}{n}\sum_{i=1}^{n} \alpha_{A_i}(\vec{z}) \tag{4}$$

The visualisation of the EAFs is a well-known method for analysing the performance of multi-objective optimisers (Knowles, 2005; López-Ibáñez et al., 2010). Moreover, by comparing the EAFs of two stochastic multi-objective optimisers, it is possible to assess differences in the expected location of their generated objective vectors (López-Ibáñez, Paquete, & Stützle, 2006). The visualisation of EAF differences is an effective and intuitive way to compare the performance of two multi-objective optimisers in the objective space (Zitzler et al., 2008).

Fig. 1 is an example of such a visualisation, where shaded areas indicate regions of the objective space where the EAF of one algorithm is larger (by at least 0.2 and larger for darker regions) than the EAF of the other algorithm, thus pointing out the location in the objective space where one algorithm is more likely to outperform the other. The visualisation also shows as solid lines the region in the objective space attained at least once by either algorithm or always by both algorithms, since any EAF differences must exist within those two lines. Finally, the dashed line shows the median attainment surface that delimits the region attained by each algorithm with an EAF value of at least 0.5.

Although EAF visualisations for more than two objectives are possible (Tušar & Filipič, 2014), they are less widespread and more computationally demanding. For simplicity, in the remainder of the paper, we will focus on problems with two conflicting objectives that, without loss of generality, must be minimised.

2.4. Automatic configuration and design of multi-objective optimisers

An emerging trend in optimisation research (and machine learning) is the increasing use of AC methods for tuning the large number of parameters and design choices present in modern optimisation algorithms (Birattari, 2009; Hoos, 2012). The use of AC methods enormously simplifies this tedious and error-prone task and provides an easy-to-use and reproducible approach.

In the context of multi-objective optimisers, AC methods are being used not only for parameter tuning (Bezerra, López-Ibáñez, & Stützle, 2020a; Dubois-Lacoste et al., 2011a; Radulescu, López-Ibáñez, & Stützle, 2013), but also for automatically designing new algorithms, from frameworks of algorithmic components, that often outperform the traditional algorithms designed by humans (Bezerra, López-Ibáñez, & Stützle, 2016; López-Ibáñez & Stützle, 2012).

There is a distinction between *multi-objective AC methods*, which are able to configure an optimiser according to multiple criteria simultaneously, such as solution cost and computation time (Blot, Hoos, Jourdan, Kessaci-Marmion, & Trautmann, 2016; Zhang, Georgiopoulos, & Anagnostopoulos, 2015), and the use of *single-objective AC methods* for configuring *multi-objective optimisers*, i.e., optimisers that return an approximation front. In this paper, we focus on the latter scenario.[1] To the best of our knowledge, the use of a quality indicator (or combination thereof) is

[1] Nevertheless, a single-objective AC method may aggregate multiple criteria into a single one and return an algorithm configuration that performs well in all indicators (as they tend to be correlated) (Bezerra, López-Ibáñez, & Stützle, 2020b).

necessary when configuring multi-objective optimisers by means of well-known AC methods (Bezerra et al., 2020a).

In any case, the use of quality indicators introduces implicit preferences beyond Pareto-optimality (Auger et al., 2012) that may not reflect the ones of the DM. Therefore, it would be better if DM's preferences were explicitly incorporated into the AC process. Although a DM may articulate their preferences by using a weighted formulation of the *HV* (Auger et al., 2009), in our experience, the specification of a weight function is far from trivial and intuitive, especially in the context of AC, where multiple stochastic outcomes of several configurations of an algorithm are compared at once.

Therefore, in this paper, we propose to elicit DM's preferences by visualising EAF differences, then converting those preferences into a HV^w that will guide the AC method.

3. From EAF differences to weighted hypervolume indicator

We describe in this section a procedure to convert EAF differences into a HV^w that may be used to measure the quality of approximation sets. More concretely, after visualising the EAF differences between two optimisers (or different configurations of the same optimiser), a DM chooses their preferred direction of the differences. The resulting HV^w will assign a better quality to objective vectors or approximation fronts that attain the areas where the EAF differences are larger and in favour to the DM's preferences, while respecting the Pareto-optimality criterion. This may be used to select objective vectors from a final approximation front, to rank alternative approximation fronts or to guide a *HV*-based optimiser (Brockhoff et al., 2013). Later in this paper, we show how it can be also used to guide the AC of multi-objective optimisers.

We are given two multisets of n approximation fronts, $\mathcal{A} = \{A_1, \ldots, A_n\}$ and $\mathcal{A}' = \{A'_1, \ldots, A'_n\}$, possibly generated by n independent runs of two different algorithms (or different configurations of the same algorithm). For the computation of the *HV* (Eq. (1)), we also require a reference point $\vec{r}$ that is weakly dominated by any approximation front under consideration. If $\vec{r}$ is set by the DM, any objective vector that does not weakly dominate $\vec{r}$ is discarded.

In order to compute the EAF differences, we first compute $\alpha_{\mathcal{A}\uplus\mathcal{A}'}$, the EAF of their multiset sum (the union of multisets that allows repeated elements). A finite representation of $\alpha_{\mathcal{A}\uplus\mathcal{A}'}$ is given by a sequence of approximation fronts $\Omega = \langle L_1, \ldots, L_{2n}\rangle$, where:

$$L_i = \min\{\vec{z} \in \mathbb{R}^m : \alpha_{\mathcal{A}\uplus\mathcal{A}'}(\vec{z}) \geq i/2n\} \tag{5}$$

and "min" denotes the minima elements according to Pareto-optimality. The size of Ω is finite and bounded by $\sum_{i=1}^{2n} |L_i| \in \Theta(2n \cdot \sum_{i=1}^{n} |A_i| + |A'_i|)$ (Fonseca et al., 2011).

Let us assume, w.l.o.g, that the DM prefers differences in favour of $\mathcal{A}$, i.e., $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{z}) = \alpha_{\mathcal{A}}(\vec{z}) - \alpha_{\mathcal{A}'}(\vec{z})$ is positive. A finite representation of these EAF differences will contain the points that bound, by below, regions with the same value of the difference. Since a change in the value of $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{z})$ implies a change in the number of approximation fronts that dominate a certain region, it will also imply a change in $\alpha_{\mathcal{A}\uplus\mathcal{A}'}(\vec{z})$. Therefore, in a second step, we revisit all points in Ω and compute their value of the EAF differences $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{p})$, $\forall \vec{p} \in L_i$, $i = 1, 2, \ldots, 2n$.

Starting from Ω and the EAF differences computed above, Algorithm 1 computes a finite representation $\mathcal{R}$ of the rectangular regions of the objective space where $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\cdot)$ is positive, i.e., differences are in favour of $\mathcal{A}$. Each region $R \in \mathcal{R}$ is defined by a tuple $\langle \vec{l}, \vec{u}, \Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{l})\rangle$, where $\vec{l}$ and $\vec{u}$ are the points in the objective space that define, respectively, the lower and upper bounds of the region. Since Ω is a sequence of approximation fronts $\langle L_1, \ldots, L_{2n}\rangle$, where each $L_{i-1} \preceq L_i$, we can process each pair of fronts L_{i-1} and L_i to find the regions delimited by them (line 2). Let us assume that

Algorithm 1: Compute the set of regions $\mathcal{R}$ corresponding to EAF differences $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{z}) = \alpha_{\mathcal{A}}(\vec{z}) - \alpha_{\mathcal{A}'}(\vec{z})$.

Input: $\Omega = \langle L_1, \ldots, L_{2n}\rangle$: finite representation of EAF differences (Eq. 5)
$\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{p})$: value of the EAF difference at each $\vec{p} \in \Omega$
$\vec{r}$: reference point
Output: $\mathcal{R}$: set of regions

```
1  R ← {}
2  for i ← 2 to |Ω| do
3      j ← 1, p_j ∈ L_{i−1}
4      k ← 1, p_k ∈ L_i
5      top ← r_2
6      while j ≤ |L_{i−1}| ∧ k ≤ |L_i| do
7          while k ≤ |L_i| ∧ p_{j,2} < p_{k,2} do        // p_j is below p_k
8              if p_{j,1} < p_{k,1} then                // p_j dominates p_k
9                  l ← (p_{j,1}, p_{k,2}), u ← (p_{k,1}, top)
10                 R ← R ∪ ⟨l, u, Δα_{A,A'}(p_j)⟩       // Add rectangle
11             top ← p_{k,2}
12             k ← k + 1
13         if k ≤ |L_i| then                            // p_j is not below p_k
14             if p_{j,1} < p_{k,1} then                // p_j does not dominate p_k
15                 l ← p_j, u ← (p_{k,1}, top)
16                 R ← R ∪ ⟨l, u, Δα_{A,A'}(p_j)⟩       // Add rectangle
17             top ← p_{j,2}
18             j ← j + 1                                // Move to next point in L_{i−1}
19             else if j > |L_{i−1}| then
20                 break                                // Done with L_{i−1}, switch to next i
21             else if top == p_{k,2} then              // p_{j−1} was not above p_k
22                 k ← k + 1                            // Move to next point in L_i
23     while j ≤ |L_{i−1}| do                           // We are done with L_i but not with L_{i−1}
24         l ← p_j, u ← (r_1, top)
25         R ← R ∪ ⟨l, u, Δα_{A,A'}(p_j)⟩               // Add rectangle
26         top ← p_{j,2}
27         j ← j + 1                                    // Move to next point in L_{i−1}
28 return R
```

all points within each L_i are sorted in increasing order (resp. decreasing order) of the first (resp. second) objective, $\forall \vec{p}_k, \vec{p}_{k+1} \in L_i$, $\vec{p}_{k,1} < \vec{p}_{k+1,1} \wedge \vec{p}_{k,2} > \vec{p}_{k+1,2}$. Algorithm 1 proceeds by examining each point $\vec{p}_j \in L_{i-1}$ in order and finding out what point $\vec{p}_k \in L_i$ bounds by above the region dominated by $\vec{p}_j$. If the region is unbounded, we use the reference point $\vec{r}$ as an upper bound. The loop in line 7 processes all points $\vec{p}_k \in L_i$ that are above the current $\vec{p}_j \in L_{i-1}$, whereas the code in lines 14–22 processes all points $\vec{p}_j \in L_{i-1}$ that are not below the current $\vec{p}_k \in L_i$. The variable *top* keeps track of the second coordinate of the lower bound from the last processed rectangle. The loop in line 23 processes all points in L_{i-1} that are not bounded to the right by any point in L_i. Since the last L_{2n} contains the points dominated by all approximation fronts of both $\mathcal{A}$ and $\mathcal{A}'$ (Eq. 5), its value of the EAF difference is always zero and we do not need to process it.

