InDM2: Interactive Dynamic Multi-Objective Decision Making Using Evolutionary Algorithms

Antonio J. Nebro[a], Ana B. Ruiz[b], Cristóbal Barba-González[a], José García-Nieto[a,*], Mariano Luque[b], José F. Aldana-Montes[a]

[a] *Departamento de Lenguajes y Ciencias de la Computación, Ada Byron Research Building, University of Málaga, 29071 Málaga, Spain*
[b] *Department of Applied Economics (Mathematics), University of Málaga, Campus El Ejido, 29071 Málaga, Spain*

Keywords:

Dynamic multi-objective optimization
Multiple criteria decision making
Preferences
Evolutionary algorithms
jMetalSP

A B S T R A C T

Dynamic optimization problems involving two or more conflicting objectives appear in many real-world scenarios, and more cases are expected to appear in the near future with the increasing interest in the analysis of streaming data sources in the context of Big Data applications. However, approaches combining dynamic multi-objective optimization with preference articulation are still scarce. In this paper, we propose a new dynamic multi-objective optimization algorithm called InDM2 that allows the preferences of the decision maker (DM) to be incorporated into the search process. When solving a dynamic multi-objective optimization problem with InDM2, the DM can not only express her/his preferences by means of one or more reference points (which define the desired region of interest), but these points can be also modified interactively. InDM2 is enhanced with methods to graphically display the different approximations of the region of interest obtained during the optimization process. In this way, the DM is able to inspect and change, in optimization time, the desired region of interest according to the information displayed. We describe the main features of InDM2 and detail how it is implemented. Its performance is illustrated using both synthetic and real-world dynamic multi-objective optimization problems.

1. Introduction

Multi-objective optimization with evolutionary algorithms and other metaheuristics has been an active research field over the last 20 years, as these techniques have shown their effectiveness to solve real-world problems in many fields [1,2]. Although most work has considered static multi-objective optimization problems (MOPs), there is a growing interest in MOPs that change somehow over time, i.e., dynamic MOPs or DMOPs [3]. A number of proposals of multi-objective optimization metaheuristics for solving DMOPs have recently been proposed [4–9].

Some examples of DMOPs are problems associated with the planning of routes in logistics (e.g., vehicle routing problem - VRP), which take into account real-traffic information that can lead to variations in some links due to congestions or accidents, or problems related to the packaging of goods for delivery, in which orders are continuously arriving. In the first example, typical objectives to optimize are the time and the distance, while in the second the optimization goals are usually to reduce the number of bins and to balance the loads.

When dealing with DMOPs, whenever there exist changes in the environment that affect the solutions of the problem (i.e., the Pareto set, the Pareto front, or both), hence in the fitness landscape, the optimization algorithm must react to adapt the search to the new features of problem [10]. This means that a dynamic multi-objective optimization metaheuristic must be able to detect when the problem changes and to apply a strategy to cope with the changes.

Although the main goal of multi-objective optimization metaheuristics is to find a set of non-dominated solutions with the features of convergence and diversity with respect to the Pareto front of the problem at hands, the final aim when solving any MOP (static or dynamic) is to identify the Pareto optimal solution that best suits the expectations of the decision maker (DM). To this end, a Pareto front approximation can be of great help, since it gives information about the problem itself

* Corresponding author.

E-mail addresses: antonio@lcc.uma.es (A.J. Nebro), abruiz@uma.es (A.B. Ruiz), cbarba@lcc.uma.es (C. Barba-González), jnieto@lcc.uma.es (J. García-Nieto), mluque@uma.es (M. Luque), jfam@lcc.uma.es (J. F. Aldana-Montes).

(i.e., the ranges of the objective functions and the conflict degree among them). However, selecting the most preferred Pareto optimal solution, analyzing and comparing a large number of solutions at the same time may be cognitively demanding for the DM, especially in the presence of many objectives. Indeed, it may be computationally expensive to generate a large number of solutions approximating the whole Pareto front, particularly when dealing with real-life problems, which may even be a wasted effort if the DM is interested in just a subset of solutions located in a particular region.

One possibility to alleviate these problems is to incorporate some information about the preferences of the DM into the multi-objective optimization metaheuristic in order to progressively focus the search process onto the subset of solutions which correspond to these preferences (known as the *region of interest*). Thus, rather than approximating the whole Pareto front, the main idea is to approximate only the region of interest.

Whereas handling preferences has been widely studied and applied for static MOPs under the research field of *Multiple Criteria Decision Making* (MCDM) [11,12], it has barely been explored in dynamic multi-objective optimization. As stated in Ref. [13], few studies have been conducted in the field of dynamic multi-objective optimization regarding approaches that introduce decision-making or preference information into the search process. This means more research is required to develop metaheuristics for dynamic multi-objective optimization that can efficiently handle preferential information. However, in a context where the Pareto front can change over time, specifying the preferences which determine the region of interest can be difficult due to the fact that the problem itself is constantly changing. Thus, a multi-objective optimization metaheuristic aimed at dealing with DMOPs and preferences should simultaneously pay particular attention to these two features: (1) visualization, to provide the DM with a graphical picture of the Pareto front approximations that are being found over time; and (2) interactivity, to allow the DM to specify new information to redefine his/her preferences (i.e., to indicate a new region of interest) according to the knowledge (s)he gains while interacting with the solution process. In this paper, we propose InDM2 (*Interactive Dynamic Multi-Objective Decision Making* using Evolutionary Algorithms), a new algorithmic proposal that fulfills these requirements.

InDM2 is the result of our experience in two fields: dynamic multi-objective optimization and MCDM. We are the developers of jMetalSP [14], a Java-based software system for dynamic multi-objective optimization with metaheuristics, which combines the jMetal framework [15,16] and the Apache Spark cluster computing system [17]. jMetalSP provides a platform that facilitates the implementation of DMOPs and the development of dynamic multi-objective optimization algorithms thanks to the reusing of the components and resources included in jMetal. The architecture of jMetalSP enables streaming data sources to be easily incorporated, whose analysis often leads to changes in a DMOP, and to add software components that receive the Pareto front approximations that the algorithms produce.

In addition, we have a strong background in MCDM and we are the designers of WASF-GA [18], a preference-based evolutionary multi-objective optimization algorithm for solving (static) MOPs. This algorithm allows the preferences of the DM to be incorporated into the optimization process, which are expressed using the reference point preferential scheme [19]. A reference point is formed by desirable aspiration values that the DM would like to reach for each of the objective functions. Internally, in WASF-GA, the reference point determines the region of interest to be approximated. From a practical point of view, the approximation set is generated by projecting the reference point onto the Pareto front using a set of evenly distributed projection directions.

Our proposal is to embed a reference point-based multi-objective optimization evolutionary algorithm into InDM2, in order to delegate the solution process of the DMOP to it. In particular, we have considered a dynamic version of two reference point-based evolutionary algorithms: the aforementioned WASF-GA and R-NSGA-II [20]. The R-NSGA-II algorithm considers as preferential information one or several reference points and modifies the crowding distance and the niching operator of NSGA-II [21] to emphasize solutions close to the reference point(s) given by the DM during the solution process.

InDM2 incorporates a strategy that enables the DM to interactively update the reference point when solving a DMOP, if desired. Also, it offers a mechanism to visualize "on the fly" the approximations of the region of interest that are being generated throughout the solution process. Additionally, we have implemented InDM2 with jMetalSP, so all the features of this framework are incorporated in the proposed algorithm, including the possibility of using Big Data technologies, such as Apache Spark. This allows, for example, access to data stored in HDFS (the Hadoop file system [22]) so as to use all the computing power of Hadoop clusters.

The main contributions of this paper can be summarized as follows:

- We present a novel algorithm, InDM2, which combines dynamic multi-objective optimization, multiple criteria decision making and interactivity. A key feature in InDM2 is the mechanism to visualize, in optimization time, the approximations of the region of interest that are being generated throughout the solution process, together with the reference point driving these approximations.
- Two reference point-based evolutionary multi-objective optimization algorithms are incorporated inside InDM2 to handle the preference information interactively. However, any other reference point-based evolutionary algorithms can be used in InDM2 as the optimization solver.
- The architecture of InDM2 allows the incorporation of strategies for reacting to changes in both the problem and the reference point.
- The proposed approach is implemented in the jMetalSP framework, so it can be used in the context of Big Data optimization by taking advantage of the features of distributed streaming data processing of Apache Spark.
- The implementation of InDM2 is open-source, so it is freely available in the jMetaSP repository at GitHub[1].
- The performance of InDM2 is validated with three FDA benchmark DMOPs [10] and with a real-world problem, which is a dynamic version of a bi-objective Traveling Salesman Problem (TSP) [23] based on real traffic data provided by the New York City Department of Traffic.

The rest of this paper is structured as follows. Section 2 explains the main background concepts and presents a review of related work in the specialized literature. In Section 3, the InMD2 algorithm is detailed. Section 4 describes the experimental use cases carried out in terms of validation. Finally, concluding remarks and future lines of research are presented in Section 5.

2. Background concepts

In order to make this paper more self-contained, this section describes the main background concepts concerning dynamic multi-objective optimization and preference handling. A review of the related works in the specialized literature is also provided. In addition, the preference-based multi-objective optimization evolutionary algorithms considered (WASF-GA and R-NSGA-II) and the jMetalSP framework are briefly described.