Fig. 2 illustrates the output of Algorithm 1 for a given input. The output consists of 7 regions (both positive and negative), and each colour represents a different value of $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\cdot)$.

Given the regions $\mathcal{R}$ computed by Algorithm 1, we now compute the HV^w of a given approximation front A as follows. First, we decompose the HV^w (Eq. (3)) into unweighted and weighted components, i.e., we compute the *HV* corresponding to A and we add

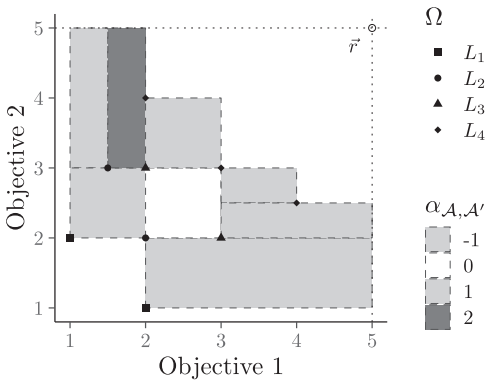


Fig. 2. Example of the output generated by Algorithm 1. The input was $\Omega = \langle L_1, L_2, L_3, L_4\rangle$ and $\vec{r}$. Each shaded area corresponds to a region $R = \langle \vec{l}, \vec{u}, \Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{l})\rangle$ of the objective space that has a non-zero value of $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{l})$ and it is bounded by $\vec{l}$ (bottom left corner) and $\vec{u}$ (top right corner).

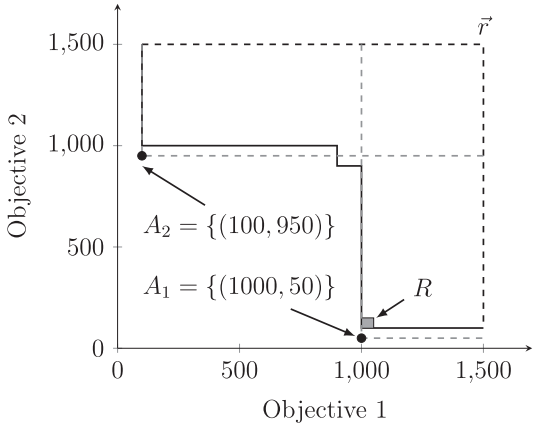


Fig. 3. Example of the effect of β (Eq. (7)). We have two approximation fronts A_1 and A_2, each of them containing only one point. The shaded area corresponds to a region $R = \langle \vec{l} = (1000, 100), \vec{u} = (1050, 150), \Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{l}) = 2\rangle$, which is much smaller in terms of area than $(r_1 - f_1^{\min}) \cdot (r_2 - f_2^{\min})$. If $\beta = 1$ in Eq. (6), then $HV^w(A_1) = 730000 < HV^w(A_2) = 770000$, even though A_1 dominates the weighted region and A_2 does not. If β is scaled as in Eq. (7) with $\psi = 0.5$, then $HV^w(A_1) = 2813225000 > HV^w(A_2) = 770000$, thus A_1 is now considered better than A_2, as expected.

an additional $HV^w_{\mathcal{R}}(A)$ for points in A that dominate the regions in $\mathcal{R}$:

$$HV^w(A) = HV(A) + HV^w_{\mathcal{R}}(A) \cdot \beta \tag{6}$$

where β is the increased weight factor assigned to the area corresponding to $HV^w_{\mathcal{R}}$. The weight factor is defined relative to the total area of the objective space as follows:

$$\beta = (r_1 - f_1^{\min}) \cdot (r_2 - f_2^{\min}) \cdot \psi, \tag{7}$$

where $\vec{r} = (r_1, r_2)$ is the reference point and $f_1^{\min}$ (resp. $f_2^{\min}$) is the minimum value in objective 1 (resp. 2) for any approximation front in $\mathcal{A} \cup \mathcal{A}'$, and $\psi \in [0, 1]$ is a factor that controls the overall weight of the differences.

The motivation behind scaling β with respect to the total area of the objective space is to increase the scale of $HV^w_{\mathcal{R}}(A)$ relative to $HV(A)$, so that there is a strong bias towards the regions in $\mathcal{R}$ even if their total area is much smaller than the total area of the objective space. Such an example is presented in Fig. 3. Parameter ψ allows controlling this scale, thus increasing or decreasing the weight of the regions. This scaling is equivalent to scaling each objective to unit length and using a weight of $\beta \in [0, 1]$ (Brockhoff et al., 2013).

Algorithm 2 computes $HV^w_{\mathcal{R}}(A)$ given an approximation (Pareto) front A and a set of regions $\mathcal{R}$. The algorithm finds which points $\vec{p} \in A$ partially dominate each region $R = \langle \vec{l}, \vec{u}, \Delta\alpha_{\mathcal{A},\mathcal{A}'}(\vec{l})\rangle \in \mathcal{R}$. A region is partially dominated by $\vec{p}$ iff $\vec{p}$ strictly dominates $\vec{u}$, $p_i < u_i$ $\forall i$. Before the start of the algorithm, points in A are sorted by descending p_2 and then by ascending p_1, and regions in $\mathcal{R}$ are sorted by descending u_2 and then by ascending u_1. Algorithm 2 considers the following three cases depicted in Fig. 4: (1) $\vec{p}$ is above $\vec{u}$, (2) $\vec{p}$ strictly dominates $\vec{u}$, and (3) $\vec{p}$ is below and to the right of $\vec{u}$.

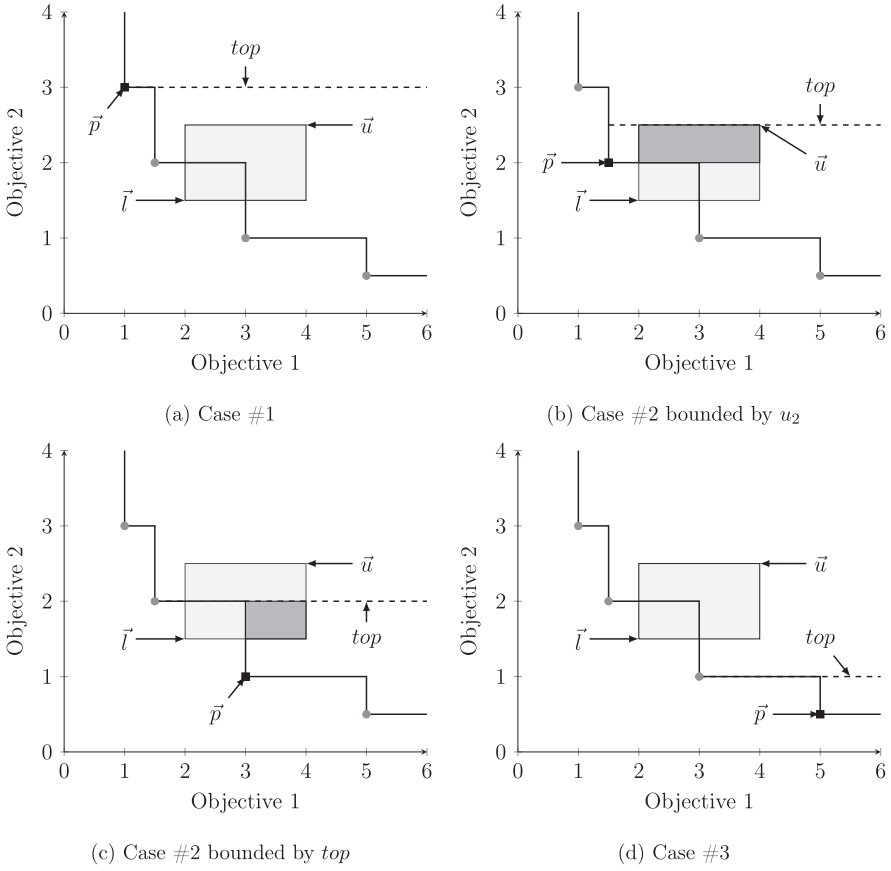
Algorithm 2: Compute $HV^w_{\mathcal{R}}$ (see Eq. 6).

Input: A: an approximation (Pareto) front as a queue sorted by descending p_2 then ascending p_1
$\mathcal{R}$: set of regions sorted by descending u_2 then ascending u_1
Output: $HV^w_{\mathcal{R}}$

```
1  HV^w_R ← 0                                                    // Initialise to zero
2  r ← 1,  ⟨l⃗, u⃗, Δα_{A,A'}(l⃗)⟩ ← R_r ∈ R                        // First region
3  top ← u_2
4  p⃗ ← pop(A)
5  while p_2 ≥ u_2 do                  // Case #1: p⃗ is above all regions
6      top ← p_2
7      if A == ∅ then return HV^w_R
8      p⃗ ← pop(A)
9  repeat
10     repeat
11         if p_1 < u_1 ∧ l_2 < top then   // Case #2: p⃗ strictly
           dominates u⃗
12             width ← u_1 − max{p_1, l_1}
13             height ← min{top, u_2} − max{p_2, l_2}
14             HV^w_R ← HV^w_R + width · height · α(l⃗)
15         else   // Case #3: p⃗ is below and to the right of
           u⃗: nothing
16         r ← r + 1
17         if ∄R_r ∈ R then break // No more regions ⇒ next
           point
18         ⟨l⃗, u⃗, Δα_{A,A'}(l⃗)⟩ ← R_r
19     until p_2 ≥ u_2
20     top ← p_2
21     if A == ∅ then return HV^w_R
22     p⃗ ← pop(A)                                                // Next point
23     r ← 1,  ⟨l⃗, u⃗, Δα_{A,A'}(l⃗)⟩ ← R_r                        // Restart regions
24 return HV^w_R
```

In case (1), the algorithm skips to the next point in A, keeping track of the second coordinate of $\vec{u}$ (*top*). In case (2), we calculate the area of the region that is dominated by $\vec{p}$ (line 14) using the variable *top* to avoid counting the same area again. Fig. 4(c) and (d) show how *top* is used to bound the area and how it is updated if the previous point also belonged to case (2). Finally, in case (3), there is nothing to do but move to the next region. In both cases (2) and (3), we move to the next region until all regions are considered (line 17) or the remaining regions are below $\vec{p}$. After that, the algorithm restarts the sequence of regions and moves to the next point, which will be below the first region, i.e., either case (2) or (3). The algorithm stops when all points in A have been visited.

By using the $HV^w_{\mathcal{R}}$ value returned by Algorithm 2 to compute HV^w (Eq. (6)), we can numerically evaluate the quality of approximation fronts in a way that respects the DM's choice of their preferred EAF differences.



(a) Case #1 (b) Case #2 bounded by u_2

(c) Case #2 bounded by *top* (d) Case #3

Fig. 4. Illustration of all possible cases considered by Algorithm 2.

4. Incorporating DM's preferences into automatic algorithm configuration

The method proposed above can be integrated with an AC method such as irace (López-Ibáñez et al., 2016) by using HV^w (Eq. (6)) as the criterion optimised by irace. However, in practice, the DM will be only interested in looking at EAF differences of already well-performing configurations and it would not be practical that the DM visualises the EAF differences of all pairs of algorithm configurations that are mutually incomparable in terms of Pareto-optimality. Therefore, we discuss here a method to identify the pair of configurations with the largest EAF differences from a given set of configurations.

Intuitively, the most interesting cases for the DM are pairs of configurations that are well-performing, incomparable in terms of Pareto-optimality and that show large EAF differences *in favour of both configurations*, although on different regions of the objective space. On the other hand, pairs of configurations where EAF differences only appear in favour of one configuration are necessarily less interesting, since that configuration is clearly better than the other one. Thus, we propose to calculate, for both configurations of each pair, the sum of the areas of weighted regions (generated from the EAF differences), so that we can select the pair with the maximum of the minimum of both areas among all pairs.

More formally, for each unique pair $\{\mathcal{A}, \mathcal{A}'\}$, where $\mathcal{A}$ and $\mathcal{A}'$ are the multiset of approximation fronts produced by two different algorithm configurations, we first compute the regions of the EAF differences in favour of either side (Algorithm 1), i.e., $\mathcal{R}_{\mathcal{A}}$ for $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\cdot)$ and $\mathcal{R}_{\mathcal{A}'}$ for $-\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\cdot)$. Then, we compute the total area of each set of regions $\mathcal{R}_{\mathcal{A}}$ and $\mathcal{R}_{\mathcal{A}'}$ by calculating (using Algorithm 2) $HV^w_{\mathcal{R}_{\mathcal{A}}}(A^T)$ and $HV^w_{\mathcal{R}_{\mathcal{A}'}}(A^T)$, where $A^T = \{(f_1^{\min}, f_2^{\min})\}$ is an approximation front with a single point that dominates all regions. As our goal is to calculate the area and not the volume of each weighted region, we use a value 1 for the $\Delta\alpha_{\mathcal{A},\mathcal{A}'}(\cdot)$ of all regions when running Algorithm 2. Finally, the DM is shown the EAF differences of the pair $\{\mathcal{A}, \mathcal{A}'\}$ that satisfies:

$$\max_{\forall\{\mathcal{A},\mathcal{A}'\}} \{\min\{HV^w_{\mathcal{R}_{\mathcal{A}}}(A^T), HV^w_{\mathcal{R}_{\mathcal{A}'}}(A^T)\}\} \tag{8}$$

The DM then chooses which side of the differences is their preferred one and, following the method described in the previous section, this choice is translated into a HV^w that will guide a subsequent run of irace.

If the pair selected by the method above does not present EAF differences that are interesting to the DM, or the DM does not feel confident in choosing either side, then the second best pair that satisfies the previous requirement (if such a pair exists) could be shown to the DM. This process could continue until the DM makes a choice or until there are no more possible pairs of configurations to be shown. If we run out of configurations to show, we execute irace once again to generate new algorithm configurations and show them to the DM. There is also the possibility that the DM is interested in a region of the objective space that cannot be attained by any algorithm configuration. In that case, the DM could provide an "ideal" approximation front to be compared with the actual fronts produced by the algorithm configurations returned by irace. Moreover, it is also possible to combine our proposed method with other preference elicitation methods (goals, desirability functions, etc) as long as the combination can be expressed as a weighted hypervolume (Brockhoff et al., 2013). For the sake of brevity, we do not discuss such cases in detail here.

5. Experiments

We test our approach on two different scenarios. The first one is a simple benchmark designed to illustrate our proposal, whereas the second scenario corresponds to a real-world multi-objective production planning problem under uncertainty.

Software implementations. We use irace[2] version 2.3 as the automatic algorithm configurator. In a nutshell, irace (López-Ibáñez et al., 2016) is an optimisation algorithm that uses an elitist iterated racing procedure to dynamically decide how many replications of each algorithm configuration are necessary to evaluate its quality. Parameter configurations are sampled from a probabilistic distribution, which is subsequently updated, as in reinforcement learning, to bias future samples towards the best parameter values found within a single run.

The eaf package[3] (López-Ibáñez et al., 2010) version 1.9 is used to compute EAF differences and visualise them. Both irace and eaf packages are implemented in R (R Development Core Team, 2008) and are publicly available.

Computational environment. All experiments are executed on an Intel(R) Xeon(R) Silver 4114 CPU 2.20 gigahertzes with 46 gigabytes of RAM running Ubuntu GNU/Linux, version 16.04.5. Each run of irace may use up to 10 CPUs in parallel. In our experiments, each irace step between interactions with the DM required 3 minutes on average in the scenario in Section 5.2 and around 280 minutes in the scenario in Section 5.3. The main factors affecting computation time are the budget assigned to irace, the runtime required by each run of the algorithm being configured and the number of parallel CPUs. In practice, some scenarios may allow almost real-time interaction, whereas others may require days of computation time between interactions.

Reproducibility. We provide codes, datasets and scripts to reproduce the experiments in Section 5.2. We also provide a version of the eaf package implementing the algorithms proposed here. The MOEA algorithm in Section 5.3 and the data regarding the real-world problem are not publicly available; however, we provide the data generated by it together with scripts to reproduce the analysis. All supplementary material is available at http://doi.org/10.5281/zenodo.3749288.

5.1. Simulation of the DM's true preferences

In order to validate our proposed interactive approach, we simulate the true preferences of a DM by means of weight distribution functions over the objective space (Brockhoff et al., 2013). Such weight functions allow a DM to specify regions or points of interest in the objective space. For example, a combination of exponential weight functions may be used to capture a preference for extreme values for both objectives:

$$w_{\text{exp}}(\vec{z}) = \sum_{i=1}^{m} \frac{1}{\mu_{\text{exp}}} \lambda e^{-\frac{(z_i - f_i^{\min})}{\mu_{\text{exp}}}} \tag{9}$$

where $\vec{z} = (z_1, \ldots, z_m)$ is a point in the objective space, $\mu_{\text{exp}} \in \mathbb{R}^+$ is the mean of the distribution (inverse of the exponential rate) and $f_i^{\min}$ is the ideal value of objective i.

Similarly, the DM may be interested in a particular point $\vec{g} \in \mathbb{R}^m$ (goal) in the objective space. Preference towards such a goal may be articulated by using as the weight distribution $w_{\text{goal}}(\vec{z})$ a bivariate normal distribution, with mean $\vec{g}$, variance $\sigma = 0.25$ and correlation 1 (Auger et al., 2009).

Given the weight distribution that describes the true preferences of the DM, we can evaluate how much a given approximation front satisfies those preferences by computing the weighted hypervolume according to the weight distribution. Since computing the exact weighted hypervolume for arbitrary weight functions is far from trivial, we instead estimate its value by Monte-Carlo sampling following the method described by Auger et al. (2009) and Brockhoff et al. (2013). The accuracy of this sampling method is independent of the weight function. In our evaluation, we use 10^6 samples to estimate the weighted hypervolume corresponding to the DM's true preferences.

5.2. DM's Preferences in the configuration of the length parameter of *W-RoTS*

We validate our proposed approach by automatically configuring a single parameter of Weighted Robust Taboo Search (W-RoTS). The effect of this parameter on the generated approximation fronts is well understood, thus we can verify whether the choices of the DM lead to the expected parameter values.

W-RoTS. Weighted Robust Taboo Search (W-RoTS) (López-Ibáñez et al., 2006) is a multi-objective local search algorithm that performs multiple runs of the single-objective robust taboo search (RoTS). Each run of RoTS solves a scalarised version (a weighted sum) of the multi-objective problem. The weights are dynamically generated using a strategy called *regular anytime* (Dubois-Lacoste, López-Ibáñez, & Stützle, 2011b) that progressively divides the range of weights into finer "levels" of maximally dispersed weights. In the bi-objective case and assuming the weighted sum $f_\lambda(\vec{x}) = \lambda f_1(\vec{x}) + (1-\lambda) f_2(\vec{x})$, with $\vec{x} \in \mathcal{X}$ and $\lambda \in [0, 1]$, the sequence of weights will be $\lambda \in \{0, 1, 0.5, 0.25, 0.75, 0.125, 0.375, 0.625, 0.875, \ldots\}$. That is, additional scalarisations will progressively fill gaps in the current approximation of the Pareto front. The computation time consumed by each run of RoTS for a given weight is controlled by a parameter ℓ (taboo search length), with higher values devoting more time to optimising the scalarised problem and, typically, returning a higher quality solution. If W-RoTS stops after a given CPU time, then higher values of ℓ will decrease the number of different scalarised problems that may be solved, thus leaving gaps in the resulting approximation front, but obtaining very high-quality solutions for the weights used; whereas lower values of ℓ will allow tackling more scalarised problems, filling the gaps yet returning lesser quality solutions for each of them. Thus, the value of ℓ has a predictable effect on the differences between the generated approximation fronts.