2.1. Concepts and notation

A *dynamic multi-objective optimization problem* (DMOP) is a time-dependent problem that can be formulated as follows:

[1] https://github.com/jMetal/jMetalSP.

$$\begin{array}{ll} \text{minimize} & \{f_1(\mathbf{x},t),\ldots,f_k(\mathbf{x},t)\} \\ \text{subject to} & \mathbf{x}\in S_t, \end{array} \quad (1)$$

where $t \in \mathbb{R}$ is the time variable ($t \geq 0$), $f_i : S_t \subseteq \mathbb{R}^n \to \mathbb{R}$, for $i = 1, \ldots, k$ ($k \geq 2$), are the objective functions to be minimized simultaneously over the dynamic feasible set S_t in the decision space $\mathbb{R}^n$, which is formed by solutions or decision vectors $\mathbf{x} = (x_1, \ldots, x_n)^T$. In the objective space $\mathbb{R}^k$, the solutions are objective vectors $\mathbf{f}(\mathbf{x}, t) = (f_1(\mathbf{x}, t), \ldots, f_k(\mathbf{x}, t))^T$, for $\mathbf{x} \in S_t$, belonging to the feasible objective region $Z_t = \mathbf{f}(S_t)$.

Similarly to static multi-objective optimization, let us briefly define some basic concepts. Because of the degree of conflict between the objective functions, it is very unlikely, if not impossible, to find a single solution where all of them can reach their individual optima. For this reason, the so-called *Pareto optimal solutions* focus the interest. In these solutions, no objective function can be improved without deteriorating, at least, one of the others. For a time value t, a solution $\mathbf{x} \in S_t$ is said to be *Pareto optimal* if and only if there is no other $\mathbf{x}' \in S_t$ such that $f_i(\mathbf{x}', t) \leq f_i(\mathbf{x}, t)$ for all $i = 1,\ldots, k$ and $f_j(\mathbf{x}', t) < f_j(\mathbf{x}, t)$ for at least one index j. In the objective space, the corresponding objective vector $\mathbf{f}(\mathbf{x}, t)$ is referred to as a *Pareto optimal objective vector*. The set of all Pareto optimal solutions in t is called the *Pareto set in t*, denoted by E_t, and the set of all Pareto optimal objective vectors in t is called the *Pareto front in t*, denoted by $\mathbf{f}(E_t)$. Additionally, given $\mathbf{z}, \mathbf{z}' \in Z_t$, we say that $\mathbf{z}$ *dominates* $\mathbf{z}'$ if and only if $z_i \leq z'_i$ for all $i = 1,\ldots, k$ and $z_j < z'_j$ for at least one index j. A *non-dominated set* refers to a set of solutions whose objective vectors are not dominated by any other of the solutions in that set.

Given a t value, the *nadir objective vector* $\mathbf{z}^{\text{nad}}_t = (z^{\text{nad}}_{t,1}, \ldots, z^{\text{nad}}_{t,k})^T$ and the *ideal objective vector* $\mathbf{z}^\star_t = (z^\star_{t,1}, \ldots, z^\star_{t,k})^T$ provide upper and lower bounds for the objective function values in the Pareto set E_t. Respectively, their components are given by $z^{\text{nad}}_{t,i} = \max_{\mathbf{x}\in E_t} f_i(\mathbf{x}, t)$ and $z^\star_{t,i} = \min_{\mathbf{x}\in E_t} f_i(\mathbf{x}, t)$ $(i = 1, \ldots, k)$. While the ideal objective vector can be easily obtained, the nadir objective vector is complicated to calculate because the set E_t is usually unknown and different approaches are used to estimate it [24,25].

2.2. Preferences in multi-objective optimization

A very common way to express preferences consists of specifying desirable objective function values, which constitute the components of a so-called reference point. This concept was introduced in Ref. [19] for static MOPs, although it can be easily adapted to DMOPs.

Given a t value, a *reference point* is given by $\mathbf{q}_t = (q_{t,1}, \ldots, q_{t,k})^T$, where $q_{t,i}$ is an aspiration level for the objective function $f_i(\cdot, t)$ provided by the DM, for all $i = 1,\ldots, k$. Usually, $\mathbf{q}_t$ is said to be *achievable* for problem (1) if $\mathbf{q}_t \in Z_t + \mathbb{R}^k_+$ (where $\mathbb{R}^k_+ = \{\mathbf{y} \in \mathbb{R}^k \mid y_i \geq 0 \text{ for } i = 1, \ldots, k\}$), that is, if either $\mathbf{q}_t \in Z_t$ or if $\mathbf{q}_t$ is dominated by a Pareto optimal objective vector in Z_t. Otherwise, the reference point is said to be *unachievable*, that is, not all of its aspiration levels can be achieved simultaneously (in some situations, a reference point is unachievable because some components cannot be achieved, although some others can be attained).

Using a reference point for a t value, an *achievement scalarizing function* (ASF) [19] can be formulated and minimized to find the Pareto optimal solution that best satisfies the expectations of the DM at the instant t. For a reference point $\mathbf{q}_t$ and a vector of weights $\boldsymbol{\mu}_t = (\mu_{t,1}, \ldots, \mu_{t,k})^T$, with $\mu_{t,i} > 0$ ($i = 1,\ldots, k$), we can consider the ASF proposed in Ref. [19] given by:

$$s(\mathbf{q}_t, \mathbf{f}(\mathbf{x}, t), \boldsymbol{\mu}_t) = \max_{i=1,\ldots,k} \left\{ \mu_{t,i}(f_i(\mathbf{x}, t) - q_{t,i}) \right\} + \rho \sum_{i=1}^{k} \mu_{t,i}(f_i(\mathbf{x}, t) - q_{t,i}), \quad (2)$$

which must be minimized over S_t:

$$\begin{array}{ll} \text{minimize} & s(\mathbf{q}_t, \mathbf{f}(\mathbf{x}, t), \boldsymbol{\mu}_t) \\ \text{subject to} & \mathbf{x}\in S_t. \end{array} \quad (3)$$

The optimal solution of (3) is always a Pareto optimal solution of (1) and that any (properly[2]) Pareto optimal solution of (1) can be obtained by solving problem (3) and varying the reference point and/or the weight vector [11]. The so-called augmentation coefficient ρ, which must have a small real positive value, is used to ensure that the solution of (3) is Pareto optimal, and not weakly Pareto optimal[3]. Note that the optimal solution of (3) does not only depend on the reference point considered, but also on the vector of weights used. Furthermore, for the same reference point, Pareto optimal solutions generated using different weights are actually different [26–29].

2.3. Related work

In Ref. [13], a revision of key challenges and future trends in dynamic multi-objective optimization is presented, where the authors mention decision-making as one of the future research lines. They cite only three papers [30–32] that incorporate decision-making or preferences of the DM into a dynamic multi-objective optimization context. Recently [33], has suggested using a set of reference points in NSGA-II-DE [34] for tracking the changing Pareto front of DMOPs, but the reference points are not used in a preferential way in this proposal.

It is evident that some ideas of interactive MCDM methods for MOPs [26,35–37], as well as concepts of preference-based evolutionary multi-objective optimization algorithms, can be used to determine an efficient way to incorporate preferences interactively in dynamic multi-objective optimization. In fact, each iteration of a classical interactive method for static multi-objective optimization can be considered as a DMOP, in which the only change at each instant t is the new preferential information provided by the DM. However, while the DM can easily learn about a static MOP at each iteration of an interactive method, it is more difficult for the DM to cope with an interactive decision-making process for solving a DMOP given that the problem itself is changing over all the time. In addition, another challenge to be met when handling preferences in DMOP is the definition of an approach that can approximate the region of interest, taking into account the changes that this region experiments over time, even if the preferential information has not been modified. Finally, it is also worth mentioning that the algorithm must react not only when a component of the DMOP changes but also when the DM elicits new preferences.

2.4. Brief descriptions of WASF-GA and R-NSGA-II

As mentioned, WASF-GA (the *Weighting Achievement Scalarizing Function Genetic Algorithm*) [18] is a preference-based evolutionary algorithm, which considers a reference point $\mathbf{q}$ as preferential information. It approximates the *region of interest of the Pareto front defined by* $\mathbf{q}$ which, in accordance with [18], is determined as follows. When $\mathbf{q}$ is achievable, the region of interest is the subset of Pareto optimal objective vectors that dominate it. However, if $\mathbf{q}$ is unachievable, the region of interest is formed by the Pareto optimal objective vectors which are dominated by it; in this case, solutions lying in this region are likely to be more appealing for the DM than the ones outside it because, in them, although the objective function values differ from the aspiration levels as little as possible, none of them are improved. The solutions outside this region may improve some of the aspiration levels (but not all of them), at the expense of a sacrifice in the rest of them, which may not be so attractive for the DM.

[2] *Properly Pareto optimal solutions* are Pareto optimal solutions with bounded trade-offs between the objectives.

[3] A solution $\mathbf{x} \in S_t$ is *weakly Pareto optimal* if there does not exist another $\boldsymbol{x}' \in S_t$ such that $f_i(\mathbf{x}', t) < f_i(\mathbf{x}, t)$, for all $i = 1,\ldots, k$.