Experimental setup. We run W-RoTS with a maximum CPU-time of 1 second with configurations $\ell = 1$ (P1C1) and $\ell = 100$ (P1C2) on an instance of the bi-objective quadratic assignment problem of size 25 and correlation between the objectives of -0.75. Each run is replicated 30 times with different random seeds. The EAF differences (Fig. 5) demonstrate the expected behaviour: P1C1 ($\ell = 1$) obtains better results along the front except for a few specific regions, whereas P1C2 ($\ell = 100$) returns higher-quality solutions at the extremes of the front. In this section, all hypervolume (weighted or not) calculations use $\vec{r} = (1779390, 1650208)$ and the ideal point is $f^{\min} = (1264374, 1166290)$.

DM preferences. We can simulate a preference of the DM for the centre of the front by means of the weight distribution w_{goal} (Section 5.1) with the goal (1 264 374, 1 166 290), which is the point we found after a few preliminary experiments. According to this preference, the mean weighted hypervolume of P1C1 ($\ell = 1$) is larger than the one of P1C2 ($\ell = 100$) (see Table 1), thus P1C1 matches more closely the DM's preferences in this case. We can also simulate a preference of the DM for the extreme values of either objective by using an exponential weight distribution w_{exp} (Eq. (9)) with $\mu_{\text{exp}} = 0.1$. According to this preference, the mean weighted hypervolume of P1C1 ($\ell = 1$) is smaller than the one of

[2] https://cran.r-project.org/package=irace.
[3] http://lopez-ibanez.eu/eaftools.

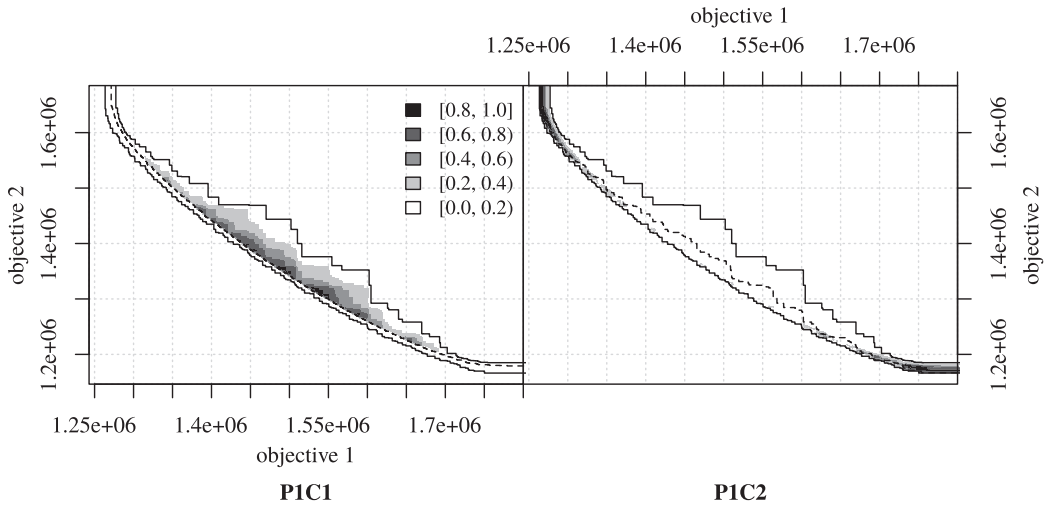


Fig. 5. EAF differences between two W-RoTS configurations, P1C1 ($\ell = 1$) and P1C2 ($\ell = 100$). Each configuration is run 30 times and each run stops after 1 CPU-second.

Table 1
Mean weighted hypervolume according to weight distributions w_{goal} and w_{exp} of the approximation fronts generated by various configurations of W-RoTS.

	P1C1	P1C2	P1iraceC1	P1iraceC2
w_{goal}	12 655 754 610	10 348 549 426	12 833 541 277	12 503 730 101
w_{exp}	45 875 372 184	49 194 740 255	50 140 237 619	52 005 727 121

P1C2 ($\ell = 100$) (see Table 1), validating that P1C2 matches more closely the preference of the DM for the extremes of the front. Boxplots of the weighted hypervolume according to both weight distributions (Figure 9) show significant differences between P1C1 and P2C2.

Automatic configuration. We now evaluate what happens if irace tunes the parameter ℓ after the DM selects either side of Fig. 5 as their preferred differences. First, we compute the regions corresponding to each side using Algorithm 1. Then, Algorithm 2 uses these regions to define two HV^w functions (Eq. (6)) with $\psi = 0.1$. These HV^ws become the optimisation criterion guiding two different settings of irace, denoted as P1iraceC1 and P1iraceC2, for the EAF differences in favour of P1C1 and P1C2, respectively. Each setting of irace is configuring parameter ℓ with an integer domain of [1,200] for a maximum budget of 500 runs of W-RoTS. We run each irace setting 30 times with different random seeds, keeping only the best configuration obtained by each run. As a result, we obtain 30 configurations of ℓ for P1iraceC1 and for P1iraceC2 with a mean value of ℓ of 3.8 (with a standard deviation of 1.06) and 23.5 (standard deviation is 8.77), respectively. Thus the different HV^ws clearly lead irace towards different values of ℓ. In order to visualise the EAF differences between these configurations, we run each of them 30 times with different random seeds.

Analysis of results. The effect of the DM's choice is visible when comparing the configurations obtained by P1iraceC1 and those obtained by P1iraceC2 (Fig. 6) in relation to the original decision of the DM (Fig. 5). There are 900 approximation fronts (30 runs of 30 configurations) on each side of Fig. 6. The plots show that the configurations returned by the two settings of irace are not only clearly different, but also that they are different precisely in the regions preferred by the DM.

The improvement obtained by irace with respect to the configuration chosen by the DM is visible in Figs. 7 and 8. Fig. 7 shows the EAF differences between P1C1 ($\ell = 1$) and the configuration with $\ell = 4$, which is the closest integer value to the mean ℓ returned by P1iraceC1 after the DM chose P1C1 ($\ell = 1$). The plot shows that there are clear differences along the front and especially on the extremes in favour of $\ell = 4$, and thus irace was able to find a configuration that outperforms the one chosen by the DM, while at the same time respecting DM's preferences. This conclusion is even stronger in Fig. 8 when comparing P1C2 ($\ell = 100$) and the configuration with $\ell = 24$, which is the closest integer value to the mean ℓ returned by P1iraceC2 when the DM chose P1C2 ($\ell = 100$). Again in this case, irace not only improves the configuration chosen by the DM, but the improvement is focused on the region of the objective space preferred by the DM.

We also verify that irace is indeed generating configurations that match the true preferences of the DM by calculating the estimated weighted hypervolume according to both the weight distributions w_{goal} (preference for the centre of the front) and w_{exp} (preference for the extremes). Table 1 shows the mean values of the weighted hypervolume for each weight distribution. In the case of w_{goal}, the value corresponding to P1iraceC1 is larger than the one of P1C1, which implies that irace was successful in improving the configuration chosen by the DM (P1C1). Moreover, the value corresponding to P1iraceC1 is larger than the value of P1iraceC2, which implies that irace obtained configurations more closely matching the true preference of the DM when guided by the HV^w computed after the DM chose the EAF differences in favour of P1C1 (consistent with w_{goal}) rather than those in favour of P1C2 (consistent with w_{exp}). A similar analysis can be made for w_{exp}, that is, P1iraceC2 obtains the largest mean value, improving over P1C2, which was the configuration chosen by the DM. Moreover, P1iraceC2 obtains larger values according to w_{exp} than P1iraceC1, because the former was guided by the HV^w computed from the EAF differences that were consistent with w_{exp}. All the observed differences are statistically significant according to the non-parametric Wilcoxon rank-sum test, as shown in Fig. 9. The fact that the configurations generated by irace optimise the weighted hypervolume that is consistent with the true preferences of the DM confirms that the proposed preference elicitation method works as expected.

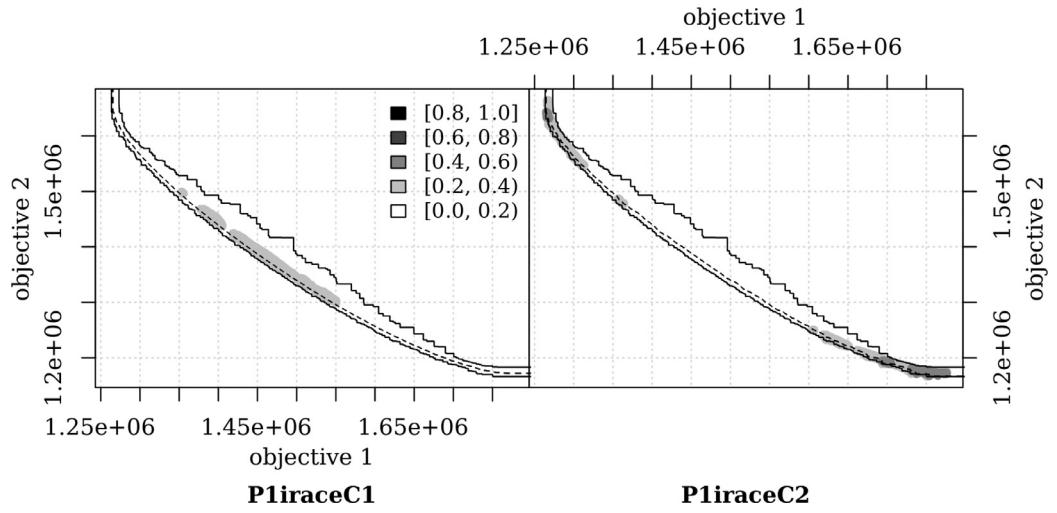


Fig. 6. EAF differences between the W-RoTS configurations returned by P1iraceC1 and by P1iraceC2. Each side contains data from the approximation fronts generated by 30 runs of 30 configurations.

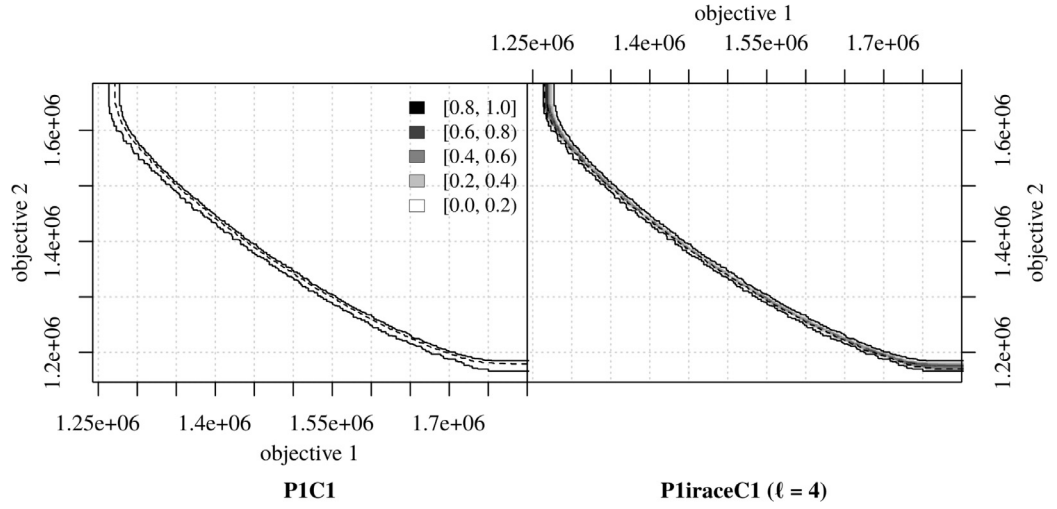


Fig. 7. EAF differences between configuration P1C1 ($\ell = 1$) and the configuration with $\ell = 4$, which is the closest integer value to the mean ℓ returned by P1iraceC1.

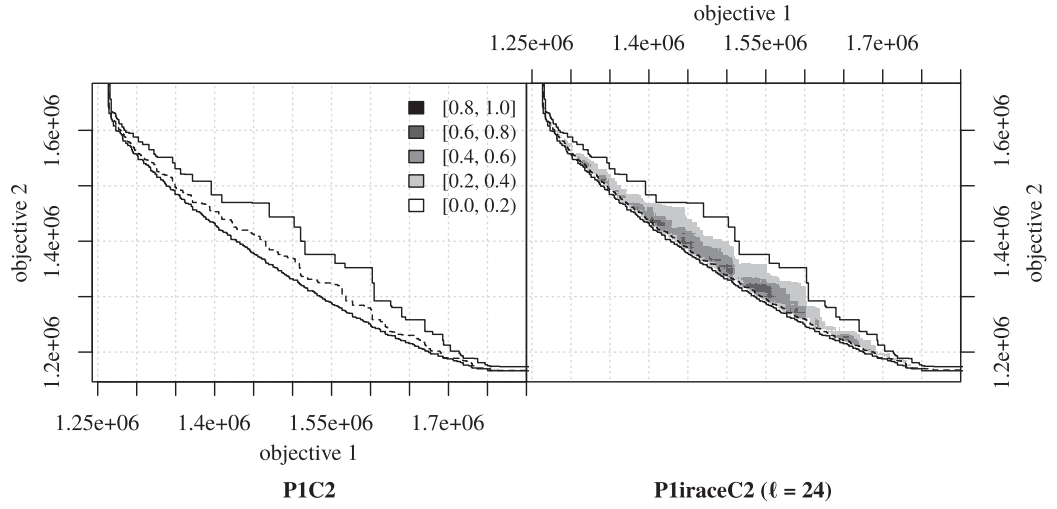
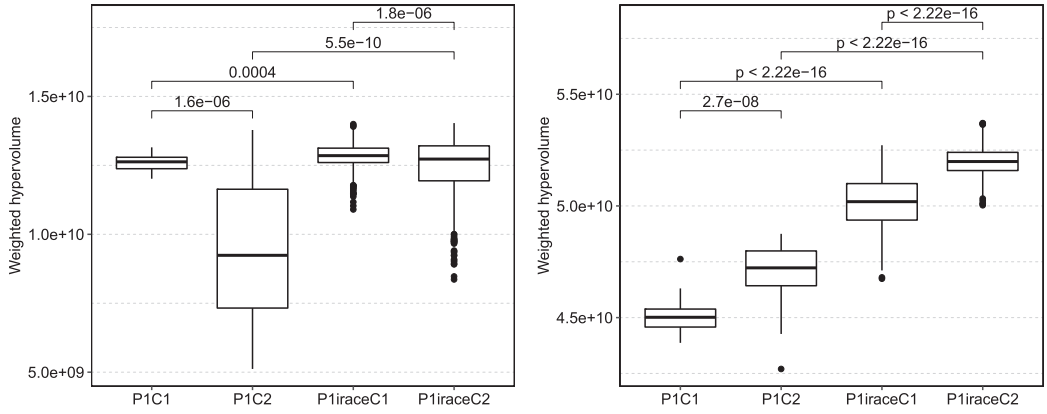


Fig. 8. EAF differences between configuration P1C2 ($\ell = 100$) and the configuration with $\ell = 24$, which is the closest integer value to the mean ℓ returned by P1iraceC2.



(a) w_{goal} with $\vec{g} = (1264374, 1166290)$ (b) w_{exp} ($\mu_{\text{exp}} = 0.1$)

Fig. 9. Boxplots of weighted hypervolume values according to two different weight distributions: w_{goal} (left) and w_{exp} (right). The same approximation fronts are evaluated on both subfigures. Sets P1C1 and P2C2 contain 30 fronts each, i.e., 30 runs of one configuration of W-RoTS. Sets P1iraceC1 and P2iraceC2 contain 900 fronts each, 30 runs of the best configuration of W-RoTS returned by each of the 30 independent runs of irace i.e., 30 runs of each configuration of W-RoTS returned by 30 runs of irace. Pairs of sets joined by a line were compared using the Wilcoxon rank-sum (MannWhitney U) test, and the resulting p-value (adjusted by Holm's method for multiple comparisons) is shown above the line.

5.3. Re-configuring a multi-objective optimiser for a real-world problem according to DM's preferences

We consider now a more realistic and challenging task: the (re-)configuration of a multi-objective evolutionary algorithm (MOEA) that tackles a real-world, multi-objective, big bucket, multi-product, multi-level (sub-products), capacitated (constraints are considered) production planning problem within a failure-prone batch manufacturing system specialised on cleaning products, conformed by multiple production lines with insufficient capacity to fully cover demand requirements. In this system, 31 products can be manufactured across 7 independent production lines. Given that some products can be manufactured in multiple production lines, the optimisation problem should determine the values of 41 decision variables. Here, a variable indicates the number of product lots to be manufactured in a specific production line. Apart from the intrinsic uncertainty present in the system due to the occurrence of production line failures, all capacity, labour and demand constraints need to be considered as well during the optimisation process, in order to set the values of those decision variables. The existing algorithm combines a MOEA with a discrete-event simulation model. See Diaz et al. (2017, 2018) for more details.

The effectiveness of this algorithm was demonstrated against other approaches, after tuning its parameters using irace guided by the (unweighted) *HV* (Diaz, Handl, & Xu, 2018). As discussed above, the *HV* implicitly incorporates unstated preferences into the AC process. Here, we demonstrate how our proposal allows DMs to explicitly incorporate their actual preferences into the AC process, leading to different configurations of the algorithm that respond to those preferences on a realistic problem.

Experimental setup. Compared to W-RoTS, the algorithm in this section has a larger number of parameters, with more complex domains, that need to be configured, as shown in Table 2. Moreover, contrary to W-RoTS, it is not obvious how to set those parameters to satisfy different DM's preferences, and such alternative parameter configurations may not even exist. Therefore, unlike Section 5.2, where the DM's choice was based upon predefined configurations that we knew in advance evidenced different outcomes, here, we first use irace to find high-performing configurations by optimising the *HV*. In this section, all hypervolume (weighted or not) calculations use $\vec{r} = (-547956, 76227)$ and the ideal point is $f^{\min} = (-752658, 0)$.

Table 2
Target algorithm parameters used by irace. See Diaz et al. (2017, 2018) for details about these parameters.

Parameter	Type	Domain
Crossover probability	real	(0, 1)
Mutation probability	real	(0, 1)
Selection operator	categorical	{NSGA-II, SPEA2, random}
Number of seeds*	categorical	{0, 2, 4, 8, 16, 32}
Number of fitness evaluations**	categorical	{10, 30}
Crowding degree for crossover	integer	[0, 100]
Crowding degree for mutation	integer	[0, 100]
Seeding operator	categorical	{1, 2, 3}

* used in the initial population.
** used to compute objective values during the optimisation procedure.

Next, we use the criterion (Eq. (8)) described in Section 4 to select from the elite configurations returned by irace, the pair that evidences the highest EAF differences. These EAF differences are visually shown to the DM, who then will select one of the configurations (or possibly ask the system to show a different pair). Following the approach proposed here, the DM's choice will be converted to a HV^w that guides further runs of irace. This would be the usual process in any realistic application scenario.

DM preferences. The true preferences of the DM are simulated by means of two w_{goal} weight distributions with different goals: $w_{\text{goal,C1}}$ uses the goal $(-700000, 5000)$, while $w_{\text{goal,C2}}$ uses $(-780000, 76200)$. These goals correspond to two extremes of the objective space, as shown in Fig. 10. This visualisation (without the goals) would be the one shown to the DM, as explained next.