To approximate the region of interest in WASF-GA, a sample of N_μ weight vectors in $(0, 1)^k$ are considered and, at each generation, the solutions which minimize the ASF given in (2) for the reference point $\mathbf{q}$ and each of the weight vectors are emphasized. In general terms, at each generation of WASF-GA, parents and offspring are classified into several fronts according to the values that each individual takes for (2). To be more precise, the first front is formed by those solutions reaching the lowest value of (2) for each of the N_μ weight vectors. The second front is constituted by the individuals with the next lowest value of (2) for each of the N_μ weight vectors, and so on until every individual has been classified. Afterwards, the population for the next generation is formed by the solutions in the lowest level fronts. To some extent, these solutions can be considered as the best individuals at the current generation for minimizing the ASF (2) with respect to the N_μ weight vectors used. In practice, the use of this ASF in WASF-GA implies that the reference point is projected onto the Pareto front at each generation, taking into account the N_μ weight vectors [11]. Thus, to preserve diversity, these weight vectors are generated so that they define evenly distributed projection directions in the objective space. For more details, see Ref. [18].

In the case of R-NSGA-II (the *Reference-Point-Based NSGA-II* algorithm) [20], it is also a reference point-based evolutionary algorithm which modifies NSGA-II in the way the individuals of the last non-dominated front are selected to be passed to the new population. The DM gives one or several reference points and the crowding distance used in NSGA-II is replaced by a preference distance, which equally emphasizes solutions whose objective vectors are close to any of the given reference points with respect to the Euclidean distance. Additionally, the niching operator is updated to control the distribution of the emphasized solutions and, by means of a parameter, very close solutions are assured to be represented by just one of them.

2.5. *The jMetalSP framework*

jMetalSP is a Java-based multi-objective optimization framework aimed at solving dynamic multi-objective Big Data optimization problems [14]. The motivation of developing this software platform has to do with the fact that many DMOPs are found in areas such as economics, engineering, computer science, logistics, etc., and these fields are also the source of Big Data applications, so it seems clear that Big Data variants of DMOPs will be common in the near future.

With these ideas in mind, jMetalSP combines the jMetal multi-objective optimization framework with Apache Spark. The former provides a large number of state-of-the-art metaheuristics, while the latter has a series of features that are useful in Big Data applications. These include a high level parallel programming model, streaming data processing from different sources, access to diverse data sources (HDFS, Cassandra, HBase, etc.), and machine learning algorithms. jMetalSP has an object-oriented architecture that allows dynamic versions of the algorithms included in jMetal to be developed and DMOPs to be implemented, which change as a consequence of the analysis of data arriving from streaming sources. We have used this feature to create a dynamic version of the WASF-GA and the R-NSGA-II algorithms, constituting the basis of our proposal.

The skeleton of a jMetalSP application is shown in Code Snippet 1. We can observe that an application is composed of a number of elements. The streaming runtime is the underlying streaming engine, which may currently be Apache Spark or Java threads. Then, the problem and the algorithm must be included. After that, one or more streaming data source entities can be incorporated, each of them able to receive data in streaming, to analyze them and, as a result, to modify the problem. Finally, a number of algorithm data consumers can be indicated. These entities receive the Pareto front approximations that are computed by the algorithm and then the consumers can for e.g., store them in files, display the fronts in the screen, perform an analysis, etc. Once the application has been configured, it can be executed.

Code Snippet 1
Skeleton of a jMetalSP application.

```
jMetalSPApplication application;

application
  . setStreamingRuntime()
  . setProblem(new DynamicProblem())
  . setAlgorithm(new DynamicAlgorithm())
  . addStreamingDataSource(new DataSource1)
  . addStreamingDataSource(new DataSource2)
  . addAlgorithmDataConsumer(new DataConsumer1())
  . addAlgorithmDataConsumer(new DataConsumer2())
  . addAlgorithmDataConsumer(new DataConsumer3())
  . run();
```

3. Description of InDM2

InDM2 is an algorithm intended to solve DMOPs, which allows the DM to interactively specify the desired region of interest to be approximated by means of a reference point. To achieve this goal, a set of requirements that are enumerated hereafter must be satisfied.

The main component of InDM2 is a preference-based dynamic multi-objective optimization metaheuristic, which is the optimization engine. Our approach is based on dynamic versions of the WASF-GA and R-NSGA-II algorithms, which have been implemented in jMetalSP as follows.

The object-oriented architecture of jMetalSP allows reusing the template for evolutionary algorithms provided by jMetal (see Code Snippet 2), which has a `run()` method that closely mimics the pseudo-code of a generic evolutionary algorithm. The implementations of the dynamic versions of WASF-GA and R-NSGA-II follow this template, and thus developing InDM2 from WASF-GA and R-NSGA-II merely requires having to redefine two methods. Firstly, the behavior of the `isStoppingConditionReached()` method must be adapted, because instead of just terminating the algorithm, this method must make the approximation of the region of interest found available to the algorithm data consumers, and the underlying optimization algorithm (based on WASF-GA or R-NSGA-II) has to start again. Secondly, at the end of each iteration, the `updateProgress()` method typically increases an evaluation counter.

In InDM2, it additionally checks whether the problem and/or the reference point(s) have been modified and, in this case, a restart strategy is applied. The restart strategies proposed are explained bellow.

Code Snippet 2
`run` () method of class `AbstractEvolutionaryAlgorithm`.

```
public class AbstractEvolutionaryAlgorithm {
  ...
  public void run() {
    offspringPopulation;
    matingPopulation;
    population = createInitialPopulation();
    population = evaluatePopulation(population);
    initProgress();
    while (!isStoppingConditionReached()) {
      matingPopulation = selection(population);
      offspringPopulation = reproduction(matingPopulation);
      offspringPopulation = evaluatePop(offspringPopulation);
      population = replacement(population, offspringPopulation);
      updateProgress();
    }
  }
}
```

To allow the DM to update the reference point(s) “on the fly” while the algorithm is running, the current implementation of InDM2 reads the new reference point(s) from the keyboard (although other more sophisticated methods could be used instead). This event is detected by the `updateProgress()` method, as stated. To help the DM give a new reference point according to his/her preferences, taking into account the results obtained, we have included a graphical algorithm’s data consumer in jMetalSP. Therefore, the approximations of the region of interest generated by InDM2 are displayed as long as they are produced, so the DM can visualize them and change the reference point(s) as desired.

By default, InDM2 is forever working. This means that, if the DMOP does not change for a period of time, our algorithm will solve it again and again, using exactly the same problem configuration. As a consequence, if the algorithm converges most of the times, the same approximation of the region of interest (or a very similar one, given that we are using stochastic approaches) may be constantly displayed. This may be confusing for the DM since change may be perceived for a period of time. To avoid this situation, the graphical data consumer only shows a new approximation in the case it has significantly changed in comparison with the previous one displayed. To check this, we calculate the inverted generational distance quality indicator (I_{IGD}) [38] from the approximation of the region of interest currently generated, to the previous one, so that the new one is only displayed if the value of this indicator is lower than a threshold τ.

All the DMOPs available in jMetalSP have a method to inquire whether or not the problem has changed or not. Thus, InDM2 only has to call this method at the end of each iteration (i.e., in the `updateProgress()` method) to know if problem parameter has changed.

Once a change in the problem and/or the reference point(s) has been detected, a key point in InDM2 is the restarting approach, i.e., how the algorithm is restarted. We have adopted a flexible approach consisting in defining a restarting strategy according to two other sub-strategies: one for removing solutions from the population, and another for filling the population with new solutions. To fill the population, given a value $N' > 0$, the strategy followed is to randomly create N' new solutions, whereas to remove solutions, the following restarting strategies are currently available:

- To remove the first N' solutions.
- To remove N' randomly chosen solutions.
- To remove the worst N' solutions according to their crowding distance.
- To remove the worst N' solutions according to their contribution to the hypervolume [39] of the last approximation of the region of interest.

It is worth mentioning that other strategies can be incorporated (e.g., those included in Ref. [4]). Our scheme is very flexible and enables InDM2 to use a wide range of combinations of restarting approaches, which differ from each other in the way the algorithm reacts to updates in the problem or to modifications in the reference point.

For the sake of a better understanding, the pseudo-code of the InDM2 algorithm is shown in Algorithm 1. After the initialization phase (from line 1 to 14), the algorithm starts the infinite loop (line 15) that carries out multiple consecutive rounds of the dynamic optimization process and the result data consumption (visualization). Each optimization round (lines 16–25) entails a maximum number of iterations ($G_{\max}$) in which the selected preference-based evolutionary algorithm (i.e., WASF-GA or R-NSGA-II) is computed (line 17) and the restarting procedures are invoked, whenever a change in the reference point (line 18) or in the problem state (line 20) is detected. At the end of each round, the approximation of the region of interest found is sent to the data consumers (line 25) and the loop starts again.