Automatic configuration. We first run irace guided by the (unweighted) *HV* and a budget of 1000 runs of the MOEA. The pair of configurations returned by irace showing the highest EAF differences (Eq. (8)) are denoted here P2C1 and P2C2. At the time irace stops, each of them has been run 16 times, thus there are 16 approximation fronts generated by each of them. Their EAF differences are shown in Fig. 10. Their mean true weighted hypervolume values according to $w_{\text{goal,C1}}$ and $w_{\text{goal,C2}}$ are shown in Table 3. As expected, P2C1 (resp. P2C2) will be chosen by the DM if the true preferences are simulated according to $w_{\text{goal,C1}}$ (resp. $w_{\text{goal,C2}}$). Although in practice the DM will only choose one of them, here we run two settings of irace, each setting guided by the HV^w computed from the regions in favour of each choice. Each run of irace has a budget of 1000 MOEA runs, and we repeat each run 30

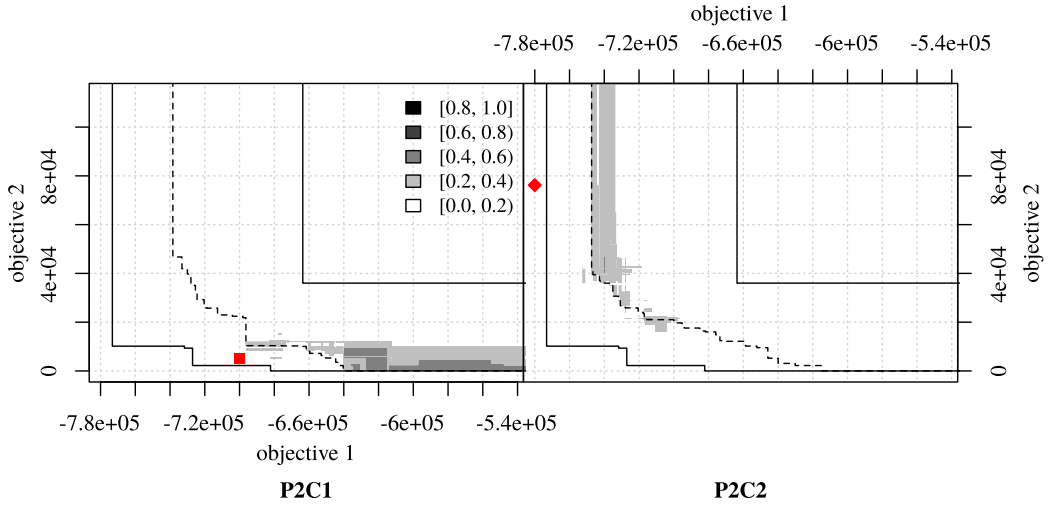


Fig. 10. EAF differences between two parameter configurations P2C1 and P2C2. These configurations were obtained by one run of irace guided by the unweighted hypervolume. Each side contains data from the approximation fronts generated by 16 MOEA runs. Points (■ and ♦) on the left and the right show the location of the goals corresponding to $w_{\text{goal,C1}}$ and $w_{\text{goal,C2}}$, respectively.

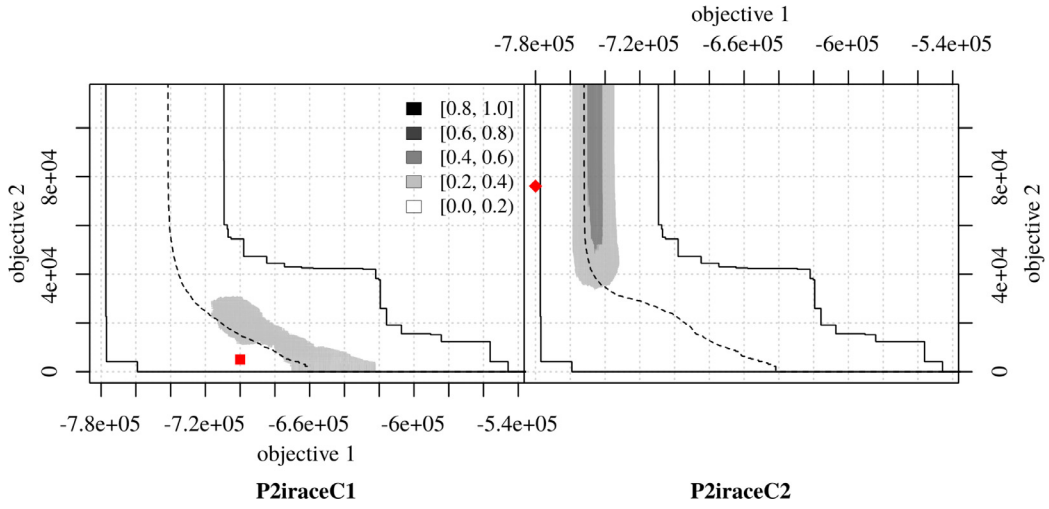


Fig. 11. EAF differences between P2iraceC1 and P2iraceC2. Each side contains data from the approximation fronts generated by 30 runs of the best configuration returned by each of 30 runs of irace.

Table 3
Mean weighted hypervolume according to weight distributions $w_{\text{goal,C1}}$ and $w_{\text{goal,C2}}$ of the approximation fronts generated by various configurations of the MOEA.

	P2C1	P2C2	P2iraceC1	P2iraceC2
$w_{\text{goal,C1}}$	5 687 432 215	4 810 012 874	6 247 075 350	5 365 817 542
$w_{\text{goal,C2}}$	3 530 670 353	3 871 195 429	3 612 257 671	4 629 175 274

times with different random seeds. P2iraceC1 denotes the 30 configurations returned by irace after the DM chooses P2C1, whereas P2iraceC2 denotes the ones returned after the DM chooses P2C2. Each of these configurations is executed 30 times with different random seeds.

Analysis of results. If we compare all configurations obtained by P2iraceC1 and P2iraceC2 (Fig. 11) and contrast those EAF differences to the ones initially shown to the DM (Fig. 10), it is fairly clear that the configurations returned by each irace variant are performing better precisely on the regions of the objective space that correspond to the choice made by the DM. It is important to understand that the regions where P2iraceC1 performs better with respect to P2iraceC2 (Fig. 11) actually dominate the regions where P1C1 performed better with respect to P2C2 (Fig. 10), which is the desired behaviour according to Pareto-optimality.

Next we compare P2C1 and an "average" configuration from P2iraceC1. In particular, we compute the mean HV^w of each configuration in P2iraceC1 using the regions in favour of P2C1, which was the HV^w used by irace when generating P2iraceC1, and select as the configuration with the closest mean HV^w to the average. Fig. 12 shows the EAF differences between P2C1 and this "average" configuration from P2iraceC1. Although the EAF differences do not show an obvious improvement, the configuration returned by irace does have a larger HV^w value than P2C1. In terms of the true preferences of the DM, the configuration returned by irace is better near the DM's preferred goal ($w_{\text{goal,C1}}$). Table 3, which presents the mean values of the weighted hypervolume for each weight

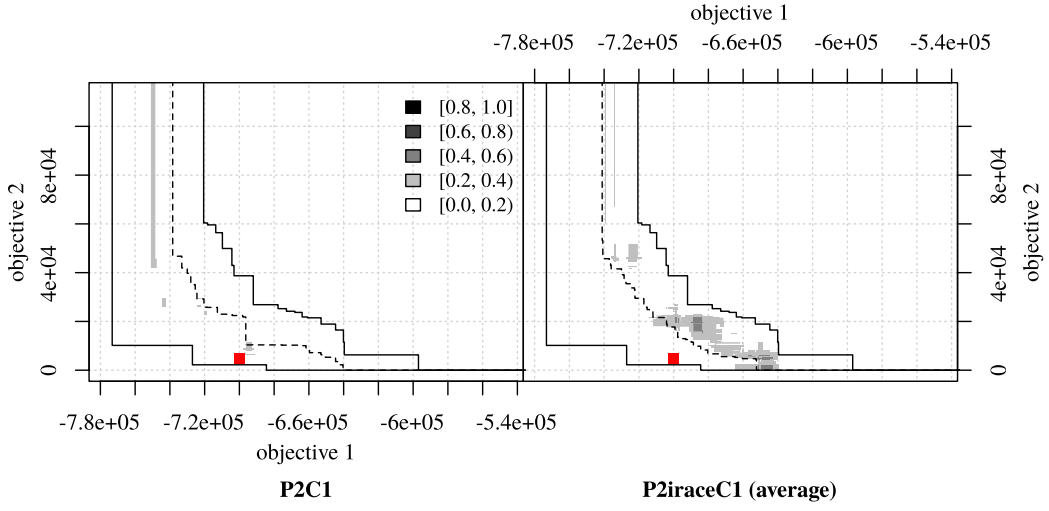


Fig. 12. EAF differences between P2C1, which has a mean HV^w of 5.53e+19, and the configuration that has the closest mean HV^w (5.95e+19) to the average (5.95e+19) across the mean HV^w values of all configurations returned by P2iraceC1. HV^w values are computed using the regions in favour of P2C1 in Fig. 10, which was chosen by the DM according to the goal corresponding to $w_{\text{goal,C1}}$ (■).