Algorithm 1 Pseudocode of InDM2

```
1: $N$; // Population size
2: $N'$; // Number of replaced solutions
3: $G_{\max}$; // Maximum number of generations
4: $c$, $m$; // Genetic operators
5: $t \leftarrow 0$; // Generation counter
6: $A_t$; // Optimization problem state
7: $\mathbf{q}_t$; // Initial reference point(s)
8: $P$; // Preference-based evolutionary algorithm
9: $\varphi_q$; // Restart strategy when the reference point changes
10: $\varphi_p$; // Restart strategy when the problem changes
11: $M \leftarrow$ {WASF-GA, R-NSGA-II};// Base optimization algorithm
12: $P_t \leftarrow initializePopulation(N)$;
13: $evaluate(P_t, A_t)$;
14: $E_t \leftarrow initializeParetoSet(P_t)$;
15: while true do
16:   while $t < G_{\max}$ do
17:     $(P_{t+1}, E_{t+1}) \leftarrow compute(M, \mathbf{q}_t, c, m, P_t, A_t)$;
18:     if $\mathbf{q}_{t+1} \neq \mathbf{q}_t$ then
19:       $P'_{t+1} \leftarrow restart(P_{t+1}, \varphi_q, \mathbf{q}_{t+1}, N')$;
20:     else if $A_{t+1} \neq A_t$ then
21:       $P'_{t+1} \leftarrow restart(P_{t+1}, \varphi_p, A_{t+1}, N')$;
22:     end if
23:     $t \leftarrow t + 1$;
24:   end while
25:   $C$: $(E_{t+1}, \tau)$;// Notify Pareto front approximation to data consumers
26:   $t \leftarrow 0$;
27: end while
```

4. Use cases

To test the performance of InDM2 in practice, we have considered two scenarios: on the one hand, we have tested it with three continuous synthetic DMOPs and, on the other hand, we have also solved a bi-objective traveling salesman dynamic problem with real-world traffic data. This way, we cover both continuous and combinatorial DMOPs. In this study, our aim is to illustrate how InDM2 works in practice when interacting with a DM, so that we can show the actual potential and benefits of our proposal from a decision-making point of view. It is worth noting that traditional comparative tables with performance metrics are less meaningful when we focus on the interaction with the DM, rather than on the Pareto front approximations.

In this study, InDM2 has been run for the DMOPs mentioned and the preferences (i.e., the reference point) have been manually changed during the execution time. The idea is to observe how InDM2 is able to adapt the optimization process to the changes (both in the problem configuration and in the preferences), and to show the effect of these changes in the Pareto front approximations obtained, which must approximate only the regions of interest associated the reference points given.

Subsequently, the two configurations of InDM2 adopting WASF-GA and R-NSGA-II as the optimization algorithms are referred to as InDM2$_W$ and InDM2$_R$, respectively.

4.1. Case use 1: synthetic continuous DMOPs

The continuous DMOPs used belong to the FDA benchmark [10]. This family of DMOPs comprises five problems with different features depending on whether their Pareto front and/or their Pareto set change over time. Each FDA problem has its own formulation and all of them have a time dependence. The time t is defined according to equation (4):

$$t = \frac{1}{n_t} \left\lfloor \frac{\sigma}{\sigma_T} \right\rfloor, \tag{4}$$

where σ is the generation counter, σ_T is the number of generations for which t remains fixed and n_t is the number of distinct steps in t. For a real number a, $\lfloor a \rfloor$ denotes the largest integer not greater than a. In Ref. [10], the authors recommend to set $\sigma_T = 5$ and $n_t = 10$.

Table 1
Configuration of InDM2 to solve the FDA2, FDA3 and FDA5 problems.

Parameter settings	
Population Size	50/100 individuals (FDA2, FDA3/FDA5)
Selection of Parents	binary tournament
Recombination	simulated binary crossover (SBX)
Recombination probability	0.9
Mutation	polynomial
Mutation probability	$1/L$ (with L = number of variables)
Maximum number of evaluations	25,000
Restart policy when reference point changes	Remove 100% of solutions Create 100% of solutions randomly
Restart policy when problem changes	Remove 50% of solutions (hypervolume contribution) Create 50% of solutions randomly
Streaming sources	
Time counter generator (σ)	Frequency: 1 s
Algorithm data consumers	
Chart visualizer	Shows approximations and reference point
Front writer	Stores the generated approximations in files
Streaming runtime	
Mechanism	*Java threads*

To show the performance of InDM2, we have focused only on three DMOP instances of the FDA family: FDA2, FDA3 and FDA5. Their problem formulations are given and, following the advice given in Ref. [10], we have used 31 variables with $x_I = x_1$, $|x_{II}| = |x_{III}| = 15$ for the three of them. The reason behind selecting these problems is that both, their Pareto fronts and their Pareto sets change in time, that is, they are Type II [10]. Therefore, the dynamic changes in the Pareto front approximations caused by changes in the reference point can be easily visualized. Furthermore FDA2, in particular, has been used to illustrate the performance of both InDM2$_W$ and InDM2$_R$; FDA3 has been solved with InDM2$_W$ using just one reference point, but it has also been solved with InDM2$_R$ to show how our proposal works when the DM gives more than one reference point at the same iteration; and finally, we have used FDA5 to show the results provided by InDM2$_W$ when solving a dynamic optimization problem with three objectives.

We have configured InDM2 with the parameter settings summarized in Table 1. The population size is 50 for the FDA2 and FDA3 problems and 100 for the FDA5 instance. The stopping condition used to report results to the data consumers is to compute a maximum of 25,000 function evaluations. The variation operators are SBX crossover and polynomial mutation, and the selection operator is binary tournament [40]. The restarting strategy used after a change in the problem is detected consists of: firstly, removing 50% of the solutions according to the hypervolume contribution and, secondly, filling the population with new randomly created solutions. The restarting strategy when a new reference point is given is similar, but in this case all the solutions in the population (100%) are removed.

To generate the time events that will lead to changes in the FDA problems, we have used a simple data streaming source that counts every second, which implies that, according to equation (4) and the default value of σ_T, the problem is updated with a frequency of 5 s. Additionally, we have used two algorithm data consumers, one for showing the Pareto front approximations generated by InDM2 (as previously said in Section 3) and another one to store the approximations generated in files. In this study, where we only consider a simple streaming data source, there is no advantage of using Apache Spark, so we have considered the default thread-based runtime.

The first problem is FDA2, which is a dynamic bi-objective problem with Pareto fronts changing from convex to non-convex, and vice-versa, and Pareto sets also changing over time. Its problem formulation is:

$$\text{minimize} \quad f(\mathbf{x}, t) = \begin{cases} (f_1(\mathbf{x}_I), g(\mathbf{x}_{II}) \cdot h(\mathbf{x}_{III}, f_1(\mathbf{x}_I), g(\mathbf{x}_{II}), t)) \\ f_1(\mathbf{x}_I) = x_1 \\ g(\mathbf{x}_{II}) = 1 + \sum_{x_i \in \mathbf{x}_{II}} x_i^2 \\ h(\mathbf{x}_{III}, f_1, g, t) = 1 - \left(\dfrac{f_1}{g}\right)^{H_2(t)} \\ H_2(t) = (H(t) + \sum_{x_i \in \mathbf{x}_{II}} (x_i^2 - H(t))^2)^{-1} \\ H(t) = 0.75 + 0.75\sin(0.5\pi t), \;\; t = \dfrac{1}{n_t}\left\lfloor \dfrac{\sigma}{\sigma_T} \right\rfloor \\ \mathbf{x}_I = (x_1) \in [0,1], \;\; \mathbf{x}_{II}, \mathbf{x}_{III} \in [-1,1]. \end{cases} \tag{5}$$

To have an idea of the possible aspiration reference values that each objective can have during the search process, we have initially run InDM2$_W$ for FDA2 using two reference points: the ideal point (0.0, 0.0) (unachievable) and the nadir point (1.0, 1.0) (achievable). The approximations produced by InDM2$_W$ for the regions of interest of both reference points are displayed in Fig. 1. In this figure, the legends "*Front i*" label the approximations of the region of interest that are shown. Each of these legends implicitly indicates in which iteration i a significant improvement in the approximation generated was detected in compar-

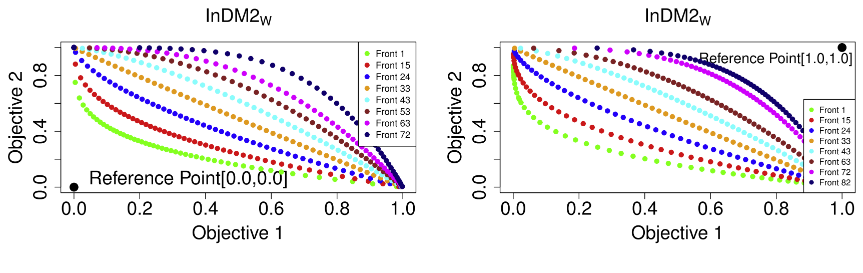

Fig. 1. Approximations of the regions of interest found by InDM2$_W$ for the FDA2 problem, using as reference points (0.0, 0.0) (left) and (1.0, 1.0) (right).

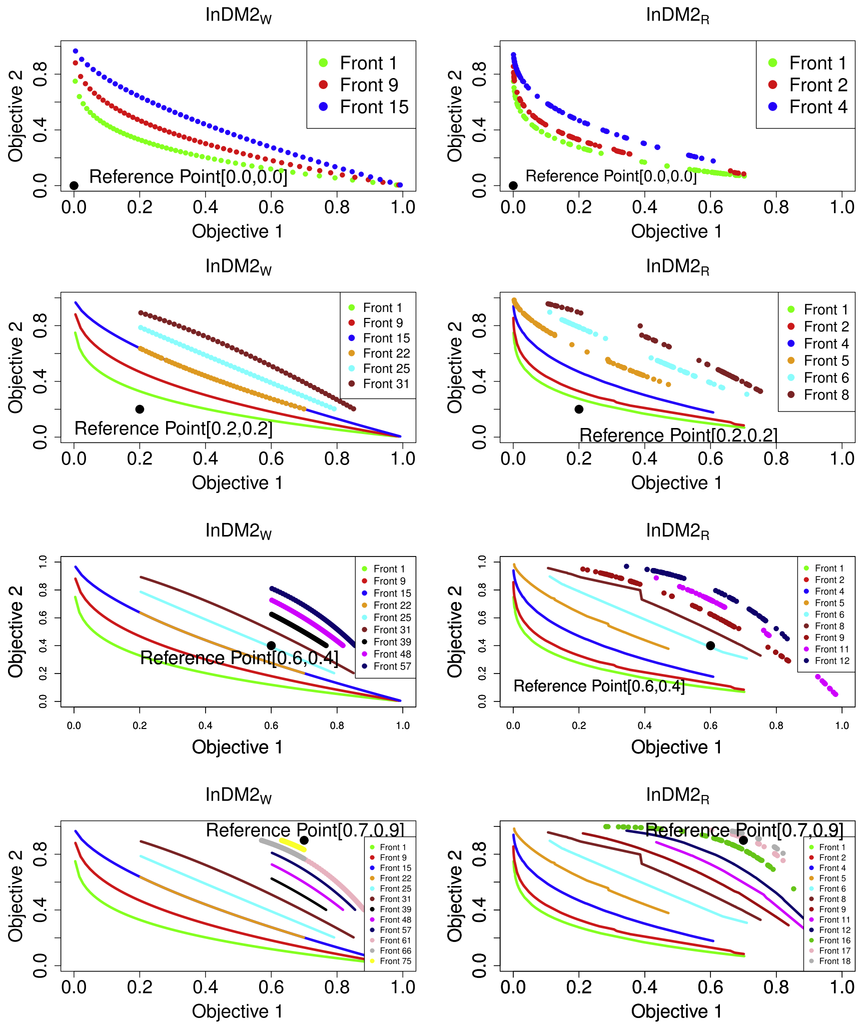

Fig. 2. Approximations of the region of interest found by InDM2$_W$ (left) and InDM2$_R$ (right) for the FDA2 problem, using different reference points.

ison to the previous one displayed (as explained in Section 3, a new approximation is graphically shown only in case a significant improvement has been detected with regards to the I_{IGD} indicator).

In Fig. 1, we can observe that, when the region of interest approximated is convex, the distribution of the solutions in the approximations found is slightly more dense in the central regions than in the extreme ones, and they become evenly spread when the approximated region tends to be a line or non-convex. This is due to the distribution of the projection directions (i.e., weight vectors) internally used in WASF-GA, which influences the distribution of the solutions found. For both reference points, InDM2$_W$ has been able to approximate the region of interest (which, as it can be seen, is the complete Pareto front in both cases) by adapting the search process to the changes in the problem configuration.

Next, we have executed both InDM2$_W$ and InDM2$_R$ for FDA2 but, in this case, we have simulated a real scenario where the DM dynamically modifies the reference point "on the fly", throughout the optimization process. Fig. 2 shows the approximations found for the given reference points along the solution process using both InDM2$_W$ (left) and InDM2$_R$ (right).

Firstly, let us describe the interactive solution process followed using InDM2$_W$. Initially, assuming that the ranges of the true Pareto front of the FDA2 problem were unknown, the DM selected (0.0, 0.0) as the first reference point. After observing the first three approximations returned by InDM2$_W$ at iterations 5, 9, and 15 (see the top left plot in Fig. 2), the DM decided to give a new reference point to reduce the region of interest approximated and set it to (0.2, 0.2). We can observe the new approximations generated in the second plot to the left in Fig. 2. In this plot, the former approximations generated for the initial reference point (0.0, 0.0) are shown with solid lines, in order to be able to distinguish them from the new approximations found for the new reference point (0.2, 0.2), which are shown with dotted lines. Later, the DM again adjusted the desired region of interest by changing the reference point twice, and (s)he gave (0.6, 0.4) as the third reference point and (0.7, 0.9) as the fourth one. Respectively, the results obtained for these two reference points are shown in the third and fourth plots on the left in Fig. 2.

We have repeated the same solution process for FDA2 with InDM2$_R$ and the approximations obtained are shown in the rightmost plots in Fig. 2. At a glance, we can observe that the approximation of the regions of interest produced by InDM2$_W$ have a better diversity that those generated by InDM2$_R$. It can be seen that the solutions generated by InDM2$_W$ always belong to the region of interest delimited by the reference points, while some of those found by InDM2$_R$ also approximate areas out side of this region. This is a consequence of the different search capabilities (convergence and diversity regarding the region of interest) of WASF-GA and R-NSGA-II. This fact highlights the impact of the preference-based evolutionary algorithm internally used in InDM2 as the optimization machine.

According to Fig. 2, we can see that three of the reference points given in the interactive solution process are unachievable (points (0.0, 0.0), (0.2, 0.2) and (0.6, 0.4)) and one of them is achievable (point (0.7,

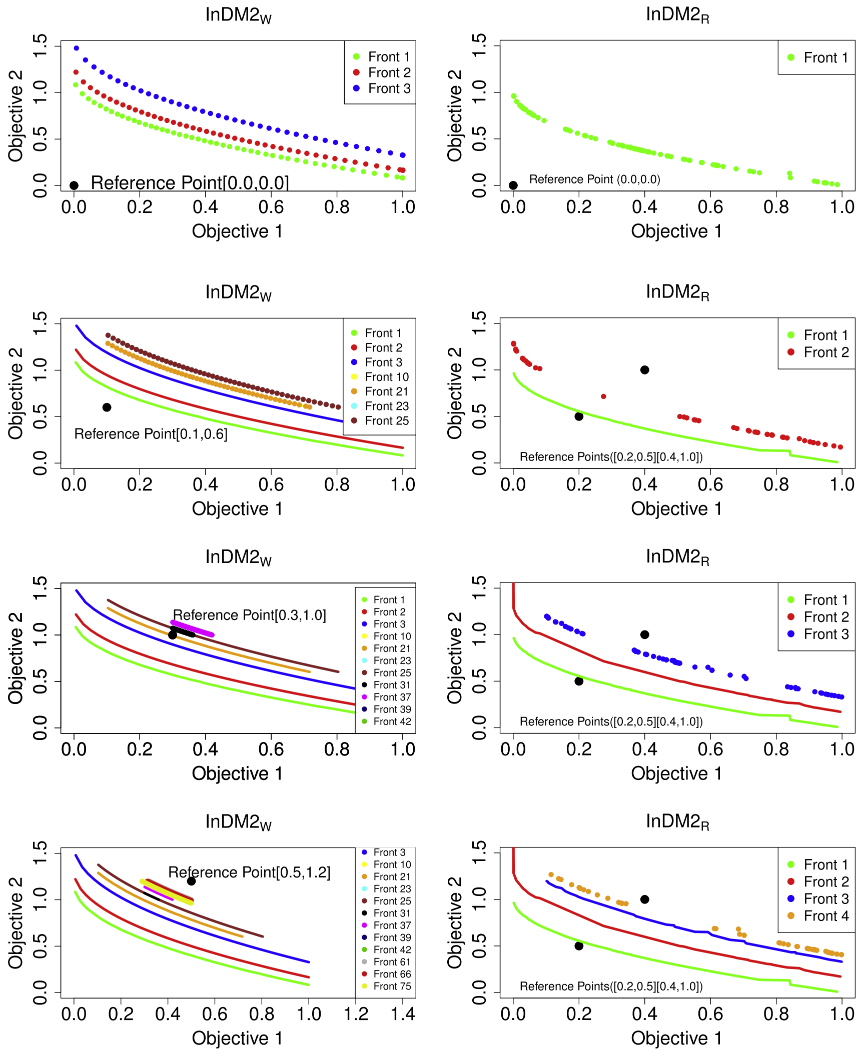

Fig. 3. Approximations of the region of interest found by InDM2$_W$ (left) and InDM2$_R$ (right) for the FDA3 problem, using different reference points.