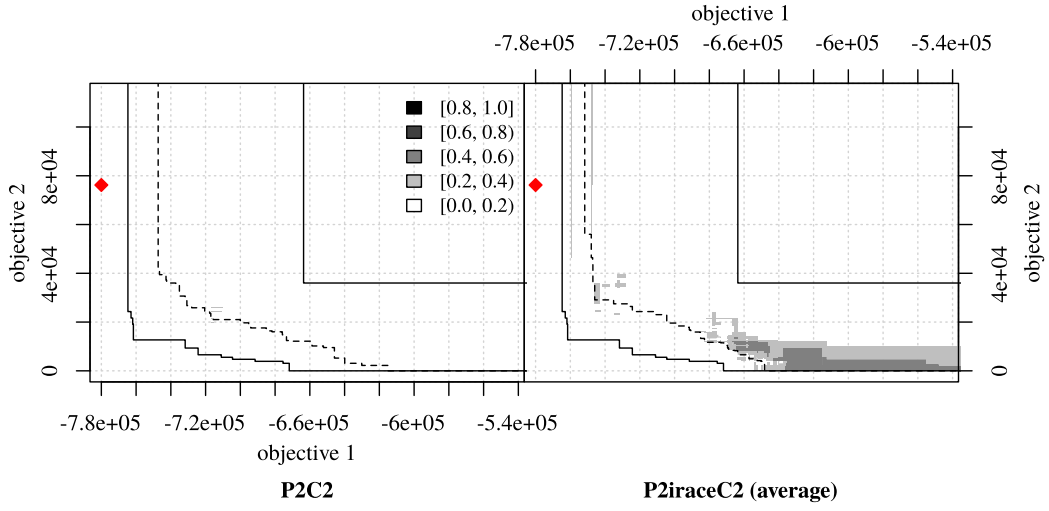
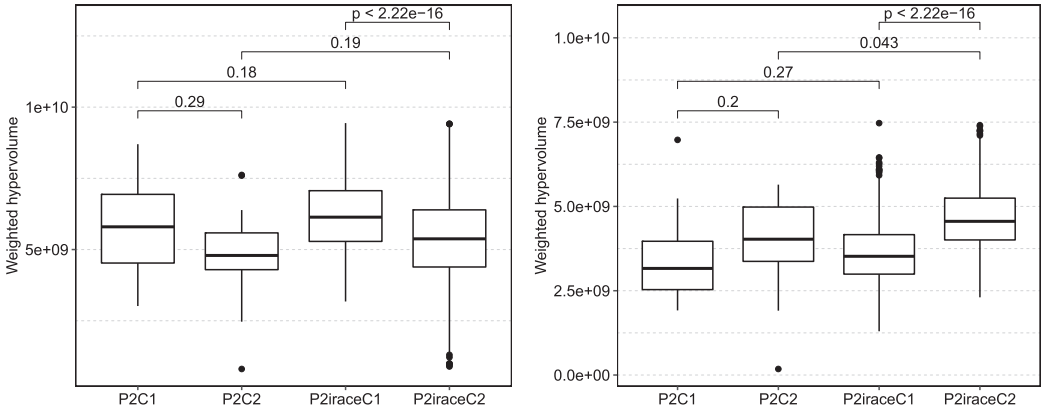


Fig. 13. EAF differences between P2C2, which has a mean HV^w of 5.44e+19, and the configuration which has the closest mean HV^w (5.60e+19) to the average (5.61e+19) across the mean HV^w values of all configurations returned by P2iraceC2. HV^w values are computed using the regions in favour of P2C2 in Fig. 10, which was chosen by the DM according to the goal corresponding to $w_{\text{goal,C2}}$ (♦).

distribution, shows the configurations in P2iraceC1 reach a better mean weighted hypervolume value according to $w_{\text{goal,C1}}$ than P2C1. However, the boxplots of P2C1 and P2iraceC1 in Fig. 14a, which illustrates the distribution of those hypervolume values by means of boxplots and the p-values resulting from Wilcoxon rank-sum tests between various pairs, shows that there is a large variance and the differences are not statistically significant. There may be no configuration of the MOEA that significantly improves P2C1 with respect to $w_{\text{goal,C1}}$ or irace was not able to find it within the budget given. Importantly, irace did not trade-off a regression with respect to $w_{\text{goal,C1}}$ for an improvement in some other region, as may be the case with the non-weighted *HV*.

A similar conclusion is obtained when comparing P2C2 and the corresponding "average" configuration from P2iraceC2. In this case, we compute the mean HV^w of each configuration in P2iraceC2 using the regions in favour of P2C2, which was the HV^w used by irace when generating P2iraceC2. Fig. 13 shows that this "average" configuration is better than the one chosen by the DM on a larger region of the objective space; however, that region is not near the DM's preferred goal ($w_{\text{goal,C2}}$). Nevertheless, according to their mean values of the weighted hypervolume given by Table 3, the configurations returned by irace (P2iraceC2) obtain a better mean weighted hypervolume value according to $w_{\text{goal,C2}}$ than P2C2. Moreover, according to Fig. 14b, which presents boxplots of those hypervolume values together with the p-values of Wilcoxon rank-sum tests between various pairs, the difference between P2iraceC2 and P2C2 is statistically significant at a 95% confidence level (*p*-value$= 0.04 < 0.05$). In other words, even though irace was not able to improve over P2C2 near the goal preferred by the DM, irace was still able to improve elsewhere without regressing with respect to $w_{\text{goal,C2}}$. If irace cannot find a configuration that improves on the regions preferred by the DM, the HV^w still allows finding improvements in other regions of the objective space.

Finally, we verify that irace is indeed generating configurations that match the true preferences of the DM by calculating the estimated weighted hypervolume according to two different w_{goal} weight distributions, one favouring objective 1 ($w_{\text{goal,C1}}$) and the other favouring objective 2 ($w_{\text{goal,C2}}$). As expected, Table 3 shows that P2iraceC1 obtains the highest values with respect to $w_{\text{goal,C1}}$, whereas P2iraceC2 does the same with respect to $w_{\text{goal,C2}}$. As



(a) $w_{\text{goal,C1}}$ with $\vec{g} = (-700\,000, 5000)$ (b) $w_{\text{goal,C2}}$ with $\vec{g} = (-780\,000, 76\,200)$

Fig. 14. Boxplots of weighted hypervolume values according to two w_{goal} weight distributions with different goals. The same approximation fronts are evaluated on both subfigures. Sets P2C1 and P2C2 contain 16 fronts each, i.e., 16 runs of one configuration of the MOEA. Sets P2iraceC1 and P2iraceC2 contain 900 fronts each, i.e., 30 runs of the best configuration of the MOEA returned by each of the 30 independent runs of irace. Pairs of sets joined by a line were compared using the Wilcoxon rank-sum (MannWhitney U) test, and the resulting p-value (adjusted by Holm's method for multiple comparisons) is shown above the line.

discussed above, according to Fig. 14, the improvement of P2iraceC1 over P2C1 with respect to $w_{\text{goal,C1}}$ is not statistically significant (p-value$= 0.18$), while the improvement of P2iraceC1 over P2C1 with respect to $w_{\text{goal,C2}}$ is close to the 95% confidence level (p-value$= 0.043 < 0.05$). However, the differences between P1iraceC1 and P2iraceC2 are statistically significant for both weight distributions and, for each of them, the difference is in favour of the configurations generated by irace when guided by the choice made by the DM (P1iraceC1 for $w_{\text{goal,C1}}$ and P2iraceC2 for $w_{\text{goal,C2}}$). In other words, the configurations generated by irace for each choice are clearly specialised towards satisfying each of the two alternative preferences of the DM, which confirms the effectiveness of the proposed preference elicitation method.

6. Conclusions and future work

In this paper, we have proposed a procedure to elicit DM's preferences from a visualisation of EAF differences by converting these differences into a weighted hypervolume (HV^w). The HV^w does not contradict the Pareto-optimality criterion and, at the same time, assigns a better quality to approximation fronts that attain regions of the objective space that are preferred by the DM.

We believe that eliciting preferences in this way is far simpler and more intuitive than directly defining a weight function, in particular, in the context of automatic algorithm configuration, where many approximation fronts returned by different configurations of stochastic bi-objective optimisers must be compared.

We demonstrate for a well-understood benchmark (the configuration of the length parameter of W-RoTS) that an AC method (irace) guided by the HV^w generated from our proposed approach is able to find improved algorithm configurations that satisfy the DM's preferences. In addition, we have shown that the proposed approach also works on a more realistic and challenging scenario arising in the context of a real-world production planing problem under uncertainty, where there is no prior knowledge about the algorithm configurations that would satisfy the DM's preferences. Our analysis shows that the resulting algorithm configurations not only improve over the ones generated by irace when guided by the unweighted HV, but also perform clearly better upon the preferred regions of the objective space, thus confirming that irace is guided by the DM's preferences.

As far as we know, our approach is the first to incorporate DM's preferences into the AC of multi-objective optimisers. Although we have focused here on irace, our proposed approach may be integrated into other AC methods.

We have limited ourselves to bi-objective problems, for which the visualisation of the EAFs is more easily understood. Nevertheless, the main ideas of our approach are applicable to problems with three objectives, although in that case, more complex and interactive visualisations should be made available to the DM (Tušar & Filipič, 2014). For higher number of objectives, it is an open question how to effectively visualise EAF differences and there is no reason to expect that the most useful methods for multi-objective optimisers (2 and 3 objectives) must be the same as for many-objective optimisers (4 and more objectives), nor vice versa.

Our proposed approach works well when the training problem instances used in the AC process are fairly homogeneous in terms of the shape of the Pareto front, so that preferences obtained on one problem instance can be translated to a different one. If this is not the case, either further interactions with the DM would be required for different classes of instances, or we would need a way to map the choice of EAF differences from one Pareto front shape to a different one. Moreover, it is also an open question how to adjust the HV^w when the DM wishes to refine their preferences through multiple sequential interactions. Thus, future work will focus on studying ways of aggregating the DM's preferences elicited by means of EAF differences across multiple interactions and across differently-shaped training problem instances.

Finally, we have only considered here the configuration of already designed multi-objective optimisers, yet the procedure is directly applicable to a wide range of automatic algorithm design tasks (Bezerra et al., 2016) and to other contexts where the DM is closely involved in the algorithm design (Trianni & López-Ibáñez, 2015).

Acknowledgement

We would like to thank the anonymous referees for all the comments and suggestions given, which help us to substantially improve our manuscript.

References

Auger, A., Bader, J., Brockhoff, D., & Zitzler, E. (2009). Articulating user preferences in many-objective problems by sampling the weighted hypervolume. In F. Rothlauf (Ed.), *Proceedings of the genetic and evolutionary computation conference, GECCO* (pp. 555–562). New York, NY: ACM Press.

Auger, A., Bader, J., Brockhoff, D., & Zitzler, E. (2012). Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications. *Theoretical Computer Science, 425*, 75–103. https://doi.org/10.1016/j.tcs.2011.03.012.

Bezerra, L. C. T., López-Ibáñez, M., & Stützle, T. (2016). Automatic component-wise design of multi-objective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation, 20*, 403–417. https://doi.org/10.1109/TEVC.2015.2474158.

Bezerra, L. C. T., López-Ibáñez, M., & Stützle, T. (2020a). Automatic configuration of multi-objective optimizers and multi-objective configuration. In T. Bartz-Beielstein, B. Filipič, P. Korošec, & E. G. Talbi (Eds.), *High-performance simulation-based optimization* (pp. 69–92). Cham, Switzerland: Springer International Publishing. https://doi.org/10.1007/978-3-030-18764-4_4.