0.9)). Note that, in practice, the achievability of the reference points used when interacting with the DM cannot be known beforehand. Only when the algorithm generates solutions and their dominance relationship with respect to the reference point used is checked, we can state if the reference point is achievable or not. The formulation of the dynamic bi-objective problem FDA3 is as follows:

$$\text{minimize}\ \ f(\mathbf{x},t) = \begin{cases} (f_1(\mathbf{x}_I,t), g(\mathbf{x}_{II},t)\cdot h(f_1(\mathbf{x}_I), g(\mathbf{x}_{II},t))) \\ f_1(\mathbf{x}_I,t) = \sum_{x_i\in\mathbf{x}_I} x_i^{F(t)} \\ g(\mathbf{x}_{II},t) = 1 + G(t) + \sum_{x_i\in\mathbf{x}_{II}} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\dfrac{f_1}{g}} \\ G(t) = |\sin(0.5\pi t)|,\ F(t) = 10^{2\sin(0.5\pi t)},\ t = \dfrac{1}{n_t}\left\lfloor \dfrac{\sigma}{\sigma_T} \right\rfloor \\ \mathbf{x}_I \in [0,1],\ \ \mathbf{x}_{II} \in [-1,1]. \end{cases} \tag{6}$$

This problem entails convex Pareto fronts and Pareto sets that change linearly. The f_1 function regulates the spread of the solutions in the objective space. Therefore, as analyzed in Ref. [10], when f_1 changes over time, the spread of solutions in the Pareto front approximations also change over time.

Fig. 3 shows the approximations found by InDM2$_W$ (left) and InDM2$_R$ (right) when solving FDA3 for different reference points. As in the previous example, these plots illustrate the impact on the generated approximations caused by the interactive modification of the reference point delimiting the region of interest to be approximated.

Regarding the interaction with the DM for InDM2$_W$, it can be seen that the reference point was progressively adjusted from (0.0, 0.0) (the first one used, shown in the first plot on the left in Fig. 3) to (0.5, 1.2) (the last one given, shown in the last plot on the left in Fig. 3). Observe that the approximations found by InDM2$_W$ have properly adapted to the changes in the region of interest, which has progressively shrank according to the preferences of the DM.

From a different point of view, to show the performance of InDM2 when using several reference points at the same time, we have simulated the use of two reference points in the interactive solution process of InDM2$_R$ with the FDA3 problem. Initially, the DM started with just one reference point, namely (0.0, 0.0), and once (s)he examined the results (first plot to the right in Fig. 3), (s)he decided to further delimit the region of interest by giving two new reference points: (0.2, 0.5) and (0.4, 1.0), at the same time. The new approximations found using these two points can be seen in the second, third and fourth plots on the right in Fig. 3. In this regard, we can observe that, as for the FDA2 problem, the solutions produced by InDM2$_R$ approximate a wider area than the region of interest, which is due to the search capabilities of R-NSGA-II.

Note that InDM2$_W$ has not been run for FDA3 with two reference points because the WASF-GA algorithm is not designed to consider several reference points when solving MOPs. As pointed out,

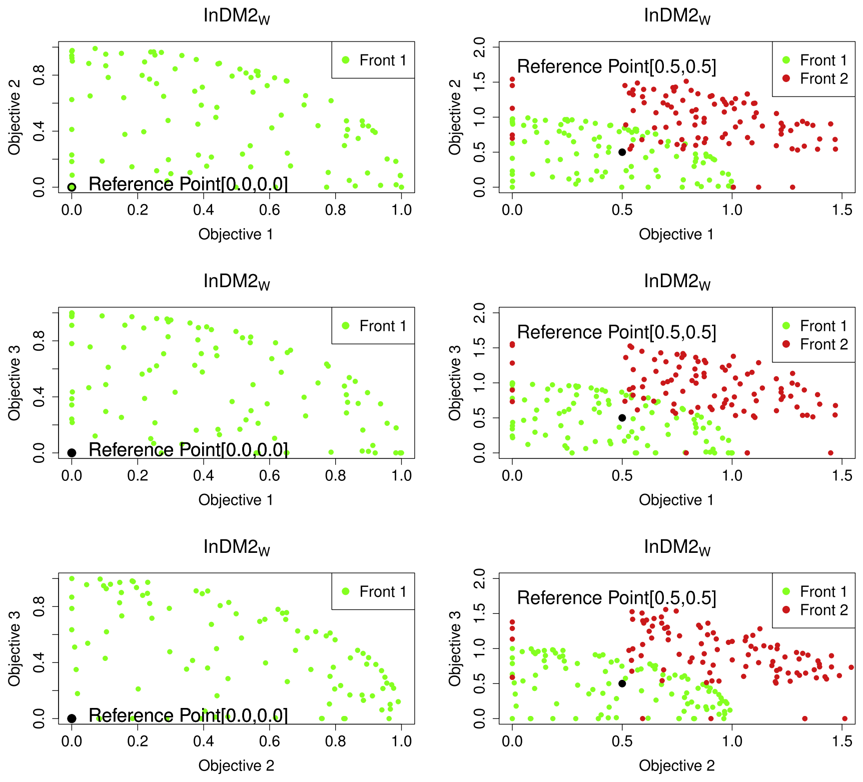

Fig. 4. Approximations of the region of interest found by InDM2$_W$ for the three-objective FDA3 problem, using as reference points (0.0, 0.0, 0.0) (left) and (0.5, 0.5, 0.5) (right).

the optimization algorithm used internally plays an important role with respect to the performance and possibilities of InDM2. In this case, the use of R-NSGA-II instead of WASF-GA has enabled the DM to give two reference points to further delimit the region of interest according to his/her preferences, although the approximations found by InDM2$_R$ have not completely adjusted to the desired region of interest.

Let us continue with the dynamic three-objective optimization problem FDA5. Its Pareto fronts change over time, although all of them are convex, and its Pareto sets change linearly.

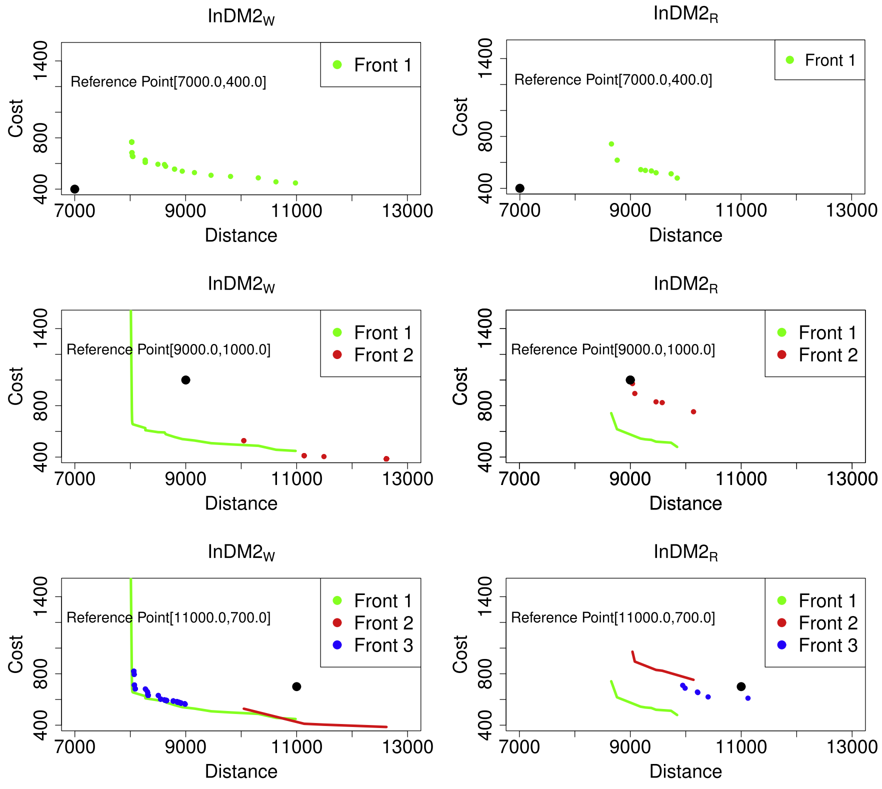

Fig. 5. Approximations of the region of interest found by InDM2$_W$ (left) and InDM2$_R$ (right) for the dynamic TSP, using different reference points.

The formal definition of FDA5 as a dynamic multi-objective optimization problem is the following one:

$$\text{minimize } f(\mathbf{x},t) = \begin{cases} (f_1(\mathbf{x}, g(\mathbf{x}_{II},t)), \dots, f_k(\mathbf{x}, g(\mathbf{x}_{II},t))) \\ f_1(\mathbf{x},g,t) = (1+g(\mathbf{x}_{II},t)) \prod_{i=1}^{M-1} \cos(\frac{y_i \pi}{2}) \\ f_k(\mathbf{x},g,t) = (1+g(\mathbf{x}_{II},t)) (\prod_{i=1}^{M-1} \cos(\frac{y_i \pi}{2})) \sin(\frac{y_{M-k+1}\pi}{2}), \\ \forall k = 2, \dots, M-1 \\ f_m(\mathbf{x},g,t) = (1+g(\mathbf{x}_{II},t)) \prod_{i=1}^{M-1} \sin(\frac{y_i \pi}{2}) \\ \textit{where}: \\ g(\mathbf{x}_{II},t) = G(t) + \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2, \\ t = \frac{1}{n_t} \left\lfloor \frac{\sigma}{\sigma_T} \right\rfloor, y_i = x_i^{F(t)}, \forall i = 1, \dots, M-1 \\ G(t) = |\sin(0.5\pi t)|\ ,\ F(t) = 1 + 100\sin^4(0.5\pi t), \\ \mathbf{x}_{II} = (x_M, \dots, x_n),\ x_i \in [0,1],\ \forall i = 2, \dots, n \end{cases} \tag{7}$$

To easily visualize the approximations found by InDM2$_W$ for FDA5, which belong to the three-dimensional objective space, we have used bi-dimensional images which show the values of each pair of objectives for all the solutions in the approximations (i.e. 2D projections of each pair of objectives). This can be seen in Fig. 4, where we show the approximations generated by InDM2$_W$ for FDA5 using (0.0, 0.0, 0.0) as the initial reference point (left) and (0.5, 0.5, 0.5) as the second given one (right), which shrank the desired region of interest. Note that all the approximations illustrated in Fig. 4 are constituted by non-dominated solutions in the three-dimensional objective space, although these bi-dimensional plots may wrongly show a domination among the solutions represented.

4.2. Dynamic Bi-Objective traveling salesman problem

Finally, we have tested the performance of InDM2 when solving a dynamic version of the Traveling Salesman Problem (TSP) [23] based on real-world data. The New York City Department of Traffic provides open real-time traffic speed data[4], which gives the length of the links, the mean speed and the mean traveling time of the cars traversing the two end points that define the links. This enables a realistic dynamic bi-objective optimization instance of the TSP to be defined, where the objectives to be minimized are the travel time and the distance. The available data are updated with a frequency of two or three times per minute, so it can actually be considered as "almost real-time data". This case study is based on the problem instance described in Ref. [14], which motivated the development of the jMetalSP framework.

To solve the resulting combinatorial dynamic bi-objective optimization problem, the solution encoding used has been a permutation of integer values representing the journeys. The variation operators applied are a swap mutation operator and a partial-mapped crossover (PMX) operator. The parameter settings used are the following: the population size is 100; the algorithm performs 250,000 function evaluations before writing out the approximation found and re-starting; the crossover and mutation probabilities are 0.9 and 0.2, respectively. A directory is used as streaming data source, which is read every 10 s. As in the former case study, we have used the thread-based runtime.

Data are gathered from the open data website of the city of New York and incorporated to jMetalSP as a streaming data source. However, as we are interested in comparing the execution of both InDM2$_W$ and InDM2$_R$ for the same instance, we have previously acquired the traffic data and, in our study, the problem is always updated with the same data with the idea of always solving exactly the same dynamic instance of the TSP with both algorithms. The updated data consist of a code that indicates the distance or time matrices, the coordinate (row and column) to change, and the new value.

In Fig. 5, the results retrieved by InDM2$_W$ (left) and InDM2$_R$ (right) are shown. In both cases, we can observe how the approximations evolve according to the changes in the region of interest, when the DM interactively gives different reference points. (S)he started with an unachievable reference point (7000, 400), and later set more pessimistic values and subsequently used the reference points (9000, 1000) and (11000, 700), both of them achievable. In general, the solutions obtained by both, InDM2$_W$ and InDM2$_R$ approximated the regions of interest, but their diversities were not as high as desired. However, the ones generated by InDM2$_W$ better fit to the desired regions. Furthermore, it can be seen that some of the solutions generated by InDM2$_R$ are dominated by the ones found by InDM2$_W$.

4.3. Discussion

From the use cases studied in the previous section, we can observe that InDM2 has performed as expected, although it is worth discussing a number of issues that have emerged after analyzing the behavior of the algorithm.

The first issue to discuss is the way of assessing the performance of the search capabilities of InDM2. In the literature, whenever a new metaheuristic technique is proposed, the results obtained for a performance metric after several independent runs are usually analyzed and compared to other algorithms. For InDM2, we have not included such study for several reasons that we now discuss. First, to the best of our knowledge, there is no algorithm with the features of InDM2 (dynamic multi-objective optimization metaheuristic handling preferences of the DM in an interactive way), so we cannot compare it with other proposals. It must be noted that InDM2 employs as the optimization engine a dynamic version of the WASF-GA and the R-NSGA-II algorithms, but any other metaheuristic based on the reference point preferential scheme could be used instead, such as for e.g. r-NSGA-II [41]. From this point of view, a comparative performance study between several dynamic multi-objective optimization metaheuristics based on reference points would make sense, but this analysis is beyond the scope of this paper. Furthermore, traditional comparative tables are less meaningful when we mainly focus on the interaction with the DM, and not only on the Pareto front approximations. Finally, evaluating the performance of dynamic multi-objective optimization algorithms is not trivial and, although some quality indicators have been proposed [3,42], the inclusion of preferential information in InDM2 adds a new element that would possibly require having to define new quality indicators.

From an overall perspective, and in accordance with our study, we can conclude that the use of WASF-GA in InDM2 has led to approximations with more evenly spread solutions in comparison with those generated when using R-NSGA-II. In some cases, InDM2$_W$ solutions dominated InDM2$_R$ solutions. However, InDM2$_W$ can only handle a unique reference point as preferential information (given that WASF-GA does so), while InDM2$_R$ has allowed the DM to indicate more than one reference point in the same iteration (thanks to the characteristics of R-NSGA-II), and this can be helpful for a DM in several situations. This highlights that the internal optimization solver used in InDM2 conditions the capability of our proposal for using just one or several reference points at the same time.

We have designed InDM2 to be flexible enough to include different restarting strategies when changes in the problem and/or in the reference point are detected, but choosing the right schemes is an open issue that requires further research. The selection of the most appropriate strategies is problem-dependent and they may be more complicated to choose in case of real-world problems whose Pareto fronts are usu-

[4] At the time of writing this paper, the traffic data can be obtained from https://data.cityofnewyork.us/view/qkm5-nuaq.

ally unknown beforehand. Thus, performing a preliminary configuration study seems to be necessary for each problem so as to have some pointers as to the best strategies to be adopted.

It is worth mentioning that the frequency of change of the reference point is up to the DM. At any moment, the DM may or may not change it, according to his/her desires. As in any interactive method, the preferences of the DM evolve while (s)he actually interacts with the method and learns about the problem itself. The DM takes an active part in the solution process: (s)he iteratively sees information about the solutions available and expresses, fine-tunes and changes his/her preference information. With this, the DM can learn about what kinds of solutions are attainable (i.e. what kinds of trade-offs exist among the objectives), and (s)he can then adjust his/her own preferences based on insight gained about the problem.

During the interactive solution process followed in InDM2, the DM can analyze the approximations obtained for the current reference point and, after a few seconds, (s)he can provide a new reference point if (s)he feels that the results generated are not appealing enough, until converging to the desired region of Pareto optimal solutions.

The bi-objective problems we have used to illustrate the working of InDM2 have allowed us to validate our proposal. For visualizing the results obtained for the three-objective problem, we have made use of 2D projections of each pair of objectives. This is an initial proposal for graphically presenting the results to the DM in order to ease the study and analysis of the solutions found in DMOPs with more than two objectives, but further research must be done in this vein and so improving the visualization features is ongoing work. Note that helping the DM to fine-tune her/his preferences in a context where the problem itself is changing over time is not an easy task, especially in the presence of a high number of objectives.

An important feature of InDM2 has to do with its implementation on top of the jMetalSP framework. For the studies we have conducted in this paper, we have used a simple streaming source that returns the value of a counter, so we have not used Apache Spark. However, using this cluster computing system is transparent to InDM2 and the potential use of the Spark streaming engine is available, and would enable us to incorporate many data sources by using providers, such as sockets, files in a directory, Kafka, Flume or Kinesis[5]. A consequence of using Spark is the advantage of working with Big Data technologies, which allows us to run InDM2 in Hadoop clusters and to access huge amounts of data stored in HDFS. In this sense, it is worth mentioning that the streaming features of Spark can perform tasks (i.e. the analysis of the received data) in parallel in a transparent way.

Finally, we would like to highlight that any future feature of jMetalSP will be automatically available in InDM2. In addition, as jMetalSP is an open-source project, the source code of InDM2 is also available for those researchers interested in using it. This gives us the possibility of receiving feedback about bugs, improvements and contributions.

5. Conclusions

We have presented InDM2, an interactive multi-objective optimization metaheuristic for solving dynamic multi-objective optimization problems. It enables the DM to interactively change the region of interest that (s)he desires to approximate by giving and updating a reference point containing her/his preferences. Our proposal incorporates a reference point-based evolutionary algorithm as a component, currently including the WASF-GA and R-NSGA-II algorithms, which indeed allow the reference points to be changed during the optimization process. To assist the DM, the approximations obtained by the algorithm are shown in a graphical window. A key component of InDM2 is that its internal design allows specifying which different restarting strategies are to be applied when changes in the reference point and/or in the problem configuration are detected, making it more versatile.

InDM2 has been described in detail and its working procedure has been analyzed by solving three continuous DMOPs and a dynamic version of the combinatorial bi-objective optimization traveling salesman problem, built using real-world traffic data from New York City. The reported figures have shown how our algorithm behaves when the problem and the reference points change. We have given an idea of the full potential of InDM2, as a dynamic algorithm handling preferences interactively, which has been able to generate approximation adjusting to the given preferences (i.e., the region of interest), in real time, while the problem also changes at the same time. We have also discussed a number of open issues related to our proposal.

A lack of real-world applications is still an open issue to illustrate how new algorithms perform, especially for DMOPs in which t changes at a high rate. Thus, apart from the future research lines indicated in Section 4.3, we are interested in studying the performance of InDM2 when solving more complex real-world DMOPs using preferences and interactively changing them by a DM.