Bezerra, L. C. T., López-Ibáñez, M., & Stützle, T. (2020b). Automatically designing state-of-the-art multi- and many-objective evolutionary algorithms. *Evolutionary Computation, 28*, 195–226. https://doi.org/10.1162/evco_a_00263.

Birattari, M. (2009). Tuning Metaheuristics: A Machine Learning Perspective. *Studies in Computational Intelligence*: 197. Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-00483-4.

Blot, A., Hoos, H. H., Jourdan, L., Kessaci-Marmion, M. E., & Trautmann, H. (2016). MO-ParamILS: A multi-objective automatic algorithm configuration framework. In P. Festa, M. Sellmann, & J. Vanschoren (Eds.), *Proceedings of the 10th international conference on learning and intelligent optimization LION 10*. In *Lecture Notes in Computer Science: vol. 10079* (pp. 32–47). Cham, Switzerland: Springer.

Brockhoff, D., Bader, J., Thiele, L., & Zitzler, E. (2013). Directed multiobjective optimization based on the weighted hypervolume indicator. *Journal of Multi-Criteria Decision Analysis, 20*, 291–317. https://doi.org/10.1002/mcda.1502.

Coello Coello, C. A. (2006). Evolutionary multi-objective optimization: a historical view of the field. *IEEE Computational Intelligence Magazine, 1*, 28–36.

Coello Coello, C. A. (2015). Multi-objective evolutionary algorithms in real-world applications: Some recent results and current challenges. In *Advances in evolutionary and deterministic methods for design, optimization and control in engineering and sciences* (pp. 3–18). Springer. https://doi.org/10.1007/978-3-319-11541-2_1.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley.

Diaz, J. E., Handl, J., & Xu, D. L. (2017). Evolutionary robust optimization in production planning: interactions between number of objectives, sample size and choice of robustness measure. *Computers & Operations Research, 79*, 266–278. https://doi.org/10.1016/j.cor.2016.06.020.

Diaz, J. E., Handl, J., & Xu, D. L. (2018). Integrating meta-heuristics, simulation and exact techniques for production planning of a failure-prone manufacturing system. *European Journal of Operational Research, 266*, 976–989. https://doi.org/10.1016/j.ejor.2017.10.062.

Dubois-Lacoste, J., López-Ibáñez, M., & Stützle, T. (2011a). A hybrid TP + PLS algorithm for bi-objective flow-shop scheduling problems. *Computers & Operations Research, 38*, 1219–1236. https://doi.org/10.1016/j.cor.2010.10.008.

Dubois-Lacoste, J., López-Ibáñez, M., & Stützle, T. (2011b). Improving the anytime behavior of two-phase local search. *Annals of Mathematics and Artificial Intelligence, 61*, 125–154. https://doi.org/10.1007/s10472-011-9235-0.

Dubois-Lacoste, J., López-Ibáñez, M., & Stützle, T. (2015). Anytime Pareto local search. *European Journal of Operational Research, 243*, 369–385. https://doi.org/10.1016/j.ejor.2014.10.062.

Fonseca, C. M., & Fleming, P. J. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. In H. M. Voigt (Ed.), *Parallel problem solving from nature, PPSN IV*. In *Lecture Notes in Computer Science: vol. 1141* (pp. 584–593). Springer, Heidelberg, Germany.

Fonseca, C. M., Grunert da Fonseca, V., & Paquete, L. (2005). Exploring the performance of stochastic multiobjective optimisers with the second-order attainment function. In C. A. Coello Coello, A. H. Aguirre, & E. Zitzler (Eds.), *Evolutionary multi-criterion optimization, EMO 2005*. In *Lecture Notes in Computer Science: vol. 3410* (pp. 250–264). Springer, Heidelberg, Germany. https://doi.org/10.1007/978-3-540-31880-4_18.

Fonseca, C. M., Guerreiro, A. P., López-Ibáñez, M., & Paquete, L. (2011). On the computation of the empirical attainment function. In R. H. C. Takahashi (Ed.), *Evolutionary multi-criterion optimization, EMO*. In *Lecture Notes in Computer Science: 6576* (pp. 106–120). Springer, Heidelberg, Germany. https://doi.org/10.1007/978-3-642-19893-9_8.

Grunert da Fonseca, V., & Fonseca, C. M. (2010). The attainment-function approach to stochastic multiobjective optimizer assessment and comparison. In T. Bartz-Beielstein, M. Chiarandini, L. Paquete, & M. Preuss (Eds.), *Experimental methods for the analysis of optimization algorithms* (pp. 103–130). Berlin, Germany: Springer.

Grunert da Fonseca, V., Fonseca, C. M., & 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, & D. Corne (Eds.), *Evolutionary multi-criterion optimization, EMO 2001*. In *Lecture Notes in Computer Science: vol. 1993* (pp. 213–225). Springer, Heidelberg, Germany. https://doi.org/10.1007/3-540-44719-9_15.

Hoos, H. H. (2012). Automated algorithm configuration and parameter tuning. In Y. Hamadi, E. Monfroy, & F. Saubion (Eds.), *Autonomous search* (pp. 37–71). Berlin, Germany: Springer. https://doi.org/10.1007/978-3-642-21434-9_3.

Jaszkiewicz, A. (2018). Many-objective Pareto local search. *European Journal of Operational Research, 271*, 1001–1013. https://doi.org/10.1016/j.ejor.2018.06.009.

Jaszkiewicz, A., Ishibuchi, H., & Zhang, Q. (2011). Multiobjective memetic algorithms. In *Handbook of memetic algorithms*. In *Studies in Computational Intelligence: vol. 379* (pp. 201–217). Springer.

Knowles, J. D. (2005). A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In A. Abraham, & M. Paprzycki (Eds.), *Proceedings of the 5th international conference on intelligent systems design and applications* (pp. 552–557). https://doi.org/10.1109/ISDA.2005.15.

Knowles, J. D., & Corne, D. (2005). Memetic algorithms for multiobjective optimization: issues, methods and prospects. In H. W. E., S. J. E., & K. N. (Eds.), *Recent advances in memetic algorithms*. In *Studies in Fuzziness and Soft Computing: vol. 166* (pp. 313–352). Berlin, Heidelberg: Springer. https://doi.org/10.1007/3-540-32363-5_14.

Knowles, J. D., Thiele, L., & 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

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Stützle, T., & Birattari, M. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives, 3*, 43–58. https://doi.org/10.1016/j.orp.2016.09.002.

López-Ibáñez, M., Paquete, L., & Stützle, T. (2006). Hybrid population-based algorithms for the bi-objective quadratic assignment problem. *Journal of Mathematical Modelling and Algorithms, 5*, 111–137. https://doi.org/10.1007/s10852-005-9034-x.

López-Ibáñez, M., Paquete, L., & Stützle, T. (2010). Exploratory analysis of stochastic local search algorithms in biobjective optimization. In T. Bartz-Beielstein, M. Chiarandini, L. Paquete, & M. Preuss (Eds.), *Experimental methods for the analysis of optimization algorithms* (pp. 209–222). Berlin, Germany: Springer. https://doi.org/10.1007/978-3-642-02538-9_9.

López-Ibáñez, M., & Stützle, T. (2012). The automatic design of multi-objective ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation, 16*, 861–875. https://doi.org/10.1109/TEVC.2011.2182651.

Lust, T., & Teghem, J. (2010). Two-phase Pareto local search for the biobjective traveling salesman problem. *Journal of Heuristics, 16*, 475–510. https://doi.org/10.1007/s10732-009-9103-9.

Miettinen, K. (1999). *Nonlinear multiobjective optimization*. Kluwer Academic Publishers.

Paquete, L., Schiavinotto, T., & Stützle, T. (2007). On local optima in multiobjective combinatorial optimization problems. *Annals of Operations Research, 156*, 83–97. https://doi.org/10.1007/s10479-007-0230-0.

Radulescu, A., López-Ibáñez, M., & Stützle, T. (2013). Automatically improving the anytime behaviour of multiobjective evolutionary algorithms. In R. C. Purshouse, P. J. Fleming, C. M. Fonseca, S. Greco, & J. Shaw (Eds.), *Evolutionary multi-criterion optimization, EMO*. In *Lecture Notes in Computer Science: vol. 7811* (pp. 825–840). Springer, Heidelberg, Germany. https://doi.org/10.1007/978-3-642-37140-0_61.

R Development Core Team (2008). R: A language and environment for statistical computing. R Foundation for Statistical ComputingVienna, Austria. http://www.R-project.org.

Trianni, V., & López-Ibáñez, M. (2015). Advantages of task-specific multi-objective optimisation in evolutionary robotics. *PLoS One, 10*, e0136406. https://doi.org/10.1371/journal.pone.0136406.

Tušar, T., & Filipič, B. (2014). Visualizing exact and approximated 3D empirical attainment functions. *Mathematical Problems in Engineering, 2014*. Article ID 569346, 18 pages

Zhang, T., Georgiopoulos, M., & Anagnostopoulos, G. C. (2015). SPRINT: Multi-objective model racing. In S. Silva, & A. I. Esparcia-Alcázar (Eds.), *Proceedings of the genetic and evolutionary computation conference, GECCO* (pp. 1383–1390). New York, NY: ACM Press. https://doi.org/10.1145/2739480.2754791.

Zitzler, E., Brockhoff, D., & Thiele, L. (2007). The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration. In S. Obayashi (Ed.), *Evolutionary multi-criterion optimization, EMO*. In *Lecture Notes in Computer Science: vol. 4403* (pp. 862–876). Springer, Heidelberg, Germany. https://doi.org/10.1007/978-3-540-70928-2_64.

Zitzler, E., Knowles, J. D., & Thiele, L. (2008). Quality assessment of Pareto set approximations. In J. Branke, K. Deb, K. Miettinen, & R. Słowiński (Eds.), *Multi-objective optimization: Interactive and evolutionary approaches*. In *Lecture Notes in Computer Science: vol. 5252* (pp. 373–404). Springer, Heidelberg, Germany. https://doi.org/10.1109/TEVC.2009.2016569.

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

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Grunert da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation, 7*, 117–132. https://doi.org/10.1109/TEVC.2003.810758.