Acknowledgments

This work is partially funded by Grants TIN2017-86049-R, TIN2014-58304 and ECO2014-56397-P (Ministerio de Ciencia e Innovación), and P11-TIC-7529 and P12-TIC-1519 (Plan Andaluz I+D+I). Cristóbal Barba-González is supported by Grant BES-2015-072209 (Spanish Ministry of Economy and Competitiveness). Ana B. Ruiz and José García-Nieto are recipient of a Post-Doctoral fellowship of "Captación de Talento para la Investigación" at Universidad de Málaga.

References

[1] K. Deb, Multi-objective Optimization Using Evolutionary Algorithms, Wiley, Chichester, 2001.
[2] C.A.C. Coello, G.B. Lamont, D.A.V. Veldhuizen, Evolutionary Algorithms for Solving Multi-objective Problems, second ed., Springer, New York, 2007.
[3] C. Raquel, X. Yao, Dynamic multi-objective optimization: a survey of the state-of-the-art, in: S. Yang, X. Yao (Eds.), Evolutionary Computation for Dynamic Optimization Problems, Springer, 2013, pp. 85–106.
[4] S. Jiang, S. Yang, A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization, IEEE Trans. Evol. Comput. 21 (2017) 65–82.
[5] X. Chen, D. Zhang, X. Zeng, A stable matching-based selection and memory enhanced MOEA/D for evolutionary dynamic multiobjective optimization, in: Proceeding of the International Conference on Tools with Artificial Intelligence, 2015, pp. 478–485.
[6] Y. Jin, C. Yang, J. Ding, T. Chai, Reference point based prediction for evolutionary dynamic multiobjective optimization, in: Proceeding of the IEEE Congress on Evolutionary Computation, 2016, pp. 3769–3776.
[7] A. Muruganantham, K.C. Tan, P. Vadakkepat, Evolutionary dynamic multiobjective optimization via kalman filter prediction, IEEE Transactions on Cybernetics 46 (2016) 2862–2873.
[8] M.G. Martinez-Penaloza, E. Mezura-Montes, Immune generalized differential evolution for dynamic multiobjective optimization problems, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2015, pp. 1918–1925.
[9] Y. Wu, Y. Jin, X. Liu, A directed search strategy for evolutionary dynamic multiobjective optimization, Soft Computing 19 (2015) 3221–3235.
[10] M. Farina, K. Deb, P. Amato, Dynamic multiobjective optimization problems: test cases, approximations, and applications, IEEE Trans. Evol. Comput. 8 (2004) 425–442.
[11] K. Miettinen, Nonlinear Multiobjective Optimization, Kluwer Academic Publishers, Boston, 1999.
[12] A. Jaszkiewicz, J. Branke, Interactive multiobjective evolutionary algorithms, in: J. Branke, K. Deb, K. Miettinen, R. Slowinski (Eds.), Multiobjective Optimization, Interactive and Evolutionary Approaches, Volume 5252 of Lecture Notes in Computer Science, Springer, 2008, pp. 179–193.
[13] M. Helbig, K. Deb, A. Engelbrecht, Key challenges and future directions of dynamic multi-objective optimisation, in: Proceeding of the IEEE Congress on Evolutionary Computation, 2016, pp. 1256–1261.
[14] C. Barba-Gonzalez, J.M. Garcia-Nieto, A.J. Nebro, J.A. Cordero, J.J. Durillo, I. Navas-Delgado, J.F. Aldana-Montes, jMetalSP: a Framework for Dynamic Multi-objective Big Data Optimization, Applied Soft Computing, to appear, 2017.
[15] J.J. Durillo, A.J. Nebro, jMetal: a Java framework for multi-objective optimization, Adv. Eng. Software 42 (2011) 760–771.
[16] A.J. Nebro, J.J. Durillo, M. Vergne, Redesigning the jMetal multi-objective optimization framework, in: Proceedings of the Companion Publication of the Annual Conference on Genetic and Evolutionary Computation, 2015, pp. 1093–1100.

[5] http://spark.apache.org/docs/latest/streaming-programming-guide.html.

[17] M. Zaharia, M. Chowdhury, M.J. Franklin, S. Shenker, I. Stoica, Spark: cluster computing with working sets, in: Proceedings of the USENIX Conference on Hot Topics in Cloud Computing, 2010, pp. 1–7.
[18] A.B. Ruiz, R. Saborido, M. Luque, A preference-based evolutionary algorithm for multiobjective optimization: the weighting achievement scalarizing function genetic algorithm, J. Global Optim. 62 (2015) 101–129.
[19] A.P. Wierzbicki, The use of reference objectives in multiobjective optimization, in: G. Fandel, T. Gal (Eds.), Multiple Criteria Decision Making, Theory and Applications, Springer, 1980, pp. 468–486.
[20] K. Deb, J. Sundar, B. Ubay, S. Chaudhuri, Reference point based multi-objective optimization using evolutionary algorithm, Int. J. Comput. Intell. Res. 2 (2006) 273–286.
[21] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2002) 182–197.
[22] T. White, Hadoop: the Definitive Guide, O'Reilly Media, Inc., 2009.
[23] C.H. Papadimitriou, The Euclidean travelling salesman problem is NP-complete, Theor. Comput. Sci. 4 (1977) 237–244.
[24] K. Deb, K. Miettinen, Nadir point estimation using evolutionary approaches: better accuracy and computational speed through focused search, in: M. Ehrgott, B. Naujoks, T.J. Stewart, J. Wallenius (Eds.), Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems, Springer, 2010, pp. 339–354.
[25] K. Deb, K. Miettinen, S. Chaudhuri, Towards an estimation of nadir objective vector using a hybrid of evolutionary and local search approaches, IEEE Trans. Evol. Comput. 14 (2010) 821–841.
[26] M. Luque, K. Miettinen, P. Eskelinen, F. Ruiz, Incorporating preference information in interactive reference point methods for multiobjective optimization, Omega 37 (2009) 450–462.
[27] K. Miettinen, M.M. Mäkelä, Comparative evaluation of some interactive reference point-based methods for multi-objective optimisation, J. Oper. Res. Soc. 50 (1999) 949–959.
[28] K. Miettinen, M.M. Mäkelä, On scalarizing functions in multiobjective optimization, OR Spectrum 24 (2002) 193–213.
[29] F. Ruiz, M. Luque, J.M. Cabello, A classification of the weighting schemes in reference point procedures for multiobjective programming, J. Oper. Res. Soc. 60 (2009) 544–553.
[30] K. Deb, U.B. Rao, S. Karthik, Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling, in: S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, T. Murata (Eds.), Evolutionary Multi-criterion Optimization, Volume 4403 of Lecture Notes in Computer Science, Springer, 2007, pp. 803–817.
[31] R. Roy, J. Mehnen, Dynamic multi-objective optimisation for machining gradient materials, CIRP Ann. - Manuf. Technol. 57 (2008) 429–432.
[32] R. Liu, W. Zhang, L. Jiao, F. Liu, J. Ma, A sphere-dominance based preference immune-inspired algorithm for dynamic multi-objective optimization, in: Proceedings of the Annual Conference on Genetic and Evolutionary Computation, 2010, pp. 423–430.
[33] Y. Jin, C. Yang, J. Ding, T. Chai, Reference point based prediction for evolutionary dynamic multiobjective optimization, in: IEEE Congress on Evolutionary Computation, 2016, pp. 3769–3776.
[34] K. Deb, A robust evolutionary framework for multi-objective optimization, in: M. Keijzer (Ed.), Conference on Genetic and Evolutionary Computation, 2008, pp. 633–640.
[35] M. Luque, F. Ruiz, R.E. Steuer, Modified interactive Chebyshev algorithm (MICA) for convex multiobjective programming, Eur. J. Oper. Res. 204 (2010) 557–564.
[36] K. Miettinen, P. Eskelinen, F. Ruiz, M. Luque, NAUTILUS method: an interactive technique in multiobjective optimization based on the nadir point, Eur. J. Oper. Res. 206 (2010) 426–434.
[37] A.B. Ruiz, K. Sindhya, K. Miettinen, F. Ruiz, M. Luque, E-NAUTILUS: a decision support system for complex multiobjective optimization problems based on the NAUTILUS method, Eur. J. Oper. Res. 246 (2015) 218–231.
[38] H. Ishibuchi, H. Masuda, Y. Tanigaki, Y. Nojima, Modified Distance Calculation in Generational Distance and Inverted Generational Distance, Springer International Publishing, pp. 110–125.
[39] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (1999) 257–271.
[40] K. Deb, Salient issues of multi-objective evolutionary algorithms, in: K. Deb (Ed.), Multi-objective Optimization Using Evolutionary Algorithms, Wiley, 2001, pp. 315–445.
[41] L. Ben-Said, S. Bechikh, K. Ghedira, The r-dominance: a new dominance relation for interactive evolutionary multicriteria decision making, IEEE Trans. Evol. Comput. 14 (2010) 801–818.
[42] E. Tantar, A.A. Tantar, P. Bouvry, On dynamic multi-objective optimization, classification and performance measures, in: Proceedings of the IEEE Congress of Evolutionary Computation, 2011, pp. 2759–2766.